Discriminative, Generative and Imitative Learning

by

Tony Jebara

B.Eng., Electrical Engineering McGill University, 1996
M.Sc., Media Arts and Sciences, MIT, 1998

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY IN MEDIA ARTS AND SCIENCES
at the
Massachusetts Institute of Technology
February 2002

Signature of Author ______________________________

Program in Media Arts and Sciences
December 18, 2001

Certified by ______________________________

Alex P. Pentland
Toshiba Professor of Media Arts and Sciences
Program in Media Arts and Sciences
Thesis Supervisor

Accepted by ______________________________

Andrew B. Lippman
Chair
Departmental Committee on Graduate Students
Program in Media Arts and Sciences

Discriminative, Generative and Imitative Learning

by

Tony Jebara

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Media Arts and Sciences

Abstract

I propose a common framework that combines three different paradigms in machine learning: generative, discriminative and imitative learning. A generative probabilistic distribution is a principled way to model many machine learning and machine perception problems. Therein, one provides domain specific knowledge in terms of structure and parameter priors over the joint space of variables. Bayesian networks and Bayesian statistics provide a rich and flexible language for specifying this knowledge and subsequently refining it with data and observations. The final result is a distribution that is a good generator of novel exemplars.

Conversely, discriminative algorithms adjust a possibly non-distributional model to data optimizing for a specific task, such as classification or prediction. This typically leads to superior performance yet compromises the flexibility of generative modeling. I present Maximum Entropy Discrimination (MED) as a framework to combine both discriminative estimation and generative probability densities. Calculations involve distributions over parameters, margins, and priors and are provably and uniquely solvable for the exponential family. Extensions include regression, feature selection, and transduction. SVMs are also naturally subsumed and can be augmented with, for example, feature selection, to obtain substantial improvements.

To extend to mixtures of exponential families, I derive a discriminative variant of the Expectation-Maximization (EM) algorithm for latent discriminative learning (or latent MED). While EM and Jensen lower bound log-likelihood, a dual upper bound is made possible via a novel reverse-Jensen inequality. The variational upper bound on latent log-likelihood has the same form as EM bounds, is computable efficiently and is globally guaranteed. It permits powerful discriminative learning with the wide range of contemporary probabilistic mixture models (mixtures of Gaussians, mixtures of multinomials and hidden Markov models). We provide empirical results on standardized data sets that demonstrate the viability of the hybrid discriminative-generative approaches of MED and reverse-Jensen bounds over state of the art discriminative techniques or generative approaches.

Subsequently, imitative learning is presented as another variation on generative modeling which also learns from exemplars from an observed data source. However, the distinction is that the generative model is an agent that is interacting in a much more complex surrounding external world. It is not efficient to model the aggregate space in a generative setting. I demonstrate that imitative learning (under appropriate conditions) can be adequately addressed as a discriminative prediction task which outperforms the usual generative approach. This discriminative-imitative learning approach is applied with a generative perceptual system to synthesize a real-time agent that learns to engage in social interactive behavior.

Thesis Supervisor: Alex Pentland
Title: Toshiba Professor of Media Arts and Sciences, MIT Media Lab

Discriminative, Generative and Imitative Learning

by
Tony Jebara

Thesis committee:

Advisor: ____________________

Alex P. Pentland
Toshiba Professor of Media Arts and Sciences
MIT Media Laboratory

Co-Advisor: ____________________

Tommi S. Jaakkola
Assistant Professor of Electrical Engineering and Computer Science
MIT Artificial Intelligence Laboratory

Reader: ____________________

David C. Hogg
Professor of Computing and Pro-Vice-Chancellor
School of Computer Studies, University of Leeds

Reader: ____________________

Tomaso A. Poggio
Uncas and Helen Whitaker Professor
MIT Brain Sciences Department and MIT Artificial Intelligence Lab

Acknowledgments

I extend warm thanks to Alex Pentland for sharing with me his wealth of brilliant creative ideas, his ground-breaking visions for computer-human collaboration, and his ability to combine the strengths of so many different fields and applications. I extend warm thanks to Tommi Jaakkola for sharing with me his masterful knowledge of machine learning, his pioneering ideas on discriminative-generative estimation and his excellent statistical and mathematical abilities. I extend warm thanks to David Hogg for sharing with me his will to tackle great challenging problems, his visionary ideas on behavior learning, and his ability to span the panorama of perception, learning and behavior. I extend warm thanks to Tomaso Poggio for sharing with me his extensive understanding of so many aspects of intelligence: biological, psychological, statistical, mathematical and computational and his enthusiasm towards science in general. As members of my committee, they have all profoundly shaped the ideas in this thesis as well as helped me formalize them into this document.

I would like to thank the Pentlandians group who has been a great team to work with: Karen Navarro, Elizabeth Farley, Tanzeem Choudhury, Brian Clarkson, Sumit Basu, Yuri Ivanov, Nitin Sawhney, Vikram Kumar, Ali Rahimi, Steve Schwartz, Rich DeVaul, Dave Berger, Josh Weaver and Nathan Eagle. I would also like to thank the TRG who have been a great source of readings and brainstorming: Jayney Yu, Jason Rennie, Adrian Corduneanu, Neel Master, Martin Szummer, Romer Rosales, Chen-Hsiang, Nati Srebro, and so many others. My thanks to all the other Media Lab folks who are still around like Deb Roy, Joe Paradiso, Bruce Blumberg, Roz Picard, Irene Pepperberg, Claudia Urrea, Yuan Qi, Raul Fernandez, Push Singh, Bill Butera, Mike Johnson and Bill Tomlinson for sharing bold ideas and deep thoughts. Thanks also to great Media Lab friends who have moved to other places but had a profound influence on me: Baback Moghaddam, Bernt Schiele, Nuria Oliver, Ali Azarbayejani, Thad Starner, Kris Popat, Chris Wren, Jim Davis, Tom Minka, Francois Berard, Andy Wilson, Nuno Vasconcelos, Janet Cahn, Lee Campbell, Marina Bers, and my UROPs Martin Wagner, Cyrus Eyster and Ken Russell. Thanks to so many folks outside the lab like Sayan Mukherjee, Marina Meila, Yann LeCun, Michael Jordan, Andrew McCallum, Andrew Ng, Thomas Hoffman, John Weng, and many others for great conversations and valuable insight.

And thanks to my family, my father, my mother and my sister, Carine. They made every possible sacrifice and effort so that I could do this PhD and supported me cheerfully throughout the whole endeavor.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

It is not knowledge, but the act of learning,... which grants the greatest enjoyment.

Karl Friedrich Gauss, 1808.

The objective of this thesis is to propose a common framework that combines three different paradigms in machine learning: generative, discriminative and imitative learning. The resulting mathematically principled framework [1] suggests that combined or hybrid approaches provide superior performance and flexibility over the individual learning schemes in isolation. Generative learning is at the heart of many approaches to pattern recognition, artificial intelligence, and perception and provides a rich framework for imposing structure and prior knowledge on a given problem. Yet recent progress in discriminative learning has demonstrated that superior performance can be obtained by avoiding generative modeling and focusing on the given task. A powerful connection will be proposed between generative and discriminative learning to combine the complementary strengths of the two schools of thought. Subsequently, we propose a connection between discriminative learning and imitative learning. Imitative learning is an automatic method for learning behavior in an interactive autonomous agent. This process will be cast and augmented with a discriminative learning formalism.

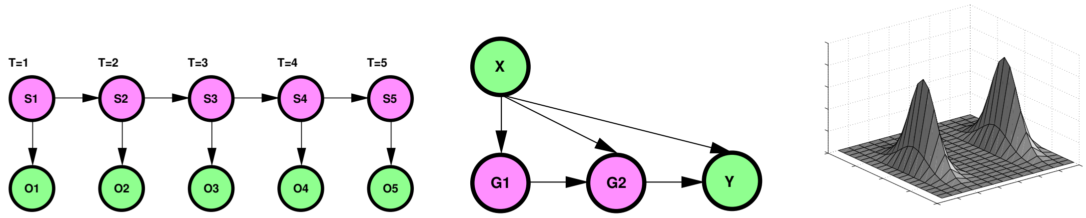
In this chapter, we begin by discussing motivation for machine learning in general. There we draw upon applied examples from pattern recognition, various AI domains and machine perception. These communities have identified various generative models specifically designed and honed to reflect the prior knowledge in their respective domains. Yet these generative models must often be discarded when one considers a discriminative approach which ironically provides superior performance despite its naive models. This motivates the need to find common formalisms that synergistically combine the different schools of thought in the community. We then describe an ambitious instance of machine learning, namely imitative learning which attempts to learn autonomous interactive agents directly from data. This form of agent learning can also benefit from being cast into a discriminative/generative paradigm. We end the chapter with a summary of the objectives and scope of this work and provide a brief overview of the rest of the chapters in this document.

[1] In the process of constructing such a framework we will expose important mathematical tools which are valuable in their own right. These include a powerful reversal of the celebrated Jensen inequality, projection approaches to learning and more.

1.1 Learning and Generative Modeling

While science can sometimes provide exact deterministic models of phenomena (i.e. in domains such as Newtonian physics), the mathematical relationships governing more complex systems are often only (if at all) partially specifiable. Furthermore, aspects of the model may have uncertainty and incomplete information. Machine learning and statistics provide a formal approach for manipulating nondeterministic models by describing or estimating a probability density over the variables in question. Within this generative density, one can specify a priori partial knowledge and refine the partially specified model using empirical observations and data. Thus, given a system with variables $x_1, \ldots, x_T$, a system can be specified through a joint probability distribution over all the significant variables within it $p(x_1, \ldots, x_T)$. This is known as a generative model since given this probability distribution, we can generate samples of various configurations of the system. Furthermore, given a full generative model, it is straightforward to condition and marginalize over the joint density to make inferences and predictions.

In many domains, greater sophistication and more ambitious tasks have made problems so intricate that complete models, theories and quantitative approaches are difficult to construct manually. This, combined with the greater availability of data and computational power have encourage many of these domains to migrate away from rule-based and manually specified models to probabilistic data-driven models. However, whatever partial amounts of domain knowledge are available are still used to seed a generative model. Developments in machine learning and Bayesian statistics have provided a more rigorous formalism for representing prior knowledge and combining it with observed data. Recent progress in graphical generative models or Bayesian networks [149] [118] [98] [104] has permitted prior knowledge to be specified structurally by identifying conditional independencies between the variables as well as parametrically by providing prior distributions over them. This partial domain knowledge [2] is then combined with observed data resulting in a more precise posterior generative model.



(a) (b) (c)

Figure 1.1: Examples of Generative Models.

In Figure 1.1, we can see two different examples of generative models. A hidden Markov model [104] [156][13] is depicted in Figure 1.1(a) as a directed graph which identifies high level conditional independence properties. These specify a Markov structure where states only depend on their predecessors and outputs only depend on the current state. A hierarchical mixture of experts [105] is portrayed in Figure 1.1(b). Similarly, a mixture model [20] is shown in Figure 1.1(c) which can also be seen as a graphical model where are parent node selects between two possible Gaussian emission distributions. The details and formalism underlying generative models will be presented in the next chapter. For now, we provide background motivation through examples from multiple

[2] A further caveat will be addressed in the following sections which warns that even the partially specified aspects of a model will often be inaccurate and suspect.

applied fields where these generative models have become increasingly popular. [3].

In the area of natural language processing, for instance, traditional rule-based or boolean logic systems (such as Dialog and Lexis-Nexis) are giving way to statistical approaches [37] [33] [122] such as Markov models which establish independencies in a chain of successive events. In medical diagnostics, the Quick Medical Reference knowledge base, initially a heuristic expert system for reasoning about diseases and symptoms has been augmented with a statistical, decision-theoretic formulation [175] [80]. This new formulation structures a diagnostics problem with a two layer bipartite graph where diseases are parents of symptoms. Another recent success of generative probabilistic models lies in genomics and bioinformatics. Once again, traditional approaches for modeling genetic regulatory networks used boolean approaches or differential equation-based dynamic models which are now being challenged by statistical graphical models [67]. Here, a model selection criterion identifies the best graph structure that matches training data. In addition, for visualization of interdependencies between variables graphical models have given a principled formalism which has proven superior to their heuristic counterparts [70] [69].

1.1.1 Learning and Generative Models in AI

In the artificial intelligence (AI) area in general, we see a similar migration from rule-based expert systems to probabilistic generative models. For example, in robotics, traditional dynamics and control systems, path planning, potential functions and navigation models are now complemented with probabilistic models for localization, mapping and control [106] [186]. Multi-robot control has also been demonstrated using a probabilistic reinforcement learning approach [193]. Autonomous agents or virtual interactive characters are another example of AI systems. From the early days of interaction and gaming, simple rule-based schemes were used, such as in in Weizenbaum's Eliza [200] program, where natural language rules were used to emulate a therapy session. Similarly, graphical virtual worlds and characters have been generated by rules, cognitive models, physical simulation, kinematics and dynamics [205] [176] [60] [11] [7] [184] [51] [36]. These traditional approaches are currently being combined with statistical machine learning techniques [23] [210] [27].

1.1.2 Learning and Generative Models in Perception

In machine perception, generative models and machine learning have become prominent tools in particular because of the complexity of the domain and sensors. In speech recognition, hidden Markov models are [156] are the method of choice due to their probabilistic treatment of acoustic coefficients and the Markov assumptions necessary for time varying signals. Even auditory scene analysis and sound texture modeling has been cast into a probabilistic learning framework with independent component analysis [14] [15]. Word distributions have also been modeled using bigrams, trigrams or Markov models and topic modeling often uses multinomial distributions. A topic spotting system is shown in Figure 1.2(c) which tracks the conversation of multiple speakers and displays related material for the users to read [89].

A similar emergence of generative models can also be found in the computer vision domain. Techniques such as physics based modeling [42], structure from motion and epipolar geometry [54] approaches have been complemented with probabilistic models such as Kalman filters [4] [87] to prevent instability and provide robustness to sensor noise. Figure 1.2(a) depicts a system that probabilistically fuses 2D motion estimates into an extended Kalman filter to obtain a rigid 3D face structure

[3] This is just a small collection examples of generative models in the various fields and is by no means a complete survey.

and pose estimate [91]. Multiple hypothesis filtering and tracking in vision have also used a generative model and Markov chain Monte Carlo via the Condensation algorithm [79]. More sophisticated probabilistic formulations in computer vision include the use of Markov random fields and loopy belief networks to perform super-resolution [56]. Generative models and structured latent mixtures are used to compute transformations and invariants in a face tracking applications [59]. Other distributions, such as eigenspaces [137] and mixture models have also been used for skin color modeling [172] and image manifold modeling [30]. For instance, in Figure 1.2(b) an eigenspace over 2D photos and 3D face scans is formed and then a generative model between the two spaces can regress 3D face shapes from 2D images in real-time [95]. Color modeling can be done with a mixture of Gaussian models to permit billiards tracking for augmented reality applications as in Figure 1.2(c) [88]. Object recognition and feature extraction has also benefited greatly from a probabilistic interpretation [171]. For example, in Figure 1.2(e), histograms of convolution operations on images provide reliable recognition for, eg. real-time augmented reality [96].

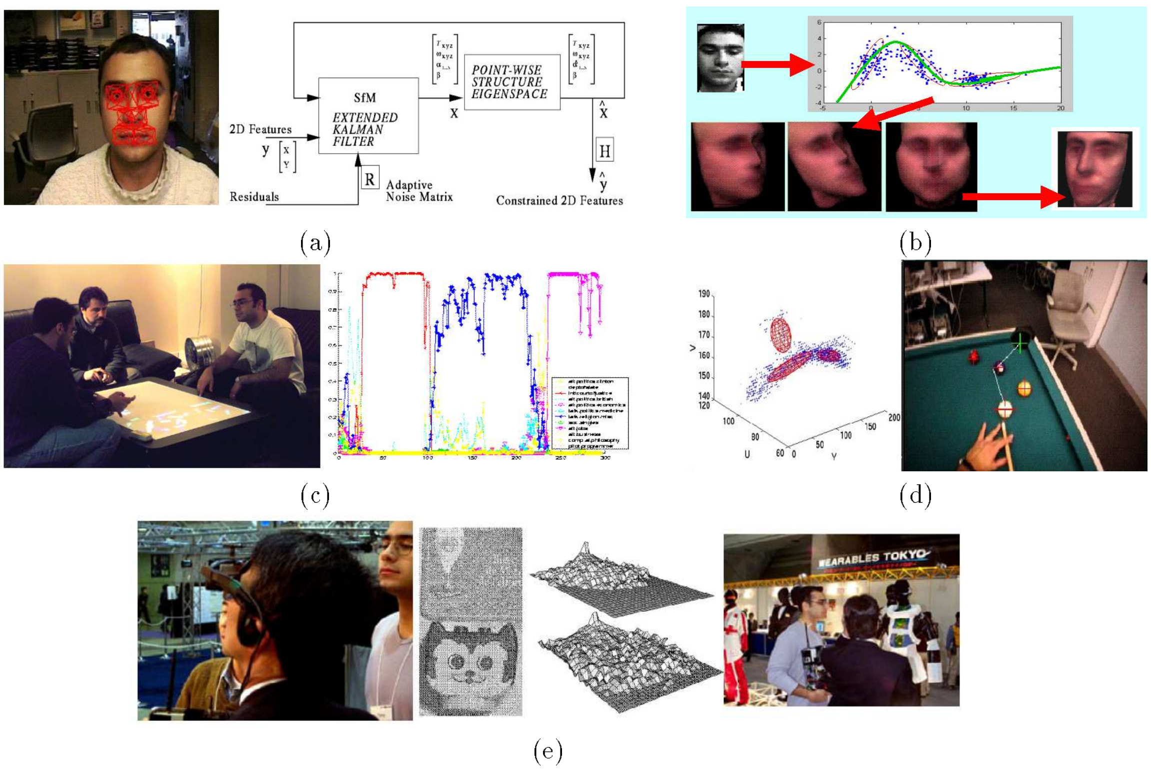


Figure 1.2: Probabilistic Perception Systems.

1.1.3 Learning and Generative Models in Temporal Behavior

Simultaneously, an evolution has been proceeding in the field as vision techniques transition from low level static image analysis to dynamic and high level video interpretation. These temporal aspects of vision (and other domains) have relied extensively on generative models, and dynamic Bayesian networks in particular. Temporal tracking has also benefited from generative models such as extended Kalman filters [5]. In tracking applications, hidden Markov models are frequently used to recognize gesture [204] [179] as well as spatiotemporal activity [61]. The richness of graphical models permit straightforward combinations of hidden Markov models with Kalman filters for switching between linear dynamical systems in modeling gaits [29] or driving maneuvers [151]. Further variations in the graphical models include coupled hidden Markov models which are appropriate for modeling interacting processes such as vehicle or pedestrian in traffic [145] [139] [141]. Bayesian networks have also been used in multi-person interaction modeling, eg. in classifying football plays [78]. Forecasting temporal activity has also been reviewed in the Santa Fe competition [62] where various approaches including hidden Markov models are compared.

1.2 Why a Probability of Everything?

Is it efficient to create a probability distribution over all variables in this 'generative' way? The previous systems make no distinction between the roles of different variables and are merely trying to model the whole phenomenon. This can be inefficient if we are simply trying to learn one (or a few) particular tasks that need to be solved and are not interested in characterizing the behavior of the complete system.

An additional caveat is the generative density estimation is formally an ill-posed problem. Density estimation, under many circumstances, can be a cumbersome intermediate step to forming a mapping between variables (i.e. from input to output). This dilemma will be further explicated in Chapter 2. Moreover, another issue is the difficulty of density estimation in terms of sample complexity and that a large amount of data may be necessary to obtain a good generative model of a system as a whole but we may only need a small sample to learn the required input-output sub-task discriminatively.

Furthermore, all the above AI and perceptual systems *work well* because of the structures, priors, representations, invariants and background knowledge designed into the system by a domain expert. This is not just due to the learning power of the estimation algorithms but that they are also *seeded* with the right framework and a priori structures for learning. Can we alleviate the amount of manual effort in this process and take some of the human knowledge engineering out of the loop? One way is to not require a very accurate generative model to be designed and not require as much domain expertise up-front. If we had discriminative learning algorithms that were more powerful, the learning would be robust to more errors in the design process and remain effective despite incorrect modeling assumptions.

1.3 Generative versus Discriminative Learning

The previous applications we described present compelling evidence and strong arguments for using generative models where a joint distribution is estimated over all variables. Ironically, though, these flexible models have been recently outperformed in many cases by relatively simpler models estimated with *discriminative* algorithms.

Unlike a generative modeling approach where modeling tools are available for combining structure, priors, invariants, latent variables and data to form a good joint density tailored to the domain at hand, discriminative algorithms directly optimize a relatively less domain-specific model for the classification or regression task at hand. For example, support vector machines [196] [35] directly maximize the margin of a linear separator between two sets of points in a Euclidean space. While the model is simple (linear), the maximum margin criterion is more appropriate than maximum likelihood or other generative model criteria.

In the domain of image-based digit recognition, support vector machines (SVMs) have produced state of the art classification performance [196] [197]. In regression [178] and time series prediction [140], SVMs improved upon generative approaches, maximum likelihood and logistic regression. In text classification and information retrieval support vector machines [48] [161] and transductive support vector machines [100] surpassed the popular naive Bayes and generative text models. In computer vision, person detection/recognition [148] [52] [142] [71] and gender classification have been dominated by SVM frameworks which surpass maximum likelihood generative models and approach human performance [138]. In genomics and bioinformatics, discriminative systems play a crucial role [202] [208] [81]. Furthermore, in speech recognition, discriminative variants of hidden Markov models have recently demonstrated superior large corpus classification performance [158] [159] [207]. Despite the more ambiguous models used in these systems, the discriminative estimation process yields improvements over the sophisticated[4] models that have been tailored for the domain in generative frameworks.

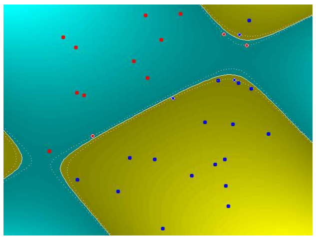


Figure 1.3: Examples of a Discriminative Classifier.

There are deeply complementary advantages in both the generative and discriminative approaches yet, algorithmically, they are not directly compatible. Within the community, one could go so far as to say that there exist two somewhat disconnected camps: the 'generative modelers' and the 'discriminative estimators' [167] [83]. We now quickly discuss the general aspects of these two schools of thought (Chapter 2 further details the two approaches and previous efforts to bridge them in the machine learning literature). Generative models provide the user with the ability to seed the learning algorithm with knowledge about the problem at hand. This is given in terms of structured models, independence graphs, Markov assumptions, prior distributions, latent variables, and probabilistic reasoning [25] [149]. The focus of generative models is to describe a phenomenon and to try to resynthesize or generate configurations from it. In the context of building classifiers, predictors, regressors and other task-driven systems, density estimation over all variables or a full generative description of the system can often be an inefficient intermediate goal. Clearly, therefore, the estimation frameworks in probabilistic generative models do not optimize parameters for a given specific task. These models are marred by generic optimization criteria such as *maximum likelihood* which are oblivious to the particular classification, prediction, or regression task at hand. Meanwhile, discriminative techniques such as support vector machines have little to offer in terms of structure and modeling power yet achieve superb performance on many test cases. This is due to their inherent and direct optimization of a task-related criterion. For example, Figure 1.3 shows an appropriate criterion for binary classification: the largest margin separation boundary (for a 3rd order polynomial model). The focus here is on *classification* as opposed to *generation* thus properly allocating computational resources directly for the task required.

Nevertheless, as previously mentioned, there are some fundamental differences in the two approaches

[4]Here, we are using the term 'sophisticated' to refer to the extra tailoring that the generative model traditionally obtains from the user to incorporate domain-specific knowledge about the problem at hand (in terms of priors, structures, etc.). Therefore, this is not a claim about the relative mathematical sophistication between generative and discriminative models.

making it awkward to combine their strengths in a principled way. It would be of considerable value to propose an elegant framework which would subsume and unite both schools of thought and thus will be one challenge undertaken in this thesis.

1.4 Imitative Learning

So far, we have provided motivation using multiple instantiations of machine learning and generative models in the applied domains of perception, temporal modeling and autonomous agents. Due to the common probabilistic platform threading across these domains, it is natural to consider combinations and couplings between these systems. One of the platforms we will investigate for combining these synergistic approaches to learning is imitative learning. The imitative learning will be used to learn an autonomous agent which exhibits interactive behavior. Imitative learning provides an easy approach [5] for learning agent behavior by providing real examples of agents interacting in a world that can be learned from and generalized. The two components of this process, passively perceiving real world behavior and learning from it are portrayed in Figure 1.4. The basic notion is to have a generative model at the perceptual level to be able to regenerate or resynthesize virtual characters while keeping a discriminative model on the temporal learning to focus resources on the prediction task necessary for action selection. This conceptual loop and its implementation will be elaborated in Chapter 8. For now, we will briefly situate imitative learning in the context of other agent learning approaches and motivate it with background and related work.

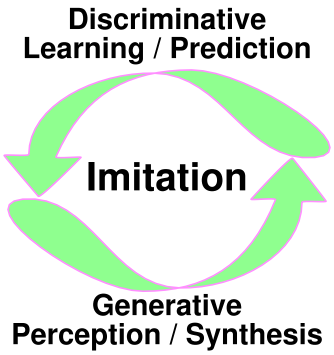


Figure 1.4: Imitative Learning through Discrimination and Probabilistic Perception.

Various approaches have been proposed for learning autonomous agents in domains such as robotics and interactive graphics. While some utilize rule-based, discriminative, or generative models, we can also distinguish among them in the manner in which the learning process is cast within the overall agent behavior model. For example, how will data be acquired, how will data be organized, how will it be labeled, what will be the task(s) of the agent, how will it generalize, and so forth. Traditional, rule-based systems for agent modeling require a cumbersome enumeration of decisions [36]. A simpler alternative is supervised learning where a teacher provides a few exemplars of the optimal behavior for a given context and a learning algorithm generalizes from the samples [160] [181] [20] [187]. Even less supervision is required in reinforcement learning [107]. This remains one of the most popular approaches to agent learning in part due to its strong ethological and cognitive science roots. Therein, an agent explores its action space and is rewarded for performing

[5] As Confucius says, there are 3 types of learning, "by reflection, which is noblest; Second, by imitation, which is easiest; and third by experience, which is the bitterest".

appropriate actions in the appropriate context. Thus, supervision is minimized due to the simplicity in providing only a reward signal and the supervision process can be done in an active online setting. Imitative learning [169] is in a sense a combination of supervised and unsupervised (i.e. no teacher) learning. While we collect data of real people interacting with the real world, we are shown many exemplars of appropriate reactionary behavior in response to the current context. Thus, the data is already labeled and needs no teaching effort for the supervision. This, of course, assumes that perceptual techniques can record and represent natural real-world activity automatically. In such an incarnation, imitative learning only involves data collection and avoids manual supervision. It is for this reason that we will investigate it further and implement it with the discriminative and generative models we will propose. While ideally, an agent would utilize a mixture of all learning methodologies and use multiple learning scenarios to introspect and bootstrap each other in a meta-learning approach [187], such an undertaking would be too ambitious. We leave the learning to learn aspect of agent behavior as an interesting topic for future research. At this point, we quickly motivate some background in imitation learning which spans multiple fields (cognitive sciences, philosophy, psychology, neuroscience, robotics, etc.) and a full survey of which would be beyond the scope of this thesis.

Early research in behavior and cognitive sciences exhibited strong interest in the role of imitative learning. However, ground breaking works of Thorndike [185] and Piaget [153] were followed by a lull in the area of movement imitation. This was in part due to the presumption that imitation or mimicry in an entity was not necessarily the sign of higher intelligence and therefore not critical to development. This prejudice slowly faded with the arrival of several studies by Meltzoff and Moore that indicated infants' ability to perform facial/manual gesture imitation from ages 12-21 days old and in some cases at an hour old [130] [131] [132] [133]. Imitative learning began to be seen as an almost innate mechanism to help the development of humans and certain species [192]. Furthermore, it was demonstrated to be absent in other, lower-order animals [191], or very limited in others [190]. In addition, through recent discoveries of mirror neurons, action-perception pathways and functional magnetic resonance imaging results [2] [157] [77] [165], a neural basis for imitative learning has been recently hypothesized [6]. Certain experiments indicated consistent firings in a mirror neuron either when an action was performed by a subject or when another individual was perceived performing the same action. In addition, imitation has also been suggested as a possible basis for language learning [164]. These results have spurred applied efforts in imitative approaches to robotics by Mataric [123], Brooks [31], etc. where imitation has gained visibility and complemented reinforcement learning [107]. Further arguments for imitation based learning include improved acquisition of complex visual gestures in human subjects [39].

However, these domains have predominantly focused on uncovering direct mappings between action and perception [169] [123]. It is through such a mapping that the imitation learning problem can be translated into a direct supervised learning one. This complex mapping is to a certain extent the Achilles' heel of imitation learning. Much of the effort of humanoid robot imitation rests in resolving Meltzoff and Moore's 'Active Intermodal Mapping' (AIM) problem. That is, the creation of a mapping of the visual perception of a teacher's movement to high-level representations that can then be matched to other high-level representations of the learner's action space and proprioceptive senses (see Figure 1.5). Effectively, the AIM problem is a change-of-coordinates task where intermediate representations allow a mapping of the learner's action space to various perceived situations and various teachers. This key challenge has driven a substantial effort in the area of humanoid robotics. Various simplifications to the AIM problem can be made to permit implementation of faster learning of robot behavior, however, many of these result in over-simplification of action and perception spaces

[6]The discovery of imitation neurons has recently generated extreme enthusiasm in psychology and has been predicted to provide that field with a leap forward equivalent to the progress in biology obtained from work in DNA [157]. It is also suggested that dysfunctional mirror neurons may explain certain phenomena such as autism.

and consequently generate uninteresting robot behavior. For example, Billard [19] over-simplifies action spaces and perception spaces into a low-dimensional discrete representation and then has a simple learning mechanism (not much more than a rote learner) to resolve the one-to-one mapping.

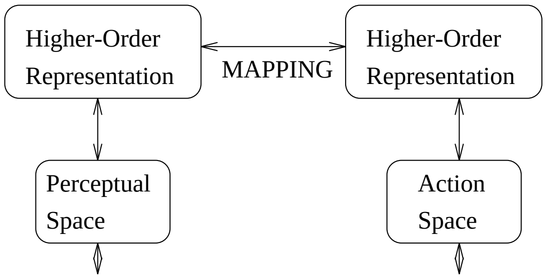


Figure 1.5: The AIM mapping from action to perception.

An alternative approach is to do away with the AIM problem altogether by either providing the teacher's perceptual data in terms of the action-space of the learner [201] [124] or by only considering virtual characters [74] [61] [101] [93] whose action space is in the perceptual space. For example, Weng [201] describes a human pushing a robot down a hallway while the robot collects images of its context. The actuators in the robot (not its cameras) measure the human's displacement and therefore to imitate the human, the displacement values need only be regurgitated (under the appropriate visual context). Hogg [101] alternatively describes a vision system which obtains perceptual measurements and needs only resynthesize behavior in the visual space to generate an action virtually. Both methods cleverly avoid a direct mapping of the perception of a teacher's activity into the learner's action-space.

Clearly, the lack of a higher-order representation of the perception and action requires some extra care. Changes in the coordinate system are not abstracted away and must be dealt with up front. Thus, the actuators in Weng's robot can not be replaced by a totally different motor system and the vision system in Hogg's virtual characters can not be pointed to a radically new scene. One way to side-step the lack of an abstraction layer is to lock the perceptual (or action) coordinate system. This may seem difficult if the learner has to acquire lessons from multiple teachers in multiple contexts and therefore prevents many types of long-term training (or developmental [126]) data scenarios. Furthermore, in a long-term training scenario, it is important to weed out irrelevant data and outliers in the behavioral data and to focus resources discriminatively on the defining exemplars in the training data.

Thus, the second challenge of this thesis is to implement imitative learning and show that it can be cast into the discriminative and generative formalisms described previously. This extends the theoretical combination of discriminative and generative learning to a practical applied task of imitation-based learning.

1.5 Objective

Therefore, we pose the following two main challenges. We will seek a combined discriminative and generative framework which extends the powerful generative models that are popular in the machine learning community into discriminative frameworks such as those present in support vector machines. We will also seek an imitative learning approach that casts interactive behavior acquisition into the discriminative and generative framework we propose. There are many features and subgoals within these two large contributions which we will also strive for. The following list enumerates some of these in further detail.

- Combined Generative and Discriminative Learning

  Ideally, the combination of generative and discriminative learning should be done at a formal level so that it is easily extensible and can be related to other domains and approaches. We will provide a discriminative classification framework that retains much of the formalism of Bayesian generative modeling via an appeal to maximum entropy which already has many connections to Bayesian approaches. The formalism also has powerful connections to regularization theory and support vector machines, two important and principled approaches in the discriminative school of thought.

- Formal Generalization Guarantees

  While empirical validation can support the combined generative-discriminative framework, we will also refer the reader to formal generalization guarantees from different perspectives. Various arguments from the literature such as sparsity, VC-dimension and PAC-Bayes generalization bounds will be compatible with the framework.

- Applicability to a Spectrum of Bayesian Generative Models

  To span a wide variety generative models we will focus on the exponential family which is central to much of statistics and maximum likelihood estimation. The discriminative methods will be consistently applicable to this large family of distributions.

- Ability to Handle Latent Variables

  While the strength of most generative models lies in the their ability to handle latent variables and mixture models, we will ensure that the discriminative method can also span these higher order multimodal distributions. Through novel bounds, we will extend beyond the classical Jensen inequality that permits much of generative modeling to apply to mixtures.

- Analytic Reversal of Jensen's Inequality

  We will present an analytic reversal of Jensen's inequality which is useful in statistics and provides a new mathematical tool. This reversal will permit various important manipulations in particular the use of discriminative estimation on latent generative models. The mathematical tool also permits dual sided bounds on many probabilistic quantities which otherwise only had a single bound.

- Computational Efficiency

  Throughout the development of the discriminative, generative and imitative learning procedures, we will consistently discuss issues of computational efficiency and implementation. These frameworks will be shown to be viable in large data scenarios and computationally as tractable as their traditional counterparts.

- Extensibility

  Many extensions will be demonstrated in the hybrid generative discriminative approach which will justify its usefulness. These include the ability to handle regression, multiclass classification, transduction, feature selection, structure learning, exponential family models and mixtures of the exponential family.

- Casting Imitative Learning into a Generative/Discriminative Framework

We will bring imitative learning into a generative/discriminative setting by describing generative models over the perceptual domain and over the temporal domain. These will be augmented by discriminatively learning a prediction model to synthesize interactive behavior in an autonomous agent.

- Combining Perception, Learning and Behavior Acquisition

  We will demonstrate a real-time system that performs perception, learning and acquires and synthesizes real-time behavior. This closes the loop we proposed and shows how a common discriminative and probabilistic framework can be extended throughout the multiple facets of a large system.

1.6 Scope

This thesis focuses on computational and statistical aspects of machine learning and machine perception. Therein, discussions of different types of learning (discriminative, generative, conditional, imitative, reinforcement, supervised, unsupervised, etc.) refer primarily to the computational and mathematical aspects of these terms. Connections to the large bodies of work in the cognitive sciences, psychology, neuroscience, philosophy, ethology, and so forth will invariably arise however these are brought in for motivation and implementation purposes and should not necessarily be taken as a direct challenge to the conventional wisdoms in those fields.

1.7 Organization

The thesis is organized as follows:

- Chapter 2

  The complementary advantages of discriminative and generative learning are discussed. We formalize the many models and methods of inference in generative, conditional and discriminative learning. The various advantages and disadvantages of each are enumerated and motivation for methods for fusing them is given.

- Chapter 3

  The Maximum Entropy Discrimination formalism is introduced as the method of choice for combining generative models in a discriminative estimation setting. The formalism is presented as an extension to regularization theory and shown to subsume support vector machines. A discussion of margins, bias and model priors is presented. The MED framework is then extended to handle generative models in the exponential family. Comparisons are made with sate of the art support vector machines and other learning algorithms. Generalization guarantees on MED are then provided by appealing to recent results in the literature.

- Chapter 4

  Various extensions to the Maximum Entropy Discrimination formalism are proposed and elaborated. These include multiclass classification, regression and feature selection. Furthermore, transduction is discussed as well as optimization issues. The chapter then motivates the need for latent models in MED and for mixtures of the exponential family. Comparisons are made with sate of the art support vector machines and other learning algorithms.

- Chapter 5

  Latent learning is motivated in a discriminative setting via reverse-Jensen bounds. The so-called Conditional Expectation Maximization framework is then proposed for latent conditional and discriminative (MED) problems. Bounds are given for exponential family mixture models and mixing coefficients. Comparisons are made with the state of the art Expectation-Maximization approaches.

- Chapter 6

  We consider the case of discriminative learning of structured mixture models where the mixture is not flat but has some additional complications that generate an intractable number of latent configurations. This is the case for many Bayesian networks and generally prevents a tractable computation of the reverse-Jensen bounds. We show how the reverse-Jensen bounds can be computed efficiently in some of these circumstances and therefore extend the applicability of latent discrimination and CEM to structured mixture models such as hidden Markov models and mixture models over an aggregated data set.

- Chapter 7

  This chapter begins with a brief discussion of work in the mathematical inequalities community including prior reversals and converses of Jensen's inequality. Then, a derivation of the reverse-Jensen inequality for use in discriminative learning is performed to justify its form and give global guarantees on the bounds.

- Chapter 8

  Imitative learning is cast as a discriminative temporal prediction problem where an agent's next action is predicted from his previous state and the world state. The implementation details for the hardware and the perceptual systems are discussed. A generative model for the audio, images and temporal structure is then presented and provides the representations for the discriminative prediction problem. Synthesized behavior is then demonstrated as well as some quantitative measurement of performance.

- Chapter 9

  The advantage of a joint framework for generative, discriminative and imitative learning is reiterated. The various contributions of the thesis are summarized. Future extensions and elaborations are proposed.

- Chapter 10

  This appendix gives a few standard derivations that are called upon in the main body of the thesis. This includes a brief discussion of convex duality as well as details of numerical procedures mentioned in Chapter 7.

Chapter 2

Generative vs. Discriminative Learning

All models are wrong, but some are useful [1].

George Box, 1979

In this chapter, we will situate discriminative and generative learning more formally in the context of their estimation algorithms and the criteria they optimize. A natural intermediate between the two is conditional learning which helps to visualize a coarse continuum between these extremes. Figure 2.1 describes a panorama of approaches as we go horizontally from the extreme of generative criteria to discriminative criteria. Similarly, another scale of variation (vertical) can be seen in the estimation procedures as we go from direct optimizations of the criteria on training data to regularized ones and finally to fully averaged ones which attempt to better approximate the behavior of the criteria on future data without overfitting.

In this chapter we begin with a sample of generative and discriminative techniques and then explore the entries in Figure 2.1 in more detail. The generative models at one extreme attempt to estimate a distribution over all variables (inputs and outputs) in a system. This is inefficient since we only need conditional distributions of output given input to perform classification or prediction and motivates a more minimalist approach: conditional modeling. However, in many practical systems, we are even more minimalist than that since we only need a single estimate from a conditional distribution. So, even conditional modeling may be inefficient which motivates discriminative learning since it only considers the input-output mapping. We conclude with some hybrid frameworks for combining generative and discriminative models and point out their limitations.

[1] At the risk of misquoting what Box truly intended to say about robust statistics, we shall use this quote to motivate combining the usefulness of generative models with the robustness, practicality and performance of discriminative estimation.

	GENERATIVE	CONDITIONAL	DISCRIMINATIVE
LOCAL	Maximum Likelihood	Maximum Conditional Likelihood Maximum Mutual Information	Empirical Risk Minimization
LOCAL + PRIOR	Maximum A Posteriori	Maximum Conditional A Posteriori	Support Vector Machines Regularization Theory
MODEL AVERAGING	Bayesian Inference	Conditional Bayesian Inference	Maximum Entropy Discrimination Bayes Point Machines

Figure 2.1: Scales of Discrimination and Integration in Learning.

2.1 Two Schools of Thought

We next present what could be called two schools of thought: discriminative and generative approaches. Alternative descriptions of the two formalisms include "discriminative versus informative" approaches [167]. Generative approaches produce a probability density model over all variables in a system and manipulate it to compute classification and regression functions. Discriminative approaches provide a direct attempt to compute the input to output mappings for classification and regression and eschew the modeling of the underlying distributions. While the holistic picture of generative models is appealing for its completeness, it can be wasteful and non-robust. Furthermore, as Box says, all models are wrong (but some are useful). Therefore, the graphical models and the prior structures that we will enforce on our generative models may have some useful elements yet should always be treated cautiously since in real-world problems, the true distribution almost never coincide with the one we have constructed. In fact, Bayesian inference does not guarantee that we will obtain the correct posterior estimate if the class of distributions we consider do not contain the true generator of the data we are observing. Here we show examples of the two schools of thought and then elaborate on the learning criteria they use in the next section.

2.1.1 Generative Probabilistic Models

In generative or Bayesian probabilistic models, a system's input (covariate) features and output (response) variables (as well as unobserved variables) are represented homogeneously by a joint probability distribution. These variables can be discrete or continuous and may also be multidimensional. Since generative models define a distribution over all variables, they can also be used for classification and regression [167] by standard marginalization and conditioning operations. Generative models or probability densities in current use typically span the class of exponential family distributions and mixtures of the exponential family. More specifically, popular models in various domains include Gaussians, naive Bayes, mixtures of multinomials, mixtures of Gaussians [20], mixtures of experts [105], hidden Markov models [156], sigmoidal belief networks, Bayesian networks [98] [118] [149], Markov random fields [209], and so forth.

For N variables of the form $(x_1, \ldots, x_n)$, we therefore have a full joint distribution of the form: $p(x_1, \ldots, x_n)$. Given a good joint distribution that accurately captures the (possibly nondeterministic) relationships between the variables, it is straightforward to use it for inference and to answer queries. This is done by straightforward manipulations of the basic axioms of probability theory such as marginalizing, conditioning and using Bayes' rule:

$$\begin{aligned} p(x_j) &= \sum_{\forall x_i, i \neq j} p(x_1, \ldots, x_n) \\ p(x_j|x_k) &= \frac{p(x_j, x_k)}{p(x_k)} \\ p(x_j|x_k) &= \frac{p(x_k|x_j)p(x_j)}{p(x_k)} \end{aligned}$$

Thus, through conditioning a joint distribution, we can easily form classifiers, regressors and predictors in a straightforward manner which map input variables to output variables. For instance, we may want to obtain an estimate of the output $\hat{x}_j$ (which may be a discrete class label or a continuous regression output) from the input $\hat{x}_k$ using the conditional distribution $p(x_j|x_k)$. While a purist Bayesian would argue that the only appropriate answer is the conditional distribution itself $p(x_j|x_k)$, in practice we must settle for an approximation to obtain an $\hat{x}_k$. For example, we may randomly sample from $p(x_j|x_k)$, or compute the expectation of $p(x_j|x_k)$ or find the mode(s) of the distribution, i.e. $\text{argmax}_{x_j} p(x_j|x_k)$.

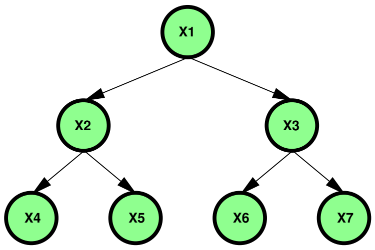


Figure 2.2: Directed Graphical Models.

There are many ways to constrain this joint distribution such that it has fewer degrees of freedom before we directly estimate it from data. One way is to structurally identify conditional independencies between variables. This is depicted, for example, with the directed graph (Bayesian network) in Figure 2.2. Here, the graph identifies that the joint distribution factorizes into a product of conditional distributions over the variables given their parents (here π_i are the parents of the variable x_i or node i):

$$p(x_1, \ldots, x_n) = \Pi_{i=1}^{n} p(x_i|x_{\pi_i})$$

Alternatively, we can parametrically constrain the distribution by giving prior distributions over the variables and hyper-variables that affect them. For example, we may restrict two variables (x_i, x_k) to be jointly a mixture of Gaussians with unknown means and a covariance equal to identity:

$$p(x_i, x_k) = \alpha \mathcal{N}\left(\begin{bmatrix} x_i \\ x_k \end{bmatrix}; \mu_1, I\right) + (1-\alpha)\mathcal{N}\left(\begin{bmatrix} x_i \\ x_k \end{bmatrix}; \mu_2, I\right)$$

Other types of restrictions exist, for example those related to sufficiency and separability [152] where a conditional distribution might simplify according to a mixture of simpler conditionals as in:

$$p(x_i|x_j, x_k) = \alpha p(x_i|x_j) + (1-\alpha)p(x_i|x_k)$$

Thus, a versatility is inherent in working in the joint distribution space since we can insert knowledge about the relationships between variables, invariants, independencies, prior distributions and so forth. This includes all variables in the system, unobserved, observed, input or output variables. This makes generative probability distributions a very flexible modeling tool.

Unfortunately, the learning algorithms used to combine such models with the observed data and produce a final posterior distribution can sometimes be inefficient. Finding the ideal generator of the data (combined with the prior knowledge) is only an intermediate goal in many settings. In practical applications, we wish to use these generators for the ultimate tasks of classification, prediction and regression. Thus, in optimizing for an intermediate generative goal, we sacrifice resources and performance on these final discriminative tasks. Section 2.2 we discuss techniques for learning from data in generative approaches.

2.1.2 Discriminative Classifiers and Regressors

Discriminative approaches make no explicit attempt to model the underlying distributions of the variables and features in a system and are only interested in optimizing a mapping from the inputs to the desired outputs (say a discrete class or a scalar prediction) [167]. Thus, only the resulting classification boundary (or function approximation accuracy for regression) are adjusted without the intermediate goal of forming a generator that can model the variables in the system. This focuses model and computational resources on the given task and provides better performance. Popular and successful examples include logistic regression [76] [68], Gaussian processes [65], regularization networks [66], support vector machines [196], and traditional neural networks [20].

Robust (discriminative) classification and regression methods have been successful in many areas ranging from image and document classification[100] [161] to problems in biosequence analysis[82] [202] and time series prediction[140]. Techniques such as Support vector machines[197], Gaussian process models[203], Boosting algorithms[57, 58], and more standard but related statistical methods such as logistic regression, are all robust against errors in structural assumptions. This property arises from a precise match between the training objective and the criterion by which the methods are subsequently evaluated. There is no surrogate intermediate goal to obtain a good generative model.

However, the discriminative algorithms do not extend well to classifiers and regressors arising from generative models and the resulting parameter estimation is hard [167]. The models discriminative techniques use (parametric or otherwise) often lack the elegant probabilistic concepts of priors, structure, uncertainty, and so forth that are so beneficial in generative settings. Instead, alternative notions of penalty functions, regularization, kernels and so forth are used. Furthermore, learning (not modeling) is the focus of discriminative approaches which often lack flexible modeling tools and methods for inserting prior knowledge. Thus, discriminative techniques feel like black-boxes where the relationships between variables is not as explicit or visualizable as in generative models.

Furthermore, discriminative approaches may be inefficient to train since they require simultaneous consideration of all data from all classes. Another inefficiency arises in discriminative techniques since each task a discriminative inference engine needs to solve requires a different model and a new training training session. Various methods exist to alleviate the extra work arising in discriminative learning. These include online learning which can be easily applied to, eg. boosting procedures [147] [55][58]. Moreover, it is not always necessary to construct all possible discriminative mappings in a system of variables which would require exponential number of models [64]. Frequent tasks, i.e. canonical classification and regression objectives can be targeted with a handful of discriminative models while a generative model can be kept around for handling occasional missing labels, unusual types of inference and so forth. Section 2.4 will discuss techniques for learning from data techniques in discriminative approaches.

2.2 Generative Learning

There are many variations for learning generative models from data. These many approaches, priors and model selection criteria include minimum description length [163], Bayesian information criterion, Akaike information criterion, entropic priors, and so on, and a survey is beyond the scope of this thesis. We will instead quickly discuss the popular classical approaches that include Bayesian inference, maximum a posteriori and maximum likelihood estimation. These can be seen as ranging from a scale of a fully weighted averaging over all generative model hypothesis (Bayesian inference), to more local computation with simple regularization/priors (maximum a posteriori) to the simplest maximum likelihood estimator which only considers performance on training data.

2.2.1 Bayesian Inference

In Bayesian inference [20] [120] [25] [135] [22] [167], the probability density function vector $\mathbf{z}$ is typically estimated from a training set of such vectors $\mathcal{Z}$. More generally, $\mathbf{z}$ need not be a vector but could correspond to multiple observable variables that are both continuous or discrete. We will assume they are vectors here without loss of generality. The (joint) Bayesian inference estimation process is shown in Equation 2.1.

$$p(\mathbf{z}|\mathcal{Z}) = \int p(\mathbf{z}, \Theta|\mathcal{Z})d\Theta = \int p(\mathbf{z}|\Theta, \mathcal{Z})p(\Theta|\mathcal{Z})d\Theta \tag{2.1}$$

By integrating over Θ, we are essentially integrating over all the pdf (probability density function) models possible. This involves varying the families of pdfs *and* all their parameters. However, often, this is impossible and instead a sub-family is selected and only its parameterization Θ is varied. Each Θ is a parameterization of a pdf over $\mathbf{z}$ and is weighted by its likelihood given the training set.

Having obtained a $p(\mathbf{z}|\mathcal{Z})$ or, more compactly a $p(\mathbf{z})$, we can compute the probability of any point $\mathbf{z}$ in the (continuous or discrete) probability space[2]. However, evaluating the pdf in such a manner is not necessarily the ultimate objective. Often, some components of the vector are given as input ($\mathbf{x}$) and the learning system is required the estimate the missing components as output[3] ($\mathbf{y}$). In other words, $\mathbf{z}$ can be broken up into two sub-vectors $\mathbf{x}$ and $\mathbf{y}$ and a conditional pdf is computed from the original joint pdf over the whole vector as in Equation 2.2. This conditional pdf is $p(\mathbf{y}|\mathbf{x})^j$ with the j superscript to indicate that it is obtained from the previous estimate of the joint density. When an input $\mathbf{x}'$ is specified, this conditional density becomes a density over $\mathbf{y}$, the desired output of the system. The $\mathbf{y}$ element may be a continuous vector, a discrete value or some other sample from the probability space $p(\mathbf{y})$. If this density is the required function of the learning system and if a final output estimate $\hat{\mathbf{y}}$ is need, the expectation or arg max of $p(\mathbf{y}|\mathbf{x}')$ is used.

$$p(\mathbf{y}|\mathbf{x})^j = \frac{p(\mathbf{z})}{\int p(\mathbf{z})d\mathbf{y}} = \frac{p(\mathbf{x},\mathbf{y})}{\int p(\mathbf{x},\mathbf{y})d\mathbf{y}} = \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{x})} = \frac{\int p(\mathbf{x},\mathbf{y}|\Theta)p(\Theta|\mathcal{X},\mathcal{Y})d\Theta}{\int p(\mathbf{x}|\Theta)p(\Theta|\mathcal{X},\mathcal{Y})d\Theta} \tag{2.2}$$

In the above derivation, we have deliberately expanded the Bayesian integral to emphasize the .

[2] This is the typical task of unsupervised learning.
[3] This is the typical task of supervised learning.

This is to permit us to differentiate the above joint Bayesian inference technique from its conditional counterpart, conditional Bayesian inference.

2.2.2 Maximum Likelihood

Traditionally, computing the integral in Equation 2.1 is not always straightforward and Bayesian inference is often approximated via maximum a posteriori (MAP) or maximum likelihood (ML) estimation as in Equation 2.3. [47] [129].

$$p(\mathbf{z}|\mathcal{Z}) \approx p(\mathbf{z}|\Theta^*, \mathcal{Z}) \text{ where } \Theta^* = \begin{cases} \arg\max p(\Theta|\mathcal{Z}) = \arg\max p(\mathcal{Z}|\Theta)p(\Theta) & \text{MAP} \\ \arg\max p(\mathcal{Z}|\Theta) & \text{ML} \end{cases} \tag{2.3}$$

Under iid (independent identically distributed data) conditions, it is easier to instead compute maximum of the logarithm of the above quantities (which results in the same arg max). Thus we expand the above for the maximum likelihood case as follows:

$$\log p(\mathcal{Z}|\Theta) = \sum_t \log p(Z_t|\Theta)$$

The above optimization of joint likelihood is thus additive in the sense that each data points contributes to it in an additive way (after the logarithm) which facilitates optimization for eg. exponential family distributions.

Thus, maximum likelihood and maximum a posteriori can be seen as approximations to Bayesian inference where the integral over a distribution of models is replaced with the mode. The a posteriori solution allows the use of a prior to regularize the estimate while the maximum likelihood approach merely optimizes the model on the training data alone which may cause overfitting. Thus, MAP also permits the user to insert prior knowledge about the parameters of the model and bias it towards solutions that are more likely to generalize well. For example, one may consider priors that favor simpler models and therefore avoid overfitting [188] or entropic priors that sparsify the model [26]. Meanwhile, the maximum likelihood criterion only considers the training examples and optimizes the model specifically for them. This is guaranteed to converge to the model for the true distribution as we obtain more and more samples yet can also overfit and show poor generalization when limited samples are available. Thus, ML is a more local solution than MAP since it is tuned only to the training data while MAP is tuned to the data as well as the prior (and approximates the weighted averaging of all models in the Bayesian inference solution more closely).

Furthermore, an important duality between ML/MAP approximations and maximum entropy also exists [102]. Standard maximum entropy approaches solve for a distribution that is closest to uniform (in a Kullback-Leibler divergence sense) which also has the same moments as the empirical distribution (i.e. entropy projection with moment constraints). Moreover, maximum likelihood and MAP have extensive asymptotic convergence properties, are efficient and straightforward to compute for exponential family distributions without latent variables, i.e. the complete data case. For the case of incomplete data, mixture models and more sophisticated generative models, maximum likelihood and MAP estimates are not directly computable, however and there exist many local maxima in the objective function. In those situations, it is standard to use iterative algorithms such as Expectation Maximization (EM) or variants.

The EM algorithm is frequently utilized to perform these maximization of likelihood for mixture models due to its monotonic convergence properties and ease of implementation [13] [12] [44] [134]

[117]. The Expectation (E) step consists of computing a bound on the log likelihood using a straightforward application of Jensen's inequality [99] [150]. The Maximization (M) step is then the usual maximum likelihood step that would be used for a complete data model. Furthermore, various generalizations of EM have also been proposed [41] [1] [144] which bring deep geometric concepts and simplifying tools to the problem. For mixture models in a Bayesian inference setting, maximization is not appropriate. However, the Jensen bounds that EM uses can be used to bound the integration. This process is known as variational Bayesian inference [3] [63] [84] which provides a more precise approximation to Bayesian inference while making computation tractable for latent models.

2.3 Conditional Learning

While generative learning seeks to estimate an probability distribution over all the variables in the system, including both inputs and outputs, it is possible to be more efficient if the task we are trying to solve is made explicit. If we know precisely what conditional distributions will be used, it is more appropriate to directly optimize the conditional distributions instead of the generative model as a whole. Since we will be using our probability model almost exclusively to compute the conditional over the outputs (response) variables given the inputs (covariates), we can directly optimize parameters and fit the model to data such that this task is done optimally. This is not quite discriminative learning since we are still fitting a probability density in the output distribution and we have a generative model of the outputs given the inputs, i.e. $p(y|x)$. In a purist discriminative setting, we would only consider the final estimate $\hat{y}$ and extract that from the distribution in a winner-take-all type of scenario. Thus, we can view conditional learning as an intermediate between discriminative and generative learning. We are still optimizing a probability distribution, but only the one that we will ultimately use for classification or regression purposes. Thus, in the spirit of minimalism, we do away with the need to learn the joint generative model, say $p(x, y)$ and focus only on the conditional distribution $p(y|x)$.

2.3.1 Conditional Bayesian Inference

Obtaining a conditional density from the unconditional (i.e. joint) probability density function in Equation 2.1 and Equation 2.2 is roundabout and can be shown to be suboptimal. However, it has remained popular and is convenient partly because of the availability of powerful techniques for joint density estimation (such as EM). If we know a priori that we will need the conditional density, it is evident that it should be estimated *directly* from the training data. Direct Bayesian conditional density estimation is defined in Equation 2.4. The vector $\mathbf{x}$ (the input or *covariate*) is always given and the $\mathbf{y}$ (the output or *response*) is to be estimated. The training data is of course also explicitly split into the corresponding $\mathcal{X}$ and $\mathcal{Y}$ vector sets. Note here that the conditional density is referred to as $p(\mathbf{y}|\mathbf{x})^c$ to distinguish it from the expression in Equation 2.2.

$$\begin{aligned} p(\mathbf{y}|\mathbf{x})^c &= p(\mathbf{y}|\mathbf{x}, \mathcal{X}, \mathcal{Y}) \\ &= \int p(\mathbf{y}, \Theta^c|\mathbf{x}, \mathcal{X}, \mathcal{Y}) d\Theta^c \\ &= \int p(\mathbf{y}|\mathbf{x}, \Theta^c, \mathcal{X}, \mathcal{Y}) p(\Theta^c|\mathbf{x}, \mathcal{X}, \mathcal{Y}) d\Theta^c \\ &= \int p(\mathbf{y}|\mathbf{x}, \Theta^c) p(\Theta^c|\mathcal{X}, \mathcal{Y}) d\Theta^c \end{aligned} \tag{2.4}$$

Here, Θ^c parameterizes a conditional density $p(\mathbf{y}|\mathbf{x})$. Θ^c is exactly the parameterization of the conditional density $p(\mathbf{y}|\mathbf{x})$ that results from the joint density $p(\mathbf{x}, \mathbf{y})$ parameterized by Θ. Initially, it seems intuitive that the above expression should yield exactly the same conditional density as before. It seems natural that $p(y|x)^c$ should equal $p(y|x)^j$ since the Θ^c is just the conditioned version of Θ. In other words, if the expression in Equation 2.1 is conditioned as in Equation 2.2, then the result in Equation 2.4 should be identical. This conjecture is wrong.

Upon closer examination, we note an important difference. The Θ^c we are integrating over in Equation 2.4 is not the same Θ as in Equation 2.1. In the direct conditional density estimate (Equation 2.4), the Θ^c only parameterizes a conditional density $p(\mathbf{y}|\mathbf{x})$ and therefore provides no information about the density of $\mathbf{x}$ or $\mathcal{X}$. In fact, we can *assume* that the conditional density parameterized by Θ^c is just a function over $\mathbf{x}$ with some parameters. Therefore, we can essentially ignore any relationship it could have to some underlying joint density parameterized by Θ. Since this is only a conditional model, the term $p(\Theta^c|\mathcal{X},\mathcal{Y})$ in Equation 2.4 behaves differently than the similar term $p(\Theta|\mathcal{Z}) = p(\Theta|\mathcal{X},\mathcal{Y})$ in Equation 2.1. This is illustrated in the manipulation involving Bayes rule shown in Equation 2.5.

$$
\begin{aligned}
p(\Theta^c|\mathcal{X},\mathcal{Y}) &= \frac{p(\mathcal{Y}|\Theta^c,\mathcal{X})p(\Theta^c,\mathcal{X})}{p(\mathcal{X},\mathcal{Y})} \\
&= \frac{p(\mathcal{Y}|\Theta^c,\mathcal{X})p(\mathcal{X}|\Theta^c)p(\Theta^c)}{p(\mathcal{X},\mathcal{Y})} \\
&= \frac{p(\mathcal{Y}|\Theta^c,\mathcal{X})p(\mathcal{X})p(\Theta^c)}{p(\mathcal{X},\mathcal{Y})}
\end{aligned}
\tag{2.5}
$$

In the final line of Equation 2.5, an important manipulation is noted: $p(\mathcal{X}|\Theta^c)$ is replaced with $p(\mathcal{X})$. This implies that observing Θ^c does not affect the probability of $\mathcal{X}$. This operation is invalid in the joint density estimation case since Θ has parameters that determine a density in the $\mathcal{X}$ domain. However, in conditional density estimation, if $\mathcal{Y}$ is not also observed, Θ^c is independent from $\mathcal{X}$. It in no way constrains or provides information about the density of $\mathcal{X}$ since it is merely a conditional density over $p(\mathbf{y}|\mathbf{x})$. This independence property does not always hold however here we are strictly assuming that the parameterization Θ^c is such that there is only a conditional functional dependence between the parameters and the input variables (i.e. no marginal distribution over X should be induced from Θ^c). The graphical models in Figure 2.3 depict the difference between joint density models and conditional density models using a directed acyclic graph [118] [98]. Note that the Θ^c model and the $\mathcal{X}$ are independent if $\mathcal{Y}$ is not observed in the conditional density estimation scenario. In graphical terms, the Θ joint parameterization is a parent of the children nodes $\mathcal{X}$ and $\mathcal{Y}$. Meanwhile, the conditional parameterization Θ^c *and* the $\mathcal{X}$ data are co-parents of the child $\mathcal{Y}$ (they are marginally independent). Equation 2.6 then finally illustrates directly estimated conditional density solution $p(\mathbf{y}|\mathbf{x})^c$.

(a) Joint Density Estimation (b) Conditional Density Estimation

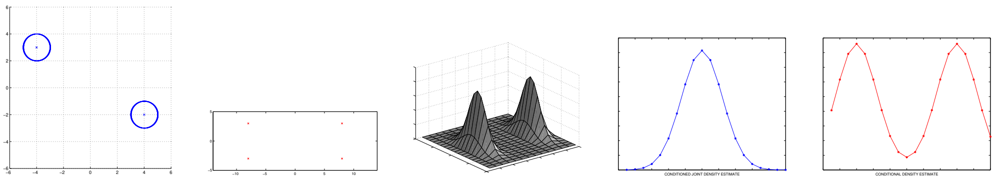
Figure 2.3: The Graphical Models

$$
\begin{aligned}
p(\mathbf{y}|\mathbf{x})^c &= \int p(\mathbf{y}|\mathbf{x},\Theta^c)p(\Theta^c|\mathcal{X},\mathcal{Y})d\Theta^c \\
&= \int p(\mathbf{y}|\mathbf{x},\Theta^c)\frac{p(\mathcal{Y}|\Theta^c,\mathcal{X})p(\mathcal{X})p(\Theta^c)}{p(\mathcal{X},\mathcal{Y})}d\Theta^c \\
&= \int p(\mathbf{y}|\mathbf{x},\Theta^c)p(\mathcal{Y}|\Theta^c,\mathcal{X})p(\Theta^c)d\Theta^c \quad / \, p(\mathcal{Y}|\mathcal{X})
\end{aligned}
\tag{2.6}
$$

If a conditional density is required, it appears superior to perform conditional Bayesian inference than to perform joint Bayesian inference and subsequently condition the answer. This is illustrated with an example below.

Example: Joint versus Conditional Bayesian Inference

In the following, we present a specific example to demonstrate this difference and to argue in favor of the conditional estimate $p(y|x)^c$ versus the conditioned joint estimate $p(y|x)^j$ (more details are in the Appendix of [97]). We demonstrate this with a simple 2-component 2D Gaussian mixture model with identity covariance and equal mixing proportions as shown in Figure 2.4(a). The likelihood for a data point $\mathbf{z} = (x, y)$ is:



(a) (b) (c) (d) (e)

Figure 2.4: Conditioned Bayesian inference vs. conditional Bayesian inference.

$p(\mathbf{z}|\Theta) = 1/2\mathcal{N}(\mathbf{z};\vec{\mu}) + 1/2\mathcal{N}(\mathbf{z};\vec{\nu})$. The prior $p(\Theta)$ over the parameters (i.e. the two means $\Theta = \{\vec{\mu}, \vec{\nu}\}$) is a wide zero-mean, spherical Gaussian distribution (with very large covariance σ^2). Thus we can infer the standard joint Bayesian distribution from a total of T training data points by Equation 2.9.

$$p(x,y) \propto \int p(x,y|\Theta)p(\mathcal{X},\mathcal{Y}|\Theta)p(\Theta)d\Theta \tag{2.7}$$

$$\propto \int p(x,y|\Theta)\Pi_{i=1}^{T} p(x_i,y_i|\Theta)p(\Theta)d\Theta \tag{2.8}$$

$$\propto \int p(x,y|\Theta)\Pi_{i=1}^{T}\left(1/2\mathcal{N}(\mathbf{z}_i;\vec{\mu}) + 1/2\mathcal{N}(\mathbf{z}_i;\vec{\nu})\right)p(\Theta)d\Theta \tag{2.9}$$

Equation 2.9 can be solved exactly if we expand the products over the two terms in the mixture model. Unfortunately, these grow exponentially fast at 2^T (from all possible assignments of the data points to the two Gaussians) but can be computed for small data sets. For the 4-point data set in Figure 2.4(b), we compute the joint Bayesian inference and plot $p(x, y)$ as shown in Figure 2.4(c). Conditioning this $p(x, y)$ on x gives us the conditional $p(y|x)^j$ (the superscript j shows that this conditional came from the joint Bayesian inference). The function $p(y|x)^j$ is plotted in Figure 2.4(d) for the value $x = -5$. We then proceed to compute $p(y|x)^c$ directly using another integration, the conditional Bayesian inference as in Equation 2.12. The resulting function $p(y|x)^c$ is *different* from $p(y|x)^j$ and is plotted in Figure 2.4(e). Note how conditional Bayesian inference captures the bimodality of the data which was lost with the regular Bayesian inference.

$$p(y|x) \propto \int p(y|x,\Theta)p(\mathcal{Y}|\Theta,\mathcal{X})p(\Theta)d\Theta \tag{2.10}$$

$$\propto \int p(y|x,\Theta)\Pi_{i=1}^{T} p(y_i|x_i,\Theta)p(\Theta)d\Theta \tag{2.11}$$

$$\propto \int \frac{p(x,y|\Theta)}{\int_y p(x,y|\Theta)dy} \Pi_{i=1}^{T} \frac{p(x_i,y_i|\Theta)}{\int_y p(x_i,y|\Theta)dy} p(\Theta) d\Theta \tag{2.12}$$

2.3.2 Maximum Conditional Likelihood

As in Bayesian inference, integration in conditional Bayesian inference (Equation 2.6) is typically intractable to evaluate in closed form. To approximate the average over many models, we will often simply pick one model at the mode of the integral. This results in the corresponding maximum conditional a posteriori (MAP^c) and maximum conditional likelihood (ML^c) solutions as shown in Equation 2.13. The a posteriori solution allows the use of a prior to regularize the estimate while the conditional likelihood approach merely optimizes the model on the training data alone which may cause overfitting.

$$p(\mathbf{y}|\mathbf{x})^c \approx p(\mathbf{y}|\mathbf{x},\Theta^*) \;\text{ where }\; \Theta^* = \begin{cases} \arg\max p(\mathcal{Y}|\Theta^c,\mathcal{X})p(\Theta^c) & \text{MAP}^c \\ \arg\max p(\mathcal{Y}|\Theta^c,\mathcal{X}) & \text{ML}^c \end{cases} \tag{2.13}$$

We typically find the maximum of the logarithm of the above quantities (which results in the same arg max). Thus we expand the above for the maximum conditional likelihood case as follows:

$$\begin{aligned} \log p(\mathcal{Y}|\Theta^c,\mathcal{X}) &= \sum_t \log p(y_t|X_t,\Theta^c) \\ &= \sum_t \log \frac{p(y_t,X_t,\Theta^c)}{p(X_t|\Theta^c)} \\ &= \sum_t \log\left(p(y_t,X_t,\Theta^c)\right) - \sum_t \log\left(\int_y p(y,X_t|\Theta^c)dy\right) \end{aligned}$$

The above optimization of conditional likelihood is very similar to the one for maximum likelihood except for the extra negative term which is often referred to as the *background probability* since it is a marginal over the input distribution. Thus, we are trying to maximize the joint likelihood of input and output while minimizing the marginal likelihood over the input data. This sets up an interesting metaphor where a class conditional model is *attracted* to data it should fit through the joint likelihood but *repelled* by the background or data that does not belong to the model's class. Many other criteria are actually conditional likelihood in disguise which sometimes causes confusion. For example, in the speech recognition literature, conditional maximum likelihood is referred to as *maximum mutual information* [158]. Currently, hidden Markov models in the speech community are being trained with these conditional criteria [158] [207] to obtain state of the art performance on large corpus data sets.

Unlike maximum likelihood which has been able to handle incomplete and latent models for years with the EM algorithm, conditional likelihood has been traditionally difficult to maximize, especially in a mixture model scenario. In fact, maximizing conditional likelihood in non-latent models will still give rise to computational difficulties since the background probability involves a log of a sum or a log of an integral over the outputs (classes or scalars) which might break concavity. Most approaches have to resort to gradient descent [20] [105]. Other variants of gradient descent and line search are also emerging in statistics [50]. Recently, the conditional version of the EM algorithm has been proposed in the CEM algorithm [92] [94] (and will be discussed further Chapter 5). As in EM, Conditional Expectation Maximization (CEM) iterates between bounding the conditional likelihood and solving the resulting simpler complete data maximization. This converges iteratively and monotonically to a maximum conditional likelihood solution. As in variational Bayes, a similar use of the CEM bounds on conditional posteriors and conditional likelihoods prior to integration can result in a better and tractable approximation to the conditional Bayesian inference. This would provide a generative model that is more optimized for the task at hand while still relying on a fully Bayesian formalism. We leave this variational conditional Bayesian inference as an interesting future direction to apply the novel bounds.

2.3.3 Logistic Regression

Maximum conditional likelihood (and in a sense maximum likelihood) is also very closely related to logistic regression, a popular technique in the statistics community [76] [127] [68]. Logistic regression is a conditional distribution of a binary output variable y given a input vector x. Typically, the conditional model is given by the following formula (where θ is a parameter vector) $p(y = 1|x) = 1/(1 + \exp(-\theta^T x))$. This generates a linear classifier which is varied by the parameter vector θ and is also referred to as a generalized linear model. There are various ways to augment the framework by computing higher order features from a given x vector. These include handling discrete x values by considering indicator features as in [43] [115].

2.4 Discriminative Learning

Discriminative learning goes beyond the conditional learning perspective and is even more minimalist. Here, only the final mapping from an input (x) to output (y) is important and the final estimate $\hat{y}$ that will be produced is considered [167]. Thus, the estimation of a conditional distribution $p(y|x)$ is also viewed as an unnecessary intermediate step just as we previously argued that the estimation of a joint distribution $p(x, y)$ was also deemed inefficient [4]. Alternatively, we may consider other quantities resulting from the classifier, for example margin distances from the decision boundary to the nearest exemplars. Thus, discriminative techniques only consider the decision boundary or the regression function approximation in evaluating the parameters for a model. Since our learning algorithm is so closely matched with the final task of the system, discriminative learning techniques will not squander resources on an intermediate goal like generative modeling. The resulting performance of the classifier and regressor will therefore be improved. Since we can no longer consider a distribution-based criterion, Bayesian and likelihood-based learning techniques are not immediately applicable.

2.4.1 Empirical Risk Minimization

As opposed to the previous sections where we started with the averaging based solutions (Bayesian integration) and moved to more empirical or local approximations (maximum likelihood), we begin here with an empirical approach to optimizing a discriminative classifier or regressor (and we will show averaging and regularization subsequently). Empirical risk minimization (ERM) is a discriminative estimation criterion which does not make assumptions about the distribution of the input or the output [129] [35] [196] [195] [197]. In ERM, we are typically given a loss function of the form $l(x_t, y_t, \Theta)$ which measures the penalty incurred for a data point (where x_t is input and y_t is desired output) when assigning the parameter Θ to our model. If we only concern ourselves with

[4] This distinction between conditional learning and discriminative learning is not currently a well established convention in the field.

an empirical local solution, we will minimize this loss function on the training data set (which has a total of T training data points). This average loss is also called the empirical risk:

$$\mathcal{R}_{emp}(\Theta) \quad = \quad \frac{1}{T}\sum_{t=1}^{T} l(x_t, y_t, \Theta)$$

This is meant to be a coarse approximation of the true loss of a classifier on the unknown distribution of the samples, also known as the expected risk:

$$\mathcal{R}(\Theta) \quad = \quad \int_{x \times y} P(x,y) l(x,y,\Theta) dx dy$$

In the limit of infinite data, the above loss functions will become almost equal for a given Θ value. Here, Θ specifies a mapping which will produce an estimated $\hat{y}_t$ from the input x_t. The loss function measures the level of disagreement between y_t and $\hat{y}_t$. Possible choices are quadratic loss, i.e. $\|y_t - \hat{y}_t\|$ or even a binary, winner take all, loss for classification. Although one can often reinterpret a loss functions as corresponding to the likelihood under a specific choice of conditional output distribution, this may be sometimes awkward. The important aspect is ERM's emphasis on the actual output and the resulting deterministic classification boundary that will be formed. For example, we may choose to compute the classification result in a winner take all sense in which case the loss will be 0 if we select the class appropriately and the loss will be 1 if we do not. This type of hard classification is fundamental to discriminative estimation. It would be awkward to represent as a conditional distribution (or logistic regressor) but one possibility might be a very sharp version of the logistic function, i.e.:

$$p(y=1|x) \quad = \quad \begin{cases} 1 & \text{if} \quad x > 0 \\ 0 & \text{if} \quad x \leq 0 \end{cases} \tag{2.14}$$

2.4.2 Structural Risk Minimization and Large Margin Estimation

Since ERM is only locally attempting to optimize the model to the training data, it does not necessarily coincide well with the true expected risk and thus may not exhibit good generalization behavior to future data. An alternative is to consider augmenting the local solution with a prior or regularizer that favors estimates that are more likely to agree with future data and are based on a measure of the model capacity. This form of regularized ERM has been called structural risk minimization (SRM) where the capacity is measured in terms of the so-called Vapnik-Chervonenkis dimension [196] [195] [197] [35] [109]. SRM will not perform as well on the training data as ERM but should generalize better to future data.

The risk or 'expected loss' (for samples outside of the training set) is then bounded above by the empirical risk plus a term that depends only on the size of training set, T and the VC-dimension of the classifier, h. This non-negative integer quantity measures the capacity of a classifier and is independent of the distribution of the data. The following bound on the expected loss holds with probability $1-\delta$:

$$\mathcal{R}(\Theta) \quad \leq \quad \mathcal{R}_{emp}(\Theta) + \sqrt{\frac{h(\log(2T/h)+1) - log(\delta/4)}{T}}$$

The SRM principle suggests that we minimize the upper bound on $\mathcal{R}(\Theta)$ to minimize the expected risk itself. Thus we seek to minimize a combination of the expected risk and the VC-dimension h.

For linear classifiers, this motivates the use of a classifier that fits data and also has large margin. Large margins mean that we also try to maximize the minimum distance from the points to the decision boundary that separates them. These principles give rise to the support vector machine (SVM). SVMs are particularly important in contemporary machine learning since they have recently provided state of the art classification and regression performance. In many senses, these are the current workhorses of discriminative estimation. However, due to their fundamentally discriminative formalism, they don't enjoy the flexibility of generative modeling (priors, invariants, structure, latent variables, etc.) which limits their applicability.

2.4.3 Bayes Point Machines

An alternative to the SRM or SVMs is to not only consider a single solution that fits to the data in conjunction with a helpful regularizer or prior but to consider a weighted combination of all possible models. Thus, as in Bayesian inference, for better generalization properties, we will consider a discriminative averaging classifier. For instance, we may attempt to average all linear classifiers that perfectly classify the training data (such hard classification is mainly a discriminative concept). This problem can be cast in the framework of Bayesian inference (or more specifically a conditional Bayesian inference problem). One popular approximation to the true Bayesian inference is called the Bayes point machine (BPM) [72] [199] [135] [168]. For tractability reasons, the BPM is not the true result of Bayesian inference but rather a single point approximation. Instead of summing the effect of all linear classifier models, the BPM uses a single model that is the closest to the mean over the continuous space of valid models (i.e. the linear classifiers with perfect classification accuracy on training).

Thus, in a Bayesian way, we would like to average over all linear models. However this averaging is not a soft probabilistic weighting but is done according to a discriminative criterion which makes a binary decision: only count the classifiers that perfectly separates the training data. This corresponds to the conditional distribution in Equation 2.14. The averaging over models does bring forth slightly better generalization properties for the Bayes point machine (BPM). Unfortunately, in practice, the performance does not exceed that of SVMs in a consistent manner. Furthermore, the BPM does not easily handle non-separable data sets where averaging multiple models according to perfect classification would yield no feasible solution whatsoever. Also, a practical consideration is that the BPM is very difficult to compute requiring computational effort that far surpasses generative modeling approaches as well as SVMs (if the latter are implemented efficiently as in [154]).

Our main concern is that the BPM and its counterparts were really designed to handle linear models or kernel-based nonlinearities. Therefore, they are not easily computable for classifiers arising from the large spectrum of generative models. For instance, exponential family and mixtures of the exponential family cannot be easily estimated in a BPM framework. Thus, they don't enjoy the flexibility of generative modeling (priors, non-separability, invariants, structured models, latent variables, etc.) which limits their applicability. Another discriminative averaging framework that addresses these limitations is Maximum Entropy Discrimination (MED) and will be introduced in the following chapter.

2.5 Joint Generative-Discriminative Learning

After having explored the spectrum of discriminative and generative modeling, we see a strong argument for a hybrid approach that combines these deeply complementary schools of thought. Fusing the versatility and flexibility of generative models with the power of a discriminative framework that focuses resources on the given task would be extremely valuable. Furthermore, as argued throughout, an averaging based approach (as opposed to local or a regularized local fit to training data) promises better generalization and a more principled Bayesian treatment.

Several approaches have been recently proposed for combining the generative and discriminative methods. These include Bayesian estimation with special priors such as automatic relevance detection [188]. However, these have only explored discriminative learning in the context of simple linear or kernel-based models and have yet to show applicability to the large spectrum of generative models.

An alternative technique involves modular combination of generative modeling with subsequent SVM classification using Fisher kernels [83]. This technique is readily applicable to a large spectrum of generative models which are first easily estimated with maximum likelihood and then mapped into features/kernels for training in an SVM. However, the piece-meal cascaded approach of maximum likelihood followed by large margin estimation does not fully take advantage of the power of both techniques. For example, since the generative models are first estimated by maximum likelihood, this non-discriminative criterion might collapse important aspects of the model and sacrifice modeling power (particularly under latent situations). For example, due to a model-mismatch, the pre-specified class of generative model may not have enough flexibility to capture all the information in the training data. It then becomes possible that the model's resources will be misused and encode aspects of the data that are irrelevant for discrimination (instead of task-related information). This ultimately results in a loss of valuable modeling power before we feed into the subsequent SVM layer making it too late to exploit some aspects of the generative model for discrimination. For instance, a maximum likelihood HMM trained on speech data may focus all modeling power on the vowels (which are sustained longer than consonants) preventing a meaningful set of features for discrimination in the final SVM stage. In other words, there is no iteration between the generative modeling and the discriminative learning since the maximum likelihood estimate is not adjusted in response to the SVM's criteria. Thus, a simultaneous computation of the generative model with a discriminative criterion would improve on this technique [5].

In the next chapter, we will present the Maximum Entropy Discrimination formalism as a hybrid generative-discriminative model with many of the desirable qualities we have so far motivated. The proposed MED framework is a principled averaging technique which is be able to span the large spectrum of generative models and simultaneously perform estimation with a discriminative criterion.

[5] This method in [83] was also formally encompassed by the Maximum Entropy Discrimination technique in [85].

Chapter 3

Maximum Entropy Discrimination

It is futile to do with more what can be done with fewer [1].

William of Ockham, 1280-1349

Is it possible to combine the strongly complementary properties of discriminative estimation with generative modeling? Can eg. support vector machines and the performance gains they provide be combined elegantly with flexible Bayesian statistics and graphical models? This chapter introduces a novel technique called Maximum Entropy Discrimination (MED) which provides a general formalism for marrying both methods [85].

The duality between maximum entropy theory [86] and maximum likelihood is well known in the literature [113]. Therefore, the connection between generative estimation and classical maximum entropy already exists. MED brings in a novel discriminative aspect to the theory and forms a bridge to contemporary discriminative methods. MED also involves another twist on the usual maximum entropy paradigm in that it considers distributions over model parameters instead of only distributions over data. Although other possible approaches for combining the generative and discriminative schools of thought exist [83, 92, 188, 72], the MED formalism has distinct advantages. For instance, MED naturally spans both ends of the discriminative-generative spectrum: it subsumes support vector machines and extends their driving principles to a large majority of the generative models that populate the machine learning community.

This chapter is organized as follows. We begin by motivating the discriminative maximum entropy framework from the point of view of regularization theory. Powerful convexity, duality and geometric properties are elaborated. We then explicate how to solve classification problems in the context of the maximum entropy formalism. The support vector machine is then derived as a special case. Subsequently, we extend the framework to discrimination with generative models and prove that the whole exponential family of generative distributions is immediately estimable within the MED framework. Generalization guarantees are then presented.

Further MED extensions such as transductive inference, feature selection, etc. are elaborated in the following chapter (Chapter 4). To make the MED framework applicable to the wide range

[1] At the risk of misquoting what Ockham truly intended to say, we shall use this quote to motivate the sparsity which arises from a constraint-based discriminative learner such as the Maximum Entropy Discrimination formalism.

of Bayesian models, latent models are also considered. These mixtures of the exponential family deserve special attention and their development in the context of MED is deferred to Chapter 5.

3.1 Regularization Theory and Support Vector Machines

We begin by developing the maximum entropy framework from a regularization theory and support vector machine perspective (this derivation was first described in [90]). For simplicity, we will only address binary classification in this chapter and defer other extensions to Chapter 4. Regularization theory is a field in its own right with many formalisms (the approach we present is only one of many possible developments). A good contact point for the machine learning reader to regularization theory can be found in [66] [155].

We begin with a parametric family of decision boundaries: $\mathcal{L}(X;\Theta)$ which we shall call *discriminant functions*. Each discriminant function (given a specific parameter Θ) takes an input X and produces a scalar output. The sign (± 1) of the scalar value will indicate which class the input X will be assigned to. For example, a simple type of decision boundary is the linear classifier. The parameters of this classifier $\Theta = \{\theta, b\}$ are the concatenation of a linear parameter vector θ and the scalar bias b. This generate the following linear classifier:

$$\mathcal{L}(X;\Theta) \quad = \quad \theta^T X + b \tag{3.1}$$

To estimate the optimal $\hat{\Theta}$, we are given a set of training examples $\{X_1, \ldots, X_T\}$ and the corresponding binary (± 1) labels $\{y_1, \ldots, y_T\}$. We would like to find a parameter setting for Θ that will minimize some form of classification error. Once we have found the best possible $\hat{\Theta}$, we can use our classifier to predict the labels of future input examples via:

$$\hat{y} \quad = \quad \text{sign}\mathcal{L}(X;\hat{\Theta}) \tag{3.2}$$

We will form a measure of classification error based loss functions $L()$ for each data point which will depend on our parameter Θ only through the *classification margin*. The margin [2] is defined as $y_t\,\mathcal{L}(X_t;\Theta)$ and is large and positive whenever the label y_t agrees with the scalar valued prediction $\mathcal{L}(X_t;\Theta)$ and negative when they disagree. We shall further assume that the loss function, $L : \Re \rightarrow \Re$, is a *non-increasing* and *convex* function of the margin. Thus a larger margin results in a smaller loss. We also introduce a regularization penalty $R(\Theta)$ on the models, which favors certain parameters over others (like a prior).

The optimal parameter setting $\hat{\Theta}$ is computed by minimizing the empirical loss and regularization penalty:

$$\min_{\Theta} \left\{ R(\Theta) + \sum_t L\left(\, y_t\,\mathcal{L}(X_t;\Theta)\,\right) \right\}$$

A more straightforward solution for $\hat{\Theta}$ is achieved by recasting the above as a constrained optimization:

$$\begin{array}{cc} \min_{\{\Theta,\,\gamma_t\,\forall t\}} & R(\Theta) + \sum_t L(\gamma_t) \\ \text{subject to} & y_t\,\mathcal{L}(X_t;\Theta) - \gamma_t \geq 0,\ \forall t \end{array} \tag{3.3}$$

[2] It should be noted that regularization theory is not limited to margin-based concepts. In general the penalty function or stabilizer terms may depend on many other regularization criteria through a wide area of possible norms and semi-norms. One interpretation of regularization theory is as an approach to solving inverse problems. It spans applications in spline-fitting to pattern recognition and employs many sophisticated mathematical constructs such as reproducing kernel Hilbert spaces [53].

Here, we have also introduced the margin quantities: γ_t as slack variables in the optimization which represent the minimum margin that $y_t\mathcal{L}(X_t;\Theta)$ must satisfy. The minimization is now over both the parameters Θ and the margins γ_t.

A (linear) support vector machine can be seen as a particular example of the above formulation. There, the discriminant function is a linear hyper-plane as in Equation 3.1. Furthermore, the regularization penalty is $R(\Theta) = \frac{1}{2}\theta^T\theta$, i.e. the norm of the parameters to encourage large margin solutions. The slack variables provide the SVM with a straightforward way to handle non-separable classification problems. Thus, for the (primal) SVM optimization problem, we have:

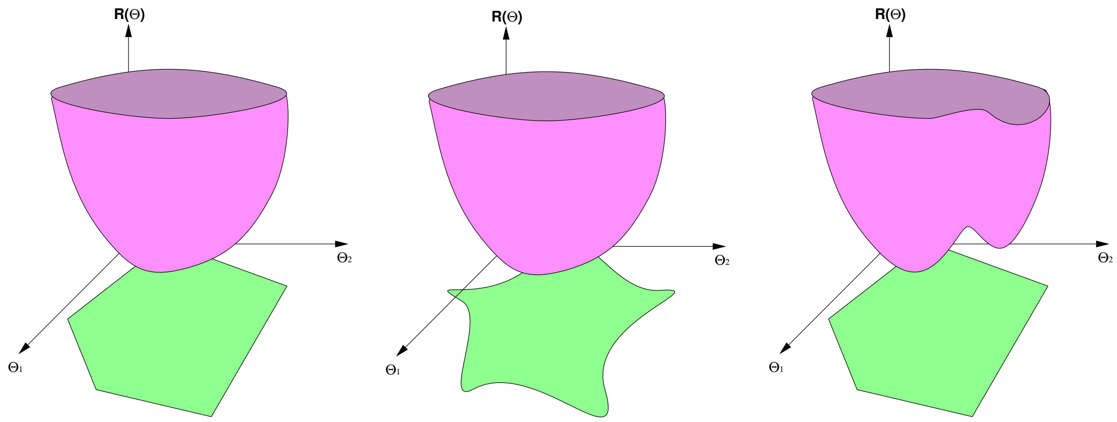
$$\begin{array}{cc} \min_{\{\theta,\,\gamma_t\,\forall t\}} & \frac{1}{2}\theta^T\theta + \sum_t L(\gamma_t) \\ \text{subject to} & y_t\,(\theta^T X_t + b) - \gamma_t \geq 0,\ \forall t \end{array}$$

At this point, we will focus on the optimization over Θ alone and ignore the optimization over the slack variables γ_t. The effect of the restriction is that the resulting classifier (or support vector machine) will require linearly separable data. In practice, we will assume the slack variables are to be held constant and set them manually to, eg. unity, $\gamma_t = 1\,\forall t$. This restrictive assumption is made to simplify the following derivations and does not result in a loss of generality. The restriction will be loosened subsequently permitting us to consider non-separable cases as well.

3.1.1 Solvability

At this point, it is crucial to investigate under what conditions the above constrained minimization problem in Equation 3.3 is solvable. For instance, can the above be cast as a convex program or can $\hat{\Theta}$ be computed uniquely?

A convex program typically involves the minimization of a convex cost function under a convex hull of constraints. Under mild assumptions, the solution is unique and a variety of strategies will converge to it (i.e. axis-parallel optimization, linear-quadratic-convex programming, etc.). In Figure 3.1, various constrained optimizations scenarios are presented. Figure 3.1(a) depicts a convex cost function with a convex hull of constraints arising from the conjunction of multiple linear constraints. this leads to a valid convex program.



(a) Valid Convex Program (b) Non-Convex Constraints (c) Non-Convex Cost Function

Figure 3.1: Convex cost functions and convex constraints.

In Figure 3.1(b) the situation is not as promising. Here, several nonlinear constraints are combined and therefore the searchable space forms a non-convex hull. This prevents guaranteed convergence and yields a non-convex program. Similarly, in Figure 3.1(c), we do not have a convex program. However, here the culprit is a non convex cost function (i.e. $R(\Theta)$ is not convex).

Therefore, for a solution to Equation 3.3, we must require that: the penalty function $R(\Theta)$ is *convex*, and that the conjunction of the classification constraints $\forall t$ *form a convex hull*. The intersection of linear constraints (under mild conditions) will always form a convex hull. In addition, it should be evident that it is *unlikely* that the intersection of multiple nonlinear constraints will form a convex hull. Therefore, it is clear that the classification constraints in the regularization framework need to be linear or at least consistently mappable to a space where they become linear.

3.1.2 Support Vector Machines and Kernels

Inspecting a support vector machine, we can immediately see that the penalty function, i.e. $R(\Theta) = \frac{1}{2}\theta^T\theta$ is convex and that the a linear hyper-plane discriminant will give rise to linear constraints and a convex hull. Thus, as is well known, the SVM is solvable via a convex program (actually more simply as a quadratic program [35]) or sequential minimal optimization [154].

But, what do we do when $\mathcal{L}(X_t;\Theta)$ is nonlinear? For example, we may wish to deal with decision boundaries that arise from generative models. These can be computed via the log-likelihood ratio of two generative models $P(X|\theta_+)$ and $P(X|\theta_-)$ (one for each class). Here the parameter space includes the concatenation of the positive generative model, the negative one and a scalar bias $\Theta = \{\theta_+, \theta_-, b\}$. This gives rise to the following nonlinear discriminant functions:

$$\mathcal{L}(X;\Theta) \quad = \quad \log \frac{P(X|\theta_+)}{P(X|\theta_-)} + b \qquad (3.4)$$

Unfortunately, these nonlinear decision boundaries generate a search space for Θ that is no longer a convex hull (compromising the uniqueness and solvability of the problem).

In some cases, nonlinear decision boundaries (i.e. nonlinear SVMs), can be handled via the so-called 'kernel trick'. If a decision boundary is nonlinear, one can consider a mapping of the data through $\Phi(X_t)$ into a higher dimensional 'feature' space. Therein, the Θ parameter vector parameterizes a higher dimensional hyper-plane effectively mimicking the nonlinearity in the original low dimensional space. Furthermore, the constraints too become linear and the search space forms a convex hull.

One subtlety here, however, is that regularization penalty is now different in the feature space than in the original space. Therefore, if we had a quadratic $R(\Theta)$ penalty function in the original space, we would obtain some possibly complicated expression for it in the feature space. This is reasonable in the case of SVMs since the VC-dimension generalization guarantees hold at the level of the feature space. This permits us to *artificially preserve* a quadratic penalty function in the feature space (which would map to a quite complicated one in the original space). The term 'kernel' simply arises since optimizing a quadratic penalty function in the feature space only requires inner products between the high dimensional vectors $\Phi(X_t)$ and these are implicitly computable using kernels $k(X_t, X_{t'})$ without the explicit mapping $\Phi(X_t)$.

However, more generally, we may have a specific regularization penalty in mind at the level of the original space and/or nonlinearities in the classifier that prevent us from considering the high-dimensional mapping 'trick'. This problematic situation is often the case for generative models and motivates an important extension (MED) to the regularization theory so far discussed.

3.2 MED - Distribution over Solutions

We will now generalize this regularization formulation by presenting the maximum entropy discrimination framework [85] [90]. First, we begin by noting that it is not necessarily ideal to solve for a single optimal setting of the parameter Θ when we could instead consider solving for a full distribution over multiple Θ values (i.e. give a distribution of solutions). The intuition is that many different settings of Θ might generate relatively similar classification performance so it would be better to estimate a distribution $P(\Theta)$ that preserves this flexibility instead of a single optimal $\hat{\Theta}$. Clearly, with a full distribution $P(\Theta)$ we can subsume the original formulation if we choose $P(\Theta) = \delta(\Theta, \hat{\Theta})$ where the delta function can be seen as point-wise probability mass concentrated where $\Theta = \hat{\Theta}$. So, this type of probabilistic solution is a superset of the direct optimization [3]. Here, we would like our $P(\Theta)$ to be large when Θ-values yield good classifiers and to be close to zero at Θ-values that yield poor classifiers. This probabilistic generalization will facilitate a number of extensions to the basic regularization/SVM approach. We modify the regularization approach as follows.

Given a distribution over $P(\Theta)$, we can easily modify the regularization approach for predicting a new label from a new input sample X that was shown in Equation 3.2. Instead of merely using one discriminant function at the optimal parameter setting $\hat{\Theta}$, we will *integrate* over all discriminant functions weighted by $P(\Theta)$:

$$\hat{y} \;=\; \mathrm{sign} \int_{\Theta} P(\Theta)\mathcal{L}(X;\Theta)d\Theta \qquad (3.5)$$

How do we estimate $P(\Theta)$? Again we consider an expectation form of the previous approach and cast Equation 3.3 as an integration. The classification constraints will also be applied in an expected sense. It is inappropriate to directly apply the $R(\Theta)$ arbitrary penalty function to infinite dimensional probability density functions such as $P(\Theta)$. Instead of considering an expectation of penalty functions, we will apply a canonical penalty function for distributions, the negative entropy. Minimizing the negative entropy is equivalent to maximizing the entropy. 'Maximum Entropy' theory was pioneered by Jaynes and others [119] to compute distributions with moment constraints. In the absence of any further information, Jaynes argues that one should satisfy the constraints in a way that is least committal or prejudiced. This gives rise to the need for a maximum entropy distribution, one that is as close to uniform as possible. Here, we assume Shannon Entropy defined as $H(P(\Theta)) = -\int P(\Theta)\log P(\Theta)d\Theta$. Traditionally in the Maximum Entropy community, distributions are computed subject to moment constraints (i.e. not discrimination or classification constraints). Here, the 'Discrimination' term is added to specify that our framework borrows from the concepts of regularization/SVM theory and is satisfying discriminative classification constraints (based on margin).

This gives us the following novel MED formulation[4] for finding a *distribution* $P(\Theta)$ over the parameters Θ:

$$\begin{array}{cc} \min_{P(\Theta)} & -H(P(\Theta)) \\ \text{subject to} & \int P(\Theta)\,[y_t\mathcal{L}(X_t,\Theta) - \gamma_t]d\Theta \geq 0 \;\; \forall t \end{array}$$

At present, negative entropy is not a very flexible as a surrogate penalty function. To generalize, cast negative entropy as a Kullback-Leibler divergence from $P(\Theta)$ to a target uniform distribution:

[3]In practice, though, other (possibly parametric) restrictions may arise on $P(\Theta)$ that prevent us from generating arbitrary delta functions in this manner.

[4]At this point we have assumed that the margins γ_t and their loss functions are held fixed (these are typically set to $\gamma_t = 1\,\forall t$). This assumption will be relaxed subsequently.

$-H(P(\Theta)) = KL(P(\Theta)\|P_{uniform}(\Theta))$. Kullback-Leibler divergence is defined as follows[5]:

$$KL(P(\Theta)\|Q(\Theta)) \;=\; \int P(\Theta)\log\frac{P(\Theta)}{Q(\Theta)}d\Theta$$

If we have prior knowledge about the desired shape of $P(\Theta)$, we may not necessarily want to favor high entropy uniform solutions. Instead we can customize the target distribution and use a non-uniform one. Thus we replace our penalty function with the Kullback-Leibler divergence to any prior, $KL(P(\Theta)\|P_0(\Theta))$. This gives us our more general *Minimum Relative Entropy Discrimination* (MRE or MRED) formulation:

Definition 1 *Find a distribution $P(\Theta)$ over the parameters Θ that minimizes $KL(P_\Theta\|P^0_\Theta)$ subject to $\int P(\Theta,\gamma)\,[y_t\mathcal{L}(X_t,\Theta)-\gamma_t]d\Theta \geq 0 \;\; \forall t$. Here P^0_Θ is the prior distribution over the parameters. The resulting decision rule is given by $\hat{y} = \mathrm{sign}(\int P(\Theta)\mathcal{L}(X;\Theta)d\Theta\,)$.*

It is traditional to continue to refer to *minimum relative entropy* approaches as *maximum entropy*. Therefore, at the risk of confusion, we shall adopt this convention in the nomenclature and refer to Definition 1 as 'Maximum Entropy Discrimination'. At this point, we evaluate the solvability of our formulation.

Figure 3.2 depicts the problem formulation. We note that now we are dealing with a possibly infinite-dimensional space since instead of solving for a parameter vector Θ, we are solving for $P(\Theta)$, a probability distribution. In the figure, the axes represent variation of two coordinates of the possibly continuous distribution, $P(\Theta)$ and $P(\Theta')$. Instead of $R(\Theta)$, a penalty function, we have the KL-divergence which is a convex function of $P(\Theta)$. Furthermore, the constraints are expectations with respect to $P(\Theta)$ which means they are *linear* in $P(\Theta)$. These linear constraints are guaranteed to combine into a convex hull for the search space of $P(\Theta)$ *regardless of the nonlinearities in the discriminant function!*

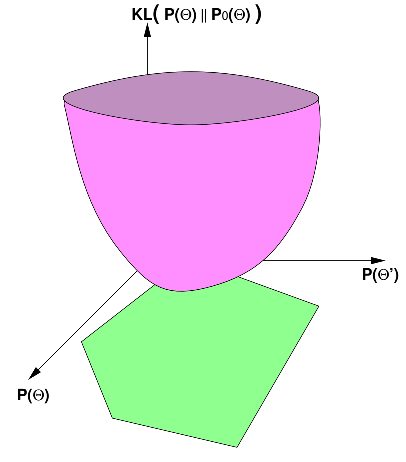


Figure 3.2: MED Convex Program Problem Formulation.

Therefore, the solution to Definition 1 is given by a valid convex program. In fact, the solution to the MED classification problem in Definition 2 is directly solvable using a classical result from maximum entropy:

[5] Often, the KL-divergence $KL(P\|Q)$ will also be written as $D(P\|Q)$.

Theorem 1 *The solution to the MED problem for estimating a distribution over parameters has the following form (cf. Cover and Thomas [40]):*

$$P(\Theta) = \frac{1}{Z(\lambda)} P_0(\Theta)\, e^{\sum_t \lambda_t [y_t \mathcal{L}(X_t|\Theta) - \gamma_t]}$$

where $Z(\lambda)$ is the normalization constant (partition function) and $\lambda = \{\lambda_1, \ldots, \lambda_T\}$ defines a set of non-negative Lagrange multipliers, one per classification constraint. λ are set by finding the unique maximum of the jointly concave objective function

$$J(\lambda) = -\log Z(\lambda) \tag{3.6}$$

The above solution arises as the dual problem to constrained optimization in Definition 1 (the primal problem) via the Legendre transform. Under mild conditions, a solution always exists and if so then it is unique. Occasionally, the objective function $J(\lambda)$ may grow without bound and prevent the existence of a unique solution however this situation is rare in practice. Furthermore, it is typically far easier to solve the dual problem since the complexity of the constraints is alleviated. It is obvious that the constraints on the Lagrange multipliers, (i.e. non-negativity) are more straightforward to realize than constraints on the possibly infinite dimensional distribution $P(\Theta)$ in the primal problem. The non-negativity of the Lagrange multipliers arises in maximum entropy problems when inequality constraints are present in the primal problem (such as those representing our classification constraints in Definition 1) [6]. At this point, we shall loosen the constraint that the margins are fixed and allow, eg. classification scenarios which are non-separable.

3.3 MED - Augmented Distributions

The MED formulation so far has made the assumption that the margin values γ_t are pre-specified and held fixed. Therefore, the the discriminant function must be able to perfectly separate the training examples with some pre-specified margin value. This may not always be possible in practice (i.e. for non-separable data sets) and will generate an empty convex hull for the solution space. Thus, we need to revisit the setting of the margin values and the loss function upon them. First, recall that we had so far ignored the loss function in the regularization framework as we derived the MED technique since we held the margins fixed. However, the choice of the loss function (penalties for violating the margin constraints) also admits a more principled solution in the MED framework.

As we had shown earlier for the case of the parameters, let us also now consider a distribution over margins, eg. $P(\gamma)$ in the MED framework [85]. Typically, for good classification (VC-dimension generalization guarantees encourage large margin solutions) performance, we will choose *margin distributions that favor larger margins.* Furthermore, by varying our choice of distribution we can effectively mimic or consider various loss functions associated with γ. Also, by choosing priors that allow a non-zero probability mass for negative margins, we can permit non-separable classification (without ad-hoc slack variables as in SVMs). This will ensure that the classification constraints will never give rise to an empty admissible set. The MED formulation will then give a solution over the joint distribution, namely $P(\Theta, \gamma)$. This gives a weighted continuum of solutions instead of specifying a single optimal value for each as in the regularization approach. There is a caveat, however, since the MED constraints apply only through expectations over the margin values. We are now satisfying a looser problem than when the margin values were set and thus this transition from margin values to margin distributions is less natural than the previous transition from parameter

[6] Equality constraints in the primal problem would generate Lagrange multipliers that are arbitrary scalars in $(-\infty, \infty)$

extrema to parameter distributions. Since there are multiple margin values (one for each training data instance t), $P(\gamma)$ is an aggregate distribution over all margins and will typically be factorized as $P(\Theta, \gamma) = P(\Theta)\Pi_t P(\gamma_t)$. This leads to the following more general MED formulation:

Definition 2 *The MED solution is the $P(\Theta, \gamma)$ over the parameters Θ and the margin variables $\gamma = [\gamma_1, \ldots, \gamma_T]$ that minimizes $KL(P_\Theta \| P_\Theta^0) + \sum_t KL(P_{\gamma_t} \| P_{\gamma_t}^0)$ subject to $\int P(\Theta, \gamma)\,[y_t \mathcal{L}(X_t, \Theta) - \gamma_t] d\Theta d\gamma \geq 0\ \forall t$. Here P_Θ^0 is the prior distribution over the parameters and and P_γ^0 is the prior over margin variables. The resulting decision rule is given by $\hat{y} = \mathrm{sign}(\int P(\Theta)\mathcal{L}(X; \Theta) d\Theta)$.*

Once again, the above solution exists under mild assumptions and is unique. Here, though, the constraints are now not just expectation constraints over the parameter distribution but also over an expectation on the margin distribution. This relaxes the convex hull since the constraints do not need to hold for a specific margin. The constraints need only hold over a *distribution* over margins that can include negative margins, thus permitting us to consider non-separable classification problems. Furthermore, in applying MED to a problem, we no longer specify ad-hoc regularization penalty (the $R(\Theta)$) and margin penalty functions (the $L(\gamma_t)$ loss-functions) but instead specify probability distributions. These distributions can sometimes be more convenient to specify and then automatically give rise to penalty functions for the model and the margins via KL-divergences. More specifically, the model distribution will give rise to the divergence term $KL(P_\Theta, P_\Theta^0)$ and the margin distribution will give rise to a divergence term $KL(P_{\gamma_t} \| P_{\gamma_t}^0)$ which correspond to the regularization penalty and the loss functions respectively. Since both terms are based on probability distributions and KL-divergence, the trade off between classification loss and regularization now on a *common probabilistic scale.*

The solution to the non-separable MED classification problem in Definition 2 is solved as follows:

Theorem 2 *The solution to the MED problem for estimating a distribution over parameters and margins (as well as further augmentations) has the following general form (cf. Cover and Thomas 1996):*

$$P(\Theta, \gamma) = \frac{1}{Z(\lambda)} P_0(\Theta, \gamma)\, e^{\sum_t \lambda_t [y_t \mathcal{L}(X_t | \Theta) - \gamma_t]}$$

where $Z(\lambda)$ is the normalization constant (partition function) and $\lambda = \{\lambda_1, \ldots, \lambda_T\}$ defines a set of non-negative Lagrange multipliers, one per classification constraint. λ are set by finding the unique maximum of the jointly concave objective function

$$J(\lambda) = -\log Z(\lambda) \tag{3.7}$$

Further details for the choices of the priors for the parameters and margins as well as other distributions will be elaborated in the following sections. It is always possible to recast the optimization problem the maximum entropy formulation has generated back into the regularization form and in terms of loss functions and regularization penalties [90]. However, MED's probabilistic formulation, is intuitive and provides more flexibility. For instance, we can continue to augment our solution space with distributions over other entities and maintain the convex cost function with convex constraints. For example, one could include a distribution over unobserved labels y_t or unobserved inputs X_t in the training set. Or, we could introduce further continuous or discrete variables into the discriminant function that are unknown and integrate over them. Thus, the distribution $P(\Theta)$ could effectively become $P(\Theta, \gamma, y, X, \ldots)$ and in principle, we will still maintain a similar convex program structure and the dual solution posed as portrayed in Theorem 2. These types of extensions will be elaborated further in Chapter 4. One important caveat remains, however, when we augment distributions: we should maintain a balance between the various priors we are trying to minimize KL-divergence to. If a prior over models $P_0(\Theta)$ is too strict, it may overwhelm a prior over other quantities such as margins, $P_0(\gamma)$ and vice-versa. Therefore, the minimization of KL-divergence will be skewed more towards one prior than the other.

3.4 Information Theoretic and Geometric Interpretation

There is an interesting geometric interpretation for the MED solution which can be described as a type of information projection. This projection is depicted in Figure 3.3 and is often referred to as a relative entropy projection or e-projection as in [1]. The multiple linear constraints form a convex hull that generates an admissible set called $\mathcal{P}$. This convex hull is also referred to as an 'm-flat constraint set'. The MED solution is the point in the admissible set that is closest in terms of divergence from the prior distribution $P_0(\Theta)$. This analogy extends to cases where the distributions are also over margins, unlabeled exemplars, missing values, structures, or other probabilistic entities that are introduced when designing the discriminant function.

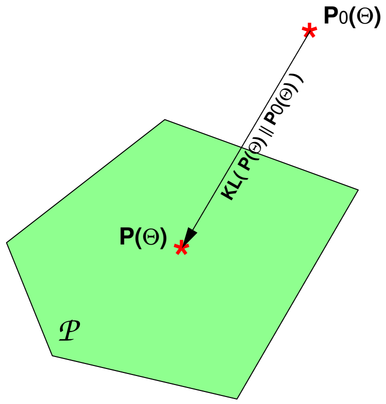


Figure 3.3: MED as an Information Projection Operation.

The MED probabilistic formalism also has interesting conceptual connections to other recent information theoretic and boosting works. One point of contact is the entropy projection and boosting (Adaboost) framework developed in [170] and [110]. Boosting uses a distribution that weights each data point in a training set and forms a weak learner based upon it. This process is iterated, updating the distribution over data and the weak learner for $t = 1 \ldots T$ iterations. All hypothesis are then combined in a weighted mixture of weak learners called the final master algorithm. Effectively, each boosting step estimates a new distribution P^{t+1} over the training data that both *minimizes* the relative entropy to a prior distribution P^t and is *orthogonal* to a 'performance vector' denoted U^t. The performance vector U^t is of the same cardinality as P^t and has values ranging between $[-1, 1]$. If the previous 'weak learner' given by prior probability distribution correctly classifies a data point, then the U vector at that training datum's index has a value close to 1 (i.e. $U_i^t = 1$). If the datum is poorly classified, then U_i^t is -1 at the corresponding index. Therefore, we update the distribution using the following exponential update rule (which follows directly from classical maximum entropy results):

$$P_i^{t+1} \quad \propto \quad P_i^t \exp(-\alpha U_i^t)$$

Instead of considering an iterative approach where individual corrective updates are made, we may enforce all the the orthogonality constraints we have up until now and generate a full convex hull to constrain the entropy projection [110]:

$$P_i^{t+1} \quad \propto \quad P_i^t \exp(-\sum_{q=1}^{t} \alpha_{t,q} U_i^q)$$

Kivenen and Warmuth [110] argue that each new distribution should provide information not present in the current weak hypothesis given the individual orthogonality constraint. When we simultaneously consider *all* orthogonality constraints up until time t, the new hypothesis should provide new information that is uncorrelated from *all* previous hypotheses. The convex hull of constraints results in the exponentiated $\sum_{q=1}^{t} \alpha_{t,q} U_i^q$ terms in the above equation which are strongly reminiscent of the MED formulation's exponentiated classification constraints (and their Lagrange multipliers). We can therefore interpret the MED formulation as minimizing a divergence to a prior while extracting as much information as possible from the training data.

Another information-theoretic point of contact can be found in the work of Tishby and others [189] [177]. Here, the authors propose minimizing the lossy coding of input data X via a compact representation $\tilde{X}$ while maintaining a constraint on the mutual information between the coding and some desired output variable Y, $I(\tilde{X};Y)$. This information-theoretic setting gives rise to the Lagrangian optimization $I(X;\tilde{X}) - \beta I(\tilde{X};Y)$. The result is an efficient representation of the input data X which extracts as much information as possible (in terms of bits to encode) from the relevance output variable. A loose analogy can be made to the MED framework which solves for a solution distribution $P(\Theta)$ which minimally encodes the prior distribution $P_0(\Theta)$ (analogous to the input vectors $\tilde{X}$ and X respectively) such that the classification constraints due to the training data (analogous to the relevance variables) are satisfied and provide as much information as possible.

An important connection also lies between MED and Kullback's early work on the so-called Minimum Discrimination Information method [113]. The definition Kullback adopts for 'discrimination' is slightly different from the one we are discussing here. It mainly involves discrimination between two competing hypotheses based on an information metric where one hypothesis has to satisfy some additional constraints while being as close to the prior hypothesis as possible. The mechanism proposed by Kullback is therefore very similar to the maximum entropy formalism that Jaynes proposes[86] and he even describes connections to Shannon's theory of communication [174]. Kullback points out various important elaborations to both these yet ultimately the Minimum Discrimination Information method once again finds a distribution that is as close as possible to a prior in terms of KL-divergence subject to various moment constraints. The information between hypothesis involves distributions over the measurable space as opposed to distributions over parameters as in MED. Furthermore, the constraints used are not margin-based (or even classification-based) as in MED and thus do not give rise to a discriminative classifier (or regressor). Nevertheless, MED seems to be a natural continuation of the Kullback's approach and can be seen as contemporary effort to combine it with the current impetus towards discriminative estimation as in the SVM literature (as well as the corresponding generalization arguments).

3.5 Computing the Partition Function

Ultimately, implementing the MED solution given by Theorem 2 hinges on our ability to perform the required calculations. For instance, we need to maximize the concave objective function to obtain the optimal setting of the Lagrange multipliers λ:

$$J(\lambda) = -\log Z(\lambda)$$

Ideally, therefore, we would like to be able to evaluate the partition function $Z(\lambda)$ either analytically or at least efficiently. More precisely, the partition function is given by:

$$Z(\lambda) = \int P_0(\Theta, \gamma) e^{\sum_t \lambda_t [y_t \mathcal{L}(X_t;\Theta) - \gamma_t]} d\Theta d\gamma \tag{3.8}$$

Given a closed form partition function permits us to have a convenient concave objective function that can then be optimized by standard techniques. Possible choices herein include convex programming, first and second order methods as well as axis-parallel methods. Implementation details as well as some novel speed improvements (such as learning which Lagrange multipliers are critical to maximization) for optimizing $J(\lambda)$ are provided in the Appendix Section 10.1.

An additional use of the partition function comes from a powerful aspect of maximum entropy (and exponential family distributions) which is inherited by MED. The property states that gradients (of arbitrary order) of the log-partition $\log Z(\lambda)$ with respect to a given variable λ_t, are equal to the expectations (of arbitrary order) of the corresponding moment constraints with respect to the maximum entropy distribution. This permits us to easily compute the expectations and variances over $P(\Theta, \gamma)$ of the MED constraints by taking first and second derivatives of $\log(Z)$. Therefore, given a closed-form MED partition function, we can conveniently obtain these expectations as follows [10] [113]:

$$\begin{aligned} \frac{\partial \log Z(\lambda)}{\partial \lambda_t} &= E_{P(\Theta,\gamma)} \{y_t \mathcal{L}(X;\Theta) - \gamma_t\} \\ \frac{\partial^2 \log Z(\lambda)}{\partial^2 \lambda_t} &= Var_{P(\Theta,\gamma)} \{y_t \mathcal{L}(X;\Theta) - \gamma_t\} \end{aligned}$$

Unfortunately, integrals are required to compute this critical log-partition function which may not always be analytically solvable. If it is indeed solvable, various strategies can then be used to optimize $J(\lambda)$. For instance, axis-parallel techniques will iteratively converge to the global maximum. In certain situations, $J(\lambda)$ may even be maximized using eg. quadratic programming. Furthermore, online *evaluation* of the decision rule after training from data also requires an integral followed by a sign operation which may not be feasible for arbitrary choices of the priors and discriminant functions. However, this is usually less cumbersome than actually computing the partition function to obtain the optimal Lagrange multipliers.

In the following sections we shall specify under what conditions the computations will remain tractable. These will depend on the specific configuration of the discriminant function $\mathcal{L}(X;\Theta)$ as well as the choice of the prior $P_0(\Theta, \gamma)$. In the following section, we discuss various choices of margin priors, bias priors, model priors and discriminant functions.

3.6 Margin Priors

We can mathematically expand the partition function in Equation 3.8 by noting that the distribution factorizes as follows:

$$Z(\lambda) = \int P_0(\Theta, \gamma)\, e^{\sum_t \lambda_t [y_t \mathcal{L}(X_t;\Theta) - \gamma_t]} d\Theta d\gamma$$

$$
\begin{aligned}
&= \int P_0(\Theta)\Pi_t P_0(\gamma)\, e^{\sum_t \lambda_t [y_t \mathcal{L}(X_t;\Theta)-\gamma_t]} d\Theta d\gamma \\
&= \int P_0(\Theta)\, e^{\sum_t \lambda_t y_t \mathcal{L}(X_t;\Theta)} d\Theta \;\times\; \Pi_t \int P_0(\gamma)\, e^{-\lambda_t \gamma_t} d\gamma_t \\
&= Z_\Theta(\lambda) \times \Pi_t Z_{\gamma_t}(\lambda_t)
\end{aligned}
$$

Recall that our optimization function, $J(\lambda)$ was expressed as the negated logarithm of the partition function:

$$
\begin{aligned}
J(\lambda) &= -\log(Z(\lambda)) \\
&= -\log(Z_\Theta(\lambda)) - \sum_t \log(Z_{\gamma_t}(\lambda_t)) \\
&= J_\Theta(\lambda) + \sum_t J_{\gamma_t}(\lambda_t)
\end{aligned}
$$

These $J_{\gamma_t}(\lambda_t)$ behave very similarly to the loss functions $L(\gamma_t)$ in the original regularization theory approach (actually, they are negated versions of the loss functions). We now have a direct way of finding penalty terms $-J_{\gamma_t}(\lambda_t)$ from margin priors $P_0(\gamma_t)$ and vice-versa. Thus, there is a dual relationship between defining an objective function and penalty terms and defining a prior distribution over parameters and prior distribution over margins.

For instance, consider the following margin prior distribution:

$$
P(\gamma_t) = ce^{-c(1-\gamma_t)}, \gamma_t \leq 1 \tag{3.9}
$$

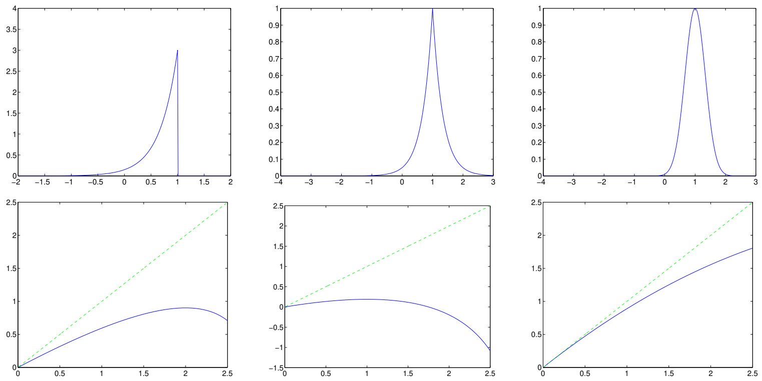
Integrating, we get the penalty function (Figure 3.4):

$$
\begin{aligned}
\log Z_{\gamma_t}(\lambda_t) &= \log \int_{\gamma_t=-\infty}^{1} ce^{-c(1-\gamma_t)} e^{-\lambda_t \gamma_t} d\gamma_t \\
&= -\lambda_t - \log(1-\lambda_t/c)
\end{aligned}
$$

In this case, a penalty is incurred for margins smaller than the prior mean of γ_t which is $1-1/c$. Margins larger than this quantity are not penalized and the associated classification constraint becomes irrelevant (i.e. the corresponding Lagrange multiplier could possibly vanish to 0). Increasing the parameter c will encourage separable solutions and when $c \to \infty$, the margin distribution becomes peaked at the setting $\gamma_t = 1$ which is equivalent to having fixed margins as in the initial MED Definition 1. The choice of the margin distribution will correspond closely to the use of slack variables in the SVM formulation as well as the choice of different loss functions in the regularization theory approach. In fact, the parameter c will play an almost identical role here as the regularization parameter c which upper bounds the Lagrange multipliers in the slack variable SVM solution.

Figure 3.4(a) shows the above prior and its associated potential term (the negated penalty term above). Various other classification margin priors and penalty terms that are analytically computable are given in Table 3.1 and Figure 3.4. Furthermore, in the figure, the dotted green line indicates the potential function that arises when the margins are fixed at unity (which assumes separability). For all plots, the value $c = 3$ was used.

Note that all these priors form concave $J()$ potential functions (or convex penalty functions) as desired for a unique optimum in the Lagrange multiplier space. It should be noted that some potential functions will force an upper bound (via a barrier function) on the λ_t while others will allow them to vary freely (as long as it is non-negative). may or may not set an upper bound on



(a) (b) (c)

Figure 3.4: Margin prior distributions (top) and associated potential functions (bottom).

	Margin prior $P_0(\gamma_t)$	Dual potential term $J_{\gamma_t}(\lambda_t)$
a)	$P_0(\gamma) \propto e^{-c(1-\gamma)}, \gamma \leq 1$	$\lambda + \log(1 - \lambda/c)$
b)	$P_0(\gamma) \propto e^{-c\|1-\gamma\|}$	$\lambda + 2\log(1 - \lambda/c)$
c)	$P_0(\gamma) \propto e^{-c^2(1-\gamma)^2/2}$	$\lambda - (\lambda/c)^2$

Table 3.1: Margin prior distributions and associated potential functions.

the value of λ_t. Other priors and penalty functions are also possible, in particular for the regression case which will be discussed later and which will require quite different margin configurations. We now move to priors for the model, in particular, priors for the bias.

3.7 Bias Priors

Bias is merely a subcomponent of the model however due to its particular interaction with the discriminant function, it will be treated separately here. More specifically, the bias, b, appears as an additive scalar in the discriminant. Recall that Θ which can be seen as a concatenation of all parameters and thus we can consider the breakdown: $\Theta = \{\Theta/b, b\}$. Recall the following form of the discriminant functions from Equation 3.1 (or Equation 3.4):

$$\mathcal{L}(X; \Theta) \quad = \quad \theta^T X + b$$

Such a bias term arises under not only in linear models but many other classification models, including generative classification, multi-class classification, and even regression models. Evidently, one can always set b to zero to remove its effect, or simply set b to a fixed constant, yet the MED approach easily permits us to consider a distribution over b, namely $P(b)$ and to tailor the solution by specifying a prior $P_0(b)$. Here, we consider two possible choices for the prior $P_0(b)$ (although many others are possible): the Gaussian prior and the non-informative prior.

3.7.1 Gaussian Bias Priors

Consider the zero-mean Gaussian prior for $P_0(b)$ given by:

$$P_0(b) \quad = \quad \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{b^2}{2\sigma^2}} \tag{3.10}$$

This prior favors bias values that are close to zero and therefore a priori assumes an even balance between the two binary classes in the decision problem. If we have a prior belief that the class frequencies are slightly skewed, we may introduce a mean into the above prior which would then favors one class over another. The resulting potential term $J_b(\lambda)$ is:

$$\begin{aligned} J_b(\lambda) &= -\log Z_b(\lambda) \\ &= -\log \int_{b=-\infty}^{b=\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{b^2}{2\sigma^2}} e^{\sum_t y_t \lambda_t b} db \\ &= -\frac{\sigma^2}{2} \left(\sum_t y_t \lambda_t \right)^2 \end{aligned}$$

The variance (or standard deviation) σ further specifies how certain we are that the classes are evenly balanced. In terms of the potential function, it constrains with a quadratic penalty the balance between Lagrange multipliers for the negative class and the positive class.

3.7.2 Non-Informative Bias Priors

Evidently, a Gaussian prior will favor values of b that are close to zero. However, in the absence of any knowledge about the bias, it would be reasonable to permit any scalar value for b with equal preference. This will give rise to a non-informative prior. This form of prior can be parameterized as a Gaussian as in Equation 3.10 but with the variance approaching infinity, i.e. $\sigma \rightarrow \infty$. This stretches out the Gaussian until it starts to behave like a uniform distribution on the axis $b \in (-\infty, \infty)$.

The resulting potential term will naturally be:

$$J_b(\lambda) \quad = \quad \lim_{\sigma \rightarrow \infty} -\frac{\sigma^2}{2} \left(\sum_t y_t \lambda_t \right)^2$$

Since we are to maximize the potential terms, if σ grows to infinity, the above objective function will go to negative infinity *unless* the term in the parentheses $\sum_t y_t \lambda_t$ is exactly zero. Therefore, the non-informative prior generates the extra constraint (in addition to non-negativity) on the Lagrange multipliers requiring that $\sum_t y_t \lambda_t = 0$:

Lemma 1 *If the bias prior $P_0(b)$ is set to a non-informative infinite covariance Gaussian, the (non-negative) Lagrange multipliers in the MED solution must also satisfy the following equality constraint:* $\sum_t y_t \lambda_t = 0$

At this point, we have the defined the priors and the computational machinery necessary for the MED formulation to give rise to support vector machines.

3.8 Support Vector Machines

As previously discussed, a support vector machine can be cast in the regularization theory framework and is solvable as a convex program due to the fundamentally linear discriminant function it employs:

$$\mathcal{L}(X;\Theta) = \theta^T X + b$$

One can also interpret the linear decision boundary generatively by considering, for example, the log-likelihood ratio of two Gaussian distributions (one per class) with equal covariance matrices.

$$\mathcal{L}(X;\Theta) = \log \frac{P(X|\theta_+)}{P(X|\theta_-)} + b$$

We shall adopt the first linear discriminant boundary since it has a more efficient parameterization and with the choice of a simple prior will exactly synthesize a support vector machine. In particular if we choose a Gaussian prior on the weights θ of our linear discriminant function, the MED formulation will produce support vector machines:

Theorem 3 *Assuming $\mathcal{L}(X;\Theta) = \theta^T X + b$ and $P_0(\Theta,\gamma) = P_0(\theta)P_0(b)P_0(\gamma)$ where $P_0(\theta)$ is $N(0,I)$, $P_0(b)$ approaches a non-informative prior, and $P_0(\gamma)$ is given by $P_0(\gamma_t)$ as in Equation 3.9 then the Lagrange multipliers λ are obtained by maximizing $J(\lambda)$ subject to $0 \leq \lambda_t \leq c$ and $\sum_t \lambda_t y_t = 0$, where*

$$J(\lambda) = \sum_t [\, \lambda_t + \log(1 - \lambda_t/c)\,] - \frac{1}{2}\sum_{t,t'} \lambda_t \lambda_{t'} y_t y_{t'} (X_t^T X_{t'})$$

The above $J(\lambda)$ objective function is strikingly similar to the SVM dual optimization problem. The only difference between the two is the above formulation has an extra potential term $\log(1 - \lambda_t/c)$ which acts as a barrier function preventing the λ values from growing beyond c. In an SVM, the Lagrange multiplier values are clamped to be no greater than c explicitly as an extra constraint in the convex program. In both formalisms, c plays an almost identical role by varying the degree of regularization and upper bounding the Lagrange multipliers. Typically, low c values increase regularization, vary the sensitivity of the solution to classification errors, robustness to outliers and permit non-separable classification problems. However, in an SVM, the c-regularization parameter arises from an ad-hoc introduction of slack variables to permit the SVM to handle non-separable data. If we let $c \rightarrow infty$, the potential term $\log(1 - \lambda_t/c)$ vanishes and MED gives rise to exactly an SVM (for separable data). In practice, even for finite c, the MED and SVM solutions are almost identical.

3.8.1 Single Axis SVM Optimization

We can greatly simplify the support vector machine by avoiding the non-informative prior on the bias. If we assume a Gaussian prior with finite covariance, the equality constraint $\sum_t \lambda_t y_t = 0$ can be omitted. The resulting convex program only requires non-negativity on the Lagrange multipliers and the updated objective function becomes:

$$J(\lambda) = \sum_t [\, \lambda_t + \log(1 - \lambda_t/c)\,] - \frac{\sigma^2}{2}\left(\sum_t y_t \lambda_t\right)^2 - \frac{1}{2}\sum_{t,t'} \lambda_t \lambda_{t'} y_t y_{t'} (X_t^T X_{t'})$$

Therefore, it is now possible to update a single Lagrange multiplier at a time in an axis-parallel manner. In fact, the update for each axis is analytic (even with the MED logarithmic barrier function in the non-separable case). The minimal working set in this case is 1 while in the SVM, updates to increase the objective must be done simultaneously on at least 2 Lagrange multipliers as in the Sequential Minimal Optimization (SMO) technique proposed by Platt [154]. This gives the MED implementation a simpler optimization problem which lead to gains in computational efficiency without any significant change from the solution produced under non-informative priors.

3.8.2 Kernels

The MED formulation for SVMs also readily extends to the Kernel case where nonlinearities (of the Kernel type) can be immediately folded in by an implicit mapping to a higher dimensional space. The updated MED objective function merely becomes:

$$J(\lambda) = \sum_t [\, \lambda_t + \log(1 - \lambda_t/c)\,] - \frac{\sigma^2}{2}\left(\sum_t y_t \lambda_t\right)^2 - \frac{1}{2}\sum_{t,t'} \lambda_t \lambda_{t'} y_t y_{t'} K(X_t, X_{t'}) \tag{3.11}$$

In the above, our standard inner products of the input vectors $X_t^T X_{t'}$ are replaced with a kernel function of the vectors $K(X_t, X_{t'})$ as is done in the SVM literature. The MED computations remain relatively unchanged since (in the linear discriminant case) all calculations only involve inner products of the input vectors.

3.9 Generative Models

At this point we consider the use of generative models in the MED framework. This fundamentally extends the regularization and SVM discriminative frameworks to the powerful modeling in Bayesian generative models. Herein lies the strength of the MED technique as a bridge between two communities with mutually beneficial tools. Consider a two class problem where we have a generative model for each class, namely $P(X|\theta_+)$ and $P(X|\theta_-)$. These two generative models can be directly combined to form a classifier by considering their log-likelihood ratios as follows:

$$\mathcal{L}(X;\Theta) = \log \frac{P(X|\theta_+)}{P(X|\theta_-)} + b \tag{3.12}$$

Here, the aggregate parameter set is $\Theta = \{\theta_+, \theta_-, b\}$ which includes both generative models as well as a scalar bias. Thus, by merely changing the discriminant function, the MED framework can be used to estimate generative models and guarantee that the decision boundary they give rise to will be optimal in a classification setting. Naturally, the above discriminant function is generally nonlinear and will give rise to a non-convex hull of constraints in a standard regularization setting. However, in the MED framework, due to the probabilistic solution $P(\Theta)$, the above discriminant functions still behave as a convex program.

Estimating $P(\Theta)$ using MED will ultimately yield $P(\theta_+, \theta_-, b)$ which can be used to specify the generative models for the data $P(X|\theta_+)$ and $P(X|\theta_-)$. These will be full generative models that can be sampled from, integrated, conditioned, etc. yet unlike a direct Bayesian framework, these generative models will be also combine to form a high-performance discriminative classifier when plugged into the $\mathcal{L}(X;\Theta)$ discriminant function. Figure 3.5(a) depicts the estimation of a maximum

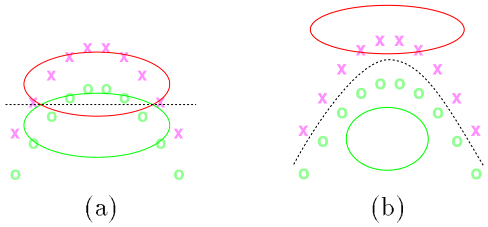


Figure 3.5: Discriminative Generative Models. In (a) we show the standard maximum likelihood estimation of two generative models from the data and the resulting poor classifier decision boundary they generate. In (b), MED moves the generators slightly such that they combine to form an accurate classification boundary.

likelihood generative model while MED moves the generators for each class (ellipses) such that the decision boundary creates good classification separation in Figure 3.5(b).

Whether or not MED estimation is feasible once again hinges upon our ability to compute the log-partition function $Z(\lambda)$. We will show that it is possible to obtain the partition function analytically whenever the generative models $P(X|\theta_+)$ and $P(X|\theta_-)$ are in the exponential family.

3.9.1 Exponential Family Models

We have argued that functions that can be efficiently solved within the MED approach include log-likelihood ratios of the exponential family of distributions. Can we compute the partition function efficiently to actually implement this estimation? First we give details on the exponential family form [9] [32]. It is well known that such distributions have important properties in maximum likelihood estimation [9] [198]. This family subsumes a wide set of distributions and its members are characterized by the following general structure (commonly referred to as the *natural* parameterization):

$$p(X|\theta) = \exp(A(X) + X^T\theta - K(\theta))$$

Where $K(\theta)$ is convex and the distribution is normalized over the space of X. Further details on the exponential family and its many interesting properties can be found in Chapter 5 and [9] [32]. In addition, each exponential family member has a conjugate prior distribution given by:

$$p(\theta|\chi) = \exp(\tilde{A}(\theta) + \theta^T\chi - \tilde{K}(\chi))$$

The conjugate distribution is itself in the exponential family and therefore, its $\tilde{K}$ is also convex.

Whether or not a specific combination of a discriminant function and an associated prior is estimable within the MED framework depends on the computability of the partition function (i.e. the objective function used for optimizing the Lagrange multipliers associated with the constraints). In general, these operations will require integrals over the associated parameter distributions. In particular, recall the partition function corresponding to the binary classification case. Consider the integral over Θ in:

$$Z_\Theta(\lambda) = \int P_0(\Theta)e^{\sum_t \lambda_t y_t L(X_t|\Theta)}d\Theta$$

If we now separate out the parameters associated with the class-conditional densities as well as the bias term (i.e. θ_+, θ_-, b) and expand the discriminant function as a log-likelihood ratio, we obtain the following:

$$Z_\Theta = \int P_0(\theta_+) P_0(\theta_-) P_0(b) e^{\sum_t \lambda_t y_t [\log \frac{P(X|\theta_+)}{P(X|\theta_-)} + b]} d\Theta$$

The above factorizes as $Z_\Theta = Z_{\theta_+} Z_{\theta_-} Z_b$. We can now substitute the exponential family forms for the class-conditional distributions and associated conjugate distributions for the priors. We assume that the prior is defined by specifying a value for χ. It suffices here to show that we can obtain Z_{θ_+} in closed form. The derivation for Z_{θ_-} is identical. We will drop the "+" symbol from Z_{θ_+} for clarity. The problem is now reduced to evaluating:

$$Z_\theta(\lambda) = \int e^{\tilde{A}(\theta) + \theta^T \chi - \tilde{K}(\chi)} e^{\sum_t \lambda_t y_t (A(X_t) + X_t^T \theta - K(\theta))} d\theta$$

We have shown earlier (see Lemma 1) that a non-informative prior over the bias term b leads to the constraint $\sum_t \lambda_t y_t = 0$. Making this assumption, we get

$$\begin{aligned} Z_\theta(\lambda) &= e^{-\tilde{K}(\chi) + \sum_t \lambda_t y_t A(X_t)} \times \int e^{\tilde{A}(\theta) + \theta^T (\chi + \sum_t \lambda_t y_t X_t)} d\theta \\ &= e^{-\tilde{K}(\chi) + \sum_t \lambda_t y_t A(X_t)} \times e^{\tilde{K}(\chi + \sum_t \lambda_t y_t X_t)} \end{aligned}$$

The last evaluation in above results from a natural property of the exponential family. The expressions for $A, \tilde{A}, K, \tilde{K}$ are known for specific distributions in the exponential family and can easily be used to complete the above evaluation, or realize the objective function (which holds for any exponential-family distribution):

$$\log Z_\theta(\lambda) = \tilde{K}(\chi + \sum_t \lambda_t y_t X_t) + \sum_t \lambda_t y_t A(X_t) - \tilde{K}(\chi)$$

Therefore, it is clear that we can compute the objective function $J(\lambda)$ for any discriminant function arising from exponential family generative models. In fact, integration doesn't even need to be performed since we have an analytic expression of our objective function in terms of the natural parameterization for all exponential family distributions. It should also be noted that the above objective function often bears a strong resemblance to the evidence term in Bayesian inference (i.e. Bayesian integration), where the Lagrange multipliers seem to act as weights on the Bayesian inference. It is straightforward at this point to perform the required optimization and find the optimal setting of the Lagrange multipliers that maximize the concave $J(\lambda)$.

3.9.2 Empirical Bayes Priors

At this point, we have proven that it is feasible to estimate generative models in the exponential family form under MED if we assume the priors are given by the conjugate distribution. However, the parameters of the conjugate priors are still not specified and we still have quite some flexibility in designing prior knowledge into the MED formulation. In the absence of any prior knowledge, and whenever possible we recommend the default prior to be either a conjugate non-informative prior or an *Empirical Bayes prior*.

Loosely put, the prior for $P_0(\Theta)$, or more specifically for $P_0(\theta_+)$ and $P_0(\theta_-)$, we will use will be the posterior distribution of the parameters given the data that Bayesian inference generates. Consider the data set $\{X_1, \ldots, X_T\}$ with binary (± 1) labels $\{y_1, \ldots, y_T\}$. Thus, the inputs can be split into the positive inputs $\{X_{1+}, \ldots, X_{T+}\}$ and the negative inputs $\{X_{1-}, \ldots, X_{T-}\}$.

We now explicate the Bayesian inference procedure. To distinguish the resulting densities from those that will be used in the MED formulation, here we will put a $\hat{P}$ symbol on the Bayesian distributions. In Bayesian inference, each class's posterior distribution is estimated only from the positive input exemplars $\{X_{1+}, \ldots, X_{T+}\}$ as follows:

$$\begin{aligned}\hat{P}(\theta_+) &= \hat{P}(\theta_+ | \{X_{1+}, \ldots, X_{T+}\}) \\ &= \hat{P}(\{X_{1+}, \ldots, X_{T+}\} | \theta_+) \hat{P}(\theta_+) \\ &= \Pi_{t+} \hat{P}(X_{t+} | \theta_+) \hat{P}_0(\theta_+)\end{aligned}$$

Similarly, the negative class's generative Bayesian posterior model is estimated only from the negative input exemplars $\{X_{1-}, \ldots, X_{T-}\}$:

$$\hat{P}(\theta_-) = \Pi_{t-} \hat{P}(X_{t-} | \theta_-) \hat{P}_0(\theta_-)$$

For this Bayesian estimate of the generative model of the data, a minimally informative prior $\hat{P}_0(\theta_\pm)$ should be used. The result is a distribution that is as *good a generator* as possible for the data set.

However, we don't want just a good generator of the data, we also want a good discriminator. Thus, we can use MED to satisfy the large-margin classification constraints. But, simultaneously, the solution should be as close as possible to the generative model in terms of KL-divergence. Therefore, we shall use the Bayesian posteriors as the MED priors!

$$\begin{aligned}P_0(\theta_+) &:= \hat{P}(\theta_+) \\ P_0(\theta_-) &:= \hat{P}(\theta_-)\end{aligned}$$

Figure 3.6 depicts the information projection solution MED will generate from the Bayesian estimate. Effectively, we will try solving for the distribution over parameters that is as close as possible to the Bayesian estimate (which is often actually quite similar to the maximum likelihood estimate in the case of the exponential family) but that also satisfies the classification constraints.

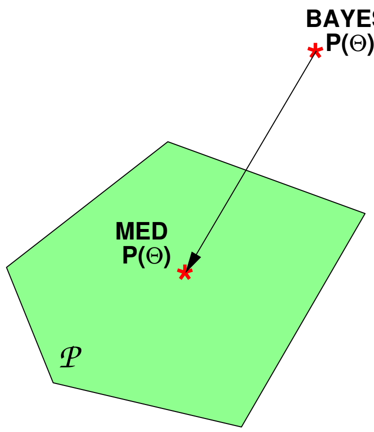


Figure 3.6: Information Projection Operation of the Bayesian Generative Estimate.

The motivation here is that in the absence of any further discriminative information, we should have as good a generator as possible. We now note a number of advantages for this type of empirical Bayes prior, that include theoretical, conceptual and practical arguments.. First, an empirical Bayes prior is a good precautionary measure to take because it allows more flexible use of MED's discriminative model as a generator whenever necessary. This may be the case when the discriminator has to cope with missing data or noise. If, therefore, we are in a prediction setting where some input variables are missing, we could reconstruct them (or integrate over them) by simply using the MED discriminative model as a surrogate for a generator distribution.

Under sparse data situations a model may easily satisfy some given discrimination constraints and many aspects of the model could remain ambiguous. In these cases, the empirical Bayesian prior provides a backup generative criterion, which further constrains the problem (albeit in ways not helpful to the task) and therefore can help consistent estimation. We also obtain an invariance in using an empirical Bayes prior that we would not get if we assume a fixed prior. For example, a fixed zero-mean Gaussian prior would produce different MED solutions if we translate the training data while an empirical Bayes prior would follow the translation of the data (with the Bayesian generative model) and consistently setup the same relative decision boundary.

Furthermore, consistency is important under the (arguably over-optimistic) situation that the generative model we are using is exactly correct and perfectly matches the training data. In that case, the Bayesian solution is optimal and MED may stray from it unless we have an empirical Bayes prior, even if we obtain infinite training data. An interesting side note is that if we use the standard margin prior distribution given by Equation 3.9, and obtain an upper bound on the Lagrange multipliers (i.e. they are $< c$), then as $c \rightarrow 0$, the MED solution uses the Bayesian posterior while as c increases, we reduce regularization (and outlier rejection) in favor of perfect classification.

Finally, on a purely practical note, an empirical Bayes prior may provide better numerical stability, for example. A discriminative MED model could put little probability mass on the training data and return a very poor generative configuration while still perfectly separating the data. This would be undesirable numerically since we would get very small values for, eg. $P(X|\theta_+)$ and $P(X|\theta_-)$. During prediction, a new test point may cause numerical accuracy problems if it is far from the probability mass in the MED discriminative solution. Therefore, whenever it does not result in a loss of discriminative power, one should maintain the generative aspects of the model.

3.9.3 Full Covariance Gaussians

We now consider the case where the discriminant function $\mathcal{L}(X;\Theta)$ corresponds to the log-likelihood ratio of two Gaussians with different (and adjustable) covariance matrices. The parameters Θ in this case are both the means and the covariances. These generative models are within the exponential family and so the previous results hold. Thus, the prior of choice $P_0(\Theta)$ must be the conjugate to the full-covariance Gaussian which is the Normal-Wishart. We shall use $\mathcal{N}$ as shorthand for the normal distribution and $\mathcal{IW}$ as shorthand for the inverse-Wishart. This choice of distributions permits us to obtain closed form integrals for the partition function $Z(\lambda)$. Here, we shall once again breakdown the parameters into the two generative models and the bias as before. Thus, we have $P(\Theta) = P(\theta_+)P(\theta_-)P(b)$. More specifically, the $\theta_\pm$ will also be broken down into the mean and covariance components of the Gaussian. Therefore, we have: $P(\Theta) = P(\mu_+,\Sigma_+)P(\mu_-,\Sigma_-)P(b)$ which gives us a density over means and covariances (this notation closely follows that of [136]).

The prior distribution has the form

$$P_0(\theta_+) \quad = \quad \mathcal{N}(\mu_+|m_+,\Sigma_+/k)\ \mathcal{IW}(\Sigma_+|kV_+,k)$$

Where the parameters that specify the prior, namely the scalar k, the vector m_+, and the matrix V_+ can be imputed manually. Also, one may let $k \rightarrow 0$ to get a non-informative prior.

We used the MAP values for k, m_0 and V_0 from the class-specific data which corresponds to the posterior distribution over the parameters given the data under a Bayesian inference procedure (i.e. an empirical Bayes procedure as described in the previous section). Integrating over the parameters, we get the partition function which factorizes $Z(\lambda) = Z_\gamma(\lambda)Z_+(\lambda)Z_-(\lambda)$. For $Z_+(\lambda)$ we obtain the following:

$$Z_+(\lambda) \quad \propto \quad N_+^{-d/2} \ |\pi S_+|^{-N_+/2} \ \Pi_{j=1}^d \Gamma\left(\frac{N_+ + 1 - j}{2}\right)$$

In the above we have defined the following intermediate variables (the scalar N_+, the vector X_+ and the matrix S_+) for brevity:

$$\begin{aligned} N_+ &\triangleq \sum_t w_t \\ \bar{X}_+ &\triangleq \sum_t \frac{w_t}{N_+} X_t \\ S_+ &\triangleq \sum_t w_t X_t X_t^T - N_+ \bar{X}_+ \bar{X}_+^T \end{aligned}$$

Here, w_t is a scalar weight given by $w_t = u(y_t) + y_t\lambda_t$ for $Z_+(\lambda)$. To solve for $Z_-(\lambda)$ we proceed in exactly the same manner as above however here, the weights are set to $w_t = u(-y_t) - y_t\lambda_t$. The $u(\cdot)$ is merely the step function. Given Z, updating λ is done by maximizing the corresponding negative entropy $J(\lambda)$ subject to $0 \leq \lambda_t \leq c$ and $\sum_t \lambda_t y_t = 0$ where:

$$J(\lambda) \quad = \quad \sum_t [l_\alpha \lambda_t + \log(1 - \lambda_t/c)] - \log Z_+(\lambda_t) - \log Z_-(\lambda_t)$$

The potential term above corresponds to integrating over the margin with a margin prior $P_0(\gamma) \propto e^{-c(l_\alpha - \gamma)}$ with $\gamma \leq s$. We pick l_α to be some α-percentile of the margins obtained under the standard MAP solution.

Optimal Lagrange multiplier values are then found via a simple constrained gradient descent procedure. The resulting MRE (normalized by the partition function $Z(\lambda)$) is a Normal-Wishart distribution itself for each generative model with the final λ values set by the maximization of $J(\lambda)$:

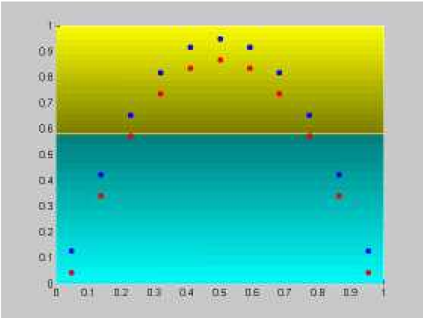
$$P(\theta_+) \quad = \quad \mathcal{N}(\mu_+; \bar{X}_+, \Sigma_+/N_+) \ \mathcal{IW}(\Sigma_+; S_+, N_+)$$

Predicting the labels for a data point X under the final $P(\Theta)$ involves taking expectations of the discriminant function under a Normal-Wishart. For the positive generative class, this expectation is:

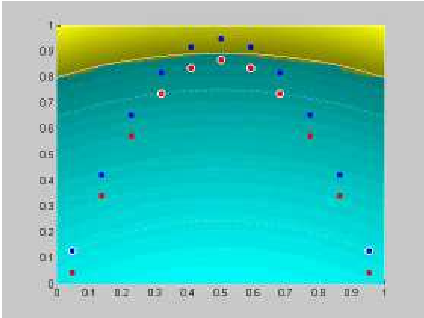
$$E_{P(\theta_+)}[\log P(X|\theta_+)] \quad = \quad \text{constant} - \frac{N_+}{2}(X - \bar{X}_+)^T S_+^{-1}(X - \bar{X}_+)$$

The expectation over the negative class is similar. This gives us the predicted label quite simply as:

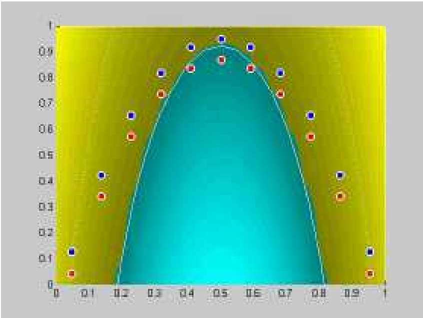
$$\begin{aligned} \hat{y} &= \text{sign} \int P(\Theta)\mathcal{L}(X;\Theta)d\Theta \\ \hat{y} &= \text{sign}\ E_{P(\Theta)}\left[\log \frac{P(X|\theta_+)}{P(X|\theta_-)} + b\right] \\ \hat{y} &= \text{sign}\left(\ E_{P(\theta_+)}[\log P(X|\theta_+)] - E_{P(\theta_-)}[\log P(X|\theta_-)] + E_{P(b)}[b]\right) \end{aligned}$$



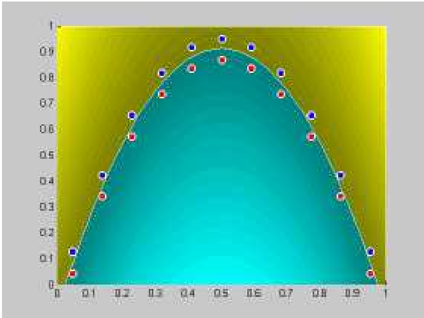
(a) ML & MED Initialization



(b) MED Intermediate



(c) MED Intermediate



(d) MED Converged

Figure 3.7: Classification visualization for Gaussian discrimination.

Computing the expectation over the bias is avoided under the non-informative case and the additive effect it has is merely estimated as in an SVM via the Karush-Kuhn-Tucker conditions. We thus obtain discriminative *quadratic* decision boundaries. These extend the linear boundaries without (explicitly) resorting to *kernels*. Of course, kernels may still be used in this formalism, effectively mapping the feature space into a higher dimensional representation. However, unlike linear discrimination, the covariance estimation in this framework allows the model to adaptively modify the kernel.

For visualization, we present the technique on a 2D set of training data in Figure 3.7. In Figure 3.7(a), the maximum likelihood technique is used to estimate a 2 Gaussian discrimination boundary (bias is estimated separately) which has the flexibility to achieve perfect classification yet produces a classifier whose performance is equal to random guessing. Meanwhile, the maximum entropy discrimination technique places the Gaussians in the most discriminative configuration as shown in Figure 3.7(b) without requiring kernels or feature space manipulations.

Experiments

In the following, we show results using the minimum relative entropy approach where the discriminant function $\mathcal{L}(X, \Theta)$ is the log-ratio of Gaussians with variable covariance matrices on standard 2-class classification problems (Leptograpsus Crabs and Breast Cancer Wisconsin). Performance is compared to regular support vector machines, maximum likelihood estimation and other methods.

The Leptograpsus crabs data set was originally provided by Ripley [162] and further tested by Barber and Williams [8]. The objective is to classify the sex of the crabs from 5 scalar anatomical

Method	Training Errors	Testing Errors
Neural Network (1)		3
Neural Network (2)		3
Linear Discriminant		8
Logistic Regression		4
MARS (degree = 1)		4
PP (4 ridge functions)		6
Gaussian Process (HMC)		3
Gaussian Process (MAP)		3
SVM - Linear	5	3
SVM - RBF $\sigma = 0.3$	1	18
SVM - 3rd Order Polynomial	3	6
Maximum Likelihood Gaussians	4	7
MaxEnt Discrimination Gaussians	2	3

Table 3.2: Leptograpsus Crabs

observations. The training set contains 80 examples (40 of each sex) and the test set includes 120 examples.

The Gaussian based decision boundaries are compared in Table 3.2 against other models from[8]. The table shows that the maximum entropy (or minimum relative entropy) criterion improves the Gaussian discrimination performance to levels similar to the best alternative models. The bias was estimated separately from training data for both the maximum likelihood Gaussian models and the maximum entropy discrimination case. In addition, we show the performance of a support vector machine (SVM) with linear, radial basis and polynomial decision boundaries (using the Matlab SVM Toolbox provided by Steve Gunn). In this case, the linear SVM is limited in flexibility while kernels exhibit some over-fitting.

Another data set which was tested was the Breast Cancer Wisconsin data where the two classes (malignant or benign) have to be computed from 9 numerical attributes from the patients' tumors (200 training cases and 169 test cases). The data was first presented by Wolberg [206]. We compare our results to those produced by Zhang [211] who used a nearest neighbor algorithm to achieve 93.7% accuracy. As can be seen from Table 3.3, over-fitting prevents good performance from the kernel based SVMs and the top performer here is the maximum entropy discriminator with an accuracy of 95.3%.

Method	Training Errors	Testing Errors
Nearest Neighbor		11
SVM - Linear	8	10
SVM - RBF $\sigma = 0.3$	0	11
SVM - 3rd Order Polynomial	1	13
Maximum Likelihood Gaussians	10	16
MaxEnt Discrimination Gaussians	3	8

Table 3.3: Breast Cancer Classification

3.9.4 Multinomials

Another popular exponential family model is the multinomial distribution. We next consider the case where the discriminant function $\mathcal{L}(X;\Theta)$ corresponds to the log-likelihood ratio of two multinomials:

$$\mathcal{L}(X;\Theta) \quad = \quad \log \frac{P(X|\theta_+)}{P(X|\theta_-)} - b$$

Where we have the generative models given by (if the X vector is consider as a set of counts):

$$P(X|\theta_+) \quad = \quad \left(\begin{array}{c} \sum_{k=1}^K X^k \\ X^1 \ldots X^K \end{array} \right) \Pi_{k=1}^K \rho_k^{X^k} \tag{3.13}$$

In the above, we are using the superscript on X^k to index into the dimensionality of the vector (the subscript will be used to index into the training set). The scalar term in the large parentheses is the multinomial coefficient (the natural extension of the binomial coefficient from coin tossing to die tossing). This scalar term is unity if the X vector is zero everywhere except for one unit entry. Otherwise it simply scales the probability distribution by a constant factor which can be rewritten as follows for more clarity (the use of gamma functions permits us to also consider continuous X vectors):

$$\left(\begin{array}{c} \sum_{k=1}^K X^k \\ X^1 \ldots X^K \end{array} \right) \quad = \quad \frac{\left(\sum_{k=1}^K X^k\right)!}{\Pi_{k=1}^K X^k!} \quad = \quad \frac{\Gamma(1+\sum_{k=1}^K X^k)}{\Pi_{k=1}^K \Gamma(1+X^k)}$$

The generative distribution in Equation 3.13 parameterizes the multinomial with the ρ vector of non-negative scalars that sum to unity, i.e. $\sum \rho = 1$. The parameters Θ in this case are both the ρ for the positive class and the negative class as well as the bias scalar b. These generative models are within the exponential family and so the previous results hold. Thus, the prior of choice $P_0(\Theta)$ must be the conjugate to the multinomial which is the Dirichlet distribution. This choice of distributions permits us to obtain closed form integrals for the partition function $Z(\lambda)$. Here, we shall once again breakdown the parameters into the two generative models and the bias as before. Thus, we have $P(\Theta) = P(\theta_+)P(\theta_-)P(b)$, to distinguish the ρ for the positive class, we will denote the parameters for the negative class as $\bar{\rho}$. The prior Dirichlet distribution has the form

$$P_0(\theta_+) \quad = \quad \frac{\Gamma(\sum_k \alpha_k)}{\Pi_k \Gamma(\alpha_k)} \Pi_k \rho_k^{\alpha_k - 1}$$

We typically assume that α_k will be pre-specified manually (or given by an empirical Bayes procedure) and will satisfy $\alpha_k > 1$. The core computation involves computing the component of the log-partition function that corresponds to the model (the computation for the bias and the margins remain the same as all the previous cases). Thus, we need:

$$Z_{\theta_+}(\lambda) Z_{\theta_-}(\lambda) \quad = \quad \int P_0(\theta_+) P_0(\theta_-) e^{\sum_t \lambda_t y_t [\log \frac{P(X_t|\theta_+)}{P(X_t|\theta_-)}]} d\theta_+ d\theta_-$$

It suffices to show how to compute Z_{θ_+}:

$$Z_{\theta_+} \quad = \quad \int P_0(\theta_+) e^{\sum_t \lambda_t y_t \log P(X_t|\theta_+)} d\theta_+$$

$$
\begin{aligned}
&= \int \frac{\Gamma(\sum_k \alpha_k)}{\Pi_k \Gamma(\alpha_k)} \Pi_k \rho_k^{\alpha_k - 1} \; e^{\sum_t \lambda_t y_t \log\left(\frac{\sum_k X_t^k}{X_t^k \ldots X_t^k} \right) + \sum_t \lambda_t y_t \log \Pi_k \rho_k^{X_t^k}} d\rho \\
&= \int \frac{\Gamma(\sum_k \alpha_k)}{\Pi_k \Gamma(\alpha_k)} \Pi_k \rho_k^{\alpha_k - 1} \; e^{\sum_t \lambda_t y_t \log \Pi_k \rho_k^{X_t^k}} d\rho \;\times\; e^{\sum_t \lambda_t y_t \log\left(\frac{\sum_k X_t^k}{X_t^1 \ldots X_t^K} \right)} \\
&= \int \frac{\Gamma(\sum_k \alpha_k)}{\Pi_k \Gamma(\alpha_k)} \Pi_k \rho_k^{\alpha_k - 1} \Pi_k \rho_k^{\sum_t \lambda_t y_t X_t^k} d\rho \;\times\; e^{\sum_t \lambda_t y_t \log\left(\frac{\sum_k X_t^k}{X_t^1 \ldots X_t^K} \right)} \\
&= \int \frac{\Gamma(\sum_k \alpha_k)}{\Pi_k \Gamma(\alpha_k)} \Pi_k \rho_k^{\alpha_k - 1 + \sum_t \lambda_t y_t X_t^k} d\rho \;\times\; e^{\sum_t \lambda_t y_t \log\left(\frac{\sum_k X_t^k}{X_t^1 \ldots X_t^K} \right)} \\
&= \frac{\Gamma(\sum_k \alpha_k)}{\Gamma(\sum_k \alpha_k + \sum_{tk} \lambda_t y_t X_t^k)} \Pi_k \frac{\Gamma(\alpha_k + \sum_t \lambda_t y_t X_t^k)}{\Gamma(\alpha_k)} \;\times\; e^{\sum_t \lambda_t y_t \log\left(\frac{\sum_k X_t^k}{X_t^1 \ldots X_t^K} \right)}
\end{aligned}
$$

We can thus form our objective function and maximize it to obtain the setting for the Lagrange multipliers λ (subject to the constraint $\sum_t \lambda_t y_t = 0$):

$$J(\lambda) = -\log Z_{\theta_+}(\lambda) - \log Z_{\theta_-}(\lambda) - \log Z_{\gamma}(\lambda)$$

The setting for the Lagrange multipliers permits us to exactly specify the final MED solution distribution $P(\Theta)$ which is used to compute the predictions for future classification:

$$\hat{y} \;=\; \mathrm{sign} \int P(\Theta) \mathcal{L}(X;\Theta) d\Theta$$

3.10 Generalization Guarantees

We now present several arguments for MED in terms of generalization guarantees. While generative frameworks have guarantees (asymptotic and otherwise) on the goodness of fit of a distribution (i.e. Bayesian evidence score, Bayesian Information Criterion, Akaike Information Criterion, etc.), they seldom have guarantees on the generalization performance of the models in a classification or regression setting. Furthermore, the guarantees may be distribution dependent which might be inappropriate if the true generative distribution of a data source is not perfectly known. Conversely, discriminative approaches that specifically target the classification or regression performance can have strong generalization arguments as we move from training data to testing data. These may also be distribution independent. The MED framework, in its discriminative estimation approach, brings classification performance guarantees to generative models. There are a number of arguments we will make, including sparsity-based generalization, references to VC-dimension based generalization and PAC-Bayesian generalization. Although generalization bounds can be quite loose for small amounts of training data, they are better than no guarantees whatsoever. Furthermore, the 'shape' of the generalization bounds have been demonstrated empirically to be useful in a discriminative setting. Finally, under large amounts of data, these bounds could be reasonably tight.

3.10.1 VC Dimension

Due to the ability of the MED framework to subsume SVMs (exactly generating the same equations in the separable case), it too benefits from the generalization guarantees that accompany them.

These are of course the VC-dimension (Vapnik-Chervonenkis) bounds on the expected risk, $\mathcal{R}(\Theta)$, of a classifier. Assuming we have a $[0,1]$ loss function $l(X_t, y_t, \Theta)$ and T training exemplars, the empirical risk can be readily computed [35] [196] [197]:

$$\mathcal{R}_{emp}(\Theta) = \frac{1}{T}\sum_{t=1}^{T} l(X_t, y_t, \Theta)$$

The true risk (for samples outside of the training set) is then bounded above by the empirical plus a term that depends only on the size of training set, T and the VC-dimension of the classifier, h. This non-negative integer quantity measures the capacity of a classifier and is independent of the distribution of the data. The following bound holds with probability $1-\delta$:

$$\mathcal{R}(\Theta) \leq \mathcal{R}_{emp}(\Theta) + \sqrt{\frac{h(\log(2T/h)+1) - log(\delta/4)}{T}}$$

The VC-dimension of a set of hyper-planes in $\Re^n$ is equal $n+1$. This does not directly motivate the use of large margin decision boundaries in an SVM. However, an SVM can be interpreted instead as a gap-tolerant classifier instead of a pure hyper-plane. A gap-tolerant classifier is a set of two parallel hyper-planes with a sphere. All points outside the sphere and all points between the planes do not contribute to the risk while points within the sphere and on either side of the planes are assigned a class (i.e. ± 1). Thus, if all points are within the sphere and outside of the two hyper-planes, the VC-dimension can be upper bounded by the radius of the resulting sphere R and the margin between the planes M. This gives the following upper bound on the VC-dimension, h, in an feature space of $\Re^n$:

$$h \leq \max\left\{\frac{D^2}{M^2}, n\right\} + 1$$

Thus, we have a 'plausible' argument for maximizing margin with a linear classifier. Although this does not translate immediately to nonlinear classifiers (if there is no direct kernel mapping back to linear hyper-planes), the motivation for large-margin in SVMs still can be used to justify using large margins in the MED formulation (namely with priors put large probability mass on larger margins values). We now move to other formal arguments for MED generalization.

3.10.2 Sparsity

The MED solution involves a constraint-based optimization where a classification constraint is present over each training data point to be classified. Each constraint is represented by the Lagrange multiplier associated with the given data point. In many cases, these constraints are likely to be redundant. This is apparent since classifying one data point correctly might automatically result in correct classification of several others. Therefore, the constraints involving some data points will be obviated by others and their corresponding Lagrange multipliers will go to zero. As in an SVM, points close to the margin (which have small margin values) have a critical role in shaping the decision boundary and generate non-zero Lagrange multipliers. These are the support-vectors in the standard SVM terminology. Meanwhile, other points that are easily correctly classified with a large margin will have zero Lagrange multipliers. Thus, the MED solution only depends on a small subset of the training data and will not change if the other data points were deleted. This gives rise to a notion of sparsity and with it we can make some generalization arguments. One argument is that the generalization error (denoted ϵ_g) is less than the expected percentage (ratio) of non-zero Lagrange multipliers over all Lagrange multipliers.

$$\epsilon_g \leq E\left[\frac{\sum_{t=1}^{T}\delta(\lambda_t > 0)}{T}\right]$$

Thus, for T data points, we simply count the number of non-zero Lagrange multipliers (using the δ function which is zero for Lagrange multipliers of value zero and unity for non-vanishing values). However, the expectation is taken over arbitrary choices of the training set which means that the upper bound on generalization error can only be approximated (using cross-validation or other techniques as in [196] [83]). Alternatively, a coarse and riskier approximation to the expectation can be done by simply counting the number of remaining non-zero Lagrange multipliers after maximizing $J(\lambda)$ on the training set in the MED solution.

3.10.3 PAC-Bayes Bounds

An alternative to VC dimension arguments for generalization includes PAC bounds (probably approximately correct, Valiant 1984). Recent contributions in terms of a PAC-Bayesian model selection criteria by McAllester [125] and Langford [116] have given theoretical generalization arguments that directly motivate the MED approach (MED was actually developed prior to the generalization results). Essentially PAC-Bayesian approaches allow the combination of a Bayesian integration of prior domain knowledge with PAC generalization guarantees without forcing the PAC framework to assume the truthfulness of the prior. We loosely adapt and state the main results of [116] here but further details are available from the original work as well as [125]. Effectively, the generalization guarantees are for model averaging where a stochastic model selection criterion is given in favor of a deterministic one. MED is a model averaging framework in that a *distribution* over models is computed (unlike, eg. an SVM). Therefore, these new generalization results apply almost immediately.

First, (as in MED) we assume a prior probability distribution $P_0(\Theta)$ over a possibly uncountable (continuous) model class. We also assume our discriminant functions $\mathcal{L}(X;\Theta)$ are *bounded* real-valued hypotheses [7]. Given a set T training exemplars of the form (X_t, y_t) sampled from a distribution D, we would like to compute the expected loss (i.e. the fraction of misclassifications) over the true distribution D. Recall in MED that a correct classification of the data is given by:

$$y_t \int P(\Theta)\mathcal{L}(X_t;\Theta)d\Theta \geq 0$$

Meanwhile, incorrect classifications are of the form:

$$y_t \int P(\Theta)\mathcal{L}(X_t;\Theta)d\Theta \leq 0$$

A more conservative empirical misclassification rate (i.e. which over counts the number of errors)

[7] The generalization guarantees were actually originally for averaging binary discriminant functions, not real ones, but can be extended to real ones in a straightforward manner. One may construct an MED classifier where the discriminant function is, eg. sigmoidal, or binary and then satisfy the requirements for these bounds to hold. Alternatively, a trivial extension is to find a bound by considering a maximal sphere around all the data which implicitly provides limits on the range of the discriminant function. This then permits a scaled version of the generalization bound.

can be made by also counting those errors below some positive margin threshold γ:

$$y_t \int P(\Theta)\mathcal{L}(X_t;\Theta)d\Theta \quad \leq \quad \gamma$$

If we compute the empirical number of misclassifications with this more conservative technique based on the margin threshold, γ, we can upper bound the expected (standard) misclassification rate. The expected misclassification rate has the following upper bound bound which holds with probability $1-\delta$:

$$\begin{aligned} E_D\left[y \int P(\Theta)\mathcal{L}(X;\Theta)d\Theta \leq 0\right] \quad &\leq \quad \frac{1}{T}\sum_t \left[y_t \int P(\Theta)\mathcal{L}(X_t;\Theta)d\Theta \leq \gamma\right] \\ &\quad +O\left(\sqrt{\frac{\gamma^{-2}D(P(\Theta)\|P_0(\Theta))\ln T + \ln T + \ln \delta^{-1}}{T}}\right) \end{aligned}$$

Ideally, we would like to minimize the expected risk of the classifier on future data (left hand side). Clearly, the bound above motivates forming a classifier that satisfies the empirical classification constraints (encapsulated by the first term on the right hand side), while minimizing divergence to the prior distribution (the second term on the right hand side). We also note that increasing the margin threshold is also useful at minimizing the expected risk. These criteria are directly addressed by the MED framework which strongly agrees with this theoretical motivation. Furthermore, increasing the cardinality of the training data set will make the bound tighter independently of the distribution of the data. Another point of contact is that [125] argues that the optimal posterior according to these types of bounds is, as in the Maximum Entropy Discrimination solution, the Gibbs distribution.

3.11 Summary and Extensions

We have presented the Maximum Entropy Discrimination framework from a regularization theory perspective and shown how it subsumes support vector machines. The solvability of the MED solution has been demonstrated. We have also shown how MED can readily be used in a generative framework where decision boundaries arise from exponential family distributions over the input space. Finally, generalization guarantees provide the framework with a theoretical grounding that reinforces its flexibility and generality. We next discuss the many extensions that can now be cascaded into the MED formalism which further motivate the usefulness of the approach.

Chapter 4

Extensions to Maximum Entropy Discrimination

Up to this point the MED formulation has already bridged generative modeling with the discriminative performance of SVMs, for example. However, the MED method can be further elaborated and spans a wide range of machine learning scenarios. In this chapter, we discuss various extensions to the framework to demonstrate its flexibility and intuitive nature. One resounding them in exploring extensions is to introduce further (possibly intermediate) variables in the discriminant function $\mathcal{L}(X;\Theta)$ and to solve for an augmented distribution $P(\Theta,\ldots)$ that includes them. The resulting partition function typically involves more integration yet if it is analytic and the number of Lagrange multipliers and the optimization complexity will remain basically unchanged. Figure 4.1 depicts the common metaphor of augmenting the probability with further variables which will be utilized. This follows the same principle used to augment the distribution with soft margin constraints as in Section 3.3. Once again, we note the caveat that as we add more distributions to the prior, we should be careful to balance their competing goals (i.e. their variances) evenly so that we still derive meaningful information from each component of the aggregate prior (i.e. the model prior, the margin prior, and the many further priors we will introduce shortly).

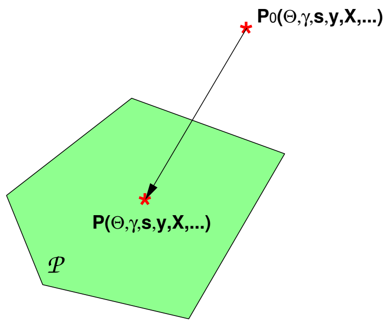
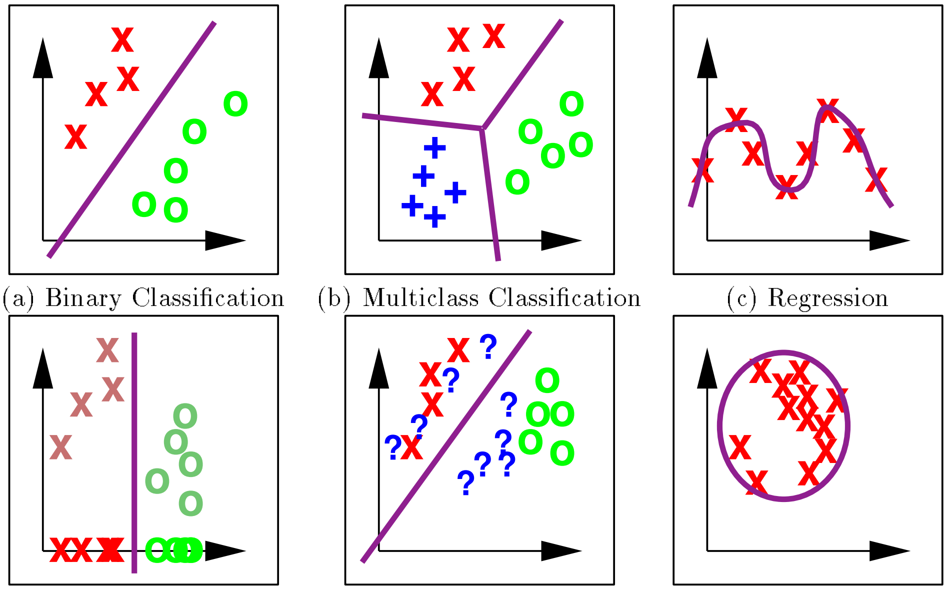


Figure 4.1: Formulating extensions to MED.

Figure 4.2 depicts the many different scenarios the MED can handle. Some extensions such as multi-class classification can be treated as several binary classification constraints [85] or through error-correcting codes [45]. In this chapter we explicate the case where the labels are no longer discrete but continuous, i.e. regression. Once again, for the case of regression (just as in binary classification) the SVM regression is subsumed. Subsequently, we discuss structure learning (as opposed to parameter estimation) in particular for feature selection applications. Furthermore, we discuss the use of partially labeled examples and transduction (for both classification and regression). Finally, we lead into a very important generalization which requires special treatment on its own: extension to mixture models (i.e. mixtures of the exponential family) and latent modeling (discussed Chapter 5).



(a) Binary Classification (b) Multiclass Classification (c) Regression

(d) Feature Selection (e) Transduction (f) Anomaly Detection

Figure 4.2: Various extensions to binary classification.

4.1 MED Regression

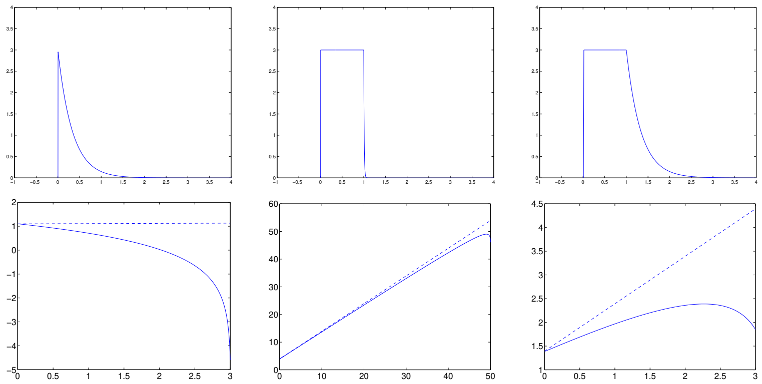
The MED formalism is not restricted to classification. Here, we present its extension to the regression (or function approximation) case using the approach and nomenclature in [178]. Dual sided constraints are imposed on the output such that an interval called an ϵ-tube around the function is described [1]. Suppose training input examples $\{X_1, \ldots, X_T\}$ are given with their corresponding output values as continuous scalars $\{y_1, \ldots, y_T\}$. We wish to solve for a distribution of parameters of a discriminative regression function as well as margin variables:

Theorem 4 *The maximum entropy discrimination regression problem can be cast as follows:*

Find $P(\Theta, \gamma)$ that minimizes $KL(P\|P_0)$ subject to the constraints:

$$\begin{aligned} \int P(\Theta,\gamma)\left[y_t - \mathcal{L}(X_t;\Theta) + \gamma_t\right] d\Theta d\gamma &\geq 0, \quad t = 1..T \\ \int P(\Theta,\gamma)\left[\gamma_t' - y_t + \mathcal{L}(X_t;\Theta)\right] d\Theta d\gamma &\geq 0, \quad t = 1..T \end{aligned}$$

[1] An ϵ-tube (as in the SVM literature) is a region of insensitivity in the loss function which only penalizes approximation errors which deviate by more than ϵ from the data.



(a) (b) (c)

Figure 4.3: Margin prior distributions (top) and associated potential functions (bottom).

where $\mathcal{L}(X_t;\Theta)$ is a discriminant function and P_0 is a prior distribution over models and margins. The decision rule is given by $\hat{y} = \int P(\Theta)\,\mathcal{L}(X;\Theta)d\Theta$. The solution is given by:

$$P(\Theta,\gamma) = \frac{1}{Z(\lambda)} P_0(\Theta,\gamma)\,\frac{e^{\sum_t \lambda_t [y_t - \mathcal{L}(X_t|\Theta) + \gamma_t]}}{e^{\sum_t \lambda'_t [y_t - \mathcal{L}(X_t|\Theta) - \gamma'_t]}}$$

where the objective function is again $J(\lambda) = -\log Z(\lambda)$.

Typically, we have the following prior for γ which differs from the classification case due to the additive role of the output y_t (versus multiplicative) and the two-sided constraints.

$$P_0(\gamma_t) \propto \left\{ \begin{array}{ccc} 1 & \text{if} & 0 \leq \gamma_t \leq \epsilon \\ e^{c(\epsilon-\gamma_t)} & \text{if} & \gamma_t > \epsilon \end{array} \right\} \tag{4.1}$$

Integrating, we obtain:

$$\begin{aligned} \log Z_{\gamma_t}(\lambda_t) &= \log\left(\int_0^\epsilon e^{\lambda_t\gamma_t} d\gamma_t + \int_\epsilon^\infty e^{c(\epsilon-\gamma_t)} e^{\lambda_t\gamma_t} d\gamma_t \right) \\ &= \log\left(\frac{e^{\lambda\epsilon}}{\lambda} - \frac{1}{\lambda} + \frac{e^{\lambda\epsilon}}{c-\lambda} \right) \\ &= \log\left(\left(\frac{e^{\lambda\epsilon}}{\lambda}\right) \left(1 - e^{-\lambda\epsilon} + \frac{\lambda}{c-\lambda}\right) \right) \\ &= \epsilon\lambda_t - \log(\lambda_t) + \log\left(1 - e^{-\lambda_t\epsilon} + \frac{\lambda_t}{c-\lambda_t}\right) \end{aligned}$$

Figure 4.3 shows the above prior and its associated penalty terms under different settings of c and ϵ. Varying ϵ effectively modifies the thickness of the ϵ-tube around the function. Furthermore, c varies the robustness to outliers by tolerating violations of the ϵ-tube.

The above margin prior tends to produce a regressor which is insensitive to errors smaller than ϵ and then penalizes errors by an almost linear loss thereafter (where c controls the steepness of the linear loss).

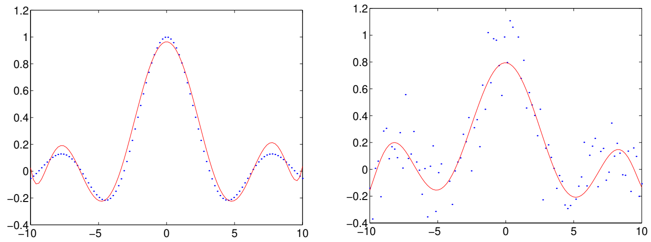


Figure 4.4: MED approximation to the sinc function: noise-free case (left) and with Gaussian noise (right).

4.1.1 SVM Regression

If we assume a linear discriminant function for $\mathcal{L}$ (or linear decision after a Kernel), the MED formulation generates the same objective function that arises in SVM regression [178]:

Theorem 5 *Assuming* $\mathcal{L}(X;\Theta) = \theta^T X + b$ *and* $P_0(\Theta,\gamma) = P_0(\theta)P_0(b)P_0(\gamma)$ *where* $P_0(\theta)$ *is* $N(0,I)$, $P_0(b)$ *approaches a non-informative prior, and* $P_0(\gamma)$ *is given by Equation 4.1 then the Lagrange multipliers* λ *are obtained by maximizing* $J(\lambda)$ *subject to* $0 \leq \lambda_t \leq c$, $0 \leq \lambda'_t \leq c$ *and* $\sum_t \lambda_t = \sum_t \lambda'_t$, *where*

$$\begin{aligned} J(\lambda) &= \sum_t y_t(\lambda'_t - \lambda_t) - \epsilon \sum_t (\lambda_t + \lambda'_t) + \sum_t \log(\lambda_t) - \log\left(1 - e^{-\lambda_t \epsilon} + \frac{\lambda_t}{c - \lambda_t}\right) \\ &\quad + \sum_t \log(\lambda'_t) - \log\left(1 - e^{-\lambda'_t \epsilon} + \frac{\lambda'_t}{c - \lambda'_t}\right) - \frac{1}{2}\sum_{t,t'} (\lambda_t - \lambda'_t)(\lambda_{t'} - \lambda'_{t'})(X_t^T X_{t'}) \end{aligned}$$

As can be seen (and more so as $c \to \infty$), the objective becomes very similar to the one in SVM regression. There are some additional penalty functions (all the logarithmic terms) which can be considered as barrier functions in the optimization to maintain the constraints.

To illustrate the regression, we approximate the sinc function, a popular example in the SVM literature. Here, we sampled 100 points from the $\mathrm{sinc}(x) = |x|^{-1} \sin |x|$ within the interval [-10,10]. We also considered a noisy version of the sinc function where Gaussian additive noise of standard deviation 0.2 was added to the output. Figure 4.4 shows the resulting function approximation which is very similar to the SVM case. The Kernel applied was an 8th order polynomial [2].

4.1.2 Generative Model Regression

As was the case for MED classification, we can also consider an MED regression scenario where the regression model is not linear (or linear after some kernel manipulation) but actually the regression model is given by a probability distribution. Thus, the regularization and epsilon-tube properties of the SVM approach can be readily applied to the estimation of generative models in a regression

[2] A Kernel implicitly transforms the input data by modifying the dot-product between data vectors $k(X_t, X'_t) = \langle \Phi(X_t), \Phi(X'_t) \rangle$. This can also be done by explicitly remapping the data via the transformation $\Phi(X_t)$ and using the conventional dot-product. This permits non-linear classification and regression using the basic linear SVM machinery. For example, an m-th order polynomial expansion replaces a vector X_t by $\Phi(X_t) = [X_t;\ X_t^2;\ \ldots X_t^m]$.

setting. Furthermore, these can be in the exponential family and also mixtures of the exponential family as will be elaborated in the subsequent chapter. We begin by modifying the discriminant function $\mathcal{L}(X;\Theta)$ from its usual linear form.

Consider a two class problem where we have a generative model for each class, namely $P(X|\theta_+)$ and $P(X|\theta_-)$. For example, these could each be a Gaussian distribution, or a mixture of Gaussians or even a complex structured model such as a hidden Markov model. To form a regressor, we directly combine two generative models by considering their log-likelihood ratios into a discriminant function as follows:

$$\mathcal{L}(X;\Theta) = \log \frac{P(X|\theta_+)}{P(X|\theta_-)} + b$$

Here, the aggregate parameter set is $\Theta = \{\theta_+, \theta_-, b\}$ which includes both generative models as well as a scalar bias. Thus, by merely changing the discriminant function, the MED framework can be used to estimate generative models that form a regression function. If the two generators are Gaussians with equal covariance, the regression function will effectively produce a linear regression. However, the above discriminant function is generally nonlinear and will give rise to a non-convex hull of constraints in a standard regularization setting. However, in the MED framework, due to the probabilistic solution $P(\Theta)$, the above discriminant functions still behave as a convex program. Furthermore, it is possible (through the iterative bounds and machinery in Chapter 5) to deal with latent discriminant regression functions of the form:

$$\mathcal{L}(X;\Theta) = \log \frac{\sum_m P(m, X|\theta_+)}{\sum_m P(m, X|\theta_-)} + b$$

4.2 Feature Selection and Structure Learning

The MED framework is not limited to estimating distributions over continuous parameters such as Θ. We can also use it to solve for a distribution over discrete parameters and thus use it for structure learning. One form of structure learning is feature selection. The feature selection problem can be cast as finding the structure of a graphical model (as in [43]) or identifying a set of components of the input examples that are relevant for a classification task. More generally, feature selection can be viewed as a problem of setting discrete structural parameters associated with a specific classification or regression method. We will use feature selection in the MED framework to ignore components of the input space (i.e. the X_t vectors) that are not relevant to the given classification or regression task. This will naturally provide computational advantages since the algorithm can ignore these inputs during run-time. However, not only does feature selection reduce the input dimensionality, we will also show that it helps improve generalization accuracy in both classification and regression (cf. [111]). The omission of certain input dimensions permits better generalization and leads to a further notion of sparsity in the input dimensionality (in addition to the sparsity from the support vectors and Lagrange multipliers as discussed in Section 3.10.2) [90] [202]. This is often critical if the input space has high dimensionality, many irrelevant features and the data set is small.

We will initially derive the feature selection in the MED framework as a feature weighting to permit a probabilistic solution. Each feature or structural parameter is given a probability value. The feature selection process then estimates the most discriminative probability distribution over the structural parameters while it also estimates the most discriminative parameter model. Irrelevant features will eventually receive extremely low probabilities of being selected. Since the feature selection process is performed jointly and discriminatively together with model estimation and both specifically optimize a classification or regression criterion, feature selection will usually improve results over, for example, an SVM (up to a point where we start removing too many features).

4.2.1 Feature Selection in Classification

The MED formulation can be extended to feature selection if we consider augmenting the distribution over models (and margins, bias, etc.) also with a distribution over feature selection switches. This 'augmentation' paradigm was initially discussed in Section 3.3 and under many conditions will preserve the solvability of the MED projection. We will now consider augmenting a linear classifier (such as an SVM) with feature selection. We first introduce extra parameters into our linear discriminant function:

$$\mathcal{L}(X;\Theta) = \sum_{i=1}^{n} \theta_i s_i X_i + \theta_0$$

Here, the familiar $\theta_1, \ldots, \theta_n$ correspond to the linear parameter vector while the θ_0 is the bias parameter (usually denoted b). In addition to the scalar θ_i parameters, we have also introduced binary switches $s_1, \ldots, s_n$ which can only be 0 or 1. These are structural parameters and will either completely turn off a feature X_i if $s_i = 0$ or leave it on if $s_i = 1$. If we were to solve for the optimal feature selection in a brute-force method, we would have to try all 2^n configurations of the discrete switch variables. However, in the MED formulation, we can instead consider a *distribution* over switches which will lead to tractable computation. The fact that now the switches are discrete (instead of continuous like the θ_i parameters) does note violate the MED formulation [85]. The partition function and the expectations over discriminant functions now also involve summations over the s_i as well as integration over the continuous parameters. Therefore, now the MED solution distribution $P(\Theta)$ includes the linear model, the switches and the bias, i.e. $\Theta = \{\theta_0, \theta_1, \ldots, \theta_n, s_1, \ldots, s_n\}$.

We will now define a prior over the desired MED solution and then discuss how to solve for the optimal projection. The prior will reflect some regularization on the linear SVM parameters as well as the degree of feature selection we would like to enforce overall. In other words, we would like to specify (in coarse terms) how many feature switches will be set to zero or remain active. One possible prior for the solution is:

$$P_0(\Theta) = P_{0,\theta_0}(\theta_0)\, P_{0,\theta}(\theta) \prod_{i=1}^{n} P_{s,0}(s_i)$$

where P_{0,θ_0} is an uninformative prior (a a zero mean Gaussian prior with infinite variance [3], $P_{\theta,0}(\theta) = \mathcal{N}(0, I)$ the usual white Gaussian prior, and

$$P_{s,0}(s_i) = \rho^{s_i}\,(1-\rho)^{1-s_i}$$

where ρ controls the overall prior probability of including a feature. Thus the prior over each feature is merely a Bernoulli distribution. The user selects ρ where a setting of $\rho = 1$ will produce the original linear classifier problem without feature selection. By decreasing ρ, more features will be removed. Given a prior distribution over the parameters in the MED formalism and a discriminant function, we can now readily compute the partition function (cf. Equation 3.8). Solving the integrals and summations, we obtain the following objective function:

$$J(\lambda) \;=\; \sum_t [\lambda_t + \log(1 - \lambda_t/c)] - \sum_{i=1}^{n} \log\left[1 - \rho + \rho e^{\frac{1}{2}(\sum_t \lambda_t y_t X_{t,i})^2}\right]$$

[3] Alternatively, we can use a finite-variance Gaussian which will give a quadratic penalty term on the final objective function of $-0.5\sigma(\sum_t \lambda_t y_t)^2$ instead of the hard equality constraint.

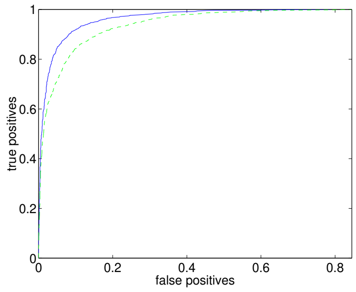


Figure 4.5: ROC curves on the splice site problem with feature selection $\rho = 0.00001$ (solid line) and without $\rho = 0.99999$ (dashed line).

which we maximize subject to $\sum_t \lambda_t y_t = 0$.

The above is maximized to obtain the optimal setting of our Lagrange multipliers. Given that setting, our linear classifier merely becomes:

$$\mathcal{L}(X) = \sum_i \left(\frac{\rho \sum_t \lambda_t y_t X_{t,i}}{\rho + (1-\rho)\exp(-1/2[\sum_t \lambda_t y_t X_{t,i}]^2)} \right) X^i + b$$

In the above, the $X_{t,i}$ indicates the i'th dimension of the t'th training set vector. The bias b is estimated separately either from the Kuhn-Tucker conditions or (under a finite variance bias prior), is set to $b = \sigma \sum_t \lambda_t y_t$. The terms in the large parentheses in the equation above are the linear coefficients of the new model and can be denoted C_i.

Experiments

To test the linear feature selection method, we used a DNA splice site classification problem which must identify between true and spurious splice sites based on a DNA sequence. The examples were fixed length DNA sequences (length 25) which were binary encoded (using a 4 bit translation of $\{A, C, T, G\}$) into a 100-element vector. The training set consisted of 500 examples and the test set contained 4724 examples. Results are depicted in Figure 4.5 which shows superior classification accuracy when feature selection is used (as opposed to no feature selection which is roughly equivalent to an SVM).

The feature selection process drives many of the linear model's coefficients to zero in an aggressive pruning manner. This provides better generalization as well as more efficiency during run-time. To picture the sparsity in the resulting model, we plot the cumulative distribution function of the magnitudes of the resulting coefficients $|C_i| < x$ as a function of x for all the 100 components of the linear classification vector. Figure 4.6 indicates that most of the weights resulting from the feature selection algorithm are indeed small enough to be neglected.

While our derivation of the above feature selection was so far only performed for linear models, we can mimic a kernel-based nonlinear classifier by mapping the feature vectors explicitly into a higher order representation (i.e. through polynomial expansions). This does not retain the efficiency of implicit kernel mappings (and infinite kernel mappings are infeasible) however we have the ability to do fine-scale feature selection as *components* of the kernel mapping can also be extinguished. The complexity of the feature selection algorithm is linear in the number of features and therefore we can easily consider small expansions (such as quadratic or cubic polynomials) by explicit mapping. The above problem was attempted with a quadratic expansion of the 100-dimensional feature vectors

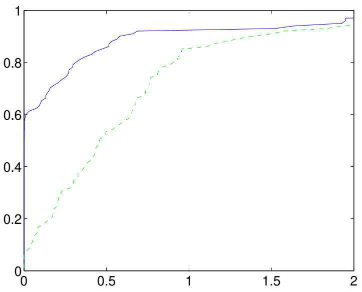


Figure 4.6: Cumulative distribution functions for the resulting effective linear coefficients with feature selection (solid line) and without (dashed line).

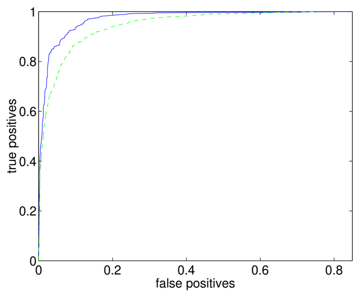


Figure 4.7: ROC curves corresponding to a quadratic expansion of the features with feature selection $\rho = 0.00001$ (solid line) and without $\rho = 0.99999$ (dashed line).

by concatenating the outer products of the original features to form a resulting ≈ 5000-dimensional feature space. Figure 4.6 shows that feature selection is still helpful (compared to a plain linear classifier), improving performance even when we have a larger and expanded feature space.

In another experiment, we used the feature selection classification to label protein chains (from the UCI repository) which were valid splice sites into one of two classes: intron-exon (ie) or exon-intron (ei). These are often also called donor and acceptor sites respectively. The chains consist of 60 base-pairs which were then represented in binary code: A=(1000), C=(0100), G=(0010) and T=(0001). Uncertain base-pairs where represented as a mixed version, i.e. A or C would be represented as (0.5 0.5 0 0). Thus, we have 240 scalar input dimensions and a binary output class. We trained on 200 exemplars and test on the remaining 1335 exemplars in the testing set. Figure 4.8 depicts the performance.

In training, linear classifiers can easily separate both classes at 100% accuracy however using all the features causes over-fitting. The regularization brought upon by varying c does not prune away features but rather ignores outliers. However, not the whole length of the protein chain is useful in determining acceptor/donor status and we should ignore dimensions instead of data exemplars. Thus, the best performance possible in a regular SVM (no feature selection, i.e. $\rho = 1$) remains around 92% on testing as we vary regularization c. Meanwhile, any small amount of feature selection quickly improves performance much more significantly. The experiments indicate that a level of $\rho = 1e - 2$ or $\rho = 1e - 3$ helps obtain the greatest generalization accuracy of about 96%. Error is halved from the SVM's count of 100+ errors to an error count of 50 with feature selection. Figure 4.9 depicts the linear model for the SVM as well as the pruned model for the feature selection technique.

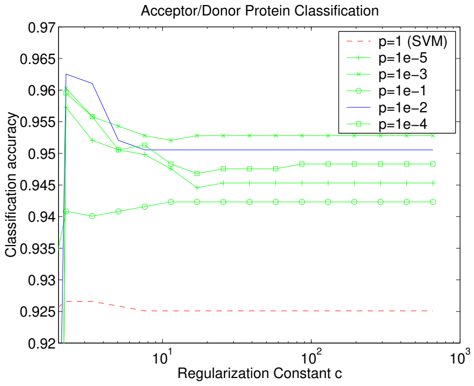


Figure 4.8: Varying Regularization and Feature Selection Levels for Acceptor/Donor Protein Classification. Testing performance on unseen 1335 protein sequences. The dashed line indicates SVM performance while the solid lines indicate varying performance improvements due to feature selection. Optimal feature selection levels for this problem appear to be between $\rho = 1e - 2$ and $\rho = 1e - 3$.

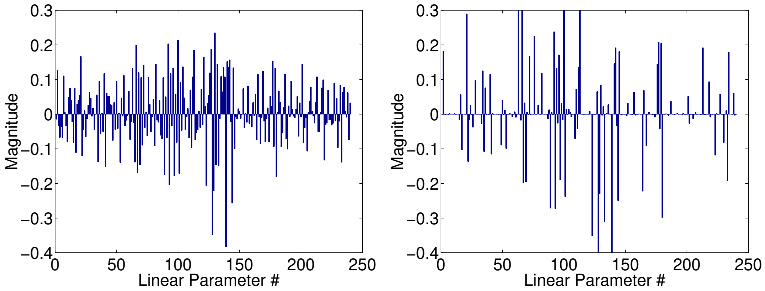


Figure 4.9: Sparsification of the Linear Model. On the left are the parameters for the SVM's linear model while on the right are the parameters for the feature selection technique's linear model. Note the sparsification in the parameters as many are set to 0 on the right. This pruning encourages better generalization.

4.2.2 Feature Selection in Regression

Feature selection can also be advantageous in the regression case where a map is learned from inputs to scalar outputs. Since some input features might be irrelevant (especially after a Kernel expansion), we again employ an aggressive pruning approach by adding a "switch" (s_i) on the parameters as before. The prior is given by $P_0(s_i) = \rho^{s_i}(1-\rho)^{1-s_i}$ where lower values of ρ encourage further sparsification. This prior is in addition to the Gaussian prior on the parameters (Θ_i) which does not have quite the same sparsification properties.

The previous derivation for feature selection can also be applied in a regression context. The same priors are used except that the prior over margins is swapped with the one in Equation 4.1. Also, we shall include the estimation of the bias in this case, where we have a Gaussian prior: $P_0(b) = \mathcal{N}(0,\sigma)$. This replaces the hard constraint that $\sum_t \lambda_t = \sum_t \lambda'_t$ with a soft quadratic penalty, making computations simpler. After some straightforward algebraic manipulations, we obtain the following form for the objective function:

$$\begin{aligned} J(\lambda) \;=\; & \sum_t y_t(\lambda'_t - \lambda_t) - \epsilon \sum_t (\lambda_t + \lambda'_t) - \frac{1}{2}\sigma(\sum_t \lambda_t - \lambda'_t)^2 + \sum_t \log(\lambda_t) - \log\left(1 - e^{-\lambda_t \epsilon} + \frac{\lambda_t}{c - \lambda_t}\right) \\ & + \sum_t \log(\lambda'_t) - \log\left(1 - e^{-\lambda'_t \epsilon} + \frac{\lambda'_t}{c - \lambda'_t}\right) - \sum_i \log\left(1 - \rho + \rho e^{\frac{1}{2}[\sum_t (\lambda_t - \lambda'_t) X_{t,i}]^2}\right) \end{aligned}$$

This objective function is optimized over (λ_t, λ'_t) and by concavity has a unique maximum. The optimization over Lagrange multipliers controls optimization of the densities of the model parameter settings $P(\Theta)$ as well as the switch settings $P(s)$. Thus, there is a *joint* discriminative optimization over feature selection and parameter settings. At the optimal setting of the Lagrange multipliers, our resulting MED regression function is then:

$$\mathcal{L}(X_{new}) \;=\; \sum_i \left(\frac{\rho \sum_t (\lambda'_t - \lambda_t) X_{t,i}}{\rho + (1-\rho)\exp\left[-1/2(\sum_t (\lambda'_t - \lambda_t) X_{t,i}]^2)\right.} \right) X_{new,i} + b$$

Where the bias b is given by $b = \sigma \sum_t (\lambda'_t - \lambda_t)$.

Experiments

Below, we evaluate the feature selection based regression (or Support Feature Machine) on a popular benchmark dataset, the 'Boston housing' problem from the UCI repository. A total of 13 features (all treated continuously) are given to predict a scalar output (the median value of owner-occupied homes in thousands of dollars). To evaluate the dataset, we utilized both a linear regression and a 2nd order polynomial regression by applying a Kernel expansion to the input. The dataset is split into 481 training samples and 25 testing samples (as in [188]).

Table 4.1 indicates that feature selection (decreasing ρ) generally improves the discriminative power of the regression. Here, the ϵ-sensitive linear loss functions (typical in the SVM literature) shows improvements with further feature selection. Just as sparseness in the number of vectors helps generalization, sparseness in the number of features is advantageous as well. Here, there is a total of 104 input features after the 2nd order polynomial Kernel expansion. However, not all have the same discriminative power and pruning is beneficial.

For the 3 trial settings of the sparsification level prior ($\rho = 0.99999, \rho = 0.001, \rho = 0.00001$), we again analyze the cumulative density function of the resulting linear coefficients $C_i < x$ as a function

Linear Model Estimator	ϵ-sensitive linear loss
Least-Squares Fit	1.7584
MED $\rho = 0.99999$	1.7529
MED $\rho = 0.1$	1.6894
MED $\rho = 0.001$	1.5377
MED $\rho = 0.00001$	1.4808

Table 4.1: Prediction Test Results on Boston Housing Data. Note, due to data rescaling, only the relative quantities here are meaningful.

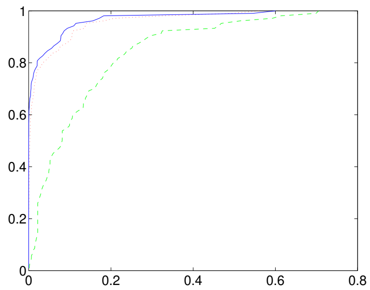


Figure 4.10: Cumulative distribution functions for the linear regression coefficients under various levels of sparsification. Dashed line: $\rho = 0.99999$, dotted line: $\rho = 0.001$ and solid line: $\rho = 0.00001$.

of x based on the features from an explicit Kernel expansion. Figure 4.10 clearly indicates that the magnitudes of the coefficients are reduced as the sparsification prior is increased.

The MED regression was also used to predict gene expression levels using data from "Systematic variation in gene expression in human cancer cell lines", by D. Ross et. al. Here, log-ratios ($\log(RAT2n)$) of gene expression levels were to be predicted for a Renal Cancer cell-line from measurements of each gene's expression levels across different cell-lines and cancer-types. Input data forms a 67-dimensional vector while output is a 1-dimensional scalar gene expression level. Training set size was limited to 50 examples and testing was over 3951 examples. The table below summarizes the results. Here, an $\epsilon = 0.2$ was used along with $c = 10$ for the MED approach. This indicates that the feature selection is particularly helpful in sparse training situations.

4.2.3 Feature Selection in Generative Models

As mentioned earlier, the MED framework is not restricted to discriminant functions that are linear or non-probabilistic. For instance, we can consider the use of feature selection in a generative model-

Linear Model Estimator	ϵ-sensitive linear loss
Least-Squares Fit	3.609e+03
MED $\rho = 0.00001$	1.6734e+03

Table 4.2: Prediction Test Results on Gene Expression Level Data.

based classifier. One simple case is the discriminant formed from the ratio of two identity-covariance Gaussians. Parameters Θ are (μ, ν) for the means of the $y = +1$ and $y = -1$ classes respectively and the discriminant is $\mathcal{L}(X;\Theta) = \log \mathcal{N}(\mu, I) - \log \mathcal{N}(\nu, I) + b$. As before, we insert switches (s_i and r_i) to turn off certain components of each of the Gaussians giving us:

$$\mathcal{L}(X;\Theta) = \sum_i s_i (X_i - \mu_i)^2 - \sum_i r_i (X_i - \nu_i)^2 + b$$

This discriminant then uses the similar priors to the ones previously introduced for feature selection in a linear classifier. It is straightforward to integrate (and *sum* over discrete s_i and r_i) with these priors (shown below and in Equation 3.9) to get an analytic concave objective function $J(\lambda)$:

$$\begin{array}{ll} P_0(\mu) = \mathcal{N}(0, I) & P_0(\nu) = \mathcal{N}(0, I) \\ P_0(s_i) = \rho^{s_i}(1-\rho)^{1-s_i} & P_0(r_i) = \rho^{r_i}(1-\rho)^{1-r_i} \end{array}$$

In short, optimizing the feature selection and means for these generative models jointly will produce degenerate Gaussians which are of smaller dimensionality than the original feature space. Such a feature selection process could be applied to many density models in principle but computations may require mean-field or other approximations to become tractable.

4.3 Transduction

In this section, we provide a Maximum Entropy Discrimination framework for solving the missing labels or transduction problem [197] [100]. In many classification problems, labeled data is scarce yet unlabeled data may be easily available in large quantities. The MED framework can be easily extended to utilize the unlabeled data in forming a discriminative classifier by integrating over the unobserved labels. In previous work, we initially presented the MED approach for transductive classification using primarily mean-field approximations [85]. Szummer [182] also presents an alternative transduction approach in terms of Kernel expansions which may also be cast in an MED formalism.

In the classification setting, the exact solution of the resulting MED projection becomes intractable (just as in SVM based transduction). We first review our mean-field approximation case as a possible local solution [85]. A global information-projection solution is also possible if the prior over unobserved labels is described by a distribution that is conjugate (and continuous) to the original distribution over models. We thus also provide a transduction algorithm which computes a global large margin solution over both labeled and unlabeled data by forcing the prior to be conjugate. We subsequently discuss the use of unlabeled data in the regression scenario which does yield a tractable global MED solution.

4.3.1 Transductive Classification

SVM transduction requires a search over binary labels of the unlabeled exemplars. The complexity of this approach grows exponentially. Joachims proposes using efficient heuristics which approximate this process yet are not guaranteed to converge to the true SVM transduction solution. Unlike SVMs, the MED approach permits a probabilistic treatment of the search over labels which is somewhat similar in spirit to relaxation methods. The discrete search problem is embedded in a continuous probabilistic setting.

First, recall that MED solves for distributions over parameters as well as other unknown quantities by augmenting the solution space. For example, when margins are unknown in a non-separable problem, we introduced them into the solution as posteriors $P(\Theta,\gamma)$ (and in the prior as well). When feature selection structure was unknown, it too was cascaded into the final MED posterior solution as $P(\Theta,\gamma,s)$. In the transduction case, unlabeled examples are given where y_t is unknown. Thus, we can hypothesize a prior/posterior distribution as well. This distribution would ideally take on the form of two delta functions at -1 and $+1$. Thus, instead of solving for only a distribution over say $P(\Theta,\gamma)$ we generalize to the non-separable transductive case via $P(\Theta,\gamma,y)$ where now projection is over a larger space.

We have the following general solution:

$$P(\Theta,\gamma,y) = \frac{1}{Z(\lambda)} P_0(\Theta,\gamma,y)\, e^{\sum_t \lambda_t [y_t \mathcal{L}(X_t|\Theta) - \gamma_t]}$$

where $Z(\lambda)$ is the normalization constant (partition function) and $\lambda = \{\lambda_1,\ldots,\lambda_T\}$ is again our set of non-negative Lagrange multipliers, one per classification constraint. λ are set by finding the unique maximum of the objective function $J(\lambda) = -\log Z(\lambda)$.

A distribution for y is required such that computation of the partition function $Z(\lambda)$ remains analytic. If we assume that the prior for (continuous) unlabeled y is given by the natural choice of a point-wise delta function, i.e. $P_0(y_t) = 1/2\delta(y_t,1) + 1/2\delta(y_t,-1)$, the integrals above become intractable. To proceed, a *mean-field* approximation is performed which effectively computes the integral over $P(\Theta)$ with the unlabeled $P(y)$ locked at a current estimate and then computes an update on the $P(y)$ while the $P(\Theta)$ is held fixed. This is equivalent to assuming that the distribution, $P(\Theta,\gamma,y)$ is forced to factorize according to $P(\Theta,\gamma)P(y)$. Details are provided in [85] and produce good generalization results when unlabeled data is useful for a classification problem. However, the mean-field approximation forces us to obtain a local solution which is no longer unique. If our two-stage iterative algorithm is poorly initialized, this may be a problem.

If, however, we could guarantee an analytic computation of $Z(\lambda)$ for the non-transductive case, this will yield a log-convex $Z(\lambda)$. Thus, one can consider $Z(\lambda)$ as an exponential family distribution. This is because it is a concave function of λ for any setting of the y-variables. The y-variables can then be considered as the data for the distribution while the λ are the parameters of the exponential family. Therefore, it is always possible to find a conjugate distribution (in y variables) such that the integral $\int_y P_0(y)Z(\lambda,y)$ is analytic. For instance, if $Z(\lambda)$ is Gaussian (or equivalently $J(\lambda)$ is quadratic, as in an SVM) we could have a conjugate distribution $P_0(y)$ which is Gaussian and end up with a final $Z(\lambda)$ which is still concave and analytic.

Assume that the data set is partitioned into two sets, the labeled and unlabeled data. There are T_l labeled components and T_u unlabeled components. Thus, we can consider our y vector of labels and our λ vector of Lagrange multipliers as being split as follows:

$$y = \begin{bmatrix} \bar{y} \\ \tilde{y} \end{bmatrix} \qquad \lambda = \begin{bmatrix} \bar{\lambda} \\ \tilde{\lambda} \end{bmatrix}$$

The $\bar{y}$ labels are known however we do not know the $\tilde{y}$ labels and only have a prior distribution over them. This prior is a scaled zero-mean spherical Gaussian, $P_0(\tilde{y}) = N(0,\kappa^{-2}I)$. This can also be interpreted as a prior over each individual unlabeled data point as $P_0(\tilde{y}) = \Pi_t P_0(\tilde{y}_t) = \Pi_t N(0,\kappa^{-2})$. We can also consider the y and λ vectors in a diagonal matrix form, as $Y = diag(y)$ and $\Lambda = diag(\lambda)$ respectively.

$$Y = \begin{bmatrix} \bar{Y} & \mathbf{0} \\ \mathbf{0} & \tilde{Y} \end{bmatrix} \qquad \Lambda = \begin{bmatrix} \bar{\Lambda} & \mathbf{0} \\ \mathbf{0} & \tilde{\Lambda} \end{bmatrix}$$

Similarly, we can consider the data matrix $\mathbf{X}$ as being a matrix of the X_t data vectors arranged as columns. It can be further divided into labeled and unlabeled vectors as follows:

$$\mathbf{X} \;=\; \left[\bar{\mathbf{X}} \;\; \tilde{\mathbf{X}}\right]$$

For simplicity, we will derive the transduction assuming a linear classifier yet drop the bias term (i.e. $\Theta = \theta$). This will limit the linear decision boundaries that can be generated to those that intersect the origin. This restriction can be circumvented by concatenating a constant scalar to our input features X. However, the formulation we will show here readily admits kernels and can also be augmented with the bias term with a little extra derivation. Thus, our classifier's discriminant function is:

$$\mathcal{L}(X;\Theta) \;=\; \theta^T X$$

Let us derive the corresponding partition function (up to a constant scalar factor):

$$\begin{aligned}
Z(\lambda) &= \int_{\tilde{y}}\int_{\theta}\int_{\gamma} P_0(\theta,\gamma,\tilde{y})\, e^{\sum_t \lambda_t[y_t\mathcal{L}(X_t|\theta)-\gamma_t]} \\
Z(\lambda) &= \int_{\tilde{y}}\int_{\theta} P_0(\theta)P_0(\tilde{y})e^{\sum_t \lambda_t y_t \theta^T X_t} \times \int_{\gamma} P_0(\gamma)e^{-\sum_t \lambda_t\gamma_t} \\
Z(\lambda) &= Z_\theta(\lambda)\times Z_\gamma(\lambda)
\end{aligned}$$

We may also consider dealing directly with our objective function $J(\lambda) = -\log Z(\lambda)$ in which case we have the following decomposition of our objective function:

$$J(\lambda) \;=\; J_\gamma(\lambda) + J_\theta(\lambda)$$

Solving, we ultimately obtain the following standard $J_\gamma(\lambda)$:

$$J_\gamma(\lambda) \;=\; \sum_{\forall t}\lambda_t + \sum_{\forall t}\log(1-\lambda_t/c)$$

The remaining component of the partition function is $J_\theta(\lambda)$ is given by:

$$\frac{1}{2}\log\left|\kappa^2 I - (\tilde{\Lambda}\tilde{\mathbf{X}})^T(\tilde{\Lambda}\tilde{\mathbf{X}})\right| - \frac{1}{2}\bar{\lambda}^T\left[(\bar{\mathbf{X}}\bar{Y})^T(\bar{\mathbf{X}}\bar{Y}) + (\bar{\mathbf{X}}\bar{Y})^T(\tilde{\mathbf{X}}\tilde{\Lambda})\left(\kappa^2 I - (\tilde{\mathbf{X}}\tilde{\Lambda})^T(\tilde{\mathbf{X}}\tilde{\Lambda})\right)^{-1}(\tilde{\mathbf{X}}\tilde{\Lambda})^T(\bar{\mathbf{X}}\bar{Y})\right]\bar{\lambda}$$

This provides our overall solution for the objective function. This is an analytic concave function with a single maximum that can be uniquely determined to obtain the answer $P(\Theta,\gamma,y)$.

Theorem 6 *Assuming a discriminant function of the form $\mathcal{L}(X;\Theta) = \theta^T X$ and given a prior over parameters and margins $P_0(\Theta,\gamma) = P_0(\theta)P_0(\gamma)$ where $P_0(\theta) \sim N(0,\kappa^{-2}I)$, $P_0(\tilde{y}) \sim N(0,I)$ and $P_0(\gamma)$ is given by $P_0(\gamma_t)$ as in Equation 3.9 then the MED Lagrange multipliers λ are obtained by maximizing $J(\lambda)$ subject to $0 \le \lambda_t \le c$:*

$$\begin{aligned}
J(\lambda) \;=\; & \sum_{\forall t}\lambda_t + \log(1-\lambda_t/c) + \frac{1}{2}\log\left|\kappa^2 I - (\tilde{\Lambda}\tilde{\mathbf{X}})^T(\tilde{\Lambda}\tilde{\mathbf{X}})\right| \\
& -\frac{1}{2}\bar{\lambda}^T\left[(\bar{\mathbf{X}}\bar{Y})^T(\bar{\mathbf{X}}\bar{Y}) + (\bar{\mathbf{X}}\bar{Y})^T(\tilde{\mathbf{X}}\tilde{\Lambda})\left(\kappa^2 I - (\tilde{\mathbf{X}}\tilde{\Lambda})^T(\tilde{\mathbf{X}}\tilde{\Lambda})\right)^{-1}(\tilde{\mathbf{X}}\tilde{\Lambda})^T(\bar{\mathbf{X}}\bar{Y})\right]\bar{\lambda}
\end{aligned}$$

It is interesting to note that in the case of no missing data, the above objective function simplifies back to the regular fully-labeled SVM case. The above objective function can be maximized via axis-parallel techniques. It is also important to use various matrix identities (i.e. some by Kailath [108] and some matrix partitioning techniques [121]) to make the optimization efficient. This optimization gives us the desired λ values to specify the distribution $P(\Theta, \gamma, y)$. This constrained optimization problem can be solved in an axis-parallel manner (similar to Platt's SMO algorithm). In fact, we modify a single λ_t value at a time in an axis-parallel type of optimization. Since there are no joint constraints on the λ vector, this step-wise optimization is feasible without exiting the convex hull of constraints. We derived an efficient update rule for any chosen lambda that will guarantee improvement of the objective function. The update will be different depending on the type of λ_t we choose, basically if it is labeled or unlabeled. It is also particularly efficient to iterate within the set of labeled and then the set of unlabeled Lagrange multipliers individually. This is because we store running versions of the large unlabeled data matrix, $\mathbf{G}$ and its inverse where:

$$\mathbf{G} \;=\; \kappa^2 I - (\tilde{\mathbf{X}}\tilde{\Lambda})^T(\tilde{\mathbf{X}}\tilde{\Lambda})$$

There are a number of ways that we can now use the current setting of the Lagrange multipliers to compute the labels of the unlabeled data. One way is to find the distribution over Θ, i.e. $P(\Theta)$ for the current setting of λ and integrate over it to obtain the classification. We now derive the computation of $P(\theta)$:

$$\begin{aligned}
P(\theta) &= \int_{\tilde{y}}\int_{\gamma} \frac{1}{Z(\lambda)} P_0(\Theta, \gamma, \tilde{y})\, e^{\sum_t \lambda_t [y_t \mathcal{L}(X_t|\Theta) - \gamma_t]} \\
P(\theta) &\propto \exp\left\{ -\frac{1}{2}\theta^T \left(I - \frac{1}{\kappa^2}(\tilde{\mathbf{X}}\tilde{\Lambda})(\tilde{\mathbf{X}}\tilde{\Lambda})^T \right)\theta + \sum_t \bar{\lambda}_t \bar{y}_t \bar{X}_t^T \theta \right\}
\end{aligned}$$

Thus, we have the $P(\theta) \sim \mathcal{N}(\mu_\theta, \Sigma_\theta)$. To obtain the classifier, we merely need the mean of the resulting Gaussian distribution over θ. Since $P(\theta) \sim \mathcal{N}(\mu_\theta, \Sigma_\theta)$, we have the following for our classifier:

$$\begin{aligned}
y_{new} &= \operatorname{sign}\left(\int_\theta P(\theta)\mathcal{L}(X_{new}|\theta) \right) \\
y_{new} &= \operatorname{sign}\left(\mu_\theta^T X_{new} \right)
\end{aligned}$$

More specifically, the mean μ_θ is given by the formula below (and the simplifications that follow):

$$\begin{aligned}
\mu_\theta &= \sum_t \bar{\lambda}_t \bar{y}_t \left(I - \frac{1}{\kappa^2}(\tilde{\mathbf{X}}\tilde{\Lambda})(\tilde{\mathbf{X}}\tilde{\Lambda})^T \right)^{-1} \bar{X}_t \\
\mu_\theta^T X_{new} &= \sum_t \bar{\lambda}_t \bar{y}_t \;\; \bar{X}_t^T X_{new} + \sum_{t'} \mathbf{m}_{t'} \tilde{X}_{t'}^T X_{new}
\end{aligned}$$

Where we have the following definition $\mathbf{m} = \kappa^2 \bar{y}^T (\bar{\Lambda}\bar{\mathbf{X}}^T \tilde{\mathbf{X}}\tilde{\Lambda}\mathbf{G}^{-1}\tilde{\Lambda})$. This vector effectively defines the linear decision boundary. It is important to choose κ large enough such that the unlabeled data influences the estimation strongly (small values of κ will cause vanishing unlabeled Lagrange multipliers, lock the unlabeled label estimates to 0 and effectively reduce to a standard SVM). Since all input vectors appear only within inner products computations, the formulation can readily accommodate kernels as well.

4.3.2 Transductive Regression

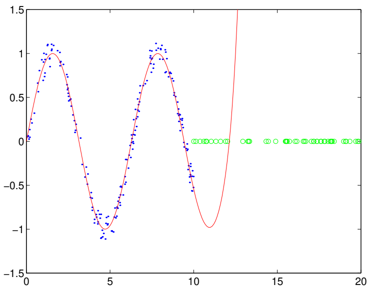
The previous assumption of a Gaussian over unlabeled data is actually much more reasonable for the regression case. In the previous section, we showed how unlabeled exemplars in a binary (± 1) classification problem can be dealt with by integrating over them with a Gaussian prior. However, the Gaussian is a continuous distribution and does not match the discrete nature of the classification labels. In regression, the outputs are scalars and are therefore much better suited to a continuous Gaussian prior assumption. If the scalar outputs do not obey a Gaussian distribution, we may consider transforming them (via the respective cumulative density functions) such that they are Gaussian. We may also consider using other continuous distribution as priors. However, the Gaussian has advantages since it has the conjugate form necessary to integrate over an MED linear regression problem (which results in a quadratic log-partition function in the non-transductive case). This guarantees that we will maintain a closed-form partition function in the transductive case.

Why would we wish to use unlabeled data in a regression scenario and when is it advantageous? The basic motivation is that transductive regression should focus the model such that its predictions on unlabeled data should be similarly distributed to its predictions on the labeled data. In other words, when we extrapolate to new test regions in the unlabeled data, the regression function should not diverge and exhibit unusual behavior. It should produce outputs that are similar to those it generates over the labeled data. This is illustrated in the following example where we fit a noisy sinusoidal data set with a high-order polynomial function. For example, note Figure 4.11. In The standard regression scenario in Figure 4.11(a), fitting a polynomial to the $\sin(x)$ function without transduction generates a good regression on the labeled data (blue dots) yet it sharply diverges on the unlabeled data (green circles) and produces predictions that are far from the typical range of $[-1, 1]$. If we instead require that the outputs on the unlabeled data obey a similar distribution, these will probably stay within $[-1, 1]$, generate similarly distributed output, and produce a better regression fit. This is illustrated in Figure 4.11(b) where the polynomial fit must obey the output distribution even when we extrapolate to the unlabeled data (at $x > 10$). It is important, however, not to go too far and have the regression function follow the prior on the unlabeled data too closely and compromise labeled data fitting as well as the natural regularization properties on the parameters. Therefore, as usual in MED with a multi-variable prior distribution, it is important to balance between the different priors.

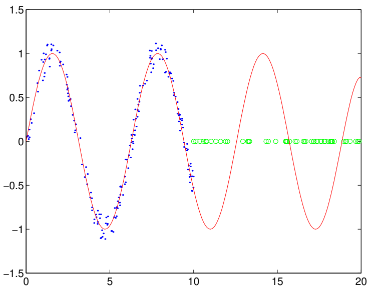
We begin the development with from a regular (non-transductive) regression setting. A support vector machine is typically cast as a large-margin regression problem using a linear discrimination function and an epsilon-tube of insensitivity with linear loss. Given input data as high dimensional vectors $X_1, \ldots, X_T$ and corresponding scalar labels $y_1, \ldots, y_T$ we wish to find a linear regressor that lies within ϵ of each output. The regressor is again the same discriminant function, $\mathcal{L}(X;\Theta) = \theta^T X + b$. Recall the objective function for the regression case in Theorem 5. If we assume, instead of a non-informative prior on $P_0(b)$ a zero-mean Gaussian prior with covariance σ we obtain the following slightly modified objective function (which must be optimized subject to $0 \leq \lambda_t \leq c$ and $0 \leq \lambda'_t \leq c$):

$$
\begin{aligned}
J(\lambda) \;=\; & \sum_t y_t(\lambda'_t - \lambda_t) - \epsilon \sum_t (\lambda_t + \lambda'_t) - \frac{1}{2}\sigma(\sum_t \lambda'_t - \lambda_t)^2 \\
& + \sum_t \log(\lambda_t) - \log\left(1 - e^{-\lambda_t \epsilon} + \frac{\lambda_t}{c - \lambda_t}\right) + \sum_t \log(\lambda'_t) - \log\left(1 - e^{-\lambda'_t \epsilon} + \frac{\lambda'_t}{c - \lambda'_t}\right) \\
& - \frac{1}{2}\sum_{t,t'} (\lambda_t - \lambda'_t)(\lambda_{t'} - \lambda'_{t'})(X_t^T X_{t'})
\end{aligned}
$$

In the case of unlabeled data, we do not know some particular y_t values and must introduce a prior



(a) Polynomial Regression



(b) Transductive Polynomial Regression

Figure 4.11: Transductive Regression vs. Labeled Regression Illustration. Here, we are attempting to fit the $\sin(x)$ function with a high order polynomial. Labeled data is shown as blue dots while unlabeled data are put on shown as green circles on the x-axis. In (a) the unlabeled data are not used and we merely fit a regression function (red line) which unfortunately diverges sharply away from the desired function when it is over the unlabeled data. In (b), the polynomial must maintain a similar distribution of outputs (roughly within [-1,1]) over the unlabeled exemplars and therefore produces are more reasonable regression function.

over these and integrate it out to obtain the the partition function. The prior we shall use over the unobserved y_t is a white Gaussian prior. This modifies the above optimization function as follows. Observe the component of $J(\lambda)$ that depends on a given y_t:

$$J(\lambda) \quad = \quad \ldots + y_t(\lambda'_t - \lambda_t) + \ldots$$

Going back to the partition-function representation of that component we have:

$$Z(\lambda) \quad = \quad \ldots \times \exp\left(-y_t(\lambda'_t - \lambda_t)\right) \times \ldots$$

If the y_t value of the above is unknown, we can integrate over it with a Gaussian distribution as a prior, i.e. $P_0(y_t) \sim N(0,1)$ [4]. The Gaussian prior gives rise to the following computation:

$$Z(\lambda) \quad = \quad \ldots \times \int_{-\infty}^{\infty} \exp\left(-\frac{1}{2}y_t^2\right) \exp\left(-y_t(\lambda'_t - \lambda_t)\right) \times \ldots$$

Ultimately our updated transduction $J(\lambda)$ function is modified as follows for the unlabeled data exemplars:

$$J(\lambda) \quad = \quad \ldots + \frac{1}{2}(\lambda'_t - \lambda_t)^2 + \ldots$$

Therefore, for the transductive regression case, we obtain the following objective overall function:

$$\begin{aligned} J(\lambda) \quad = \quad & \sum_{t \in labeled} y_t(\lambda'_t - \lambda_t) + \sum_{t \in unlabeled} \frac{1}{2}(\lambda'_t - \lambda_t)^2 - \epsilon \sum_t (\lambda_t + \lambda'_t) - \frac{1}{2}\sigma(\sum_t \lambda'_t - \lambda_t)^2 \\ & + \sum_t \log(\lambda_t) - \log\left(1 - e^{-\lambda_t \epsilon} + \frac{\lambda_t}{c - \lambda_t}\right) + \sum_t \log(\lambda'_t) - \log\left(1 - e^{-\lambda'_t \epsilon} + \frac{\lambda'_t}{c - \lambda'_t}\right) \\ & -\frac{1}{2}\sum_{t,t'}(\lambda_t - \lambda'_t)(\lambda_{t'} - \lambda'_{t'})(X_t^T X_{t'}) \end{aligned}$$

[4] A uniform prior for $P_0(y_t)$ is also feasible.

The final $P(\Theta)$ computation is straightforward to solve for once we have the optimal setting of λ by maximizing $J(\lambda)$. This effectively generates a simple linear regression model which takes into account the unlabeled data. In practice, the y_t values don't have a white Gaussian distribution so we transform these into a white Gaussian (via standard histogram fitting techniques or just a whitening affine correction) and then solve the MED regression. The transformation is then inverted to obtain y_t values appropriate for the original problem.

Figure 4.12 depicts results on an the Ailerons data set (by R. Camacho) which addresses a control problem for flying an F16 aircraft. The inputs are 40 continuous attributes are given that describe the status of the airplane (i.e. pitch, roll, climb-rate) while the output is the control action for the ailerons of the F16. An implicit second-order polynomial (quadratic) kernel was used as a regression model. For the labeled case, we trained on 96 labeled data points (using standard SVM regression). The MED transductive regression case used 96 labeled and 904 unlabeled examples for training. Figure 4.12 depicts better regression accuracy for transduction techniques at appropriate levels of regularization (while the non transductive regression remains somewhat fixed despite varying regularization levels).

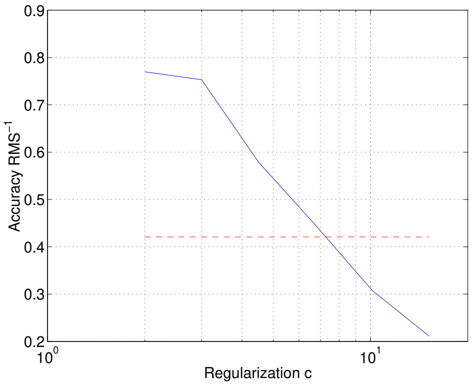


Figure 4.12: Transductive Regression vs. Labeled Regression for F16 Flight Control. The above show the inverse RMS error for the labeled regression case (dashed red line) and the transductive regression case (solid blue line) at varying c-regularization levels.

It appears the the transduction is mostly useful when the labeled data is ambiguous and can cause large errors when extrapolating out of a region that was well sampled to new test data unlabeled data. The Gaussian prior on unobserved variables effectively constrains the extrapolation caused by over-fitting by preventing unlabeled examples from having extreme outputs. If the unlabeled examples are, however, in the convex hull of the labeled ones, transductive regression is unlikely to be beneficial.

4.4 Other Extensions

In this sections we will motivate some extensions at a cursory level for completeness. More thorough derivations and results concerning anomaly detection, latent anomaly detection, tree structure learning, invariants, and theoretical concepts can be found in [85] [90] and [128]. The MED framework is not just limited to learning continuous model parameters, eg. Gaussian means and covariances. It can also be used to learn discrete structures as well. For instance, one may consider using MED to learn both the parameters and the structure of a graphical model. For instance, Θ may be partitioned into a component that learns the discrete independency structure of the graphical model and a component that learns the continuous parameters of the probability tables. Since the MED solves for a continuous distribution over the discrete and continuous model components, its estimation will remain straightforward.

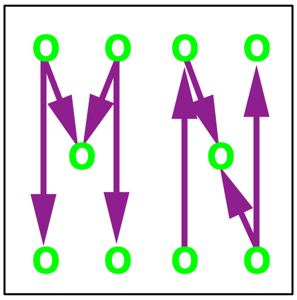


Figure 4.13: Tree Structure Estimation.

For example, consider solving for tree structures where a classifier results from the likelihood ratio of two tree distributions. We have a space of dimensionality D and therefore D nodes in each tree to connect. In Figure 4.13 we show an example where 5 nodes are to be connected in different tree structures. One configuration is on the left while the other is on the right. The resulting discriminant function has the abstract form:

$$\mathcal{L}(X;\Theta) \quad = \quad \log \frac{P(X|\theta_+, E_+)}{P(X|\theta_-, E_-)} + b$$

Here, the Θ model description will be composed of both set of $\theta_\pm$ continuous parameters for each tree as well as a structure component $E_\pm$ which specifies the configuration of edges which will be present between the various nodes. The classification constraints will then involve not only an integration but also a summation over discrete structures:

$$\sum_{E_+}\sum_{E_-}\int_{\theta_+}\int_{\theta_-}\int_{\gamma_t}\int_b [y_t\mathcal{L}(X_t;\Theta) - \gamma_t] d\theta_+ d\theta_- d\gamma_t db \quad \geq \quad 0 \;\; \forall t$$

Similarly, computation of the partition function $Z(\lambda)$ will require integration of the exponentiated constraints and the prior over $P_0(\theta_+, \theta_-, E_+, E_-, \gamma, b)$. Since there is an exponential number of tree structures that could connect D nodes, summing over all E_+ and E_- would be intractable. However, due to some interesting results in graph theory (namely the matrix tree theorem), summing over all possible tree structures of a graph can be done efficiently. This is reminiscent of Section 4.2 where we discussed an alternative form of structure learning. There, we also solved for a discrete component of the model, namely feature selection. We similarly had to sum over an exponential number of feature selection configurations. However, in this problem and the earlier one, embedding the computation into a probabilistic MED setting makes it solvable in an efficient way. Further details on tree structure estimation will be omitted here yet are provided in [85] and [128].

4.5 Mixture Models and Latent Variables

We have so far seen many variations of the MED formalism and its flexible application in different scenarios and with different models. One key benefit MED enjoys is the ability to combine generative modeling virtues such as prior distributions with discriminative requirements that are typically manifested as constraints. However, we have so far restricted the generative models in MED to simple and typically uni-model distributions. For example, we have shown exponential family classification using, e.g. Gaussian and multinomial models. However, to thoroughly harness that power of generative modeling, we must go beyond such simple models and consider mixture models or latent variable models. These models include sigmoid belief networks, latent Bayesian networks, hidden Markov models which play a critical role in many applied domains (speech recognition, vision, etc.). These could also be potential clients for the MED formalism and could benefit strongly from a discriminative estimation technique (as opposed to their traditional maximum-likelihood incarnations).

Therefore, ideally, we would also like to handle latent mixture models in an MED discriminative setting. However, this type of problem is decidedly more difficult than the cases we have seen so far. Latent models and mixtures give rise to logarithms of sums and negated logarithms of sums. Therefore, computational difficulties arise and the various MED integrals and optimizations become intractable. In maximum likelihood and Bayesian estimation, these intractabilities are readily addressed by the use of Jensen inequalities to simplify expressions and generate iterative variants of the closed-form computations in the non-latent case. The Expectation-Maximization (EM) algorithm is essentialy such a tool which iteratively solves a simpler version of an intractable latent-variable problem using Jensen lower bounds on the logarithm of a sum. However, the EM algorithm is designed for maximum likelihood estimation and is insufficient for MED discrimination since the later involves negated logarithms of sums (the negation flips Jensen lower bounds and creates undesirable upper bounds). Therefore, latent discrimination will require novel bounds and a discriminative variant of the EM algorithm. The next chapter will propose such a discriminative counterpart and will develop novel bounding tools which will permit discrimination on latent models. Although the treatment in the next chapter will often focus on maximizing conditional likelihood, the tools that will be developed for CML while also be useful for latent discrimination in MED.

Chapter 5

Latent Discrimination and CEM

Entities should not be multiplied unnecessarily [1] .

William of Ockham, 1280-1349

We have discussed several frameworks for optimizing the discriminative power of generative models. These include maximum conditional likelihood, conditional Bayesian inference, and, the aforementioned Maximum Entropy Discrimination. All emphasize accuracy on a the given task either through margins or a 'background' probability. However, computational problems quickly arise when these are applied to latent models, i.e. mixture models and models with hidden variables. It is these very mixture models which are the workhorses of generative machine learning. Structures such as mixtures of Gaussians, many Bayesian networks, hidden Markov models, and so forth are actually latent generative models and not just simple exponential family distributions. The latent aspects of such models can prevent them from being mathematically tractable in a discriminative setting.

Statistical model estimation and inference often require the maximization, evaluation, and integration of complicated mathematical expressions. One approach for simplifying the computations is to find and manipulate variational upper and lower bounds instead of the expressions themselves. A prominent tool for computing such bounds is Jensen's inequality which subsumes many information-theoretic bounds (cf. Cover and Thomas 1996 [40]). In maximum likelihood (ML) estimation under incomplete data, Jensen is used to derive an iterative Expectation-Maximization (EM) algorithm [13] [12] [44]. For graphical models, intractable inference and estimation is performed via variational bounds [103]. Bayesian integration also uses Jensen and EM-like bounds to compute integrals that are otherwise intractable [3] [63].

For discriminative frameworks to handle latent models, we need a discriminative version of the EM algorithm and the bounds it uses. It is well known that in generative frameworks (eg. likelihood) the EM algorithm uses Jensen's inequality to lower bound latent log-likelihoods. The resulting tractable variational bounds can then be integrated or maximized in a straightforward manner. However, unlike their generative counterparts, discriminative frameworks involve both latent log-likelihoods and *negated* log-likelihoods. The latter are not lower boundable by Jensen's inequality and instead

[1] At the risk of misquoting what Ockham truly intended to say, we shall use this quote to motivate the use of bounds on latent likelihoods which, if treated exactly and multiplied exactly, would cause an exponential explosion in the number terms.

require a type of reverse-Jensen inequality [94]. We derive this inequality for arbitrary mixtures of the exponential family. This includes complex mixtures such as those arising in hidden Markov models. The resulting bounds on the parameters and the mixing proportions then permit straightforward maximization-steps or integration.

More specifically, we will show how a mixture model (or incomplete distribution) can be lower bounded (through *Jensen's inequality*) and also upper bounded (through a *reverse-Jensen inequality*) by a *single* complete e-family distribution over parameters. For discriminative learning, the reverse bounds are tight enough to permit efficient estimation that approaches EM computationally yet yields superior classification and regression accuracy. We will call the resulting discriminative EM variant the Conditional Expectation Maximization (CEM) algorithm.

This chapter is organized as follows. We first describe the set of probabilistic models we will restrict ourselves to: mixtures of the exponential family. We then argue why latent models cannot be handled tractably in the same manner as exponential families. This is because they typically form intractable expressions when multiplied. A formal treatment of these models exposes why generative (and discriminative) criteria therein become intractable. We then describe the EM algorithm as a way to address this intractability in latent models. EM addressed the intractable estimation problem by by iteratively maximizing a lower bound on log-likelihood via *Jensen's inequality*. Discriminative criteria, however, also require a reverse upper bound. We propose such a bound, the *reverse-Jensen inequality*, and develop it for a variety of cases. This includes mixtures of Gaussians, mixtures of multinomials, mixing coefficients. The next chapter considers more sophisticated mixtures or structure models such as hidden Markov models and aggregated data set bounds. A rigorous derivation and proof of the reverse-Jensen bound is provided in Chapter 7 which justifies its use in latent discrimination and CEM.

5.1 The Exponential Family and Mixtures

We will restrict the treatment of latent variables (via Jensen and our reverse-Jensen bounds) to mixtures of the exponential family (e-family) [9] [32]. In practice this class of densities covers a very large portion of contemporary statistical models. Mixtures of the e-family include Gaussians Mixture Models, Multinomials, Poisson, Hidden Markov Models, Sigmoidal Belief Networks, Discrete Bayesian Networks, etc. [34]. The e-family (which is closely related to generalized linear models) has the following form:

$$P(X|\Theta) \quad = \quad \exp(\mathcal{A}(X) + X^T\Theta - \mathcal{K}(\Theta))$$

Here, the e-family is shown in its *natural* parameterization. Many alternative parameterizations exist however the natural one will be easiest to manipulate for our purposes (i.e. for computing the reverse-Jensen inequality and performing discriminative estimation). The $\mathcal{K}(\Theta)$ function is the *cumulant generating function* [32] and is convex in Θ, the multi-dimensional parameter vector. Typically the data vector X is constrained to live in the gradient space of $\mathcal{K}$, i.e. $X \in \frac{\partial}{\partial\Theta}\mathcal{K}(\Theta)$ or $X \in \mathcal{K}'(\Theta)$ for short. In fact, a duality exists in that the domain of the function $\mathcal{A}(X)$ is the gradient space of $\mathcal{K}(\Theta)$ and vice versa. A more specific property of the exponential family is that the cumulant generating function is not just an arbitrary convex but also given by the following Laplace transform. This is directly due to the normalization property of the distribution (which directly generates convexity of

$\mathcal{K}(\Theta)$ [2]):

$$\mathcal{K}(\Theta) = \log \left(\int_X \exp(\mathcal{A}(X) + X^T \Theta) dX \right)$$

The e-family has special properties (i.e. conjugates, convexity, linearity, etc.) [9] [32] [34] [6]. Furthermore, one very important property of this class of distributions is that products of exponential family distributions remain in the family. These intrinsic properties will be exploited for the derivation of the reverse-Jensen bound in this chapter and the next. The table below lists example $\mathcal{A}$ and $\mathcal{K}$ functions for Gaussian, multinomial[3] and other distributions.

E-Distribution	$\mathcal{A}(X)$	$\mathcal{K}(\Theta)$	Constraints
Gaussian (mean)	$-\frac{1}{2}X^T X - \frac{D}{2}\log(2\pi)$	$\frac{1}{2}\Theta^T\Theta$	
Gaussian (covariance)	$-\frac{1}{2}\log(2\pi)$	$-\frac{1}{2}\log(\Theta)$	$\forall\Theta > 0$
Multinomial	$\log(\Gamma(\eta+1)) - \log(\nu)$	$\eta\,\log(1+\sum_{d=1}^{D}\exp(\Theta_d))$	
Exponential	0	$-\log(-\Theta)$	$\forall\Theta < 0$
Gamma	$-\exp(X) - X$	$\log\Gamma(\Theta)$	$\forall\Theta > 0$
Poisson	$\log(X!)$	$\exp(\Theta)$	

Table 5.1: Sample exponential family distributions.

This crucial property (along with others) makes maximum likelihood estimation of the parameters of an e-family distribution with respect to an iid data set fully tractable, unique and straightforward. This is because log-likelihood remains concave in the parameters for the e-family (and products thereof). It is also straightforward to integrate over an exponential family distribution (with conjugate priors) to obtain a fully Bayesian estimate of its parameters. Thus, it is widely acknowledged that this family enjoys tractable and straightforward estimation properties.

5.1.1 Mixtures of the Exponential Family

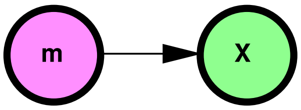


Figure 5.1: Graph of a Standard Mixture Model

A generalization beyond the class of exponential family distributions is to consider *mixtures* of the e-family. We begin by considering a simple (flat) mixture model (Chapter 6 considers more complex mixture cases). Here, a convex combination of different e-family distributions with different parameters is shown. The mixture is done by introducing a latent variable which is denoted as m here. Thus, we have an incomplete data representation and since m is unobserved, multiple e-family

[2] We will actually be mildly restricting our derivations to $\mathcal{K}$ functions satisfying Lemma 7.4. However, *all standard* e-family distributions are still spanned.

[3] The multinomial shown is traditionally cast as in Equation 3.13 subject to the constraint $\sum_{d=1}^{D+1}\rho_d = 1$. In the table, we map this into exponential family form where $\eta = \sum_{d=1}^{D+1} X_d$ and $\nu = \Pi_{d=1}^{D+1}\Gamma(X_d+1)$. In the standard case where X is a choice vector with all zeros and a single unit entry, the expressions simplify to $\eta = 1$ and $\nu = 1$ and then we only have $\mathcal{A}(X) = 0$.

models models are mixed [4]. Thus, we can consider Figure 5.1 as a Bayes net describe the relationship between the latent variable m which acts as a parent to the emission variable X. This generates the following distribution:

$$p(X|\Theta) = \sum_m p(m)p(X,\Theta|m) = \sum_m \alpha_m \exp(\mathcal{A}_m(X_m) + X_m^T\Theta_m - \mathcal{K}_m(\Theta_m)) \tag{5.1}$$

In the above, we are allowing the X_m to vary with the latent variable although in a regular mixture model, we typically have $X_m = X \; \forall m$. This mathematical generalization or extra flexibility is done here for convenience later on when we will need to consider different types of mixtures where X_m will vary with m, i.e. the X_m will be a vector of the same cardinality as X but explicitly computed as a result of some function of both X and m. Furthermore, the α_m are scalars that sum to unity representing the prior on each model in the mixture. We will assume that the α_m are fixed now yet this assumption will be loosened in Section 5.6.

For now, merely think of the above as a standard mixture model such as a mixture of Gaussians [20]. For example, we may consider the distribution of the weight and height of the males and females of a species which would form two distinct clusters. Thus, the latent variable m would be binary and determines the hidden male/female variable which select between the two different Gaussian distributions on, eg. weight and height. This type of mixture can be called a flat mixture since there are no additional relationships between the clusters or a hierarchy between the hidden variables m. These types of latent distributions arise frequently in many machine learning tasks and include include mixtures of Gaussians, mixtures of experts, mixtures of multinomials, and so forth. These latent probability distributions need to get maximized, integrated, marginalized, conditioned, etc. to solve various inference, prediction, and parameter estimation tasks. However, such manipulations can sometimes be quite complex or intractable.

5.2 Mixtures in Product and Logarithmic Space

Why is it that non-exponential family models, such as mixture models or latent models, cannot be treated exactly or in closed form? This is the case for both generative *and* discriminative frameworks. In MED, just as in maximum likelihood, we can show that any exponential family member can be estimated in a straightforward manner. However, as we consider mixtures of the exponential family, there are many interpretations for how intractabilities that arise. We shall begin by illustrating these problems using two different metaphors: 'product space' or 'logarithmic space'. First, consider the likelihood of an independent identically distributed (iid) data set in 'product space':

$$p(\{X\}|\Theta) \;=\; \Pi_{t=1}^T p(X_t|\Theta)$$

If the generative model $P(X_t|\Theta)$ for each data point is *in* the exponential family, the above aggregate likelihood for the whole data set remains computationally tractable. In fact, products of the exponential family are in the exponential family. In other words, the e-family forms a closed set under multiplication (but not under addition unfortunately). For example, we would obtain the following if we had an e-family distribution:

$$\begin{aligned} p(\{X\}|\Theta) &= \Pi_{t=1}^T \exp(\mathcal{A}(X_t) + X_t^T\Theta - \mathcal{K}(\Theta)) \\ &= \exp\left(\sum_t \mathcal{A}(X_t) + (\sum_t X_t)^T\Theta - T\mathcal{K}(\Theta)\right) \end{aligned}$$

[4] Note we use Θ to denote an aggregate model encompassing all individual $\Theta_m \; \forall m$.

Thus, the aggregate $P(\{X\}|\Theta)$ is in the exponential family itself and it would be just as easy to integrate over it or maximize the likelihood as it would be to work with a single data point $P(X_t|\Theta)$. For example, maximizing the likelihood over the parameter Θ would simply be given by the unique closed-form solution of:

$$\frac{\partial \mathcal{K}(\Theta)}{\partial \Theta} = \frac{1}{T}\sum_t X_t$$

However, if the generative model $P(X_t|\Theta)$ is *not in* the exponential family but rather a mixture model or a latent distribution, it must be represented as a sum or marginalization over a (possibly continuous) set of simpler distributions. Unfortunately, summations (unlike products) of exponential family distributions are *not* in the exponential family. For example, consider the case where we are summing M exponential family distributions to get $P(X_t|\Theta)$, in other words $\sum_{m=1}^{M} P(m, X_t|\Theta)$. In this case, there are M possible configurations of the simpler distributions and expanding we get:

$$p(\{X\}|\Theta) = \Pi_{t=1}^{T}\left(\sum_{m=1}^{M} p(m, X_t|\Theta)\right)$$

Here we see quite clearly that if we were to expand the above product over sums we would effectively have a summation over *many* terms, in fact an exponential number equal to M^T. At this point, integration or other calculations would be solvable as we bring them into the sum and apply them to the simple non-latent distributions in isolation. However, having to deal with a total of M^T terms prevents this approach from being tractable except in toy problems. Thus, exact Bayesian integration (as depicted in Example 2.3.1) would become very cumbersome.

It is often the case that we will manipulate the log-likelihood instead of the likelihood itself (as in maximum likelihood estimation). Taking the logarithm will put us in 'logarithmic space' which might *deceptively* appear to simplify the expressions:

$$\log p(\{X\}|\Theta) = \log \Pi_{t=1}^{T} p(X_t|\Theta) \tag{5.2}$$

$$= \sum_{t=1}^{T} \log p(X_t|\Theta) \tag{5.3}$$

$$= \sum_{t=1}^{T} \log\left(\sum_{m=1}^{M} p(m, X_t|\Theta)\right) \tag{5.4}$$

Now, we only have a summation over a tractable number of terms, namely T terms in total. However, the presence of the log-sum is equally intractable and prevents direct integration or maximization steps. Thus, non-exponential family distributions (i.e. mixtures) prevent easy maximization calculations for ML (i.e. maximizing by taking gradients and setting to zero). Alternatively, in Bayesian inference, we will attempt to perform integrals to compute, for example, the evidence:

$$p(\{X\}) = \int p(\{X\}, \Theta) d\Theta \tag{5.5}$$

$$= \int \Pi_{t=1}^{T}\left(\sum_{m=1}^{M} p(m, X_t|\Theta)\right) P(\Theta) d\Theta \tag{5.6}$$

Once again, integrating over a product of sums is a very difficult task. In fact, the intractability resulting from products of sums is equivalent to the intractability resulting from the addition of logarithms of sums. Therefore, it is highly desirable to find surrogate distributions which are easier to manipulate than these latent distributions or log-sums. These can be either upper or lower bounds on the log-sum so that we can give guarantees on the integrals or iteratively maximize the objective functions (such as likelihood). Conceptually, we would like to *pull* the logarithm into the sum so that we have a sum of logs instead of a log-sum. This intractability is directly avoided when we introduce bounds (upper and lower) which permit tractable multiplication, maximization and integration. In this chapter we will derive these upper and lower bounds and provide an analytic formula for obtaining them. Basically, the multi-modal, complex expressions we have shown above will be replaced with simple (exponential family) distributions that upper and lower bound them to avoid intractabilities. We will focus on the log-sum interpretation in the next sections but all results are directly equivalent in the product space as well. Product space is more convenient for integration while logarithmic space is more convenient for maximization. We now discuss the celebrated EM [44] algorithm which does just that: it lower bounds the latent log-sums to alleviate the computational intractability.

5.3 Expectation Maximization: Divide and Conquer

The Expectation-Maximization (EM) algorithm [5] finds its roots as the successor of old heuristic approaches to fitting mixtures of models and clustering algorithms such as k-means. In the case of mixtures of models or latent maximum likelihood estimation, notions of divide and conquer [102] were used to motivate EM-type procedures. One can view maximum likelihood with a mixture model as several independent models available to collectively describe a training data set. Therefore, we divide the data among these models in some sort of competition where each model gravitates to data that it accounts for most strongly. Iterating this division of data and conquering (i.e. reestimating individual models) breaks down what would be a difficult fit to a complex training set of points into several small fitting operations to partitions of the training data. In k-means, models gravitate to clusters of points by competing in a winner take all scenario. EM is also such a divide and conquer operation but models do not greedily take over a given point (winner-take-all) but share a soft responsibility assignment to each data point. In other words, models that describe a point better are given a higher responsibility or weight. Computationally, then, this division permits each model to be estimated from data as if it were alone, simplifying the intractabilities of the log-sum in Equation 5.4, for instance. Each model is estimated on its own over a weighted configuration of the data according to its previous responsibility levels. Effectively, the summation inside the logarithm is *pulled outside*, avoiding the difficulties in the m-steps (or the intractable integration in Bayesian inference over Equation 5.6).

EM's strategy of conquer and divide *works* not because it is intuitive but because of the mathematical properties of maximum log-likelihood, namely the concavity of the log function and the direct applicability of Jensen's inequality in forming a guaranteed lower bound on log-likelihood [44] [13] [12]. In fact, if we change the objective function, this same conquer and divide strategy will not work. Figure 5.2 depicts what EM is effectively doing. In an E-step, we compute a lower bound on the log-likelihood using Jensen's inequality and in an M-step we maximize that lower bound. By iterating these two procedures, we are bounding and maximizing the objective function which is therefore guaranteed to increase monotonically.

Unfortunately, discriminative criteria cannot use EM's simple conquer and divide strategy because Jensen's inequality does not generate a lower bound on their objective functions. This will be elaborated in the following sections but we provide a high level discussion here. Computationally,

[5] Here we are deferring a formal treatment in favor of an intuitive explanation of EM.

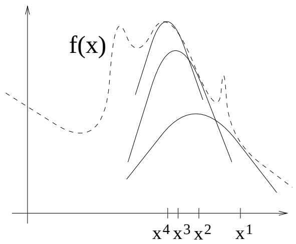
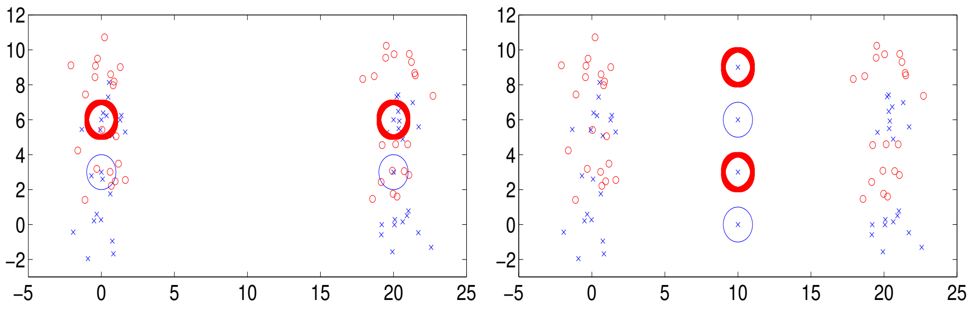


Figure 5.2: Expectation-Maximization as Iterated Bound Maximization.

what differentiates the discriminative criteria from ML is that they not only require Jensen-type lower bounds but also need the corresponding upper bounds. The Jensen bounds only partially simplify their expressions and some intractabilities remain. For instance, latent distributions need to be bounded above and below in a discriminative setting[85]. Metaphorically, discriminative learning requires lower bounds to cluster positive examples and upper bounds to *repel* away from negative ones. Thus, conquer and divide doesn't work and we need a discriminative variant of the conquer-and-divide strategy.

5.4 Latency in Conditional and Discriminative Criteria

Why can't we apply EM in a discriminative setting? The combination of ML estimates with EM and Jensen have indeed produced straightforward and monotonically convergent estimation procedures for mixtures of the e-family [44] [34] [103]. However, EM, like ML, does not explicitly address the task of the learning system. It is a are rather non-discriminative *modeling* technique for estimating a generative model. Consequently, EM and ML suffer when assumptions in the model are inaccurate.



ML Classifier: $l = -8.0$, $l^c = -1.7$ CML Classifier: $l = -54.7$, $l^c = 0.4$

Figure 5.3: Maximum Likelihood versus Maximum Conditional Likelihood. Thick Gaussians represent o's, thin ones represent x's.

For visualization, observe the binary classification[6] problem in Fig. 5.3. Here, we have a training data set which consists of o's (positive class) and x's (negative class). These have been sampled from 8 identity-covariance Gaussians (4 of each class). Each Gaussian has equal probability. We will

[6]The derivations herein extend to multi-class classification and regression as well.

fit this data with a two-class generative model which incorrectly has 2 Gaussians per class (again with equal probability each and identity covariance). Two solutions are shown, ML in Figure 5.3(a) and CML in Figure 5.3(b). Each of the 4 Gaussians in the model is depicted by its iso-probability contour. The 2 thick circles represent the positive (o's) Gaussian models while the 2 thin circles represent the negative (x's) Gaussian models. We also see the values of the joint log-likelihood l and conditional log-likelihood l^c for each solution. Note how ML has the larger l value of the two while CML has the larger l^c value of the two configurations. These distributions induce a classification boundary, i.e. points where the positive 2-Gaussian model is greater than the negative one will be assigned to the positive class and vice-versa.

In Figure 5.3(a) this results in a decision boundary that splits the figure in half with a horizontal line across the middle because the positive Gaussians overtake the top and the negative ones overtake the bottom half of the figure. Counting the number of correct classifications, we see that the ML solution performs as well as random chance, getting roughly 50% accuracy. This is because the ML model is trying to cluster the data and put the Gaussian models at our disposal such that their probability mass is on top of the data samples (that belong to their class). In fact, fitting the positive data with the positive model is done independently of the fit of the negative data with the negative model. This is precisely how EM iterates and the objective it maximizes is likelihood. It's objective is to get as good a generator of the data so classification performance is sacrificed.

Meanwhile, in Figure 5.3(b), the decision boundary that is generated by the model creates 4 horizontal classification strips (as opposed to splitting the figure in half). These classify the data as positive,negative,positive and negative respectively as we go from top to bottom, because the 4 Gaussians are arranged in this vertical interleaving order. The accuracy for this fit is roughly 100%. This is because CML attempts to form a good output (class label) distribution given the input and this is more appropriately suited to classification. CML, in estimating a conditional density, propagates the classification task into the estimation criterion. It is clear, however, that the model is not a good generator of the data since the Gaussians don't put their probability mass on the samples but are simply arranged down the middle of the figure which provides a very low value of likelihood l. Clearly, therefore, optimizing l^c using CML is more appropriate for finding a good classifier model. What is needed is a discriminative or conditional counterpart of EM that seeks to optimize l^c instead.

Let us now describe the above classification scenario more mathematically. In such examples, we are given training examples X_i and corresponding binary labels c_i. The goal is to classify the data with a latent variable e-family model (a mixture of Gaussians here). We use m to represent the latent missing variables. Let us now consider the objective functions which will immediately exhibit the aforementioned intractable log-sum structures. For the generative log-likelihood objective function l we have the following formula:

$$l = \sum_i \log \sum_m p(m, c_i, X_i|\Theta)$$

For the more discriminative conditional likelihood approach we have the conditional log-likelihood l^c which also has log-sums as well as *negated* log-sums:

$$\begin{aligned} l^c &= \sum_i \log \sum_m p(m, c_i|X_i, \Theta) \\ &= \sum_i \log \sum_m p(m, c_i, X_i|\Theta) - \sum_i \log \sum_m \sum_c p(m, c, X_i|\Theta) \end{aligned}$$

Alternatively, we could have considered an even more discriminative MED approach whose discriminant function would again result in complications due to the log-sum:

$$\begin{aligned}\mathcal{L}(X|\Theta) &= \log \frac{p(X|\Theta_+)}{p(X|\Theta_-)} \\ &= \log \sum_m p(m, X|\Theta_+) - \log \sum_m p(m, X|\Theta_-)\end{aligned}$$

In the above latent log-likelihoods and discriminant functions we recognize the presence of logarithms of sums (and *negated* log-sums). As before these cause intractabilities (which remain in the product space interpretation as well). EM can handle log-sums through Jensen's inequality and manipulate lower bounds on log-likelihood. Each log-sum in the objective function l will be lower bounded and all the lower bounds will add to form an aggregate lower bound. However, in conditional and discriminative criteria we observe *negated* log-sums. Negation will flip the Jensen inequality lower bounds. Thus, applying Jensen on the negated components of these objective functions or discriminant functions will actually produce upper bounds. Thus, the log-sum terms will be lower bounded while the negated log-sum terms will be upper bounded. Adding lower bounds and upper bounds is useless sinc it will *not* generate the desired aggregate lower bound on the discriminative quantities (i.e. conditional likelihood or MED discriminant functions).

We next show the Jensen inequality which will lower bound log-sums to produce an EM algorithm. For discrimination, we also derive the complementary upper bounds [7] through a reverse-Jensen inequality. These reverse-bounds are structurally similar to Jensen bounds, allowing easy migration of ML techniques to discriminative settings. The bounds could also be useful as mathematical tools for non-statistical problems. We will focus the development of these bounds on mixtures of the exponential family which will have desirable properties permitting us to establish guaranteed bounds on discriminative quantities like conditional likelihood and Maximum Entropy Discrimination.

5.5 Bounding Mixture Models

As was mentioned earlier, optimizing likelihood or conditional likelihood becomes intractable when we have latent models or mixture models (equivalently). The EM algorithm was shown to be well suited to maximize likelihood since it replaces the complicated log-sum expressions that arise with simple lower bounds that can be maximized in a straightforward way. Yet, for discrimination and conditional likelihood maximization, we have negated log-sums and cannot use lower bounds alone. Therefore, it is clear that we need to bound latent quantities or log-sums on both sides with lower *and* upper bounds. Once again, we assume that we have a generative model that is given by a mixture of exponential family as described earlier: $p(X) = \sum_m p(m)p(X|m)$. In a log-likelihood based optimization, each data point therefore, gives rise to a log-sum term $\log(\sum_m p(m)p(X|m))$ which causes intractabilities.

We propose upper and lower bounds on the log-sum which appear as follows (in a logarithmic space):

$$\sum_m \tilde{w}_m \log p(\tilde{Y}_m|m,\Theta) + \tilde{k} \;\leq\; \log \sum_{m=1}^{M} p(m, X|\Theta) \;\leq\; \sum_m (-w_m) \log p(Y_m|m,\Theta) + k \tag{5.7}$$

Taking exp() of both sides allows us to consider these bounds in product space:

$$\exp(\tilde{k})\Pi_m p(\tilde{Y}_m|m,\Theta)^{\tilde{w}_m} \;\leq\; \sum_{m=1}^{M} p(m, X|\Theta) \;\leq\; \exp(k)\Pi_m p(Y_m|m,\Theta)^{-w_m} \tag{5.8}$$

[7] A similar yet weaker bound was shown for Gaussian mixture regression in [92].

Upon inspection, it is clear that the left hand side and right hand side of the inequalities share a very similar structure. This homogeneous structure is critical if we are to use lower bounds and upper bounds in a combined way to simplify log-sums. Furthermore, through these bounds it should be evident that we are no longer dealing with a log-sum but rather a sum of logs (we have 'pulled' the log into the summation). This structure is far easier to handle computationally. In product space, we are no longer dealing with a sum of e-family distributions but rather products of e-families (which remain in the e-family) and therefore immediately inherit the straightforward computational and estimation properties therein. We have yet to specify the parameters of the upper and lower bounds, namely the $\tilde{k}, \tilde{w}_m, \tilde{Y}_m$ and the k, w_m, Y_m respectively (where the subscript m ranges from $1..M$, the number of latent models). This will be done shortly after we give a conceptual description of the parameters and their roles.

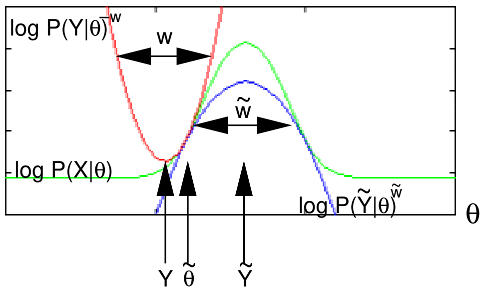


Figure 5.4: Dual-Sided Bounding of Latent Likelihood

The left hand side of the above inequality follows a direct consequence of Jensen's inequality (which computes guaranteed $\tilde{k}, \tilde{Y}_m, \tilde{w}_m$). The right hand side is a direct consequence of the reverse-Jensen inequality (which computes guaranteed k, Y_m, w_m). The bounds basically give a weight to the data points (i.e. w_m) and translate their coordinates (i.e. from X_m to Y_m) to avoid representing them as a sum of distributions. Therefore, we have replaced the sum of the exponential family members by bounds which are only products of the exponential family. As we said earlier, products of the e-family will remain in the e-family and intractable latent log-likelihood quantities will be bounded by simple e-family distributions above and below. The bounding is illustrated in Figure 5.4 on a 1d example (where the e-family being used here is a Gaussian mean). The middle curve (which is neither concave nor convex) is the original latent log-likelihood while the upper bound is a convex (as a consequence of the negated weight $-w$) complete likelihood and the lower bound is a concave (as a consequence of a positive weight $\tilde{w}$) complete likelihood. The upper and lower bounds both make contact with the original log-sum quantity at the point $\tilde{\Theta}$ which is also referred to as the contact point. If we were performing iterative maximization (such as in EM), this would be the current model estimate which would get iteratively updated after a bounding and maximization step. Table 5.5 summarizes the meaning of the parameters. In the next two sections, we provide the equations to compute them for a given log-sum of the form shown in Equation 5.1 at a current operating or contact point $\tilde{\Theta}$.

5.5.1 Jensen Bounds

Recall the definition of Jensen's inequality: $f(E\{\Box\}) \geq E\{f(\Box)\}$ for concave f. The log-summations in l, l^c, and $\mathcal{L}(X|\Theta)$ all involve a concave $f = \log$ around an expectation, i.e. a *log-sum* or proba-

Parameter	Role
Jensen	
$\tilde{w}_m$	Scalar weight on the virtual data for the m'th model Θ_m
$\tilde{Y}_m$	Virtual data vector computed from the datum $X_m = X$ for the m'th model Θ_m
$\tilde{k}$	An additive constant ensuring the lower bound equals the log-sum at $\Theta = \tilde{\Theta}$
Reverse-Jensen	
w_m	Scalar weight on the virtual data for the m'th model Θ_m
Y_m	Virtual data vector computed from the datum $X_m = X$ for the m'th model Θ_m
k	An additive constant ensuring the upper bound equals the log-sum at $\Theta = \tilde{\Theta}$

Table 5.2: Jensen and Reverse-Jensen Bound Parameters.

bilistic mixture over latent variables. We apply Jensen as follows:

$$\log \sum_m p(m, X|\Theta) \quad \geq \quad \sum_m \left(\frac{p(m, X|\tilde{\Theta})}{\sum_n p(n, X|\tilde{\Theta})} \right) \log \frac{p(m, X|\Theta)}{p(m, X|\tilde{\Theta})} + \log \sum_m p(m, X|\tilde{\Theta}) \tag{5.9}$$

It is traditional to denote the terms in the parentheses above by h_m and refer to them as the *responsibilities* [102]. These can be thought of as the weights each model has for the data point which is shared by each model according to how likely it was generated by it.

$$h_m \quad := \quad \frac{p(m, X|\tilde{\Theta})}{\sum_n p(n, X|\tilde{\Theta})}$$

The Jensen inequality application above can be recast into the form shown in Equation 5.7. This manipulation readily shows how the log-sum intractability is removed. Recall the lower bound we wished to form on the log-sum:

$$\log \sum_m p(m, X|\Theta) \quad \geq \quad \sum_m \tilde{w}_m \log p(\tilde{Y}_m|m, \Theta) + \tilde{k}$$

We expand this form in the e-family notation:

$$\log \sum_m \alpha_m \exp(\mathcal{A}_m(X_m) + X_m^T \Theta_m - \mathcal{K}_m(\Theta_m)) \quad \geq \quad \sum_m \tilde{w}_m \left(\mathcal{A}_m(\tilde{Y}_m) + \tilde{Y}_m^T \Theta_m - \mathcal{K}_m(\Theta_m) \right) + \tilde{k}$$

Here, $\tilde{k}$ is a scalar additive constant, $\tilde{w}_m$ are positive scalar weights on the data and $\tilde{Y}_m$ are the new virtual data vectors (translated from the original X_m). This forms a variational lower bound on the log-sum which makes tangential contact with it at $\tilde{\Theta}$ and is much easier to manipulate. Basically, the log-sum becomes a sum of log-exponential family members. Plugging in the results from Equation 5.9 into the above expanded e-family notation gives us the parameters of the lower bound in exponential family form as:

$$\begin{aligned}
\tilde{k} &= \log p(X|\tilde{\Theta}) - \sum_m \tilde{w}_m (\mathcal{A}(\tilde{Y}_m) + \tilde{Y}_m^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m)) \\
\tilde{Y}_m &= -\frac{1}{\tilde{w}_m} h_m \left(\left. \frac{\partial \mathcal{K}_m(\Theta_m)}{\partial \Theta_m} \right|_{\tilde{\Theta}_m} - X_m \right) + \left. \frac{\partial \mathcal{K}_m(\Theta_m)}{\partial \Theta_m} \right|_{\tilde{\Theta}_m} \quad = \quad X_m \\
\tilde{w}_m &= h_m
\end{aligned}$$

Note the the positive scalar h_m terms (the *responsibilities*) which arise from Jensen's inequality. These quantities are relatively straightforward to compute. We only require *local* evaluations of log-sum values at the current $\tilde{\Theta}$ to compute a *global* lower bound.

If we bound all log-sums in the log-likelihood, we have a lower bound on the objective l which we can maximize easily. Iterating maximization and lower bound computation at the new Θ produces a local maximum of log-likelihood as in EM [8] . However, applying Jensen on log-sums in l^c and $\mathcal{L}(X|\Theta)$ is not as straightforward. Some terms in these expressions involve *negative* log-sums and so Jensen is actually solving for an *upper bound* on those terms. If we want overall lower and upper bounds on l^c and $\mathcal{L}(X|\Theta)$, we need to compute reverse-Jensen bounds.

5.5.2 Reverse-Jensen Bounds

Reversals and converses of Jensen's inequality have been explored in the mathematics and statistics community and are summarized in [150] and [46]. However, these reversals do not have the correct form for direct use in discriminative and conditional latent learning so we have derived our own reversal (which is detailed in Chapter 7). It seems strange we can reverse Jensen (i.e. $f(E\{\square\}) \leq E\{f(\square)\}$) but it is possible. Among other things, we exploit the convexity of the $\mathcal{K}$ functions in the e-family instead of exploiting the concavity of $f = \log$. However, not only does the reverse-bound have to upper-bound the log-sum, it should also have the same form as the Jensen-bound above, i.e. a sum of log-exponential family terms. That way, upper and lower bounds can be combined and used together homogeneously and ML tools can be quickly adapted to the new bounds. The derivation for the reverse-Jensen inequality is long and is deferred to Chapter 7. For now, we simply show how to calculate the inequality's required parameters (w_m, Y_m, k) that will guarantee a global upper bound on the log-sum. Recalling Equation 5.7, we note that we had:

$$\log \sum_m p(m, X|\Theta) \quad \leq \quad \sum_m (-w_m) \log p(Y_m|m,\Theta) + k$$

Expanding out in exponential family form we obtain:

$$\log \sum_m \alpha_m \exp(\mathcal{A}_m(X_m) + X_m^T \Theta_m - \mathcal{K}_m(\Theta_m)) \quad \leq \quad \sum_m -w_m(\mathcal{A}(Y_m) + Y_m^T \Theta_m - \mathcal{K}_m(\Theta_m)) + k$$

Here, we give the parameters of the bound directly, refer to Chapter 7 for their algebraic derivation. This bound again makes tangential contact at $\tilde{\Theta}$ yet is an upper bound on the log-sum[9].

$$\begin{aligned}
k &= \log p(X|\tilde{\Theta}) + \sum_m w_m(\mathcal{A}(Y_m) + Y_m^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m)) \\
Y_m &= \frac{h_m}{w_m}\left(\left.\frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} - X_m \right) + \left.\frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} \\
w'_m &= \min w'_m \text{ such that } \frac{h_m}{w'_m}\left(\left.\frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} - X_m \right) + \left.\frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} \in \frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m} \\
w_m &= 4\, G(h_m/2) \left(X_m - \mathcal{K}'(\tilde{\Theta}_m)\right)^T \mathcal{K}''(\tilde{\Theta}_m)^{-1} \left(X_m - \mathcal{K}'(\tilde{\Theta}_m)\right) \; + \; w'_m
\end{aligned}$$

[8] Other justifications for the convergence of EM include appeals to Kullback-Leibler divergence (1951) [114] and Bregman distances. However, these concepts are newer than Jensen's inequality (1906) [98] which pre-dates them and is sufficient to prove EM's convergence.

[9] We can also find multinomial bounds on α-priors instead of Θ parameters as in Section 5.6.

In the above, we have introduced the function $G(\gamma)$ which is simply:

$$G(\gamma) \;\; = \;\; \begin{cases} \gamma + \frac{1}{4\log(6)} + \frac{25/36}{\log(6)^2} - 1/6 & \gamma \geq 1/6 \\ \frac{1}{4\log(1/\gamma)} + \frac{(\gamma-1)^2}{\log(\gamma)^2} & \gamma \leq 1/6 \end{cases} \tag{5.10}$$

This bound effectively re-weights (w_m) and translates (Y_m) incomplete data to obtain complete data. The w_m are positive weights and we pick the smallest w_m and w'_m which still satisfy the last two simple conditions. We can always set w_m larger than the conditions require but smaller w_m values mean a tighter bound. The first condition requires that the w'_m generate a valid Y_m that lives in the gradient space of the $\mathcal{K}$ functions (a typical e-family constraint) [10]. Thus, from *local* computations of the log-sum's values, gradients and Hessians at the current $\tilde{\Theta}$, we can compute *global* upper bounds.

The Appendix contains a tighter formulation than the one above. The $G(\gamma)$ function is actually an upper bound on a tighter reverse-Jensen solution for w_m which involves numerical table lookups. This is detailed in the derivation of the reverse-Jensen inequality in the next Chapter and in the Appendix. Using these lookup tables provides slightly tighter bounds. Furthermore, in practice, we often compute a tighter version of the w_m by omitting the multiplicative 4 in front of the $G(\gamma)$ function (i.e. make it 1 or less). This still seems to empirically generate reasonable bounds yet these have no analytic guarantees.

In earlier work [94], we derived and used a simpler (yet looser) version of the above bounds which is given by:

$$w_m \;\; = \;\; \left(X_m - \mathcal{K}'(\tilde{\Theta}_m)\right)^T \mathcal{K}''(\tilde{\Theta}_m)^{-1} \left(X_m - \mathcal{K}'(\tilde{\Theta}_m)\right) \; + \; w'_m$$

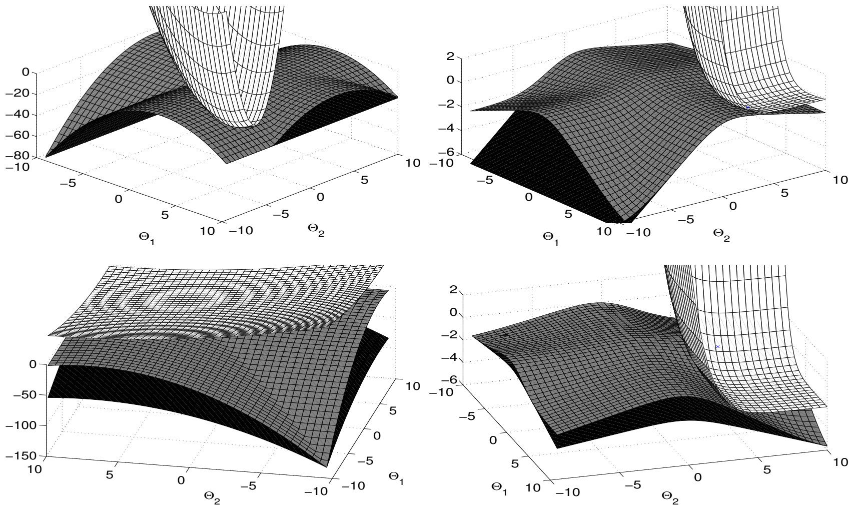
Visualization

In Figure 5.5 we plot the bounds for a two-component unidimensional Gaussian mixture model case and a two component binomial (unidimensional multinomial) mixture model. The Jensen-type bounds as well as the reverse-Jensen bounds are shown at various configurations of $\tilde{\Theta}$ and X. Jensen bounds are usually tighter but this is inevitable due to the intrinsic shape of the log-sum. In addition to viewing many such 2D visualizations, we computed higher dimensional bounds and sampled them extensively, empirically verifying that the reverse-Jensen bound remained above the log-sum.

5.6 Mixing Proportions Bounds

Recall that we had Equation 6.1 to contend with as our mixture of e-family distributions. The α_m were assumed to be constants. The variational bound we solved for is on the Θ model parameters which are allowed to change. However, it may be necessary to vary the α_m as parameters (i.e. if these are free and must be optimized). Thus, we could conceive of bounds on the mixing proportions and use these to estimate optimal mixing coefficients. It is possible to find a reverse-Jensen bound over both α and Θ simultaneously, just as EM and Jensen will handle both jointly. This is done by simply *rewriting* the mixing proportions in their natural exponential family form and seeing that they can be concatenated into the Θ parameters. This development mirrors that proposed in [105] (in the appendix thereof).

[10] In the Gaussian case, the bounds can be made tighter by letting w_m be a full matrix and ignoring the contribution of w'_m.



(a) Gaussian Case (b) Multinomial Case

Figure 5.5: Jensen (black) and reverse-Jensen (white) bounds on the log-sum (gray).

Let us return to our mixture model and note that α will be allowed to vary:

$$p(X|\alpha, \Theta) \;=\; \sum_{m=1}^{M} \alpha_m \exp(\mathcal{A}_m(X_m) + X_m^T \Theta_m - \mathcal{K}_m(\Theta_m))$$

For simplicity, let us define the following:

$$\mathcal{T}_m \;:=\; \exp(\mathcal{A}_m(X_m) + X_m^T \Theta_m - \mathcal{K}_m(\Theta_m))$$

This allows us to simplify the expression for the mixture model:

$$p(X|\alpha, \Theta) \;=\; \sum_{m=1}^{M} \alpha_m \mathcal{T}_m$$

We will now show that $p(X|\alpha, \Theta)$ can be written as a sum of exponential family members whose parameters are in α and are therefore boundable via Jensen and reverse-Jensen. First note that we are dealing with mixing proportions, and thus the α sum to unity, i.e $\sum_m \alpha_m = 1$. Furthermore, consider the change of variable (from the M-dimensional α-space to a natural, compact $M-1$-dimensional η-space):

$$e^{\eta_m} = \frac{\alpha_m}{\alpha_M} \quad \forall m \in [1...(M-1)]$$

We can thus rewrite $p(X|\alpha, \Theta)$ as follows:

$$p(X|\alpha, \Theta) \;=\; \sum_{m=1}^{M} \mathcal{T}_m \alpha_m \;=\; \sum_{m=1}^{M} \mathcal{T}_m \frac{\alpha_m}{\alpha_M} \alpha_M \;=\; \sum_{m=1}^{M} \mathcal{T}_m \exp\left(\log \frac{\alpha_m}{\alpha_M}\right) \alpha_M$$

$$
\begin{aligned}
&= \sum_{m=1}^{M} \mathcal{T}_m \exp\left(\log\frac{\alpha_m}{\alpha_M}\right) \frac{\alpha_M}{\sum_{m=1}^{M}\alpha_m} \\
&= \sum_{m=1}^{M} \mathcal{T}_m \exp\left(\log\frac{\alpha_m}{\alpha_M}\right) \frac{1}{1+\sum_{m=1}^{M-1}\frac{\alpha_m}{\alpha_M}} \\
&= \sum_{m=1}^{M} \mathcal{T}_m \exp\left(\log\frac{\alpha_m}{\alpha_M} - \log\left[1+\sum_{m=1}^{M-1}\frac{\alpha_m}{\alpha_M}\right]\right) \\
&= \sum_{m=1}^{M-1} \mathcal{T}_m \exp\left(\log\frac{\alpha_m}{\alpha_M} - \log\left[1+\sum_{m=1}^{M-1}\frac{\alpha_m}{\alpha_M}\right]\right) + \mathcal{T}_M \exp\left(\log\frac{\alpha_M}{\alpha_M} - \log\left[1+\sum_{m=1}^{M-1}\frac{\alpha_m}{\alpha_M}\right]\right) \\
&= \sum_{m=1}^{M-1} \mathcal{T}_m \exp\left(\eta_m - \log\left[1+\sum_{m=1}^{M-1} e^{\eta_m}\right]\right) + \mathcal{T}_M \exp\left(0 - \log\left[1+\sum_{m=1}^{M-1} e^{\eta_m}\right]\right)
\end{aligned}
$$

Now consider the vector η (which is a reparameterization of α) as an $(M-1)$-tuple. We can rewrite the above as a multinomial e-family member. Recall that for a multinomial we have the following K and A functions[11]:

$$
\begin{aligned}
A(x_m) &= 0 \quad \forall m & (5.11) \\
K(\eta) &= \log\left(1+\sum_{i=1}^{M-1}\exp(\eta_i)\right) & (5.12)
\end{aligned}
$$

This allows us to rewrite the above as:

$$
p(X|\alpha,\Theta) = \sum_{m=1}^{M-1} \mathcal{T}_m \exp\left(\eta_m - K(\eta)\right) + \mathcal{T}_M \exp\left(0 - K(\eta)\right)
$$

In addition, we shall introduce virtual data vectors $\mathbf{x}$ as $(M-1)$-tuples and define them. For each $m \in [1, M-1]$ consider virtual data vectors $\mathbf{x}_m$ which are all-zero except each has a single 1 at dimension m. Furthermore, the vector $\mathbf{x}_M$ is all zeros. Also note that the function $A(\mathbf{x}) = 0$ for a typical multinomial model. We can therefore rewrite the latent likelihood parameterized by α in the following familiar form:

$$
\begin{aligned}
p(X|\alpha,\Theta) &= \sum_{m=1}^{M} \mathcal{T}_m \exp\left(\eta^T\mathbf{x}_m - K(\eta)\right) \\
&= \sum_{m=1}^{M} \mathcal{T}_m \exp\left(A(\mathbf{x}_m) + \eta^T\mathbf{x}_m - K(\eta)\right)
\end{aligned}
$$

Thus we realize that we can rewrite the above as an exponential family form and the same bounding techniques can be applied for both Jensen and reverse-Jensen. The derivations for the Θ-case can be adapted to α which is effectively a multinomial e-family member. Let us now plug back the definition of $\mathcal{T}$ into the above:

$$
\begin{aligned}
p(X|\alpha,\Theta) &= \sum_{m} \exp\left(A(\mathbf{x}_m) + \eta^T\mathbf{x}_m - K(\eta)\right) \mathcal{T}_m \\
&= \sum_{m} \exp\left(A(\mathbf{x}_m) + \eta^T\mathbf{x}_m - K(\eta)\right) \exp(\mathcal{A}_m(X_m) + X_m^T\Theta_m - \mathcal{K}_m(\Theta_m))
\end{aligned}
$$

[11] Here we are omitting the calligraphic font for K and A to differentiate them from the ones in the definition of $\mathcal{T}_m$.

Note that the above product of e-family terms over Θ with the multinomial over η remains in the e-family. We can thus now consider an agglomerative model (i.e. $\bar{\Theta}$ where the bar indicates some aggregated model) that contains the parameters Θ and η which is also in the exponential family and has the general form as in Equation 6.1. However, now the α parameters have been folded into the exponential family distributions over Θ. This makes it possible to jointly upper bound the logarithm of $p(X|\alpha,\Theta)$ over both α and Θ using the reverse-Jensen approach.

Recall that we typically have $X_m = X$ for a mixture model as we argued earlier in the definition of a mixture of exponential families. In the above, we encounter a situation where the data $\mathbf{x}_m$ is varying for each value of m unlike the regular mixture model scenario. This justifies our initial precautionary measure of indexing the data with the latent variable m in the definition of the exponential family.

5.7 The CEM Algorithm

Equipped with Jensen and reverse-Jensen bounds, it is now straightforward to implement a maximum conditional likelihood algorithm or a discriminative learning approach with latent variables. The CEM (Conditional Expectation Maximization) algorithm mirrors the EM algorithm in its approach to maximizing joint likelihood. EM iterates by lower bounding and then maximizing the joint log-likelihood. CEM iterates by lower bounding and then maximizing the conditional log-likelihood. Due to the guarantees behind both the Jensen and reverse-Jensen bounds, CEM converges monotonically to a local maximum of conditional likelihood. It should be noted though that CEM's convergence is slower than EM's since the reverse-Jensen bounds are looser. However, this is an almost inevitable byproduct of the shape of the negated log-sum.

To maximize conditional likelihood (or joint likelihood minus marginal likelihood), we begin with the following type of expression which needs to be maximized over the parameters Θ:

$$\begin{aligned} l^c &= l^j - l^m \\ l^c &= \sum_i \log \sum_m p(m, c_i, X_i|\Theta) - \sum_i \log \sum_m \sum_c p(m, c, X_i|\Theta) \end{aligned}$$

The above is the conditional log-likelihood of a data set with a (simple) mixture model. We lower bound the joint likelihood term with Jensen and upper bound the marginal likelihood term with the reverse-Jensen inequality. This gives us the following overall lower bound on conditional log-likelihood (this step can be called the CE-step of the CEM algorithm):

$$l^c \geq \sum_{mi} h_{mi} \left(\Theta_{mc_i}^T X_i - \mathcal{K}(\Theta_{mc_i})\right) - \sum_{mci} (-w_{mci}) \left(\Theta_{mc}^T Y_{mci} - \mathcal{K}(\Theta_{mc})\right) \; + \; \textit{constant terms}$$

It is then straightforward to maximize the right hand side by taking derivatives and setting to zero (this is the M-step):

$$\frac{\partial \mathcal{K}(\Theta_{mc})}{\partial \Theta_{mc}} \;=\; \frac{1}{\sum_i \delta(c_i, c) h_{mi} + w_{mci}} \left(\sum_i h_{mi} \delta(c_i, c) X_i + w_{mci} Y_{mci} \right) \quad \forall m, c$$

The above M-step has a unique solution due to the convexity of the $\mathcal{K}$ cumulant generating functions. In fact, the above step corresponds to merely maximizing a *non-latent* exponential family distribution where the data has been *weighted* (through the h_{mi} and w_{mci} scalar terms) and has also been *translated* since the Y_{mci} are translated versions of the original data.

Visualization

As we mentioned earlier, CEM involves weighted and translated virtual data in the M-steps. EM only involves weighted terms in the M-step since it only employs the Jensen inequality. The use of the reverse-Jensen inequality applies to the negated marginal likelihood which provides a repulsion term through what is often called the background probability. Thus, CEM bounds the negated marginal likelihood and its *negated* repulsion forces by translating the data to another position and then treating it as an attractive force (as in a standard maximum likelihood setting).

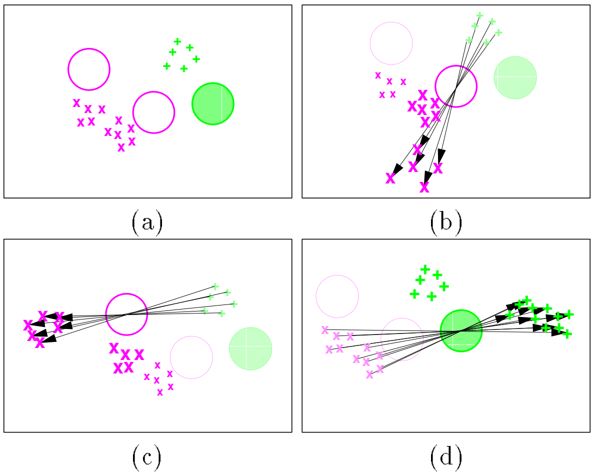


Figure 5.6: CEM Data Weighting and Translation. Figure (a) depicts the incomplete data and the models that describe it. Figures (b) and (c) depicts the complete data as seen by one of the 'x' models where the data is weighted and the other class' data is translated. Similarly, (c) depicts the what the model for the '+' class sees.

An interesting analogy can be made with the translated and weighted data which is depicted in Figure 5.6. Here, we show a mixture model where 2 (light) Gaussians are assigned to the 'x' class and 1 (dark) Gaussian is assigned to the '+' class. Basically, CEM re-weights the data if it is in the model's class but if it is in another class, the data gets translated and re-weighted. The translation involves adding a scaled gradient vector $\mathcal{K}'(\tilde{\Theta}_{mc})$ to the data. In the Gaussian case, the translation effectively moves the data point *through the mean* of the model and puts it on the other side. Therefore, instead of *repelling* from the negative data (or the incorrect class), a model gravitates towards it after it has been translated to the other side. The data and models are seen in Figure 5.6(a). Figure 5.6(b) shows how one of the 'x' Gaussians sees 'x' data nearby it with high weight and 'x' data far away with less weight (as in EM). However, the other class data (the '+' points) are seen translated to the other side of the model with a weight of their own. Thus, the model will effectively repel away from them. Similarly, Figure 5.6(c) depicts the weighting for the other 'x' model which has different weights on the correct data and has different virtual data from the repulsion of the other class. Finally, Figure 5.6(d) depicts the lonely '+' model which gets all the '+' data with equal weight as well as a repulsion term from the 'x' data.

Since Gaussians are an exponential family model that is *self-dual*, their data vectors and their parameter vectors lie in the same space (i.e. the gradient space of $1/2\Theta^T\Theta$ is Θ). Therefore, the analogy of translating *through* the means of the Gaussian can be made here. In other distributions, the analogy does not maintain the same geometric interpretation but can be made more loosely at a higher level through the gradients of the cumulant-generating function.

Experiments

We now show some experiments that compare CEM with EM (and therefore pit conditional likelihood against joint likelihood). In these experiments, we used the straightforward reverse-Jensen bounds that we have described in the previous section. It is quite possible that alternative schemes (which will be described later) such as annealing (see Section 5.7.1) and the so-called data-set bound (see Section 6.7) could be helpful in obtaining faster convergence or quasi-global optima and this remains to be explored.

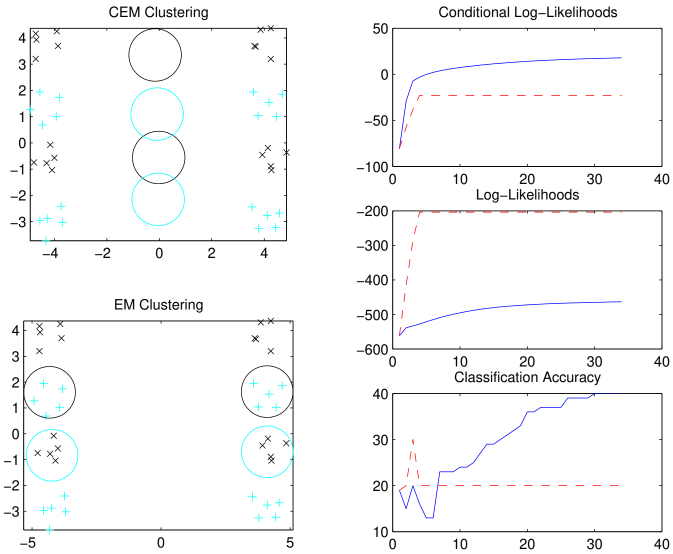


Figure 5.7: CEM vs. EM Performance on a Gaussian Mixture Model. The clustering CEM computes as well as the one EM finds are shown on the left. The plots depict conditional likelihood, likelihood and classification accuracy on the right. CEM's performance is shown as a solid blue line while EM's performance is shown as a dashed red line.

In Figure 5.7 we depict the toy problem we initially posed. The model being used is a mixture of 2 Gaussians per class where the mixing proportions are equal and the Gaussians have identity covariance. Here, both EM and CEM are initialized with the same configuration. Both EM and CEM converge monotonically for their respective objective functions We note that EM quickly converges to a maximum likelihood solution and CEM takes a few more iterations to converge to the maximum conditional likelihood solution. However, in this problem, maximum likelihood is a very poor criterion and the resulting *classifier* that EM generates has a classification accuracy of 50% (random chance). Meanwhile, CEM produces a classifier that has 100% accuracy.

We compare that above performance with that of (batch) gradient ascent in Figure 5.8. Here, we applied gradient ascent to both the maximum likelihood problem and the maximum conditional likelihood problem. Therefore, we are not using bounds. While gradient ascent seems to converge nicely for the maximum likelihood problem, it gets stuck in local minima when applied to the maximum conditional likelihood problem. Thus, the reverse-Jensen bounds can have advantages over a purely local optimization technique like gradient ascent.

In another evaluation of CEM an EM, we used a standardized UCI data set, the Yeast data set. Here, there are 9 classes of yeast that are to be classified based on continuous features. The inputs

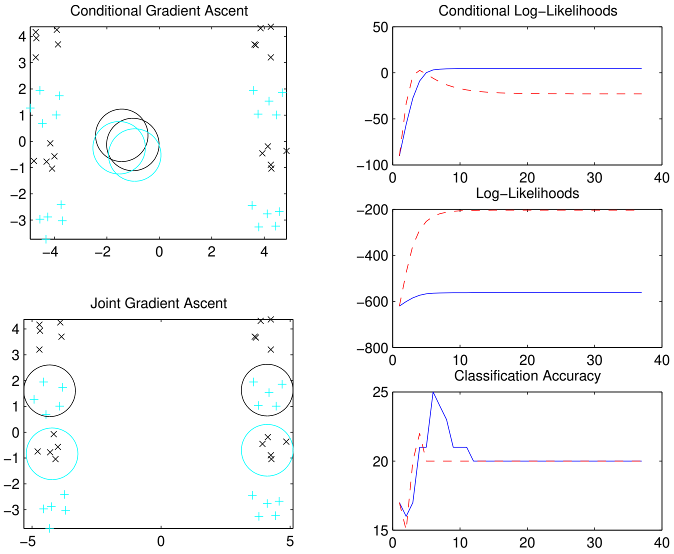


Figure 5.8: Gradient Ascent on Conditional and Joint Likelihood. Here, we are optimizing the same Gaussian Mixture Model. Gradient ascent seems to work on joint likelihood but gets stuck on the conditional. The plots depict conditional likelihood, likelihood and classification accuracy on the right. Gradient ascent on conditional likelihood is shown as a solid blue line while ascent on joint likelihoods is shown as a dashed red line.

were scaled and translated to have zero mean and identity covariance. We assumed a rather poor model: an equal mixture of 2 Gaussians per class. Each Gaussian has identity covariance which restricts the power of the model considerably. Figure 5.9 depicts the resulting performance of EM and CEM. Although CEM takes a long time to converge, it provides a better conditional likelihood score as well as better classification accuracy. The training data included 600 exemplars while the test data included 884. Table 5.7 summarizes the results which suggests that CEM is better suited for classification tasks (here, random guessing would produce an accuracy of about 10%).

Training	Log-Likelihood	Conditional Log-Likelihood	Accuracy
EM	-5946	444.1	58.3%
CEM	-7404	859.0	67.2%
Testing	Log-Likelihood	Conditional Log-Likelihood	Accuracy
EM	-9210	424.4	51.2%
CEM	-11121	835.4	54.0%

Table 5.3: CEM & EM Performance for Yeast Data Set.

5.7.1 Deterministic Annealing

In the EM algorithm, it is known that local minima problems can be avoided if one uses a technique called *Deterministic Annealing* [194] which effectively replaces the current models for each

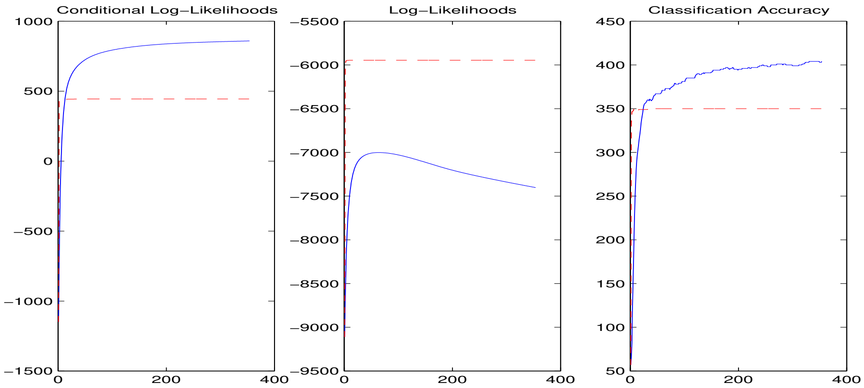


Figure 5.9: CEM vs. EM Performance on the Yeast UCI Dataset. The plots depict conditional likelihood, likelihood and classification accuracy on the right. CEM's performance is shown as a solid blue line while EM's performance is shown as a dashed red line.

exponential family distribution with a temperature-scaled version as follows:

$$\exp(\ldots) \rightarrow \exp\left(\frac{1}{Temp} \times (\ldots)\right)$$

Effectively, Jensen's inequality is softened by bounding a different pdf which has smoother properties. A similar technique is also be feasible with the reverse-Jensen inequality. This gives a deterministically annealed type of a maximum conditional likelihood algorithm, i.e. annealed CEM. Annealing is the simple operation of replacing every exponentiation operation with a softened exponentiation where we divide by a scalar $Temp$ temperature value. We follow a standard annealing schedule where this temperature value starts off large and is slowly decremented until the temperature goes to unity. This typically softens the bounds and should improve convergence to obtain quasi-global optima.

5.8 Latent Maximum Entropy Discrimination

The MED framework can also directly benefit from the reverse-Jensen bounds and it is straightforward to see how these permit it to handle mixture models. Furthermore, discrimination is much better suited for classification and regression than conditional likelihood. We begin by placing mixtures of the e-family in the MED discriminant function:

$$\mathcal{L}(X;\Theta) \quad = \quad \log \frac{\sum_m P(m, X|\Theta_+)}{\sum_m P(m, X|\Theta_-)} + b$$

Recall that the constraints to be satisfied involve expectations over the discriminant function as follows:

$$\int P(\Theta, \gamma_t)\,[y_t \mathcal{L}(X_t;\Theta) - \gamma_t] \quad \geq \quad 0$$

If we expand the above constraints, it becomes clear that the integrals they produce are intractable due to the presence of the summation over the hidden variable. Therefore, we will not have an analytic partition function and objective to optimize. The solution is to invoke Jensen and reverse-Jensen bounds as follows (here, without loss of generality, we assume that $y_t = 1$, if $y_t = -1$, we swap the application of the Jensen and reverse-Jensen bound):

$$\int P(\Theta, \gamma_t) \left[y_t \log \left(\sum_m P(m, X_t|\Theta_+) \right) - y_t \log \left(\sum_m P(m, X_t|\Theta_-) \right) + b - \gamma_t \right] \geq 0$$

$$\int P(\Theta, \gamma_t) \left[\log \left(\exp(\tilde{k}) \Pi_m p(\tilde{Y}_m|m, \Theta_+)^{\tilde{w}_m} \right) - \log \left(\exp(k) \Pi_m p(Y_m|m, \Theta_-)^{-w_m} \right) + b - \gamma_t \right] \geq 0$$

Above, we have lower bounded the left hand side. If we satisfy the 'greater than zero' constraint with the lower bound we have just created (using Jensen and reverse-Jensen), then we must automatically satisfy it for the original quantity (the true expectation constraint). Therefore, we have introduced bounds that give stricter constraints in the MED framework to avoid the intractabilities. The logarithm operator no longer acts on a summation and the integrals can be computed analytically (provided that the $P(\Theta, \gamma)$ is in a conjugate form to the discriminant function's probability model).

Effectively, we have replaced the discriminant function $\mathcal{L}(X_t; \Theta)$ with a lower bound on it whenever $y_t = 1$, and an upper bound whenever $y_t = -1$. Therefore we can now propose an iterative MED algorithm that is locally optimal (but not unique). We assume that we start with an estimated mode of the model $P(\Theta, \gamma)$ which we will denote Θ_t:

- Step 1: Each data point's discriminant function is bounded individually using the Jensen and reverse-Jensen bounds. This is done at the current estimated mode of the model $P(\Theta, \gamma)$, i.e. a given Θ_t.
- Step 2: We solve the MED optimization using the bounds and obtain the current Lagrange multipliers. These new Lagrange multipliers λ_{t+1} are then used to compute $P(\Theta, \gamma)$ and again generate a new estimated mode Θ_{t+1}.

The above steps are iterated until convergence. In practice, the above variational bound will become more accurate as the MED solution distribution $P(\Theta)$ becomes more peaked. This is typically the case as we converge to a final solution and the Lagrange multipliers in the objective function $J(\lambda)$ settle to their locally optimal configuration.

The latent MED technique discussed above has an interesting geometric interpretation. Figure 5.10 depicts the process where we interleave the Jensen and reverse-Jensen bounds with an iterated MED projection calculation. Since it is impossible to do the projection tractably with the full latent model, we cannot find the closest distribution from the MED prior $P_0(\Theta)$ to the true admissible set $\mathcal{P}$ and its large (and complicated) convex hull. Instead, we pick an arbitrary point $P_t(\Theta)$ (typically the EM algorithm's maximum likelihood estimate is used to initialize this posterior distribution). We then find a more constrained convex hull which is formed by invoking the Jensen and reverse-Jensen bounds at the mode of the current $P_t(\Theta)$, namely $\hat{\Theta}_t$. This smaller convex hull admits a closed-form solution since it only involves e-family discriminant functions. Thus, we can find the closest point to the prior by a direct MED projection which yields $P_{t+1}(\Theta)$. This process is iterated and the small convex hull (which lies within the large original hull, i.e. the admissible set), is slowly updated until we reach a local minimum where the MED projection no longer modifies the solution $P_{t++}(\Theta)$.

5.8.1 Experiments

To compare the latent MED against the standard EM framework, we employed a simple mixture of Gaussians model. The means are permitted to vary while the mixing proportions are held fixed

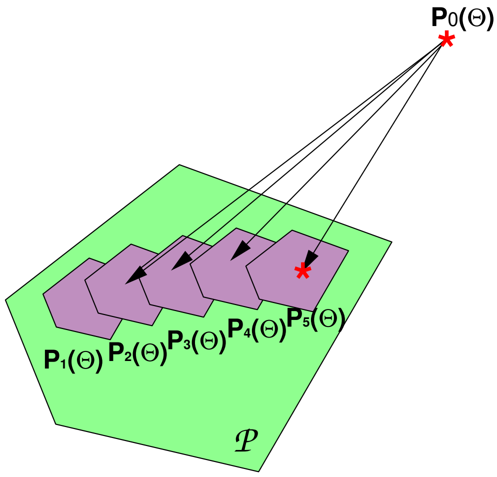


Figure 5.10: Iterated Latent MED Projection with Jensen and Reverse-Jensen Bounds. Direct projection to the admissible set $\mathcal{P}$ would give rise to an intractable MED solution due to the mixture of e-family discriminant function. Instead, a stricter convex hull within the admissible set is found using the Jensen and reverse-Jensen inequalities which give rise to a simple e-family discriminant function and therefore permit closed form projection. The process is iterated until we converge to a locally optimal point that is as close to the prior as possible while remaining in the admissible set.

and the covariances are locked at identity. The data set used was the Pima Indians Diabetes data set (available from the UCI repository). The input forms a 7-dimensional feature space while the output is a binary class. Training was performed on 200 input points while testing was performed on the remaining 332 points. Table 5.8.1 depicts the performance of EM and the latent MED technique (using the reverse-Jensen bounds in a CEM-type of iterative loop).

	Training Accuracy	Testing Accuracy
EM - 2 Gaussian Mixture	73 %	70%
EM - 3 Gaussian Mixture	71 %	72%
EM - 4 Gaussian Mixture	68 %	67%
MED - 2 Gaussian Mixture	74 %	79%
MED - 3 Gaussian Mixture	78 %	78%
MED - 4 Gaussian Mixture	77 %	77%

Table 5.4: EM and latent MED Performance on the Pima Indians Data Set.

In Table 5.8.1 we also present the results with a standard support vector machine and note that the latent MED performance achieves comparable to the polynomial kernel based SVM, or equivalently a non-latent MED with a polynomial kernel. While the latent MED solution used the same generative model as the EM algorithm (a mixture of Gaussians), the discriminative aspect of the estimation in MED provides better classification performance. In all the SVM and the MED experiments, the regularization constant c was set to 10. Furthermore, in the SVM experiments, the input space was scaled such that it remained within the unit cube for normalization reasons (while the EM and latent MED implementations operated on the raw original data).

	Training Accuracy	Testing Accuracy
SVM - 1st Order	76 %	79%
SVM - 2nd Order	77 %	80%
SVM - 3rd Order	79 %	78%
SVM - 4th Order	84 %	76%

Table 5.5: SVM with Polynomial Kernel Performance on the Pima Indians Data Set.

5.9 Beyond Simple Mixtures

It is clear that the mixture of exponential family distributions we have specified in this chapter does not capture all latent model situations that are typical in machine learning. For example, it is difficult to represent the latencies that would arise in a structured graphical model with this flat mixture. Chapter 6 expands on the mixture model we have seen such that it can encompass latent Bayesian networks and structured mixture model situations. This will permit us to consider structures such as hidden Markov models. The Jensen and reverse-Jensen inequalities will be reiterated for such models and we shall show efficient algorithms for computing the bounds' parameters in polynomial time. Subsequently, Chapter 7 goes into the derivation details of the reverse-Jensen inequality which was put forward without proof in this chapter. Therein we show the proof for the case of structured mixtures as in Chapter 6 since it subsumes the case of flat mixtures in this chapter.

Chapter 6

Structured Mixture Models

In the previous chapter, we addressed the problem of discrimination with a standard flat mixture model. The intractabilities that resulted from this model were avoided by utilizing upper and lower bounds that mapped the mixture of exponential family model into a standard exponential family form permitting monotonically convergent maximum conditional likelihood and iterative latent MED applications. However, the mixture models we described earlier were limited and cannot span the full spectrum of latent models such as latent Bayesian networks, hidden Markov models and so forth. Additional flexibility is required in the mixture model such that it can accommodate these so-called structured models. A structured model, as depicted in Figure 6.1 differs from a flat mixture model in that the latent variables are not a simple parent of the observable variables. Instead, the latent variables and the observables may have other dependencies, such as a Markov structure and so forth. From a mathematical point of view, the flat mixture model we previously considered, only had parameters that were independent across each element in the summation (within the log-sum). In many latent model situations, particularly when we are dealing with a structured graphical model, the elements in the mixture will have tied parameter models.

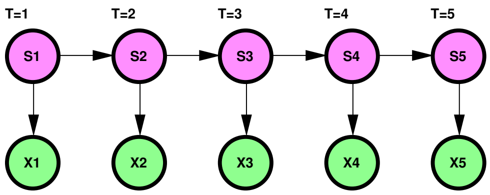


Figure 6.1: Graph of a Structured Mixture Model (HMM)

In this chapter, we will begin by motivating a more complicated class of mixtures of the exponential family that goes beyond Equation 5.1. This is done by noting that a hidden Markov model cannot be described in the mixture model framework we previously encountered and subsequently proposing a more appropriate *structured mixture* or, alternatively, what we call a *mixture of mixtures*. The Jensen and Reverse-Jensen inequalities are then explicated for this case and effectively map the latent model back to a standard exponential family form. We then go into details on the computation of the reverse-Jensen inequality for a hidden Markov model as well resolve important issues of efficiency (i.e. via dynamic programming and other efficient algorithms). Then, we show some illustrations of the CEM algorithm applied to an HMM. Further results on CEM with HMMs are elaborated in Chapter 8. We then discuss the case of latent Bayesian networks in general. Finally, as an interesting academic exercise (which may be skipped by the reader) we show how summation or data set of log mixture models can be mapped into a single structured mixture model.

6.1 Hidden Markov Models

Consider the case above where we are dealing with a hidden Markov model as in Figure 6.1. This model and any many latent Bayesian networks like it can be seen as a mixture of exponential families. An important result [34] is that tree structured dependencies between variables who are themselves in the exponential family form an aggregate exponential-family distribution. Therefore, we can extend the Jensen bound and reverse-Jensen bounds to Bayesian networks or directed acyclic graphs (DAGs) which have a general tree-structure. However, it remains crucial to map the probability distribution of the HMM (or another latent Bayesian network) into a mixture of naturally parameterized exponential family distributions to fully take advantage of this property and to apply the reverse-Jensen bounds (just as it was crucial to work with natural parameterizations to clearly see the bounds on flat mixtures in the previous chapter).

A hidden Markov model [1] or doubly-stochastic-automaton is an interesting mixture of exponential family distributions and is another important 'client model' for the reverse-Jensen bounds we have proposed for discriminative learning. An HMM typically involves a sequence X of T output vectors $(o_1, ..., o_T)$ living in a D-dimensional space. A M-state Markov model has state vectors $(s_1, ..., s_T)$ which identify for each $t \in [1..T]$ which state $m \in [1..M]$ the model is in. It can generally be described by the following pdf (if we assume we know S, i.e. the states are not hidden):

$$p(X, S) = p(s_1)p(o_1|s_1)\Pi_{t=2}^{T} p(s_t|s_{t-1})p(o_t|s_t)$$

The choices for the component distributions (i.e. the transition distributions $p(s_t|s_{t-1})$ and emission distributions $p(o_t|s_t)$) are either multinomials, Gaussians or another member of the exponential family. Selecting a multinomial model for $p(s_t|s_{t-1})$ gives us a stochastic finite state automaton (an HMM). Selecting a Gaussian model for $p(s_t|s_{t-1})$ generates a linear dynamic system (an LDS) or a Kalman filter. If the output emission $p(o_t|s_t)$ distribution is chosen to be Gaussian, we expect continuous vector outputs from the automaton. If the output distribution is multinomial, the automaton generates discrete symbols. Either way, as long as the component distributions are in the exponential family, any product of the exponential distributions remains in the exponential family as well.

Since we deal with hidden Markov models where S is not observed (it is a latent or hidden variable) a marginalization is involved. Thus, we sum over all possible settings of the S (i.e. a total of about M^T configurations). This summation need not be inefficient, though, and can be computed via recursion and tree-based algorithms. This is represented as follows:

$$p(X) = \sum_S p(X, S) = \sum_{s_1=1}^{M} \cdots \sum_{s_T=1}^{M} p(X, S)$$

This is the likelihood of data under a hidden Markov model. It is well known that EM and Jensen can generate a lower bound on the probability distribution above which has a simple (non-latent) e-family form. To perform maximum conditional likelihood or discrimination, though, we need to

[1] The HMM can be taken more generally as a hidden state machine (which we can take noisy measurements of) evolving with Markovian dynamics. Therefore, a linear dynamical system (LDS) can be treated in a similar way.

upper bound the log-likelihood. First, we shall begin by expanding the above:

$$p(X) = \sum_S p(X,S) = \sum_{s_1=1}^{M} \cdots \sum_{s_T=1}^{M} p(s_1)p(X_1|s_1)\Pi_{t=2}^{T} p(s_t|s_{t-1})p(X_t|s_t)$$

At this point, we will be more specific for the sake of clarity and define the actual distributions of the HMM. The following development doesn't cause a loss of generality. We shall assume that we are dealing with an HMM with Gaussian emissions (where the means of each Gaussian are to be estimated while the covariances are locked at identity) and we have multinomial models to describe the transition matrix from hidden state to hidden state (unlike a Kalman filter which would require a Gaussian on the sate transition probabilities). In the above, the state transition probability is typically given as a multinomial (or transition matrix). This state transition can be expressed in the following standard form or in its natural exponential family form:

$$\begin{aligned} p(s_t|s_{t-1}=m) &= \Pi_{i=1}^{M}((\alpha_m)_i)^{(s_t)_i} \\ &= \exp\left\{A(\mathbf{x}_t) + \eta_m^T \mathbf{x}_t - K(\eta_m)\right\} \end{aligned}$$

As before, we will use the multinomial in its exponential family form to compute the reverse-Jensen bound. Therein, the $\mathbf{x}_t$ is a vector of length $M-1$ which is all zero and contains a '1' at the $k'th$ index value if the current state is $s_t = k$. If the current state is $s_t = M$, then the $\mathbf{x}$ is all zeros. Once again, the η_m is related to the α_m multinomial model through a simple transformation. The $A()$ function and the $K()$ function correspond to the standard ones for the multinomial as given in Table 5.1. To be more specific, we write the transition probabilities as follows where the η_m and the $\mathbf{x}$ values are indexed by the appropriate s_{t-1} and s_t state-labels respectively:

$$p(s_t|s_{t-1}=m) = \exp\left\{A(\mathbf{x}(s_t)) + \eta(s_{t-1})^T \mathbf{x}(s_t) - K(\eta(s_{t-1}))\right\}$$

The Gaussian in exponential family form is straightforward and gives the emission probability:

$$\begin{aligned} p(X_t|s_t=m) &= \exp\left\{\mathcal{A}(X_t) + \theta_m^T X_t - \mathcal{K}(\theta_m)\right\} \\ &= \exp\left\{\mathcal{A}(X_t) + \theta(s_t)^T X_t - \mathcal{K}(\theta(s_t))\right\} \end{aligned}$$

For simplicity, we can assume the prior over initial states $p(s_1)$ is fixed and equal for all states as $p(s_1) = \frac{1}{M}$. Thus, at this point, we can write the HMM's likelihood as a mixture of exponential family distributions as follows:

$$\begin{aligned} p(X) = \sum_{s_1=1}^{M} \cdots \sum_{s_T=1}^{M} \frac{1}{M} \exp\Bigg(& \sum_{t=1}^{T} \mathcal{A}(X_t) + \theta(s_t)^T X_t - \mathcal{K}(\theta(s_t)) \\ & + \sum_{t=2}^{T} A(\mathbf{x}(s_t)) + \eta(s_{t-1})^T \mathbf{x}(s_t) - K(\eta(s_{t-1}))\Bigg) \end{aligned}$$

The above mixture cannot be cast in the form introduced in Chapter 5 (in Equation 5.1) since each of the M elements in the mixture there had its own exponential family model, Θ_m. The above mixture has approximately M^T components yet only $2M$ models (M Gaussian mean parameters, θ_m, and M multinomial transition parameters η_m). Therefore, we need to have a mixture model form which involves summing many e-family distributions where the models can be replicated while the data varies. In other words, consider the form below where we have indexed data vectors X with both m and n and sum over $m = 1..M$ as well as $n = 1..N$.

$$p(X|\Theta) = \sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \Theta_m - \mathcal{K}_m(\Theta_m)) \quad (6.1)$$

Equation 6.1 does appear to be a strange mixture model indeed. The latent variables (i.e. the indexes m and n) no longer have the simple intuitive statistical meaning that the latencies in Equation 5.1 carried. However, the *mathematical form* is the important generalization and we now have the flexibility to describe the probability distribution of an HMM (or other latent Bayesian networks) using the notation in Equation 6.1. Here, we can reuse a given exponential family model Θ_m in different components in the mixture while varying the data that it interacts with (namely the X_{mn}). The X_{mn} are various vectors of the same cardinality as Θ_m which can be explicitly computed from the original observations, say X and the index variables m and n. For example, we may think of them as the result of an arbitrary function that operates on X, i.e. $X_{mn} = f_{mn}(X)$. The form above will be called a mixture of mixtures. It is clear that the setting of $N = 1$ will make the above mixture model identical to the original one in Equation 5.1 in Chapter 5. We will now go into details of the above form and derive the parameters for the corresponding Jensen and reverse-Jensen inequalities. Subsequently, we will explicitly put the HMM in the form of Equation 6.1 and show how these parameters can be obtained tractably and efficiently by taking advantage of the independency structure of the HMM's directed graphical model.

6.2 Mixture of Mixtures

We have motivated the use of the double mixture or mixture of mixtures. We will see that the more elaborate mixture can be bounded with the same form of upper and lower bounds as the flat mixture model in Chapter 5. We will compute the same (or analogous) parameters for the bounds as derived previously and obtain similar terms such as $\tilde{w}_m, \tilde{Y}_m, \tilde{k}$ for the Jensen bounds as well as w_m, Y_m, k for the reverse-Jensen bounds. The actual computation of the bounds is slightly different than the one shown in Chapter 5 and is a natural generalization of the previous formulas (which are exactly the same as those for the mixture of mixtures under the setting of $N = 1$). Once again, we will work in log-space and consider bounds on this (more elaborate) log-sum:

$$\log p(X|\Theta) = \log \left(\sum_{m=1}^{M} \sum_{n=1}^{N} \alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \Theta_m - \mathcal{K}_m(\Theta_m)) \right) \quad (6.2)$$

6.2.1 Jensen Bounds

Applying Jensen to the log mixture of mixtures $\log p(X|\Theta)$ *lower bounds* it with the same function form as in Chapter 5 except with different parameter definitions:

$$\log \sum_m \sum_n \alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \Theta_m - \mathcal{K}_m(\Theta_m)) \geq \sum_m \tilde{w}_m(\mathcal{A}(\tilde{Y}_m) + Y_m^T \Theta_m - \mathcal{K}_m(\Theta_m)) + \tilde{k}$$

The right hand term is the familiar $Q(\Theta, \tilde{\Theta})$ function that is derived in the EM algorithm. This bound is not static, it is a variational lower bound which is always below the original function yet makes tangential contact with it at a specified current configuration of $\tilde{\Theta}$. For a given $\tilde{\Theta}$, we again have to find settings for the parameters of the bound $(\tilde{w}_m, \tilde{Y}_m, \tilde{k})$. Here, $\tilde{k}$ is a scalar, $\tilde{w}_m$ are positive scalar weights on the data and $\tilde{Y}_m$ are virtual data vectors (i.e. translated mixtures of the original data points):

$$\begin{aligned}
\tilde{k} &= \log p(X|\tilde{\Theta}) - \sum_m \tilde{w}_m(\mathcal{A}(\tilde{Y}_m) + \tilde{Y}_m^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m)) \\
\tilde{Y}_m &= -\frac{1}{\tilde{w}_m}\sum_n h_{mn}\left(\left.\frac{\partial \mathcal{K}_m(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} - X_{mn}\right) + \left.\frac{\partial \mathcal{K}_m(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} = \sum_n \frac{h_{mn}}{\sum_n h_{mn}} X_{mn} \\
w_m &= \sum_n h_{mn}
\end{aligned}$$

For convenience, the above uses h_{mn} which are referred to as *responsibilities*. These are positive scalars defined as follows:

$$h_{mn} = \frac{\alpha_{mn}\exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m))}{\sum_m \sum_n \alpha_{mn}\exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m))} \tag{6.3}$$

We obtain the parameters for $\tilde{k}$ and $\tilde{Y}_m$ by making sure that the bound is equal to the original function $p(X|\Theta)$ at $\tilde{\Theta}$ and both their gradients are equal as well. The formula for $\tilde{w}_m$ naturally results from using Jensen's inequality subsequently. It is clear that from only *local* calculations, we obtain a *global lower bound* on $\log p(X|\Theta)$.

6.2.2 Reverse Jensen Bounds

Now, if we wish to *upper bound* the log-sum instead, we use *reverse Jensen*:

$$\log \sum_m \sum_n \alpha_{mn}\exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \Theta_m - \mathcal{K}_m(\Theta_m)) \leq \sum_m (-w_m)(\mathcal{A}_m(Y_m) + Y_m^T \Theta_m - \mathcal{K}_m(\Theta_m)) + k$$

In this case the parameters are:

$$\begin{aligned}
k &= \log p(X|\tilde{\Theta}) + \sum_m w_m(\mathcal{A}_m(Y_m) + Y_m^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m)) \\
Y_m &= \frac{1}{w_m}\sum_n h_{mn}\left(\left.\frac{\partial \mathcal{K}_m(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} - X_{mn}\right) + \left.\frac{\partial \mathcal{K}_m(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} \\
w'_m &= \min w'_m \text{ such that } \frac{1}{w'_m}\sum_n h_{mn}\left(\left.\frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} - X_{mn}\right) + \left.\frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} \in \frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m} \\
w_m &= 4\ G\left(\frac{\sum_n h_{mn}\mathcal{Z}_{mn}^T \mathcal{Z}_{mn}}{2\max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}}\right) \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} + w'_m \\
&\qquad \text{where}\, \mathcal{Z}_{mn} = \mathcal{K}''(\tilde{\Theta}_m)^{-1/2}(X_{mn} - \mathcal{K}'(\tilde{\Theta}_m))
\end{aligned}$$

Once again, the function $G(\gamma)$ is given by Equation 5.10. This bound effectively re-weights (w_m) and translates (Y_m) incomplete data to obtain complete data. The w_m are positive weights and we pick the smallest w_m and w'_m which still satisfy the last two simple conditions. We can always set w_m larger than the conditions require but smaller w_m values mean a tighter bound. Furthermore, increasing the values of the terms $\max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}$ or increasing $\sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}$ will also yield guaranteed yet more conservative (looser) bounds. This may be necessary if these terms are too complicated to compute exactly while upper bounds on them may be efficient to estimate.

The first condition requires that the w'_m generate a valid Y_m that lives in the gradient space of the $\mathcal{K}$ functions (a typical e-family constraint) [2]. Thus, from *local* computations of the log-sum's values, gradients and Hessians at the current $\tilde{\Theta}$, we can compute *global* upper bounds.

We recognize that the recipes for Y_m and k here are quite similar to those of the Jensen bound except that we have solved for different w_m values to generate the dual inequality. The similarity is due to the fact that *both* bounds are variational and make tangential contact at $\tilde{\Theta}$. Thus, their values and gradients are equal to the original function $\log P(X|\Theta)$ at $\tilde{\Theta}$ hence the natural coupling of the linear parameters Y_m and k. It is clear that from only *local* calculations, we obtain a *global upper bound* on $\log p(X|\Theta)$.

In fact, the reverse-Jensen inequality's computational effort (and tightness) hinges on our ability to compute w_m (through summations and maximizations over the transformed data). This is because the Y_m is given by the following closed form formula thereafter (the scalar k parameter is trivial and usually irrelevant). The simple formula for Y_m that *does not require any summations over the data* and is given by a straightforward manipulation of the definition for Y_m in the reverse-Jensen inequality and the definition for $\tilde{Y}_m$ in the traditional Jensen inequality:

$$Y_m = \left(\frac{\tilde{w}_m}{w_m} + 1\right) \mathcal{K}'(\tilde{\Theta}_m) - \frac{\tilde{w}_m}{w_m} \tilde{Y}_m$$

We can also use this definition to quickly isolate the requirements on the positive scalar w'_m which guarantees that the virtual data Y_m remains in the gradient space of the cumulant generating function. For example, in the case of the Gaussian mean, w'_m is 0 since we have no constraints on the gradient space: $\mathcal{K}(\Theta) = 1/2\theta^T\theta$ which spans all gradients. In the case of the multinomial, the cumulant generating function is $\mathcal{K}(\Theta) = \eta \log(1 + \sum_i \exp(\theta_i))$. Therefore, each of the Y_m vector's elements is restricted to $[0, \eta)$ and the sum of the elements of Y_m is also restricted to the range $[0, \eta)$.

The Appendix contains a tighter formulation than the one above. The $G(\gamma)$ function is actually an upper bound on a tighter reverse-Jensen solution for w_m which involves numerical table lookups (this is detailed in the derivation of the reverse-Jensen inequality in the next Chapter and in the Appendix). Furthermore, in practice, we omit the multiplicative 4 (i.e. make it 1 or less) in the definition for w_m which scales the $G(\gamma)$ function and still obtain very reasonable bounds (which we cannot analytically guarantee, however).

We can also employ a simpler (yet looser) version of the above bounds as described in [94] as follows:

$$w_m = \max_n \left(X_{mn} - \mathcal{K}'(\tilde{\Theta}_m)\right)^T \mathcal{K}''(\tilde{\Theta}_m)^{-1} \left(X_{mn} - \mathcal{K}'(\tilde{\Theta}_m)\right) + w'_m \tag{6.4}$$

The reverse-Jensen inequality as well as the above looser bound are both derived thoroughly in Chapter 7. The above simpler bound is based on a sloppy curvature check. Although curvature constraints are easy ways to construct bounds, they may be too conservative and generate loose bounds. It is best to avoid curvature checking in bound derivations or to defer it to only when absolutely necessary. In the following, we outline some heuristics to avoiding the full computation of w_m which may be crucial when we are dealing with latent models that are, for example, structured and do not permit easy computation of the sum and max of the inner-products $\mathcal{Z}_{mn}^T \mathcal{Z}_{mn}$.

[2] In the Gaussian case, the bounds can be made tighter by letting w_m be a full matrix and ignoring the contribution of w'_m.

Simplifying Heuristics

There are many heuristics to avoid the extra computations involved in obtaining the w_m parameters for the reverse-Jensen inequality. One way is to avoid the maximization over the data which may be cumbersome (i.e. in hidden Markov models where the max is over an exponential number of data configurations). For example, we can use the the following simple (yet tighter) bound which is *no longer globally guaranteed* but only locally guaranteed:

$$w_m = \sum_n h_{mn} \left(X_{mn} - \mathcal{K}'(\tilde{\Theta}_m)\right)^T \mathcal{K}''(\tilde{\Theta}_m)^{-1} \left(X_{mn} - \mathcal{K}'(\tilde{\Theta}_m)\right) + w'_m \quad (6.5)$$

Here, only an average over the data is needed (i.e. no longer a maximization). Thus, we can implement an HMM with a forward-backward type of algorithm instead of also having to solve the maximization problem $\max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}$. At the end of this chapter we derive both the max and the summation for the HMM case.

An additional possible simplification is to avoid computing the w_m parameter altogether by merely setting it to $\tilde{w}_m$ from the traditional Jensen inequality, i.e.:

$$w_m = \tilde{w}_m + w'_m$$

This assumes that the upper bound has the same width and overall shape (modulo a flip and a translation) as the corresponding lower bound which may sometimes be a reasonable assumption. Evidently, this lazily requires *no extra computation* beyond the usual Jensen inequality. A few such iterations are acceptable at the beginning of an iterative algorithm until the optimization and help accelerate convergence in the early stages. It is wise, however, to eventually switch to the guaranteed bounds (involving the $G()$ function thereafter) to avoid divergence.

Visualization

In this visualization, we merely show the log-sum with several random scalar values of x_t data points under a single Gaussian model, a single Poisson distribution and finally a single exponential model. Figures 6.2, 6.3, and 6.4 depict the Jensen lower bound, the reverse-Jensen as well as the original distribution. Here, for visualization purposes we have chosen to represent the bounds in product-space for the Gaussian (and logarithmic space for the others).

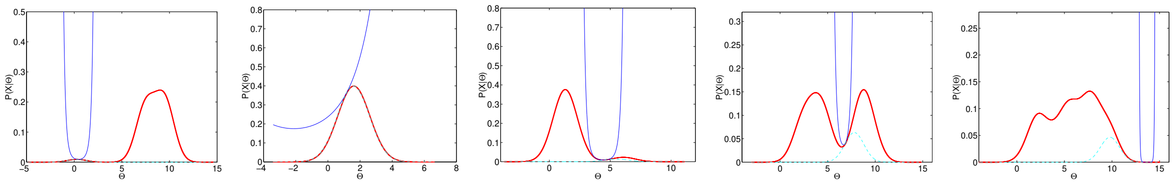


Figure 6.2: Jensen (dashed cyan) and reverse-Jensen (solid blue) bounds on the original mixture of mixtures distribution (thick red dots). The Gaussian model is shown with a mixture of data points in a product scale (as opposed to the logarithmic scale).

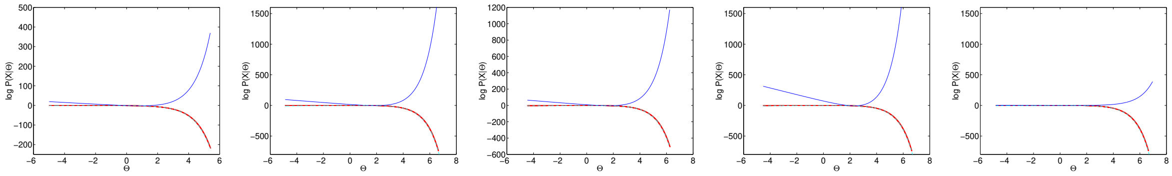


Figure 6.3: Jensen (dashed cyan) and reverse-Jensen (solid blue) bounds on the original mixture of mixtures distribution (thick red dots). The Poisson model is shown with a mixture of data points in a logarithmic scale.

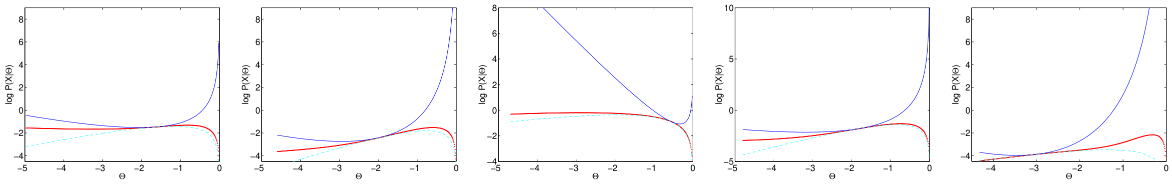


Figure 6.4: Jensen (dashed cyan) and reverse-Jensen (solid blue) bounds on the original mixture of mixtures distribution (thick red dots). The Exponential distribution is shown with a mixture of data points in a logarithmic scale.

6.3 Reverse Jensen Inequality for Hidden Markov Models

At this point, we resume our development of hidden Markov models and more explicitly cast them in the desired mixture of mixtures form (Equation 6.1). Since applying Jensen's inequality to HMMs is straightforward from the standard EM, Baum-Welch literature [13] [12] [156] [102], we will only go into the details of the reverse-Jensen inequality case. Recall that we had the following probability density function for the HMM:

$$\begin{aligned} p(X) &= \sum_{s_1=1}^{M} \cdots \sum_{s_T=1}^{M} \frac{1}{M} \exp\left(\sum_{t=1}^{T} \mathcal{A}(X_t) + \theta(s_t)^T X_t - \mathcal{K}(\theta(s_t)) \right. \\ &\qquad \left. + \sum_{t=2}^{T} A(\mathbf{x}(s_t)) + \eta(s_{t-1})^T \mathbf{x}(s_t) - K(\eta(s_{t-1})) \right) \end{aligned}$$

We shall now introduce indicator functions that will clarify the above notation. Consider the case when $s_t = m$, this can be represented by a delta-function that is only unity when $s_t = m$ and is zero otherwise, i.e. $\delta(s_t = m)$. This allows us to simplify the above as:

$$\begin{aligned} p(X) &= \sum_{s_1=1}^{M} \cdots \sum_{s_T=1}^{M} \frac{1}{M} \exp\left(\sum_{m=1}^{M} \left\{ \sum_{t=1}^{T} \delta(s_t = m)\mathcal{A}(X_t) + \delta(s_t = m)\theta_m^T X_t - \delta(s_t = m)\mathcal{K}(\theta_m) \right\} \right. \\ &\qquad \left. + \sum_{m=1}^{M} \left\{ \sum_{t=2}^{T} \delta(s_{t-1} = m)A(\mathbf{x}(s_t)) + \delta(s_{t-1} = m)\eta_m^T \mathbf{x}(s_t) - \delta(s_{t-1} = m)K(\eta_m) \right\} \right) \end{aligned}$$

Therefore, in the above, we are summing many exp() functions each of which is an exponential family member. The e-family member consists of an inner product with an aggregate data vector and an aggregate $\mathcal{K}$-type partition function. Our parameters for this hidden Markov model are M multinomial parameters and M Gaussian emission parameters. The aggregate parameter vector Θ can be seen as all these parameter vectors spliced together. The above can thus also be expressed as:

$$p(X) \quad = \quad \sum_{s_1=1}^{M} \cdots \sum_{s_T=1}^{M} \frac{1}{M} \exp\left(\mathbf{A}(\mathbf{X}_s) + \Theta^T \mathbf{X}_s - \mathbf{K}(\Theta)\right)$$

To compute reverse-Jensen bound over this structure we must be efficient since explicitly enumerating all the terms in the sum will cause intractabilities. To compute the reverse-bounds, it is suffices to show how to compute the w_m parameters. The Y_m and k parameters of the reverse-bound will not result in intractable computation once we have the w_m. We assume that obtaining the usual Jensen bounds is also feasible, giving us $\tilde{w}_m$, $\tilde{Y}_m$ and $\tilde{k}$. If we have solved for w_m for the reverse-Jensen case, obtaining the equivalent Y_m is given by the following[3] efficient formula:

$$Y_m \quad = \quad \left(\frac{\tilde{w}_m}{w_m} + 1\right) \mathcal{K}'(\tilde{\Theta}_m) - \frac{\tilde{w}_m}{w_m} \tilde{Y}_m$$

Obtaining the corresponding k scalar parameter for the reverse-Jensen case is then trivial. The crucial parameters to compute are the w_m parameters which determine the width of the reverse-Jensen bounds for the M Gaussian and the M multinomial models. Thus, we need to solve for $2M$ scalars. To distinguish the multinomial width parameters from those of the Gaussian we will use $\bar{w}_m$ where $m = 1..M$ to index the multinomial models and use w_m where $m = 1..M$ to index the Gaussian models. We now show how to estimate these *computationally critical* w_m for the Gaussian parameters of the HMM as well as the corresponding $w_{\bar{m}}$ for the multinomial parameters of the HMM.

6.3.1 Bounding the HMM Gaussians

Now, let us consider each component of the Θ vector individually. First, consider the m'th Gaussian emission component being dot-producted in the exponential:

$$\exp\left(\ldots + \theta_m^T \sum_{t=1}^{T} \delta(s_t = m) X_t - \ldots\right)$$

Therefore, the $\mathcal{Z}_{\bar{m}s}$ vector (from the previous reverse-Jensen bound definition and its nomenclature) that corresponds to it is:

$$\mathcal{Z}_{ms} \quad = \quad \mathcal{K}''(\tilde{\theta}_m)^{-1/2} \left(\sum_{t=1}^{T} \delta(s_t = m) X_t \; - \; \sum_{t=1}^{T} \delta(s_t = m) \mathcal{K}'(\tilde{\theta}_m)\right)$$

However, in the Gaussian case, we have $\mathcal{K}(\theta_m) = \frac{1}{2}\theta_m^T \theta_m$ so the above quickly simplifies into:

$$\mathcal{Z}_{ms} \quad = \quad \sum_{t=1}^{T} \delta(s_t = m)(X_t - \tilde{\theta}_m)$$

[3] The formula for Y_m here is straightforward to derive, simply by starting from the definitions of the Jensen and Reverse-Jensen bounds.

Now, to compute the w_m (and later the $w_{\bar{m}}$ for the multinomial) that correspond to the necessary reverse-Jensen bounds we need to *efficiently* evaluate the following expressions or upper bounds on them:

$$\sum_s h_s \mathcal{Z}_{ms}^T \mathcal{Z}_{ms}$$
$$\max_s \mathcal{Z}_{ms}^T \mathcal{Z}_{ms}$$

Above we recognize the responsibility terms h_s which are the probability posteriors over the latent state paths at the previous model setting $\tilde{\Theta}$ given the observations $\{X\}$ or $X_1, \ldots, X_T$. Thus, they can be written as $h_s = p(s_1, \ldots, s_T|\{X\}, \tilde{\Theta})$. Let us now expand out the required the inner products in the above computations:

$$\mathcal{Z}_{ms}^T \mathcal{Z}_{ms} = \sum_{t=1}^T \delta(s_t = m)(X_t - \tilde{\theta}_m)^T \times \sum_{t=1}^T \delta(s_t = m)(X_t - \tilde{\theta}_m)$$

For one of the terms of the reverse-Jensen bound, we need the max over all such possible inner products. Thus we need compute the maximum over any path of the above quantity:

$$\max_s \mathcal{Z}_{ms}^T \mathcal{Z}_{ms} = \max_s \sum_{t=1}^T \delta(s_t = m)(X_t - \tilde{\theta}_m)^T \times \sum_{\tau=1}^T \delta(s_\tau = m)(X_\tau - \tilde{\theta}_m)$$

Expanding the above, we obtain:

$$\max_s \mathcal{Z}_{ms}^T \mathcal{Z}_{ms} = \max_s \sum_{t=1}^T \sum_{\tau=1}^T \delta(s_t = m)\delta(s_\tau = m)(X_t - \tilde{\theta}_m)^T (X_\tau - \tilde{\theta}_m)$$

The discrete optimization required now will probably require integer programming to solve exactly (probably some variant of the knapsack problem). However, we can find an upper bound on the desired quantity with a simple a quadratic program. To do so, we simply introduce the following λ-scalar variables:

$$\lambda_t := \delta(s_t = m)$$

We do this for all time steps $t \in [1, \ldots, T]$. These λ_t are bounded over $\lambda_t \in [0, 1]$ and thus are a super-set of the binary variables. Therefore, maximizing over this larger space of variables will yield the true $\max_s \mathcal{Z}_{ms}^T \mathcal{Z}_{ms}$ or an upper bound over it. Thus, we can replace the above optimization with:

$$\max_s \mathcal{Z}_{ms}^T \mathcal{Z}_{ms} \leq \max_{\lambda_1, \ldots, \lambda_T} \sum_{t=1}^T \sum_{\tau=1}^T \lambda_t \lambda_\tau (X_t - \tilde{\theta}_m)^T (X_\tau - \tilde{\theta}_m) \quad \forall \lambda_t \in [0, 1]$$

The above is a quadratic program over T variables with $2T$ inequality constraints which can be solved in a straightforward manner (subject to the box constraints on $\lambda_t \in [0, 1]$). It is also evident that the maximum usually lies outside the convex hull of constraints and will typically cause the λ_t to rail to their extremes at the end of the optimization (except for degenerate situations). Therefore, we get a more conservative value for the maximum data magnitude which is still a guaranteed upper bound.

Now, we detail the computation of the other component of the w_m bound-parameter: the 'expected magnitude' (instead of the max):

$$\sum_s h_s \mathcal{Z}_{ms}^T \mathcal{Z}_{ms} = \sum_s h_s \sum_{t=1}^{T} \delta(s_t = m)(X_t - \tilde{\theta}_m)^T \times \sum_{t=1}^{T} \delta(s_t = m)(X_t - \tilde{\theta}_m)$$

Expanding the above, we obtain:

$$\begin{aligned}\sum_s h_s \mathcal{Z}_{ms}^T \mathcal{Z}_{ms} &= \sum_{t=1}^{T}\sum_{\tau=1}^{T} \left(\sum_s h_s \delta(s_t = m)\delta(s_\tau = m) \right) (X_t - \tilde{\theta}_m)^T (X_\tau - \tilde{\theta}_m) \\ &= \sum_{t=1}^{T}\sum_{\tau=1}^{T} p(s_t = m, s_\tau = m|\{X\}, \tilde{\Theta})\,(X_t - \tilde{\theta}_m)^T (X_\tau - \tilde{\theta}_m)\end{aligned}$$

In the above, we have introduced the marginal distribution $p(s_t = m, s_\tau = m|\{X\}, \tilde{\Theta})$ over the state at time t being m and the state at time τ being m after the HMM observations have been accounted for. This marginal distribution over a state sequence sub-string is straightforward to compute efficiently since the HMM is a tree-structured graphical model which permits efficient computation of marginal distributions (which is based on the Baum-Welch forward backward algorithm). In the case where $\tau = t$, the probability is merely $p(s_t = m|\{X\}, \tilde{\Theta})$ which is equal to the traditional normalized $\hat{\alpha}_t(m)\hat{\beta}_t(m)$ product from the familiar forward-backward algorithm [156].

> For completeness, we will show how to compute the marginal distribution $p(s_t, s_\tau|\{X\})$, where we will assume, without loss of generality, that $t < \tau$. Several quantities are easy to compute after the forward-backward algorithm. Once of them is the marginal over a single state: $p(s_t|\{X\}) = \hat{\alpha}_t\hat{\beta}_t$. Similarly, there is a direct formula for the marginal over a pair of subsequent states (where c_t is Rabiner's so-called scaling factor): $p(s_{t+1}, s_t|\{X\}) = c_{t+1}\hat{\alpha}(s_t)p(s_{t+1}|s_t)p(X_{t+1}|s_{t+1})\hat{\beta}(s_{t+1})$. Using Bayes' rule gives conditionals on pairs of subsequent states $p(s_{t+2}|s_{t+1}, \{X\}) = p(s_{t+2}, s_{t+1}|\{X\})/p(s_{t+1}|\{X\})$. Thus, we can obtain the following three-way marginal by multiplying a pairwise marginal with a conditional: $p(s_{t+2}, s_{t+1}, s_t|\{X\}) = p(s_{t+2}|s_{t+1}, \{X\})p(s_{t+1}, s_t|\{X\})$. To obtain $p(s_{t+2}, s_t|\{X\})$ we merely sum the later marginal over s_{t+1}. This reasonably efficient process is iterated until we reach $p(s_t, s_\tau|\{X\})$ and does not require intractable computation.

The above computation can be implemented efficiently with approximately $O(T^2M^2)$ operations since we need to consider bivariate probability distributions $p(s_t, s_\tau)$ which are of order $M \times M$ for every pair of time points in the trellis of length T. Although this is a factor of T slower than the computation of the forward backward algorithm in regular hidden Markov models which is typically $O(TM^2)$, the above computation is tractable (particularly for short trellis lengths).

The computations outlined thus readily give us *both* components (or legitimate upper bounds on them) of the w_m bound parameter efficiently without doing intractable calculations. These only require a simple quadratic program as well as the results from EM's Baum-Welch computations. We now move to bounding the multinomial parameters.

6.3.2 Bounding the HMM Multinomials

Next consider the $\bar{m}$'th multinomial component which is being dot-producted inside the exponential function as follows:

$$\exp\left(\ldots + \eta_{\bar{m}}^T \sum_{t=2}^{T} \delta(s_{t-1} = \bar{m})\mathbf{x}(s_t) - \ldots\right)$$

Therefore, the $\mathcal{Z}_{\bar{m}s}$ vector (from the previous reverse-Jensen bound nomenclature) that corresponds to it is:

$$\begin{aligned}
\mathcal{Z}_{\bar{m}s} &= K''(\tilde{\eta}_{\bar{m}})^{-1/2} \left(\sum_{t=2}^{T} \delta(s_{t-1} = \bar{m})\mathbf{x}(s_t) \;-\; \sum_{t=2}^{T} \delta(s_{t-1} = \bar{m})K'(\tilde{\eta}_{\bar{m}})\right) \\
&= \sum_{t=2}^{T} \delta(s_{t-1} = \bar{m})\; K''(\tilde{\eta}_{\bar{m}})^{-1/2}\; (\mathbf{x}(s_t) \;-\; K'(\tilde{\eta}_{\bar{m}}))
\end{aligned}$$

Now, to compute the $w_{\bar{m}}$ that correspond to the necessary reverse-Jensen bounds, we need to efficiently evaluate the following:

$$\sum_s h_s \mathcal{Z}_{\bar{m}s}^T \mathcal{Z}_{\bar{m}s}$$
$$\max_s \mathcal{Z}_{\bar{m}s}^T \mathcal{Z}_{\bar{m}s}$$

As before, we can expand the inner products as follows:

$$\mathcal{Z}_{\bar{m}s}^T \mathcal{Z}_{\bar{m}s} = \sum_{t=2}^{T}\sum_{\tau=2}^{T} \delta(s_{t-1} = \bar{m})\delta(s_{\tau-1} = \bar{m})\; (\mathbf{x}(s_t) \;-\; K'(\tilde{\eta}_{\bar{m}}))^T\, K''(\tilde{\eta}_{\bar{m}})^{-1}\; (\mathbf{x}(s_\tau) \;-\; K'(\tilde{\eta}_{\bar{m}}))$$

Since the $\mathbf{x}$ data here correspond to only a choice of one of the M-multinomial models, we can rewrite it as follows:

$$\mathbf{x}(s_t) = \sum_{q=1}^{M} \delta(s_t = q)\mathbf{x}_q$$

For notational convenience, we also define the following:

$$\mathbf{z}_q = K''(\tilde{\eta}_m)^{-1/2}\,(\mathbf{x}_q - K'(\tilde{\eta}_m))$$

Thus, the desired inner product term becomes:

$$\begin{aligned}
\mathcal{Z}_{\bar{m}s}^T \mathcal{Z}_{\bar{m}s} &= \sum_{t=2}^{T}\sum_{\tau=2}^{T} \delta(s_{t-1} = \bar{m})\delta(s_{\tau-1} = \bar{m}) \left(\sum_{q=1}^{M} \mathbf{z}_q \delta(s_t = q)\right)^T \left(\sum_{r=1}^{M} \mathbf{z}_r \delta(s_\tau = r)\right) \\
&= \sum_{t=2}^{T}\sum_{\tau=2}^{T} \delta(s_{t-1} = \bar{m})\delta(s_{\tau-1} = \bar{m}) \left(\sum_{q=1}^{M}\sum_{r=1}^{M} \delta(s_t = q)\delta(s_\tau = r)\mathbf{z}_q^T \mathbf{z}_r\right)
\end{aligned}$$

To obtain the desired $\bar{w}_m$ reverse-Jensen bound parameter for the multinomials, we need to compute two terms using the above inner-products. These are the expected term (the sum over all the inner products weighted by the h_s probabilities) and the max type term. The $\max_s$ term is elucidated first:

$$\max_s \mathcal{Z}_{\bar{m}s}^T \mathcal{Z}_{\bar{m}s} = \max_s \sum_{q=1}^{M} \sum_{r=1}^{M} \sum_{t=2}^{T} \sum_{\tau=2}^{T} \delta(s_{t-1} = \bar{m}) \delta(s_{\tau-1} = \bar{m}) \delta(s_t = q) \delta(s_\tau = r) \, \mathbf{z}_q^T \mathbf{z}_r$$

Although an integer programming solution may be possible for the above maximization, we can instead solve for an upper bound on it (which will still produce a guaranteed bound) by reformulating it as follows. We first define scalars as follows:

$$\lambda_{rq} := \sum_{t=2}^{T} \sum_{\tau=2}^{T} \delta(s_{t-1} = \bar{m}) \delta(s_{\tau-1} = \bar{m}) \delta(s_t = q) \delta(s_\tau = r)$$

It is clear that these scalars are all positive, i.e. $\lambda_{rq} \geq 0$ which gives us M^2 constraints. Furthermore, we have the following constraint on these variables:

$$\sum_r \sum_q \lambda_{rq} \leq (T-1)^2$$

Therefore we can solve for an upper bound on the desired quantity via a simple linear program over the M^2 surrogate variables λ_{rq}. By allowing these surrogate λ variables to vary as continuous parameters over the constrained range, we are solving for a more flexible maximization than the original integer programming problem over state paths. Therefore optimizing over the λ will yield a more conservative upper bound on the quantity of interest (which still produces a legitimate reverse-Jensen bound overall). The solution can thus be solve via the simple linear form below (with the corresponding $M^2 + 1$ inequality constraints imposed on the λ-variables):

$$\max_s \mathcal{Z}_{\bar{m}s}^T \mathcal{Z}_{\bar{m}s} \leq \max_\lambda \sum_{q=1}^{M} \sum_{r=1}^{M} \lambda_{rq} \, \mathbf{z}_q^T \mathbf{z}_r$$

Now, we turn our attention to the computation of the expected inner product (versus the maximum):

$$\begin{aligned}
\sum_s h_s \mathcal{Z}_{\bar{m}s}^T \mathcal{Z}_{\bar{m}s} &= \sum_s h_s \sum_{q=1}^{M} \sum_{r=1}^{M} \sum_{t=2}^{T} \sum_{\tau=2}^{T} \delta(s_{t-1} = \bar{m}) \delta(s_{\tau-1} = \bar{m}) \delta(s_t = q) \delta(s_\tau = r) \, \mathbf{z}_q^T \mathbf{z}_r \\
&= \sum_{q=1}^{M} \sum_{r=1}^{M} \sum_{t=2}^{T} \sum_{\tau=2}^{T} \sum_s h_s \delta(s_{t-1} = \bar{m}) \delta(s_{\tau-1} = \bar{m}) \delta(s_t = q) \delta(s_\tau = r) \, \mathbf{z}_q^T \mathbf{z}_r \\
&= \sum_{q=1}^{M} \sum_{r=1}^{M} \sum_{t=2}^{T} \sum_{\tau=2}^{T} p(s_{t-1} = \bar{m}, s_{\tau-1} = \bar{m}, s_t = q, s_\tau = r | \{X\}, \tilde{\Theta}) \, \mathbf{z}_q^T \mathbf{z}_r
\end{aligned}$$

Once again we have to simply use the marginal distribution over a state sub-string $p(s_{t-1} = \bar{m}, s_{\tau-1} = \bar{m}, s_t = q, s_\tau = r | \{X\}, \tilde{\Theta})$ which is readily obtainable for a tree-structure or chain structure HMM model without intractable computation. The clique size being considered is over 4 variables here which leads to more work then in a standard EM setting, it appears that computing the expected data magnitude as opposed to expected data vectors (as EM does) requires squared clique sizes. More specifically, the implementation of the above requires $O(T^2 M^4)$ operations since we need to consider 4-variable probability distributions $p(s_t, s_{t-1}, s_\tau, s_{\tau-1})$ which are of order M^4

for every pair of time points in the trellis of length T. Although this is slower than the computation of the forward backward algorithm in a regular hidden Markov model (which is typically $O(TM^2)$) the above computation is still tractable.

The above computations thus give us both the w_m and the $\bar{w}_m$ parameters for the Gaussian and the Multinomial components of the reverse-Jensen bound without resorting to intractable computation. All computations remain efficient via appeals to linear programming, quadratic programming and forward-backward types of dynamic programming.

6.4 Conditional Hidden Markov Models

Given both Jensen and reverse-Jensen bounds for hidden Markov models, it is now feasible to use the CEM algorithm and perform conditional likelihood maximization in a straightforward way. In this section, we will describe one scenario where it becomes necessary to optimize a conditional likelihood quantity. There are many other situations where a conditional expression will arise due to the problem formulation and result in a negated log-sum type of expression. Negated log-sums and conditional likelihood expressions would occur, for example, when learning a classifier based on multiple competing hidden Markov models, or when learning a mixture-of-experts regression function whose gates are hidden Markov models. In this section we will focus on and develop what is traditionally called an input-output hidden Markov model [16] where the objective is to regress certain components of a time series from others which are observable. Both inputs and outputs are coupled through a hidden state which evolves with Markov dynamics.

We begin with a standard hidden Markov model as portrayed in the previous sections and assume that we are given a time sequence of vectors. For an input-output hidden Markov model, we split the emission vectors into two components: x and y which are to be treated as input and output respectively. Given a training sequence of such $x_1, \ldots, x_T$ and $y_1, \ldots, y_T$ vector-pairs over time, we would like to estimate a hidden Markov model such that, on future test data, we can reliably predict a $\hat{y}_1, \ldots, \hat{y}_\tau$ sequence from a given $\hat{x}_1, \ldots, \hat{x}_\tau$ sequence alone. An example application would be to learn the mapping from various passive biological measurements such as heart-rate, temperature, motion energy, etc. to a more complicate output measurement such as blood-glucose which would not always be easily measurable.

We shall assume that we are dealing with an HMM with a total of M underlying states which gives rise to an $M \times M$ transition matrix. This transition matrix is equivalently described by M multinomial distributions of dimensionality equal to M. We will further assume that the emission model for x and y is jointly Gaussian with a diagonal covariance matrix. Therefore, it is necessary to estimate the M Gaussian covariance terms, the M Gaussian mean terms and the $M \times M$ state transition matrix.

Since we wish to regress y *from* x, it is natural to form a conditional distribution of $p(y|x)$ and a natural objective function is the conditional log-likelihood on the training data below:

$$l^c = \log p(x, y) - \log p(x)$$

We can view the first term in the above expression as the log-likelihood of the whole hidden Markov model on the training data. The second (negated) term, is the likelihood of the hidden Markov model after it has been marginalized and only applies the input data x. Figuratively, we have the following graphical model scenario:

Recall that $p(x, y)$ and $p(x)$ in the above expressions are latent quantities which involve a summation over all possible paths in the state space of the hidden Markov model. In other words, we can expand

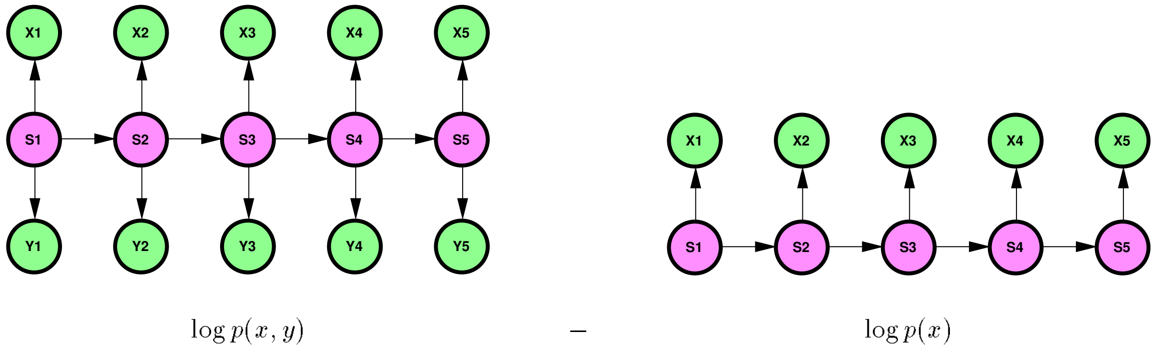


Figure 6.5: Conditional HMM Estimation. Effectively, we maximize the joint $p(x, y)$ log-likelihood of the HMM over both x and y (inputs and outputs) minus the marginal $p(x)$ log-likelihood where the HMM has been summed over to obtain a marginal only over the x input.

the conditional likelihood quantity as follows.

$$l^c = \log \sum_S p(S, x, y) - \log \sum_S p(S, x)$$

We lower bound the left hand term with Jensen's inequality while the reverse-Jensen inequality is applied to the right hand term $\log p(x)$. This gives us an overall lower bound on the conditional log-likelihood l^c which can be maximized in closed form.

We then iterate bounding and maximization steps until we converge to a local maximum of conditional likelihood. To perform regression, we merely use novel $\hat{x}_1, \ldots, \hat{x}_\tau$ data to estimate a probability distribution over the hidden states S. This can then be used to compute an estimated output sequence $\hat{y}_1, \ldots, \hat{y}_\tau$ simply by using the alpha-beta values in the forward-backward algorithm to weight each Gaussian in the emission model appropriately for each time step.

6.4.1 Experiments

To compare the estimates of the conditional likelihood criterion or CEM against the usual maximum likelihood or EM framework, we used a standard meteorological time series prediction task. This small toy data set constitutes of 420 monthly of measurements of the precipitation, temperature and water flow in the San Francisco area from 1932 to 1966. Therefore, we have a 4 dimensional time series when we concatenate the month as well (which is represented as a sinusoidal whose values range between -1.0 and 1.0). The input-output HMM we wish to build will regress the precipitation value from the remaining three inputs. Training was performed on the first 360 samples of the time series while testing was performed on the remaining 60 samples. Figure 6.6 depicts a snapshot of the training data, i.e. the 4-dimensional time series of weather measurements.

We trained both an EM and a CEM based input-output hidden Markov model with 4 states assuming diagonal-covariance Gaussian emission models. The Gaussians are over 4 dimensions (3 inputs and the precipitation level as output). Table 6.4.1 summarizes the resulting log likelihoods and the RMS error for the EM and CEM algorithms. In this particular example, the CEM algorithm was much slower requiring over 2 orders of magnitude more time to attain convergence than the EM algorithm. This is due to extra computations on top of the usual forward-backward algorithm and the extra looseness of the reverse-Jensen bounds which require more iterations. However, the resulting testing

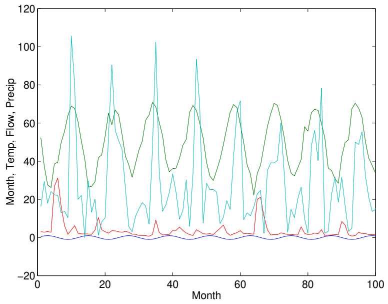


Figure 6.6: Training data from the San Francisco Data Set. Here, the 4-dimensional time series depicts the monthly levels of precipitation, water flow, temperature as well as the month of year as a sinusoid.

performance which is notable in the conditional likelihood on testing and the RMS error on testing shows CEM performed favorably when compared to EM based prediction. For an appropriate and quantitative measure of performance, the conditional likelihood on testing data is the best choice since the given task here is to predict precipitation values *from* the other inputs. Therefore, it is inappropriate to penalize or reward good estimates of the input components. Thus joint likelihood on test data is an inappropriate measurement of the quality of the estimate. Only the desired outputs need to be properly predicted and hence the conditional likelihood score (as well as RMS error) depict that a better regression estimate is obtained by the CEM algorithm.

Training	Log-Likelihood	Conditional Log-Likelihood	
EM	-5.985	-3.473	
CEM	-9.683	-3.066	
Testing	Log-Likelihood	Conditional Log-Likelihood	RMS Error
EM	-6.466	-3.463	195.2
CEM	-9.360	-3.110	184.6

Table 6.1: CEM & EM Performance for HMM Precipitation Prediction.

Figure 6.7 shows a qualitative performance difference between the EM-based hidden Markov model and the CEM-based hidden Markov model. Note how the CEM version tracks the precipitation values (during testing) more closely than the EM and also has a more accurate range of variation than the conservative near-constant precipitation estimates EM generates.

In Chapter 8, a more ambitious application of the input-output HMM regression will be attempted. However, due to the extremely large size of the data and its long trellis length, simplifying approximations will be needed to reduce the computation time of the CEM algorithm such that it operates in roughly the same time as the EM algorithm.

6.5 Discriminative Hidden Markov Models

In this section we will give details of an implementation of MED for hidden Markov models in a discriminative regression setting. Unlike the standard regression or conditional regression in the

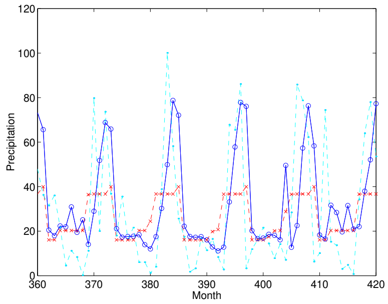


Figure 6.7: Test predictions for the San Francisco Data Set. The true precipitation values are depicted with the dashed cyan line and the filled dots. The EM hidden Markov model's predictions are outlined with the dashed red line connecting the 'x' points. The CEM hidden Markov model's predictions are outlined by the solid blue line connecting the hollow 'o' circles. Note how the CEM predictions track the true precipitation values more closely while the EM predictions do not span the desired range of variation and do not align well with the desired output.

previous input-output HMM, we will here use a discriminative epsilon-tube insensitivity with linear loss function to form the regression function. The implementation for a classification setting is straightforward given the regression development here. While classification has many applications (i.e. speech recognition, bioinformatics, etc.) we will portray the discriminative regression HMM as a novel way to incorporate time invariance in regression settings. Recall in section 4.1 where we developed MED for regression applications and discussed the use of generative and latent generative models therein. We now consider the following regression discriminant function:

$$\mathcal{L}(X;\Theta) \quad = \quad \log \frac{\sum_S p(X,S|\theta_+)}{\sum_S p(X,S|\theta_-)} + b$$

Where the numerator and denominator are hidden Markov models which define a probability density over each X datum which is itself a *sequence* of observations instead of a single vector. Recall the regression constraints that arise in the MED framework:

$$\begin{aligned}
\int P(\Theta,\gamma)\,[y_t - \mathcal{L}(X_t;\Theta) + \gamma_t]\, d\Theta d\gamma \quad &\geq \quad 0, \quad t = 1..T \\
\int P(\Theta,\gamma)\,[\gamma_t' - y_t + \mathcal{L}(X_t;\Theta)]\, d\Theta d\gamma \quad &\geq \quad 0, \quad t = 1..T
\end{aligned}$$

The expectations over the discriminant functions above are intractable and therefore we will employ variational bounds instead (Jensen and reverse-Jensen). In the first constraint, we will need to *upper bound* $\mathcal{L}(X_t;\Theta)$ while in the second constraint we will need to *lower bound* $\mathcal{L}(X_t;\Theta)$. This will result in the creation of more *restrictive* constraints that generate a convex hull that is contained in the convex hull of the original MED problem. In the first set of constraints, we will use reverse-Jensen on the numerator and the Jensen bound on the denominator. The situation is reversed for the other set of constraints. Once the variational bounds are used, the MED solution distribution $P(\Theta)$ will factorize across all the (positive and negative) HMM parameters (i.e. the emissions and transitions) if our prior over these $P_0(\Theta)$ is conjugate and factorized itself. Assume we have invoked the inequalities at a given $\tilde{\Theta}$ operating point and now have the following constraints instead:

$$\int P(\Theta,\gamma)\left[y_t + \sum_m w_{tm}\log p(Y_{tm}|m,\theta^+) - k_t + \sum_m \tilde{w}_{tm}\log p(\tilde{Y}_{tm}|m,\theta^-) + \tilde{k}_t - b + \gamma_t\right] d\Theta d\gamma \quad \geq \quad 0$$

$$\int P(\Theta,\gamma)\left[\gamma_t' - y_t + \sum_m \tilde{w}_{tm}\log p(\tilde{Y}_{tm}|m,\theta^+) + \tilde{k}_t + \sum_m w_{tm}\log p(Y_{tm}|m,\theta^-) - k_t + b\right] d\Theta d\gamma \;\;\geq\;\; 0$$

In the above, we have used Jensen and reverse-Jensen on each training sequence X_t in the data $t = 1..T$ to obtain the parameters of the bounds: $\tilde{w}_{tm}, \tilde{Y}_{tm}, \tilde{k}_t$ and w_{tm}, Y_{tm}, k_t for each HMM model (both the positive and the negative one). Here, m indexes over the M Gaussian emission models and the M multinomial (transition matrix) models for each HMM respectively. To compute the bounds we convert the multinomial in natural parameterization for the Jensen/reverse-Jensen computations and back to the form in section 3.9.4 for the MED computations. The Gaussian is only over means and therefore is always in its natural parameterization. The above constraints give rise to the following partition function where the margin (with exponential prior) and bias (with Gaussian prior) components are the same as any simple (linear) MED regression:

$$Z(\lambda) \;\;=\;\; Z_\gamma(\lambda)\, Z_b(\lambda)\, Z_{\theta_+}(\lambda)\, Z_{\theta_-}(\lambda)$$

The only novel computation involves solving for the Z_{θ_+} and Z_{θ_-} components since the other components were already explored in previous regression problems. A closed form partition function is needed so that we can estimate the optimal setting of the Lagrange multipliers. Each of the models, θ_+ and θ_- contains the M Gaussian emission parameters and M multinomial parameters of the positive (numerator) HMM. We will assume white Gaussian priors over the emission parameters and uniform Dirichlet priors (i.e. $\alpha_k = \alpha = 1/M$ in the notation of section 3.9.4) over the multinomials. It suffices to show how to compute the partition function component for one of the HMMs, say Z_{θ_+}, which is given by:

$$Z_{\theta_+}(\lambda) \;\;=\;\; \int_{\theta_+} P_0(\theta_+) e^{\sum_t \lambda_t \sum_m w_{tm}\log p(Y_{tm}|m,\theta^+) - \lambda_t k_t + \sum_t \lambda_t' \sum_m \tilde{w}_{tm}\log p(\tilde{Y}_{tm}|m,\theta^+) + \lambda_t'\tilde{k}_t} d\theta_+$$

The above partition function component can be further broken down as follows:

$$Z_{\theta_+}(\lambda) \;\;=\;\; Z_{linear+}(\lambda) \;\times\; \Pi_{n=1}^M Z_{gauss+,n}(\lambda) \;\times\; \Pi_{q=1}^M Z_{multi+,q}(\lambda)$$

There is a simple log-linear component which is given by the following:

$$Z_{linear+}(\lambda) \;\;=\;\; e^{-\sum_t \lambda_t k_t + \sum_t \lambda_t' \tilde{k}_t}$$

There are $n = 1..M$ Gaussian partition function components given here in logarithmic form (where the $\mathcal{A}(Y) = -1/2Y^TY - D/2\log(2\pi)$, as usual for a Gaussian e-family model):

$$\log Z_{gauss+,n}(\lambda) \;\;=\;\; \log\int P_0(\mu_n) e^{\sum_t \lambda_t w_{tn}(\mathcal{A}(Y_{tn}) + Y_{tn}^T\mu_n - 1/2\mu_n^T\mu_n) + \sum_t \lambda_t'\tilde{w}_{tn}(\mathcal{A}(\tilde{Y}_{tn}) + \tilde{Y}_{tn}^T\mu_n - 1/2\mu_n^T\mu_n)} d\mu_n$$

$$\begin{aligned}\log Z_{gauss+,n}(\lambda) \;\;=\;\; & \sum_t \lambda_t w_{tn}\mathcal{A}(Y_{tn}) + \sum_t \lambda_t'\tilde{w}_{tn}\mathcal{A}(\tilde{Y}_{tn}) - \frac{1}{2}\log\left(1 + \sum_t \lambda_t w_{tn} + \sum_t \lambda_t'\tilde{w}_{tn}\right) \\ & + \frac{1}{2}\frac{\left(\sum_t \lambda_t w_{tn}Y_{tn} + \sum_t \lambda_t'\tilde{w}_{tn}\tilde{Y}_{tn}\right)^T\left(\sum_t \lambda_t w_{tn}Y_{tn} + \sum_t \lambda_t'\tilde{w}_{tn}\tilde{Y}_{tn}\right)}{1 + \sum_t \lambda_t w_{tn} + \sum_t \lambda_t'\tilde{w}_{tn}}\end{aligned}$$

We also have to consider the contribution of the $q = 1..M$ multinomial components. We first convert the natural parameterization of the data, i.e. the $M-1$-dimensional vectors Y_{tq}, into the standard multinomial form with M-dimensional vectors U_{tq}. This is done by concatenating an extra element to the vector such that it sums to unity. We then integrate to obtain the following partition function which is reminiscent of the derivation in Section 3.9.4 (and we use the superscript k to index into the dimensionality of the U_{tq} vectors):

$$Z_{multi+,q} \quad \propto \quad \frac{\Pi_{k=1}^{M} \Gamma(\alpha_k + \sum_t \lambda_t w_{tq} U_{tq}^k + \sum_t \lambda'_t \tilde{w}_{tq} \tilde{U}_{tq}^k)}{\Gamma(\sum_{k=1}^{M} \alpha_k + \sum_t \lambda_t w_{tq} U_{tq}^k + \sum_t \lambda'_t \tilde{w}_{tq} \tilde{U}_{tq}^k)} e^{\sum_t \lambda_t w_{tq} \frac{\Gamma(1+\sum_{k=1}^{M} U_{tq}^k)}{\Pi_{k=1}^{M} \Gamma(1+U_{tq}^k)} + \sum_t \lambda'_t \tilde{w}_{tq} \frac{\Gamma(1+\sum_{k=1}^{M} \tilde{U}_{tq}^k)}{\Pi_{k=1}^{M} \Gamma(1+\tilde{U}_{tq}^k)}}$$

For computing the partition function for the negative HMM, we merely use its own Jensen and reverse-Jensen parameters and permute the roles of λ and λ'. The negative of the logarithm of the aggregate partition function is then maximized. There are redundancies in the above computations and they are all computable efficiently if only a single Lagrange multiplier is modified at a time. This permits a fast axis-parallel implementation.

Once we have sufficiently optimized over the Lagrange multipliers, we can use the current setting of these to compute $P(\Theta)$. To obtain the next $\tilde{\Theta}$ setting, we merely find the maximum of $P(\Theta)$. For the Gaussian models, the maximum is merely at the mean which involves the following simple update rule (the negative model's parameters are updated similarly with their bounds and the role of the λ_t and λ'_t swapped):

$$\mu_{n+} \quad = \quad \frac{\sum_t \lambda_t w_{tn} Y_{tn} + \sum_t \lambda'_t \tilde{w}_{tn} \tilde{Y}_{tn}}{1 + \sum_t \lambda_t w_{tn} + \sum_t \lambda'_t \tilde{w}_{tn}}$$

For the q'th multinomial, the natural parameters are θ_{+q} which permit a simple update rule involving the gradients of the cumulant-generating function $\mathcal{K} = \log(1+\sum_i \exp(\theta_i))$. The following rule merely finds the max of the MED solution distribution $P(\theta_{+q})$ in e-family form (the $\vec{1}$ is used to denote a vector of ones of the same size as Y_{tq}):

$$\frac{\partial \mathcal{K}(\theta_{+q})}{\partial \theta_{+q}} \quad = \quad \frac{\alpha \vec{1} + \sum_t \lambda_t w_{tq} Y_{tq} + \sum_t \lambda'_t \tilde{w}_{tq} \tilde{Y}_{tq}}{1 + \sum_t \lambda_t w_{tq} + \sum_t \lambda'_t \tilde{w}_{tq}}$$

Once the max has been found, we can re-convert the natural representation of θ_{+q} back into the usual multinomial parameterization (using ρ as in Section 3.9.4). Thus, we iterate, updating the $\tilde{\Theta}$ contact point, recomputing the bounds, and estimating the Lagrange multipliers. This is continued until convergence (a heuristic stopping criterion is used). Utilizing the MED solution $P(\Theta)$ for prediction is still intractable even after training is performed since we must compute expectations over the HMM. Instead, during run-time, we merely use the maximum $\hat{\Theta} = \text{argmax} P(\Theta)$ which gives us two fixed HMM models. The log-ratio of their likelihoods (with the bias scalar parameter) is then used to compute the regression to obtain an approximate output:

$$\begin{aligned} \hat{y} &= \int P(\Theta) \mathcal{L}(X;\Theta) d\Theta \\ \hat{y} &\approx \mathcal{L}(X;\hat{\Theta}) \end{aligned}$$

We can thus form a discriminative regression model that inherits the dynamic-time-warping properties of an HMM while optimizing an epsilon-tube insensitive scalar prediction. The above expressions can be trivially re-applied to also create discriminative HMMs for classification where the parameters of each HMM are estimated such that we obtain a large-margin decision boundary between the two generative models (unlike in the standard maximum-likelihood setting).

6.6 Latent Bayesian Networks

We shall now discuss the case of Bayesian networks in general, of which hidden Markov models are a specific example. The hidden Markov model has a chain dependency structure which maintains some important computational properties that were useful for computing Jensen and reverse-Jensen bounds efficiently. It is well known that Bayesian networks which have tree structures can also take advantage of efficient algorithms to avoid intractabilities and therefore may also be possible to estimate discriminatively (with the Jensen and reverse-Jensen bounds). Recall that tree structured dependencies between variables who are themselves in the exponential family form an aggregate exponential-family distribution [34]. Now, if we have latent or hidden variables in the Bayes net graph, these tree structures need to be summed over. Therefore, the log-likelihood of the latent Bayes net can be represented as a mixture of exponential family distributions. Again, this can be written in the mixture of mixtures form of Equation 6.1. Subsequently, its logarithm (the log-likelihood of the latent Bayes net) can be upper and lower bounded via Jensen and reverse-Jensen. Although we can compute these bounds merely by unfolding the latencies in the mixture of mixtures, this may be highly inefficient. The computations for the w_m would become intractable since they would require enumerating all latent configurations in the Bayesian networks. Just as we demonstrated for the hidden Markov model where efficient algorithms avoid the exponentially large explicit mixture model, we need an efficient algorithm to compute the reverse-Jensen bounds for general tree-structured Bayes nets by taking advantage of the conditional independency properties of the graphical model. Otherwise, naive computation of Jensen and reverse-Jensen bounds will require intractable amounts of work for latent Bayesian networks.

It is possible to compute lower bounds efficiently by taking advantage of the independency structure in the networks and that is done for the regular Jensen inequality parameters by using EM and the so-called Junction Tree Algorithm [102]. In fact, the forward-backward algorithm that we modified for the reverse-Jensen bound is a special case of the more general junction-tree algorithm. Both are methods that take advantage of the structure of the graph to efficiently compute expectations. For solving the reverse-Jensen upper bound's parameters, we also need such efficient procedures that take advantage of the graph structure. Just as was shown for the HMM, the general case of latent tree-structured Bayesian networks may also have a similar efficient reverse-Jensen algorithms and this direction definitely merits further investigation.

6.7 Data Set Bounds

So far, we have considered only bounds (lower and upper) on the log-likelihood of a single data point or single sequence (in the HMM case). However, it is often the case that we will want to bound the likelihood of a data set of many points. If this is the case, the reverse-Jensen inequality can be applied in a more straightforward way. The main benefit is that the reverse-Jensen inequality will be applied once to form the bound instead of being applied separately for each data point and this may generate a more efficient and more appropriately shaped bound.

Consider the following augmentation to the bounding of the log-sum. To obtain the aggregate data set's conditional likelihood, we must sum the likelihood over each data point as follows:

$$l^c = \sum_i \log \sum_m p(m, c_i, X_i|\Theta) - \sum_i \log \sum_m \sum_c p(m, c, X_i|\Theta)$$

The conditional log-likelihood (l^c) above can be seen as the joint log-likelihood (l^j) minus the marginal log-likelihood (l^m):

$$l^c = l^j - l^m$$

Where in the above we have the joint log-likelihood:

$$l^j = \sum_i \log \sum_m p(m, c_i, X_i|\Theta)$$

We also have the *negated* marginal log-likelihood:

$$l^m = \sum_i \log \sum_m \sum_c p(m, c, X_i|\Theta)$$

Since we want overall lower bounds on the conditional likelihood, we need to lower bound the joint log-likelihood above and upper bound the marginal log-likelihood (since it gets negated). We can also write the marginal log-likelihood more compactly as follows (which obscures the fact that it is latent):

$$l^m = \sum_i \log p(X_i|\Theta)$$

We begin by adding a constant term to the marginal likelihood which has no effect on the shape of the bound. This is in the same spirit as the incremental likelihood derivation for EM in Bishop [20]. This additive constant does not vary since it is based on the fixed previous $\tilde{\Theta}$ parameter values (i.e. at the point of tangential contact) and has no effect on the shape of the bounds:

$$\Delta l^m = \sum_i \log \frac{p(X_i|\Theta)}{p(X_i|\tilde{\Theta})}$$

Now, we shall do the reverse of EM and pull the summation over data *into* the logarithm operator. This will generate an upper bound on incremental marginal log-likelihood:

$$\sum_i \log \frac{p(X_i|\Theta)}{p(X_i|\tilde{\Theta})} \leq \log \frac{\sum_i \gamma_i p(X_i|\Theta)^{\beta_i}}{\sum_i \gamma_i p(X_i|\tilde{\Theta})^{\beta_i}} \tag{6.6}$$

One thing to note in the above bound is that the γ_i terms can be scaled by an arbitrary amount and this will not change the shape of the bound. However, we still need to pick the γ_i and the β_i to insure that we have a guaranteed upper bound. These will become obvious shortly. First consider the right hand side above and manipulate as follows:

$$\begin{aligned}
\log \frac{\sum_i \gamma_i p(X_i|\Theta)^{\beta_i}}{\sum_i \gamma_i p(X_i|\tilde{\Theta})^{\beta_i}} &= \log \frac{\sum_i \gamma_i p(X_i|\Theta)^{\beta_i} \times \frac{p(X_i|\tilde{\Theta})^{\beta_i}}{p(X_i|\tilde{\Theta})^{\beta_i}}}{\sum_j \gamma_j p(X_j|\tilde{\Theta})^{\beta_j}} \\
\log \frac{\sum_i \gamma_i p(X_i|\Theta)^{\beta_i}}{\sum_i \gamma_i p(X_i|\tilde{\Theta})^{\beta_i}} &= \log \sum_i \left(\frac{\gamma_i p(X_i|\tilde{\Theta})^{\beta_i}}{\sum_j \gamma_j p(X_j|\tilde{\Theta})^{\beta_i}} \right) \frac{p(X_i|\Theta)^{\beta_i}}{p(X_i|\tilde{\Theta})^{\beta_i}}
\end{aligned}$$

It is clear that if γ_i are all positive, then the terms in the parentheses are always positive and sum to unity (if we sum over the index i). Thus, due to concavity of the log() function, we can apply

Jensen's inequality:

$$\log \frac{\sum_i \gamma_i p(X_i|\Theta)^{\beta_i}}{\sum_i \gamma_i p(X_i|\tilde{\Theta})^{\beta_i}} \geq \sum_i \left(\frac{\gamma_i p(X_i|\tilde{\Theta})^{\beta_i}}{\sum_j \gamma_j p(X_j|\tilde{\Theta})^{\beta_i}} \right) \log \frac{p(X_i|\Theta)^{\beta_i}}{p(X_i|\tilde{\Theta})^{\beta_i}} \tag{6.7}$$

$$\log \frac{\sum_i \gamma_i p(X_i|\Theta)^{\beta_i}}{\sum_i \gamma_i p(X_i|\tilde{\Theta})^{\beta_i}} \geq \sum_i \left(\frac{\gamma_i p(X_i|\tilde{\Theta})^{\beta_i}}{\sum_j \gamma_j p(X_j|\tilde{\Theta})^{\beta_i}} \right) \beta_i \log \frac{p(X_i|\Theta)}{p(X_i|\tilde{\Theta})} \tag{6.8}$$

By inspection, we can see that Equation 6.8 is very similar to Equation 6.6 (when left and right hand sides are flipped). However, for them to be *identical*, we will need to guarantee that every term multiplying the logarithm in the right hand side of Equation 6.8 is unity. Thus we need:

$$\left(\frac{\gamma_i p(X_i|\tilde{\Theta})^{\beta_i}}{\sum_j \gamma_j p(X_j|\tilde{\Theta})^{\beta_i}} \right) \beta_i = 1$$

The above can be simplified to (recall that the γ_i are invariant up to a global scale factor):

$$\gamma_i \propto \frac{1}{\beta_i p(X_i|\tilde{\Theta})^{\beta_i}}$$

We can pick the scale factor above arbitrarily and it should be chosen for numerical precision reasons (to avoid underflow or overflow computationally). One possible choice is to normalize the γ_i as follows:

$$\gamma_i = \frac{\frac{1}{\beta_i p(X_i|\tilde{\Theta})^{\beta_i}}}{\sum_j \frac{1}{\beta_j p(X_j|\tilde{\Theta})^{\beta_j}}}$$

We should also note that the β_i are always greater than or equal to unity. This is because we had:

$$\left(\frac{\gamma_i p(X_i|\tilde{\Theta})^{\beta_i}}{\sum_j \gamma_j p(X_j|\tilde{\Theta})^{\beta_i}} \right) \beta_i = 1$$

Clearly, the terms in the parentheses that multiply β_i are less than or equal to unity so we can conclude that $\beta_i \geq 1$.

It is evident that we have guaranteed the bound originally proposed in Equation 6.6 and we still have some freedom in selecting the β_i (as long as they are greater than unity). We shall exploit this freedom to find the tightest bound possible. At this point we have the following bound:

$$\Delta l^m \leq \log \frac{\sum_i \gamma_i p(X_i|\Theta)^{\beta_i}}{\sum_i \gamma_i p(X_i|\tilde{\Theta})^{\beta_i}}$$

$$\Delta l^m \leq \log \frac{\sum_i \frac{1}{\beta_i} \frac{p(X_i|\Theta)^{\beta_i}}{p(X_i|\tilde{\Theta})^{\beta_i}}}{\sum_i \frac{1}{\beta_i}}$$

The above bound holds for any distribution and for any β_i that are greater than unity. However, it should be noted that β_i also sum inversely to unity since:

$$\frac{1}{\beta_i} = \frac{\gamma_i p(X_i|\tilde{\Theta})^{\beta_i}}{\sum_j \gamma_j p(X_j|\tilde{\Theta})^{\beta_i}}$$

$$\sum_i \frac{1}{\beta_i} = \sum_i \frac{\gamma_i p(X_i|\tilde{\Theta})^{\beta_i}}{\sum_j \gamma_j p(X_j|\tilde{\Theta})^{\beta_i}} = 1$$

Thus, we can write the bound as follows:

$$\Delta l^m \leq \log \frac{\sum_i \frac{1}{\beta_i} \frac{p(X_i|\Theta)^{\beta_i}}{p(X_i|\tilde{\Theta})^{\beta_i}}}{\sum_i \frac{1}{\beta_i}}$$

$$\Delta l^m \leq \log \sum_i \frac{1}{\beta_i} \left(\frac{p(X_i|\Theta)}{p(X_i|\tilde{\Theta})} \right)^{\beta_i}$$

Expanding the above to include the latent values, we obtain:

$$\Delta l^m \leq \log \sum_i \frac{1}{\beta_i} \left(\frac{\sum_{mc} p(m,c,X_i|\Theta)}{\sum_{mc} p(m,c,X_i|\tilde{\Theta})} \right)^{\beta_i}$$

However, the above expression is still undesirable since the power operation (i.e. $()^{\beta_i}$) applies to a mixture (over the summation over c and m). Let us consider bounding these terms to simplify them further. We will invoke Jensen again since the power operation (i.e. $pow(.,\beta_i)$ or $(.)^{\beta_i}$) is a convex function for $\beta_i \geq 1$. This permits us to upper bound the expressions involving the power operation and thus generates an overall upper bound on Δl^m. Manipulating the mixture of terms being taken to a power and applying Jensen, we obtain:

$$\left(\frac{\sum_{mc} p(m,c,X_i|\Theta)}{\sum_{mc} p(m,c,X_i|\tilde{\Theta})} \right)^{\beta_i} = \left(\sum_{mc} \frac{p(m,c,X_i|\Theta)}{\sum_{nd} p(n,d,X_i|\tilde{\Theta})} \right)^{\beta_i}$$

$$\left(\frac{\sum_{mc} p(m,c,X_i|\Theta)}{\sum_{mc} p(m,c,X_i|\tilde{\Theta})} \right)^{\beta_i} = \left(\sum_{mc} \frac{p(m,c,X_i|\tilde{\Theta})}{\sum_{nd} p(n,d,X_i|\tilde{\Theta})} \frac{p(m,c,X_i|\Theta)}{p(m,c,X_i|\tilde{\Theta})} \right)^{\beta_i}$$

$$\left(\frac{\sum_{mc} p(m,c,X_i|\Theta)}{\sum_{mc} p(m,c,X_i|\tilde{\Theta})} \right)^{\beta_i} \leq \sum_{mc} \frac{p(m,c,X_i|\tilde{\Theta})}{\sum_{nd} p(n,d,X_i|\tilde{\Theta})} \left(\frac{p(m,c,X_i|\Theta)}{p(m,c,X_i|\tilde{\Theta})} \right)^{\beta_i}$$

It is clear that the left hand side and the right hand side are equal at the old value of $\tilde{\Theta}$ ensuring that we have generated a variational bound that makes tangential contact appropriately. We can now plug in these upper bounds on the individual terms indexed by i and get an overall stricter upper bound on Δl^m:

$$\Delta l^m \leq \log \sum_i \frac{1}{\beta_i} \left(\frac{\sum_{mc} p(m,c,X_i|\Theta)}{\sum_{mc} p(m,c,X_i|\tilde{\Theta})} \right)^{\beta_i}$$

$$\Delta l^m \leq \log \sum_i \frac{1}{\beta_i} \sum_{mc} \frac{p(m,c,X_i|\tilde{\Theta})}{\sum_{nd} p(n,d,X_i|\tilde{\Theta})} \left(\frac{p(m,c,X_i|\Theta)}{p(m,c,X_i|\tilde{\Theta})} \right)^{\beta_i}$$

The above again holds for any arbitrary distribution as long as β_i are greater than one and their inverses sum to unity. If we more specifically assume that $p(m,c,X_i|\Theta)$ is in the exponential family, we can write it compactly as follows:

$$p(m,c,X_i|\Theta) = \alpha_{mc} \exp\left(\mathcal{A}_{mc}(X_i) + \Theta_{mc}^T X_i - \mathcal{K}_{mc}(\Theta_{mc}) \right)$$

In that case, the bound can be written more succinctly as:

$$\Delta l^m \leq \log \sum_{imc} \frac{1}{\beta_i} \frac{p(m,c,X_i|\tilde{\Theta})^{1-\beta_i}}{\sum_{nd} p(n,d,X_i|\tilde{\Theta})} \alpha_{mc}^{\beta_i} \exp\left(\beta_i \mathcal{A}_{mc}(X_i) + \beta_i \Theta_{mc}^T X_i - \beta_i \mathcal{K}_{mc}(\Theta_{mc}) \right)$$

This bound is variational and makes contact at the old $\tilde{\Theta}$ configuration in the space of pdfs. We can now invoke the reverse-Jensen inequality on the above log-sum *only one single time* to obtain a bound on the whole data set. We also have the freedom of adjusting the β_i (as long as they are greater than unity and their inverses sum to unity) to ensure that this bound is as tight as possible (i.e. select the β_i to minimize the resulting w_m values).

6.8 Applications

In the above we have only shown toy illustrations of the conditional hidden Markov models with CEM. Clearly, many important applications are now within reach and it would be interesting to investigate the performance on a real-world data set. Chapter 8 describes a behavior-imitation problem where the hidden Markov model is used to learn how an agent interacts with the outside world. This is basically a stimulus-response type of model that is trained from real data by observing human activity for several hours. Since we only wish to predict and resynthesize the agent's behavior conditioned on the outside world measurements, we train the HMM with a conditional likelihood criterion. The imitation learning application seems well suited to the CEM algorithm and we show real-world results using CEM instead of EM (after describing the implementation details of the experiment). However, the next chapter, namely Chapter 7, is the derivation of the reverse-Jensen inequality which was so far put forth without proof. The mathematical reader may find the derivation interesting while the more applied reader may wish to skip directly to Chapter 8 to see a real-world application of latent discriminative learning with structured graphical models.

Chapter 7

The Reverse Jensen Inequality

Difficulties strengthen the mind, as labor does the body.

Seneca, 3 B.C. - 65 A.D.

In this chapter we shall give a terse derivation of the reverse-Jensen inequality we used in Chapter 5. The derivation is made to generate bounds that are as tight as possible and thus involves many details. These are shown in full to permit future extensions and tightenings. We first review some literature in the contemporary mathematical inequalities community and show current converses and reversals of Jensen. These do not suit the requirements of discriminative learning since the functional form of the bounds is not easy to combine with regular Jensen or EM-type bounds and therefore we derive our own reversal.

7.1 Background in Inequalities and Reversals

The celebrated Jensen inequality has been formalized a century ago [99] and its roots trace even further back in history. This result has been at the heart of many inequalities in statistics [40] and mathematics [150]. For example, Holder's inequality can be easily derived from Jensen's inequality. Jensen in its simplest form states that a convex function of an expectation is upper bounded by the expectation of the convex function. An excellent review of Jensen's inequality, other inequalities, convex functions and various statistical applications can be found in Pecaric et al.'s text [150]. Furthermore, several converses and reversals of Jensen's inequality are provided in the text. Some are simple manipulations of Jensen's inequality (i.e. assuming that the mixture is weighted by negative scalars instead of positives ones). Others refinements require further assumptions and properties on the convex function and the elements it operates upon. Similarly, *Log sum inequality* and variants of Jensen are also discussed in [40] however many of these are usually just trivial consequences of Jensen's inequality. A more recent reversal of Jensen that has been important in statistics has been proposed in [46]. For instance, the Jensen inequality can be reversed if the function it is applied to has a limited range of variation or the elements it applies to are clamped to a limited domain. However, the bound proposed in [46] is fundamentally curvature based which can sometimes prevent tight bounds. The uses for [46] are to find general dual-sided bounds on Csiszar Φ-divergences in terms of the many other divergences in the family. These permit one to map between divergences, i.e. Kullback-Leibler divergence, harmonic divergence, variational divergence, Renyi-entropy, etc.

Unfortunately, the above bounds in the literature do not directly suit our purposes. They not only generate reversals which are not homogeneous with the regular Jensen inequality bounds (and therefore cannot be combined easily with the EM-type bounds), but they also put requirements on the elements in the expectation which are not appropriate for our pruposes. The elements in our log-sum problems are exponential family probability densities which have special properties but not necessarily the ones explored so far in the literature. We therefore start with a blank slate and derive our own customized reversal.

7.2 Derivation of the Reverse Jensen Inequality

We begin by recalling the inequality given without proof in the previous chapter:

$$\log \sum_{mn} \alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \Theta_m - \mathcal{K}_m(\Theta_m)) \leq \sum_m -w_m(\mathcal{A}_m(Y_m) + Y_m^T \Theta_m - \mathcal{K}_m(\Theta_m)) + k$$

$$\begin{array}{rcl}
k & = & \log p(X|\tilde{\Theta}) + \sum_m w_m(\mathcal{A}_m(Y_m) + Y_m^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m)) \\
Y_m & = & \frac{1}{w_m} \sum_n h_{mn} \left(\left. \frac{\partial \mathcal{K}_m(\Theta_m)}{\partial \Theta_m} \right|_{\tilde{\Theta}_m} - X_{mn} \right) + \left. \frac{\partial \mathcal{K}_m(\Theta_m)}{\partial \Theta_m} \right|_{\tilde{\Theta}_m} \\
w'_m & = & \min w'_m \text{ such that } \frac{1}{w'_m} \sum_n h_{mn} \left(\left. \frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m} \right|_{\tilde{\Theta}_m} - X_{mn} \right) + \left. \frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m} \right|_{\tilde{\Theta}_m} \in \frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m} \\
w_m & = & 4 \ G \left(\frac{\sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}}{2 \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}} \right) \ \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \ + w'_m \\
\mathcal{Z}_{mn} & = & \mathcal{K}''(\tilde{\Theta}_m)^{-1/2}(X_{mn} - \mathcal{K}'(\tilde{\Theta}_m)) \\
h_{mn} & = & \frac{\alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m))}{\sum_{mn} \alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m))} \\
G(\gamma) & = & \left\{ \begin{array}{ll} \gamma + \frac{1}{4 \log(6)} + \frac{25/36}{\log(6)^2} - 1/6 & \gamma \geq 1/6 \\ \frac{1}{4 \log(1/\gamma)} + \frac{(\gamma - 1)^2}{\log(\gamma)^2} & \gamma \leq 1/6 \end{array} \right.
\end{array}$$

We will now derive the reverse-Jensen bound above in the so-called double-mixture case. This derivation subsumes that of the reverse-Jensen bound for the single mixture case. This derivation will follow a somewhat general recipe for bounding quantities such as log-sums. It begins by making the variational bound touch the original function tangentially at the current operating point. We then find a mapping on the non-linear $\mathcal{K}$ functions into a quadratic space which makes the computations tractable. The bound is then simplified by noting its convexity properties. Subsequently, we note that a log-partition function of a Gibbs distribution arises (i.e. a log-sum of exponentiated linear terms) which needs to be bounded by a quadratic. This derivation is done in a one dimensional simple case and then generalized by sweeping the 1d bound up to the multidimensional case. Subsequently, curvature constraints are checked on the bound which are simpler to deal with and subsume the original inequality. Ultimately, a very simple formula for the bound's parameters arises which guarantees that it remains above the original log-sum function.

Again, we start with the log-sum which needs to be upper bounded:

$$\log \sum_{mn} \alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \Theta_m - \mathcal{K}_m(\Theta_m)) \leq \sum_m -w_m(\mathcal{A}(Y_m) + Y_m^T \Theta_m - \mathcal{K}_m(\Theta_m)) + k \quad (7.1)$$

At the current tangential contact point $\tilde{\Theta}$ of the variational bound, the two sides above are equal:

$$\log \sum_{mn} \alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m)) \;=\; \sum_m -w_m(\mathcal{A}(Y_m) + Y_m^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m)) + k$$

This allows us to isolate k:

$$k \;=\; \log \sum_m \sum_n \alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m)) \;+\; \sum_m w_m(\mathcal{A}(Y_m) + Y_m^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m))$$

Also, at the contact point $\tilde{\Theta}$, the gradients of both sides of Inequality 7.1 must be equal, again allowing us to isolate Y_m:

$$\sum_n h_{mn} \left(X_{mn} - \left.\frac{\partial \mathcal{K}_m(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} \right) \;=\; -w_m \left(Y_m - \left.\frac{\partial \mathcal{K}_m(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} \right) \quad \forall m$$

Now, if we reinsert the definitions for k and Y_m into Inequality 7.1 and rearrange terms we obtain the following expression below. Note, the only variables that remain to be computed are the w_m scalar values.

$$\begin{aligned}\sum\nolimits_m w_m \big(\mathcal{K}(\Theta_m) - \mathcal{K}(\tilde{\Theta}_m) - (\Theta_m - \tilde{\Theta}_m)^T \mathcal{K}'(\tilde{\Theta}_m)\big) \;\geq\; & \log \frac{\sum_{mn} \alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \Theta_m - \mathcal{K}_m(\Theta_m))}{\sum_{mn} \alpha_{mn} \exp(\mathcal{A}_m(X_{mn}) + X_{mn}^T \tilde{\Theta}_m - \mathcal{K}_m(\tilde{\Theta}_m))} \\ & + \sum\nolimits_{mn} h_{mn} (\Theta_m - \tilde{\Theta}_m)^T \big(\mathcal{K}'(\tilde{\Theta}_m) - X_{mn}\big)\end{aligned}$$

It is interesting to note that the terms in parentheses that are multiplying the w_m scalar values are Bregman distances (or Bregman divergences). This is a direct result of the convexity of the cumulant generating functions $\mathcal{K}(\Theta)$ in the exponential family. In the case of Gaussian distributions, these Bregman distances are a Euclidean distance metric with origin at $\tilde{\Theta}_m$. In the more general case of exponential family distributions, these Bregman distances are actually Kullback Leibler divergences from a distribution at Θ to one at the origin $\tilde{\Theta}$. Therefore, it is clear that the left hand side of the above is only zero at the contact point since Bregman distances are always positive except at the origin. This is encouraging, since it indicates that we should be able to obtain non-vacuous bounds with finite w_m values (i.e. due to the positivity of Bregman divergences).

At this point, we realize that it is difficult to try to solve for w_m for arbitrary $\mathcal{K}$ functions. We shall show next how we can avoid dealing explicitly with these functions and map all cases to the $\mathcal{K}$ functions that are of quadratic form.

7.3 Mapping to Quadratics

For brevity we begin by defining the following shorthand for the Bregman distances that arose earlier:

$$\mathcal{F}_m(\Theta_m) \;=\; \mathcal{K}(\Theta_m) - \mathcal{K}(\tilde{\Theta}_m) - (\Theta_m - \tilde{\Theta}_m)^T \mathcal{K}'(\tilde{\Theta}_m)$$

The $\mathcal{F}$ functions are convex and have a minimum (which is zero) at $\tilde{\Theta}_m$. We can now replace $\mathcal{K}$ functions with $\mathcal{F}$ to simplify the above expressions obtaining the following:

$$\sum_m w_m \mathcal{F}(\Theta_m) \;\geq\; \log \frac{\sum_m \sum_n \exp\left(D_{mn} + (\Theta_m - \tilde{\Theta}_m)^T Z_{mn} - \mathcal{F}(\Theta_m)\right)}{\sum_m \sum_n \exp\left(D_{mn} + \tilde{\Theta}_m^T Z_{mn} - \mathcal{F}(\tilde{\Theta}_m)\right)} - \sum_m \sum_n h_{mn} (\Theta_m - \tilde{\Theta}_m)^T Z_{mn}$$

Above, we have defined the constant scalars D_{mn} and constant vectors Z_{mn} as follows:

$$\begin{aligned} D_{mn} &= \log \alpha_{mn} + \mathcal{A}_m(X_{mn}) - \mathcal{K}_m(\tilde{\Theta}_m) + \tilde{\Theta}_m^T X_{mn} \\ Z_{mn} &= X_{mn} - \mathcal{K}'(\tilde{\Theta}_m) \end{aligned}$$

We can immediately recognize the constants above are actually closely related to the responsibilities as follows:

$$h_{mn} = \frac{\exp D_{mn}}{\sum_{mn} \exp D_{mn}}$$

Which permits the further simplification:

$$\sum_m w_m \mathcal{F}(\Theta_m) \geq \log \sum_{mn} h_{mn} \exp\left((\Theta_m - \tilde{\Theta}_m)^T Z_{mn} - \mathcal{F}(\Theta_m)\right) - \sum_{mn} h_{mn}(\Theta_m - \tilde{\Theta}_m)^T Z_{mn} \quad (7.2)$$

Through straightforward redefinitions, we seem to have a simplified expression. However, we still have hidden and possibly complicated non-linearities in Θ-space due to the $\mathcal{F}$ functions preventing an easy solution for the w_m. This situation can be alleviated if we transform the space. Let us define a mapping from Θ-space to Φ-space such that the non-linear $\mathcal{F}$ functions are simple quadratics:

$$\mathcal{F}_m(\Theta_m) = \mathcal{G}_m(\Phi_m) = \frac{1}{2}(\Phi_m - \tilde{\Theta}_m)^T(\Phi_m - \tilde{\Theta}_m)$$

The above is a mapping from a convex non-negative function to another convex non-negative function which is always possible by a stretching of the axes or a change of variable. Geometrically, we are mapping a bowl into another bowl. Both maintain their minimum at $\tilde{\Theta}$ which is 0. Thus, the origin can be consider to remain fixed at $\tilde{\Theta}$ as we stretch the domain or perform a change of variables. If the convex function $\mathcal{F}(\Theta)$ is restricted in its range, we can consider a convex-hull restriction on the domain of the quadratic[1] to make it obey the same range. We begin by using this transformation on the left hand side of Equation 7.2 to produce the following:

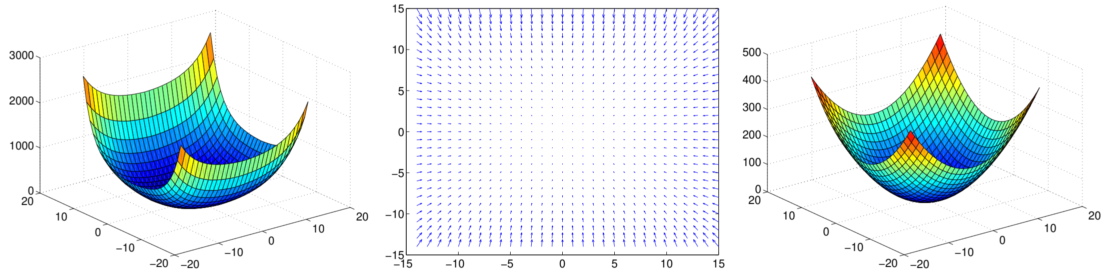
$$\sum_m w_m \mathcal{G}(\Phi_m) \geq \log \sum_{mn} h_{mn} \exp\left((\Theta_m - \tilde{\Theta}_m)^T Z_{mn} - \mathcal{F}(\Theta_m)\right) - \sum_{mn} h_{mn}(\Theta_m - \tilde{\Theta}_m)^T Z_{mn} \quad (7.3)$$

Figure 7.1 portrays this mapping in the 2D case. Essentially, we are performing a convexity-preserving map. Here, an arbitrary (strictly) convex function is being turned into a quadratic. The quadratic arises by default for the $\mathcal{F}(\Theta)$ functions when we are dealing with exponential family members in the log-sum that are Gaussians (with variable means and fixed covariance). Therefore, we can transform any e-family member to a Gaussian for the purposes of computing the w_m parameters of the reverse-Jensen inequality. Before we transform the rest of the expression out of Θ into a quadratic domain of Φ, we will point out an important property associated with the above convexity-preserving map.

7.4 An Important Condition on the E-Family

We shall now identify an important condition concerning the above mapping to a quadratic which we will put forth here without proof. Instead, we show that it holds for a wide variety of exponential family models: for example the Gaussian mean, Gaussian covariance, multinomial, gamma, Poisson, and exponential distributions.

[1] A quadratic has an unrestricted range $\in [0, \infty)$ and hence is a superset.



(a) General Convex Bowl in Θ (b) Mapping $\Theta \rightarrow \Phi$ (c) Quadratic Bowl in Φ

Figure 7.1: Convexity-Preserving Map. Given a convex bowl or Bregman divergence in Θ-space as in (a), we can always find a displacement map from $\Theta \rightarrow \Phi$ that will stretch axes as in (b) to obtain a quadratic (Euclidean distance) bowl in Φ-space in (c). Thus, a change of variables exists which permits a simpler solution in a quadratic space.

Lemma 2 *Under mild conditions and for standard exponential family models, we can always find a change of variable mapping from the variable Θ to Φ given by equating the Bregman distance at $\tilde{\Theta}$ arising from the cumulant generating function $\mathcal{K}(\Theta)$ to a quadratic function in Φ with a minimum at $\tilde{\Theta}$:*

$$\frac{1}{2}(\Phi-\tilde{\Theta})^T(\Phi-\tilde{\Theta}) \quad = \quad \mathcal{K}(\Theta)-\mathcal{K}(\tilde{\Theta})-(\Theta-\tilde{\Theta})^T\mathcal{K}'(\tilde{\Theta})$$

Select an arbitrary point Θ^* in the domain of Θ. From all possible mappings induced by the above, there will always be a mapping from $\Theta \rightarrow \Phi$ such that the following upper bound holds globally:

$$\begin{aligned}\left(\Phi-\tilde{\Theta}\right)^T\left(\mathcal{K}''(\tilde{\Theta})\right)^{-1/2}\left(\mathcal{K}'(\Theta^*)-\mathcal{K}'(\tilde{\Theta})\right) \quad &\geq \quad \left(\Theta-\tilde{\Theta}\right)^T\left(\mathcal{K}'(\Theta^*)-\mathcal{K}'(\tilde{\Theta})\right)\\ &\quad -\left(\mathcal{K}(\Theta)-\mathcal{K}(\tilde{\Theta})-(\Theta-\tilde{\Theta})^T\mathcal{K}'(\tilde{\Theta})\right)\end{aligned}$$

The above holds for any choice of the arbitrary point Θ^* in the domain of Θ (Θ^* can be any point in the domain of $\mathcal{K}$). Furthermore, the upper bound makes tangential contact at $\Theta = \tilde{\Theta}$ (Θ can be any point in the domain of $\mathcal{K}$).

The property in Lemma 2 is guaranteed to hold at the contact point when $\Theta = \tilde{\Theta}$ as both sides of the inequality go to zero. Furthermore, it is easy to show that the right hand side will eventually diverge negatively as we move away from the contact point $\tilde{\Theta}$ since the right hand side is concave and the left hand side is increasing away from the contact point. Thus, the bound is also trivial to guarantee at distant values of Θ. An exact proof is elusive since the above property does not arise only from the convexity of $\mathcal{K}$ but some more specific attributes. For instance, the $\mathcal{K}$ cumulant generating function is the logarithm of a Laplace transform and this route may be used to construct an argument. Alternatively, other specific e-family assumptions such as *steepness* [9] may be useful. Instead of deriving a formal proof for when the condition is valid, we will instead give visual examples of it holding for many standard distributions in the exponential family. For the unidimensional case, the mapping is unique (modulo a mirror flip around the origin) and we can easily compute the explicit mapping for Φ as follows:

$$\Phi \quad = \quad \tilde{\Theta}+\mathrm{sign}(\Theta-\tilde{\Theta})\sqrt{2}\sqrt{\mathcal{K}(\Theta)-\mathcal{K}(\tilde{\Theta})-(\Theta-\tilde{\Theta})\mathcal{K}'(\tilde{\Theta})}$$

For the details of the cumulant generating functions and their duals $\mathcal{A}(X)$ for these distributions refer to Table 5.1 in the previous Chapter. In Figures 7.2, 7.3, 7.4, 7.5, 7.6 and 7.7 we show examples of the bounds implied by Lemma 2. We have thus argued for a convexity-preserving map that also has a powerful bound associated with it. We will now use this property on our formula for w_m to drastically simplify it.

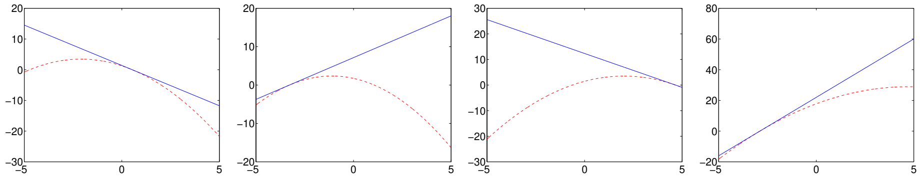


Figure 7.2: **Lemma Bound for the Gaussian Mean Distribution.** Four random configurations for $\tilde{\Theta}$ and Θ^* are shown. The solid line is the lemma's upper bound above the dashed red line. The Gaussian mean case is the most trivial one to consider and it is trivial to construct a mapping from $\Theta \rightarrow \Phi$ which is only affine. This is Gaussians have a quadratic $\mathcal{K}$ function to begin with (regardless of dimensionality). It is straightforward to see that for various choices of $\tilde{\Theta}$ and Θ^*, we have an upper bound as desired.

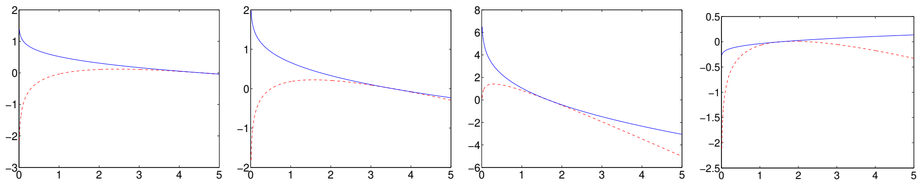


Figure 7.3: **Lemma Bound for the Gaussian Covariance Distribution.** Four random configurations for $\tilde{\Theta}$ and Θ^* are shown. The solid line is the lemma's upper bound above the dashed red line. If we consider varying the covariance of the Gaussian, $\mathcal{K}$ is not quadratic however the mapping is possible. It is straightforward to see that for various choices of $\tilde{\Theta}$ and Θ^*, we have an upper bound as desired.

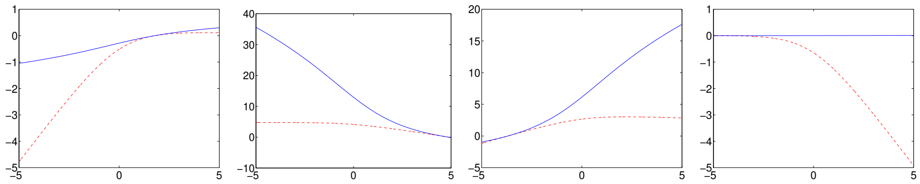


Figure 7.4: **Lemma Bound for the Multinomial Distribution.** Four random configurations for $\tilde{\Theta}$ and Θ^* are shown. The solid line is the lemma's upper bound above the dashed red line.

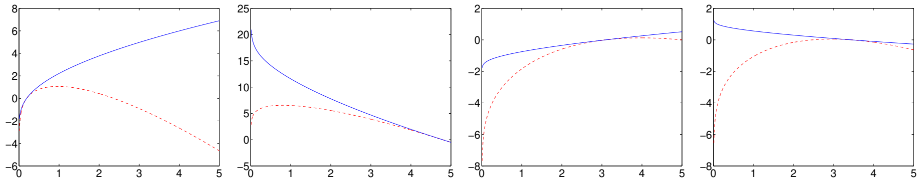


Figure 7.5: **Lemma Bound for the Gamma Distribution.** Four random configurations for $\tilde{\Theta}$ and Θ^* are shown. The solid line is the lemma's upper bound above the dashed red line.

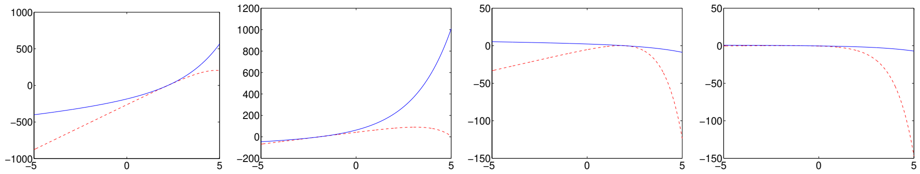


Figure 7.6: **Lemma Bound for the Poisson Distribution.** Four random configurations for $\tilde{\Theta}$ and Θ^* are shown. The solid line is the lemma's upper bound above the dashed red line.

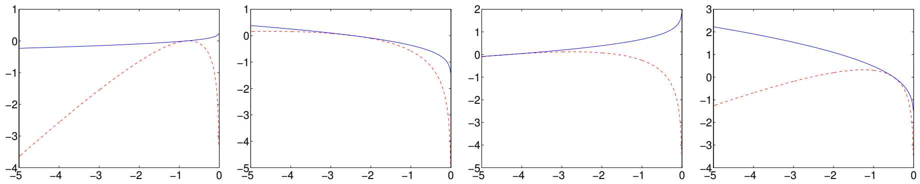


Figure 7.7: **Lemma Bound for the Exponential Distribution.** Four random configurations for $\tilde{\Theta}$ and Θ^* are shown. The solid line is the lemma's upper bound above the dashed red line.

7.5 Applying the Mapping

We shall now invoke the above Lemma on the current Equation 7.3 for w_m we have progressed to:

$$\sum_m w_m \mathcal{G}(\Phi_m) \geq \log \sum_{mn} h_{mn} \exp\left((\Theta_m - \tilde{\Theta}_m)^T Z_{mn} - \mathcal{F}(\Theta_m)\right) - \sum_{mn} h_{mn}(\Theta_m - \tilde{\Theta}_m)^T Z_{mn}$$

We recognize in the above that the $\mathcal{F}(\Theta_m)$ are the Bregman distances arising from the cumulant generating functions. Furthermore, the Z_{mn} can be represented as $\mathcal{K}'(\Theta^*_{mn}) - \mathcal{K}'(\tilde{\Theta}_m)$ for a particular choice of Θ^*_{mn} (due to the data vectors X_{mn} living in the gradient space $\mathcal{K}'$). Thus, the terms being exponentiated in Equation 7.3 are exactly the right hand side of the bound in Lemma 2. Thus, we replace them with their upper bounds to obtain a *stricter* guaranteed constraint on the w_m values:

$$\sum_m w_m \mathcal{G}(\Phi_m) \geq \log \sum_{mn} h_{mn} \exp\left((\Phi_m - \tilde{\Theta}_m)^T \mathcal{K}''(\tilde{\Theta}_m)^{(-1/2)} Z_{mn}\right) - \sum_{mn} h_{mn}(\Theta_m - \tilde{\Theta}_m)^T Z_{mn}$$

For simplicity, we shall define the following surrogate 'whitened' data vectors which are translated and scaled versions of the original data X_{mn}:

$$\mathcal{Z}_{mn} = \mathcal{K}''(\tilde{\Theta}_m)^{-1/2} Z_{mn}$$

At this stage have done away with all but the last few Θ terms at the end of the right hand side:

$$\sum_m w_m \mathcal{G}(\Phi_m) \geq \log \sum_{mn} h_{mn} \exp\left((\Phi_m - \tilde{\Theta}_m)^T \mathcal{Z}_{mn}\right) - \sum_{mn} h_{mn}(\Theta_m - \tilde{\Theta}_m)^T Z_{mn} \quad (7.4)$$

Before we also bound away the last terms depending directly on Θ and replace them with Φ, we shall invoke a constraint that was dictated in the definition of the bounds.

7.6 Invoking Constraints on Virtual Data

Recall that the bounds generate virtual data points Y_m and these were constrained (as the original X_{mn} data points) to live in the gradient space of the cumulant generating function. From the original definition of the exponential family we must have the following requirement:

$$Y_m \in \frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}$$

This makes perfect sense since the e-family distributions we will be generating to bound the log-sum mixture must be valid and it is a common constraint that the data terms are in the gradient space of the cumulant generating function. Furthermore, Y_m is acted upon by the $\mathcal{A}(X)$ function and the domain of the function only admits vectors in the gradient space as well. Let us insert the definition of Y_m we have derived into this constraint:

$$\frac{1}{w_m} \sum_n h_{mn} \left(\left.\frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} - X_{mn} \right) + \left.\frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} \in \frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}$$

Clearly, we need a w_m such that the above is satisfied. Furthermore, it is evident that setting $w_m \to \infty$ will *always* satisfy the above. The smallest w_m that satisfies the above will be called

w'_m. By convexity of the gradient space $\mathcal{K}'$, any value for w_m from $[w'_m, \infty)$ will thus satisfy the constraint above. In *practice*, it is relatively easy to compute w'_m analytically for exponential family distributions using the above formula. Let us assume that we find this w'_m and then have the particular gradient space vector it generates be a $\mathcal{K}'(\Theta^*_m)$ in the gradient space. This gives us the following equation:

$$\frac{1}{w'_m}\sum_n h_{mn}\left(\left.\frac{\partial\mathcal{K}(\Theta_m)}{\partial\Theta_m}\right|_{\tilde{\Theta}_m} - X_{mn}\right) + \left.\frac{\partial\mathcal{K}(\Theta_m)}{\partial\Theta_m}\right|_{\tilde{\Theta}_m} = \left.\frac{\partial\mathcal{K}(\Theta_m)}{\partial\Theta_m}\right|_{\Theta^*_m}$$

We now rewrite the Equation above as follows:

$$\begin{aligned}
\left(\frac{\sum_n h_{mn}}{w'_m}+1\right)\mathcal{K}'_m(\tilde{\Theta}_m) - \frac{1}{w'_m}\sum_n h_{mn}X_{mn} &= \mathcal{K}'_m(\Theta^*_m)\\
\frac{\sum_n h_{mn}}{w'_m}\mathcal{K}'_m(\tilde{\Theta}_m) - \frac{1}{w'_m}\sum_n h_{mn}X_{mn} &= \mathcal{K}'_m(\Theta^*_m) - \mathcal{K}'_m(\tilde{\Theta}_m)\\
\frac{1}{w'_m}\sum_n h_{mn}\left(\mathcal{K}'_m(\tilde{\Theta}_m) - X_{mn}\right) &= \mathcal{F}'_m(\Theta^*_m)\\
-\sum_n h_{mn}Z_{mn} &= w'_m\mathcal{F}'_m(\Theta^*_m)
\end{aligned}$$

Thus, we see that the scaled negated mixture of Z_{mn} vectors lies in the gradient space of $\mathcal{F}(\Theta_m)$. We shall now consider the formula for w_m we have up to this point, namely Equation 7.4 and replace the above into it:

$$\begin{aligned}
\sum_m w_m\mathcal{G}(\Phi_m) &\geq \log\sum_{mn} h_{mn}\exp\left((\Phi_m-\tilde{\Theta}_m)^T\mathcal{Z}_{mn}\right) - \sum_{mn} h_{mn}Z^T_{mn}(\Theta_m-\tilde{\Theta}_m)\\
\sum_m w_m\mathcal{G}(\Phi_m) &\geq \log\sum_{mn} h_{mn}\exp\left((\Phi_m-\tilde{\Theta}_m)^T\mathcal{Z}_{mn}\right) + \sum_m w'_m\mathcal{F}'_m(\Theta^*_m)^T(\Theta_m-\tilde{\Theta}_m)
\end{aligned}$$

Next we subtract $\sum_m w'_m\mathcal{F}_m(\Theta_m)$ from both sides obtaining:

$$\begin{aligned}
\sum_m w_m\mathcal{G}(\Phi_m) - \sum_m w'_m\mathcal{F}_m(\Theta_m) \geq\ & \log\sum_{mn} h_{mn}\exp\left((\Phi_m-\tilde{\Theta}_m)^T\mathcal{Z}_{mn}\right)\\
&+\sum_m w'_m\left(\mathcal{F}'_m(\Theta^*_m)^T(\Theta_m-\tilde{\Theta}_m) - \mathcal{F}_m(\Theta_m)\right)
\end{aligned}$$

The terms multiplying the w'_m in the right hand side are actually the lower bounds in Lemma 2. We can invoke an upper bound on them and get an *even stricter* constraint on the formula for w_m:

$$\begin{aligned}
\sum_m w_m\mathcal{G}(\Phi_m) - \sum_m w'_m\mathcal{F}_m(\Theta_m) \geq\ & \log\sum_{mn} h_{mn}\exp\left((\Phi_m-\tilde{\Theta}_m)^T\mathcal{Z}_{mn}\right)\\
&+\sum_m w'_m\left(\mathcal{F}'_m(\Theta^*_m)^T\mathcal{K}''(\tilde{\Theta}_m)^{(-1/2)}(\Phi_m-\tilde{\Theta}_m)\right)
\end{aligned}$$

Now, in the right hand side, we re-replace the $w'_m\mathcal{F}'_m(\Theta^*_m)$ (it was only introduced to reflect similarity with the Lemma) with its original definition:

$$\begin{aligned}
\sum_m w_m\mathcal{G}(\Phi_m) - \sum_m w'_m\mathcal{F}_m(\Theta_m) \geq\ & \log\sum_{mn} h_{mn}\exp\left((\Phi_m-\tilde{\Theta}_m)^T\mathcal{Z}_{mn}\right)\\
&-\sum_{mn} h_{mn}Z^T_{mn}\mathcal{K}''(\tilde{\Theta}_m)^{(-1/2)}(\Phi_m-\tilde{\Theta}_m)
\end{aligned}$$

The above is further simplified when we use the definition for $\mathcal{Z}_{mn}$ and the equality $\mathcal{F}(\Theta) = \mathcal{G}(\Phi)$:

$$\sum_m (w_m - w'_m)\mathcal{G}(\Phi_m) \quad \geq \quad \log \sum_{mn} h_{mn} \exp\left((\Phi_m - \tilde{\Theta}_m)^T \mathcal{Z}_{mn}\right) - \sum_{mn} h_{mn} \mathcal{Z}_{mn}^T(\Phi_m - \tilde{\Theta}_m)$$

At this point we have *succeeded in eliminating* the complexities that arise from the nonlinear Θ representation of the exponential family and we only have *quadratic* $\mathcal{G}$ functions. Effectively, all e-family computations are now computationally equivalent to the simple Gaussian mean case with its quadratic cumulant generating function. We should also recognize the logarithm is over a summation of exponentiated linear models. This is equivalent to the *log-partition function of a Gibbs distribution* and also similar to the logistic function. Thus, we can map any reverse-Jensen problem into finding quadratic upper bounds in Φ space on the log-partition function of a Gibbs distribution.

We next provide a very simple formula for w_m which was originally proposed in [94]. This bound here is based on a direct *curvature check* and is not as tight as possible. A longer and more involved derivation follows in the next section (Section 7.8) which will generate a much tighter bound.

7.7 A Simple yet Loose Bound

To solve for the w_m we are effectively bounding the log-partition of a Gibbs distribution with a quadratic function. This is an unusual side effect of solving the reverse-Jensen bounds. Earlier work in Gaussian Processes[65] also necessitated quadratic bounds on this form of log-partition function. The doctoral dissertation of M. Gibbs involved finding such a bound for use in Gaussian Processes however the bound that was solved for was approximate and not provable. Here, we shall show how to arrive at a guaranteed bound for this log-partition function which can ultimately be used to compute the w_m terms of the reverse-Jensen bound. The derivations below are simple and the resulting formula is guaranteed. However, we shall employ a crude method for obtaining a bound, namely a direct *curvature check*. In the following section we will refine this crude manipulation and obtain a tighter bound.

For convenience, we will now work with a surrogate variable r_m instead of w_m where $r_m = w_m - w'_m$. We therefore currently have the following equation:

$$\sum_m r_m \mathcal{G}(\Phi_m) \geq \quad \log \textstyle\sum_{mn} h_{mn} \exp\left((\Phi_m - \tilde{\Theta}_m)^T \mathcal{Z}_{mn}\right) - \textstyle\sum_{mn} h_{mn} \mathcal{Z}_{mn}^T(\Phi_m - \tilde{\Theta}_m)$$

For clarity, let us define the right hand side as a function of Φ:

$$L(\Phi) \quad := \quad \log \sum_{mn} h_{mn} \exp\left((\Phi_m - \tilde{\Theta}_m)^T \mathcal{Z}_{mn}\right) - \sum_{mn} h_{mn} \mathcal{Z}_{mn}^T(\Phi_m - \tilde{\Theta}_m)$$

Similarly, we can define $H(\Phi)$ as the left hand side of the above as follows:

$$\begin{aligned} H(\Phi) \quad &:= \quad \sum_m r_m \mathcal{G}(\Phi_m) \\ &= \quad \frac{1}{2} \sum_m r_m \left(\Phi_m - \tilde{\Theta}_m\right)^T \left(\Phi_m - \tilde{\Theta}_m\right) \end{aligned}$$

It should be evident that both $L(\Phi)$ and $H(\Phi)$ are zero and have zero gradients at the contact point when $\Theta = \tilde{\Theta}$. We wish to upper bound $L(\Phi)$ with $H(\Phi)$ (a quadratic) which makes tangential contact at $\Phi = \tilde{\Phi} = \tilde{\Theta}$. This requires finding valid r_m scalars which satisfy the following inequality at all values of the Φ parameter:

$$H(\Phi) \quad \geq \quad L(\Phi)$$

Instead of working in Φ-space, we can simplify things through a change of variable (a mere translation) and work in Ψ-space where we define $\Psi_m = \Phi_m - \tilde{\Phi}_m$. In other words, find the r_m that satisfy:

$$H(\Psi) \quad \geq \quad L(\Psi)$$

Where, naturally, we now have:

$$\begin{aligned} H(\Psi) &= \frac{1}{2}\sum_m r_m \Psi_m^T \Psi_m \\ L(\Psi) &= \log\left(\sum_{mn} h_{mn} \exp(\mathcal{Z}_{mn}^T \Psi_m)\right) - \sum_{mn} h_{mn} \mathcal{Z}_{mn}^T \Psi_m \end{aligned}$$

It should be noted that both $H(\Psi)$ and $L(\Psi)$ have a zero-valued minimum at $\Psi = 0$. This form of tangential contact between $H(\Psi)$ and $L(\Psi)$ is necessary if we are to have a variational bound. Note, here Ψ vector is a stacked version of all the individual Ψ_m vectors:

$$\Psi \quad = \quad [\Psi_1 \; \Psi_2 \; \ldots \; \Psi_m \; \ldots \; \Psi_M]^T$$

It is now trivial to take second derivatives of both sides of Equation 7.5 and get bounds on curvature. This effectively provides a Loewner ordering on the curvature matrices which indicates that $r_m = \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}$ which is a very simple yet typically loose bound. More specifically, we obtain:

$$\begin{aligned} \frac{\partial^2}{\partial \Psi^2} H(\Psi) &\geq \frac{\partial^2}{\partial \Psi^2} L(\Psi) \\ \begin{bmatrix} r_1 I & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & r_M I \end{bmatrix} &\geq \begin{bmatrix} \frac{\sum_n h_{1n} \exp(\mathcal{Z}_{1n}^T \Psi) \mathcal{Z}_{1n} \mathcal{Z}_{1n}^T}{\sum_{nm} h_{mn} \exp(\mathcal{Z}_{mn}^T \Psi)} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{\sum_n h_{1n} \exp(\mathcal{Z}_{Mn}^T \Psi) \mathcal{Z}_{Mn} \mathcal{Z}_{Mn}^T}{\sum_{nm} h_{mn} \exp(\mathcal{Z}_{mn}^T \Psi)} \end{bmatrix} \\ & - \begin{bmatrix} \frac{\sum_n h_{1n} \exp(\mathcal{Z}_{1n}^T \Psi) \mathcal{Z}_{1n}}{\sum_{nm} h_{mn} \exp(\mathcal{Z}_{mn}^T \Psi)} \\ \vdots \\ \frac{\sum_n h_{1n} \exp(\mathcal{Z}_{Mn}^T \Psi) \mathcal{Z}_{Mn}}{\sum_{nm} h_{mn} \exp(\mathcal{Z}_{mn}^T \Psi)} \end{bmatrix} \begin{bmatrix} \frac{\sum_n h_{1n} \exp(\mathcal{Z}_{1n}^T \Psi) \mathcal{Z}_{1n}}{\sum_{nm} h_{mn} \exp(\mathcal{Z}_{mn}^T \Psi)} \\ \vdots \\ \frac{\sum_n h_{1n} \exp(\mathcal{Z}_{Mn}^T \Psi) \mathcal{Z}_{Mn}}{\sum_{nm} h_{mn} \exp(\mathcal{Z}_{mn}^T \Psi)} \end{bmatrix}^T \end{aligned}$$

The subtractive outer-product can be ignored to obtain a stricter bound. Furthermore, by inspection (i.e. realizing that a convex combination of outer products is bounded by its max), we can clearly see that:

$$\frac{\sum_n h_{mn} \exp(\mathcal{Z}_{mn}^T \Psi) \mathcal{Z}_{mn} \mathcal{Z}_{mn}^T}{\sum_{nm} h_{mn} \exp(\mathcal{Z}_{mn}^T \Psi)} \leq \max_n \mathcal{Z}_{mn} \mathcal{Z}_{mn}^T$$

In addition, we now that the trace of a matrix multiplied by identify is greater than the matrix itself in the Loewner ordering sense:

$$\begin{aligned} trace(A)I &\geq A \\ \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} I &\geq \mathcal{Z}_{mn} \mathcal{Z}_{mn}^T \end{aligned}$$

This simplifies the above into:

$$\begin{aligned} \frac{\partial^2}{\partial \Psi^2} H(\Psi) &\geq \frac{\partial^2}{\partial \Psi^2} L(\Psi) \\ \begin{bmatrix} r_1 I & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & r_M I \end{bmatrix} &\geq \begin{bmatrix} \max_n \mathcal{Z}_{1n}^T \mathcal{Z}_{1n} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathcal{Z}_{Mn}^T \mathcal{Z}_{Mn} \end{bmatrix} \end{aligned}$$

Giving the following settings for the individual r_m:

$$r_m = \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad \forall\ m$$

Unfortunately, this bound is not quite as adaptive as we would like since it does not depend on the h_{mn} values. In other words, a $\mathcal{Z}_{mn}$ sample might have extremely low weight (i.e. $h_{mn} \rightarrow 0$) which means we should almost ignore it. Thus, we would like to have a bound that depends on the h_{mn} terms as well.

7.8 A Tighter Bound

Thus, it is preferable to get tighter bounds which can be done if we avoid this quick curvature test. The bounds should still be simple to allow easy implementation. Let us begin by manipulating $L(\Psi)$ more conservatively and obtain intermediate variational upper bounds on it that are tighter than the curvature check. If $H(\Psi)$ is greater than this upper bound, it must be an upper bound on $L(\Psi)$ itself. First, let us pull the additive linear term into the logarithm as follows:

$$L(\Psi) = \log \sum_{mn} h_{mn} \exp \left(\Psi_m^T \mathcal{Z}_{mn} - \sum_{mn} h_{mn} \Psi_m^T \mathcal{Z}_{mn} \right)$$

Now, let use define a stacked version of the $\mathcal{Z}_{mn}$ vectors as follows:

$$\mathbf{Z}_{mn} = [\vec{0}\ \vec{0}\ \ldots\ \mathcal{Z}_{mn}\ \ldots\ \vec{0}]^T$$

The $\mathbf{Z}_{mn}$ vector is of the same dimensionality as the Ψ vector and is obtained by padding the corresponding $\mathcal{Z}_{mn}$ with a total of $M-1$ zero-vectors. Therefore, we naturally have:

$$\mathbf{Z}_{mn}^T \Psi = \mathcal{Z}_{mn}^T \Psi_m$$

This allows us to rewrite the above as:

$$L(\Psi) = \log \sum_{mn} h_{mn} \exp \left(\Psi^T \left(\mathbf{Z}_{mn} - \sum_{mn} h_{mn} \mathbf{Z}_{mn} \right) \right)$$

Again, to further simplify notation, define the following vectors:

$$\mathbf{U}_{mn} = \mathbf{Z}_{mn} - \sum_{mn} h_{mn}\mathbf{Z}_{mn}$$

This lets us rewrite the $L(\Psi)$ function compactly as:

$$L(\Psi) = \log \sum_{mn} h_{mn} \exp\left(\Psi^T \mathbf{U}_{mn}\right)$$

To compute our reverse-Jensen bound, we must set the scalars r_m such that the inequality $H(\Psi) \geq L(\Psi)$ holds for all values of Ψ. More specifically we need:

$$\frac{1}{2}\sum_m r_m \Psi_m^T \Psi_m \geq \log \sum_{mn} h_{mn} \exp\left(\Psi^T \mathbf{U}_{mn}\right) \quad \forall\, \Psi$$

Selecting r_m values such that the above holds for *any* Ψ vector requires considering all Ψ configurations in the whole multidimensional Euclidean space. For now, we shall simplify this multidimensional problem by only consider a one-dimensional ray in Ψ-space.

7.8.1 Bounding a One Dimensional Ray

However, for now, let us consider guaranteeing the bound when Ψ is along a certain given direction, i.e. $\hat{\Psi}$. Therefore, we can write the Ψ vector as:

$$\Psi = \beta\hat{\Psi}$$

Thus, our $L(\Psi)$ function becomes:

$$L(\beta\hat{\Psi}) = \log \sum_{mn} h_{mn} \exp\left(\beta\hat{\Psi}^T \mathbf{U}_{mn}\right)$$

Now, let us see if we can find a quadratic bound on the above uni-dimensional function alone. In other words:

$$\frac{1}{2} q_{\hat{\Psi}} \beta^2 \geq \log \sum_{mn} h_{mn} \exp\left(\beta\hat{\Psi}^T \mathbf{U}_{mn}\right) \quad \forall\, \beta \tag{7.5}$$

In the above, we have to find a scalar $q_{\hat{\Psi}}$ that satisfies the inequality for all β. The subscript on the q indicates that it corresponds to a certain choice of direction $\hat{\Psi}$. This will give us a quadratic bound along this directional slice. The hope is that these individual uni-dimensional bounds can be aggregated and then used to find the required $H(\Psi)$ in the full multi-dimensional problem. For convenience, define the scalars v_{mn} as follows:

$$v_{mn} = \hat{\Psi}^T \mathbf{U}_{mn}$$

At this point, however, the problem has simplified to solving the following one-dimensional bound for a value of $q_{\hat{\Psi}}$:

$$\frac{1}{2} q_{\hat{\Psi}} \beta^2 \geq \log \sum_{mn} h_{mn} \exp\left(\beta v_{mn}\right) \tag{7.6}$$

It should be evident from the definitions of v_{mn} (or more specifically the original definition of $\mathbf{U}_{mn}$) that the following holds:

$$\sum_{mn} h_{mn} v_{mn} \quad = \quad 0$$

Consequently, when β is zero, both the left hand side above and the right hand side are zero and their derivatives are zero too (i.e. tangential contact). This one-dimensional log Gibbs partition function is depicted in Figure 7.8 for various random configurations of the v_{mn} and h_{mn} as well as the lower bounds that arise from the epigraph for visualization. These types of structures need to be upper bounded by a quadratic.

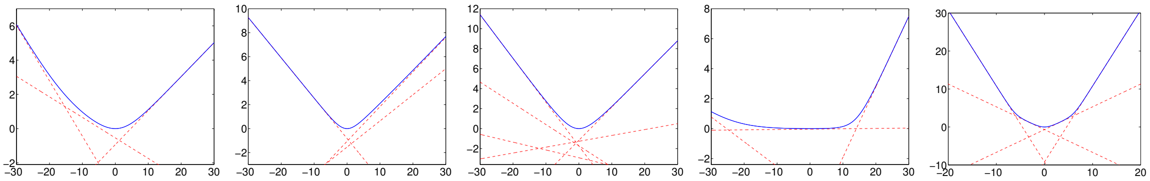


Figure 7.8: One Dimensional Log Gibbs Partition Functions. The solid line is the Gibbs partition function which is lower bounded by its epigraph (dashed red lines). These were plotted from $l(\beta) = \log \sum_{mn} h_{mn} \exp(\beta v_{mn})$ for random selections of h_{mn} and v_{mn}.

For brevity, we will use $l(\beta)$ to denote the right hand side of Equation 7.5.

$$l(\beta) \quad = \quad \log \sum_{mn} h_{mn} \exp\left(\beta v_{mn}\right)$$

Instead of working directly with $l(\beta)$, we will try to find an intermediate upper bound on this quantity which has the following simpler form:

$$l(\beta) \quad \leq \quad \log\left(\gamma \exp\left(\beta u\right) + \gamma \exp\left(-\beta u\right) + 1 - 2\gamma\right) \tag{7.7}$$

This intermediate bound is interesting since it serves to symmetrize $l(\beta)$ such that the sign of β is no longer important. This intermediate has two scalar parameters: u and γ. The right hand term above is a variational bound which makes tangential contact with the left hand side at $\beta = 0$. In other words, both sides are zero and have zero gradient at $\beta = 0$. We must now find legitimate parameters u and γ which will guarantee that the right hand side's term is an upper bound. To do so, it suffices to find parameters which guarantee that the curvature of the right hand term is an upper bound on the curvature of the left hand term (since the 0th and 1st order terms are equal at $\beta = 0$). This is because of the flat tangential contact at $\beta = 0$. Thus, we need to guarantee the following (this is the above Inequality 7.7 with the logarithm extracted out):

$$\sum_{mn} h_{mn} \exp\left(\beta v_{mn}\right) \quad \leq \quad \gamma \exp\left(\beta u\right) + \gamma \exp\left(-\beta u\right) + 1 - 2\gamma$$

Taking second derivatives with respect to β of both sides, we obtain the following (this is a stricter curvature test for the above inequality):

$$\sum_{mn} h_{mn} v_{mn}^2 \exp\left(\beta v_{mn}\right) \quad \leq \quad \gamma \exp\left(\beta u\right) u^2 + \gamma \exp\left(-\beta u\right) u^2 \tag{7.8}$$

Let us now set the u parameter as follows: $u = \max_{mn} |v_{mn}|$. Now, instead of satisfying Equation 7.8 we will satisfy the following two stricter constraints. These 2 cases subsume Equation 7.8 since in each case we are subtracting a positive quantity from the left hand side and still trying to maintain an upper bound. These cases are as follows:

- Case 1: When β is *positive*, we know that: $\exp(\beta u) \geq \exp(\beta v_{mn})$. This is because we already defined $u = \max_{mn} |v_{mn}|$ and β is positive. Thus we impose the following stricter guarantee and expand it:

$$\begin{aligned} \gamma \exp(\beta u) u^2 &\geq \sum_{mn} h_{mn} v_{mn}^2 \exp(\beta v_{mn}) \\ \gamma \exp(\beta u) u^2 &\geq \sum_{mn} h_{mn} v_{mn}^2 \exp(\beta u) \\ \gamma &\geq \frac{\sum_{mn} h_{mn} v_{mn}^2}{u^2} \end{aligned}$$

- Case 2: When β is *negative*, we know that $\exp(-\beta u) \geq \exp(\beta v_{mn})$. Therefore, we impose the stricter guarantee below and expand it:

$$\begin{aligned} \gamma \exp(-\beta u) u^2 &\geq \sum_{mn} h_{mn} v_{mn}^2 \exp(\beta v_{mn}) \\ \gamma \exp(-\beta u) u^2 &\geq \sum_{mn} h_{mn} v_{mn}^2 \exp(-\beta u) \\ \gamma &\geq \frac{\sum_{mn} h_{mn} v_{mn}^2}{u^2} \end{aligned}$$

We can thus satisfy Inequality 7.7 by setting the following definition for the parameters γ and u and guarantee that the intermediate bound is valid. Thus, we have an intermediate upper bound on the log Gibbs partition function which simplifies its possibly large number of terms (in the summation $\sum_{mn}$) to just two parameters. Figure 7.9 depicts the bound for various random cases.

$$u := \max_{mn} |v_{mn}| \qquad \gamma := \frac{\sum_{mn} h_{mn} v_{mn}^2}{u^2}$$

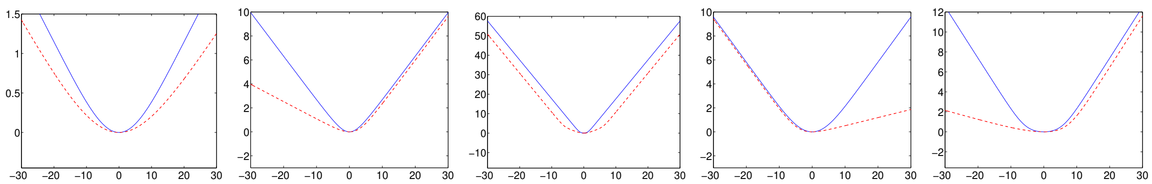


Figure 7.9: Intermediate Bound on Log Gibbs Partition Functions. The solid line is the simple intermediate upper bound which is always guaranteed greater than Gibbs partition function in the dashed red line.

Thus, instead of finding a $q_{\hat{\Psi}}$ that satisfies the original uni-dimensional bound in Equation 7.6 we will merely find $q_{\hat{\Psi}}$ that satisfies the stricter intermediate bounds we have just computed:

$$\frac{1}{2} q_{\hat{\Psi}} \beta^2 \geq \log(\gamma \exp(\beta u) + \gamma \exp(-\beta u) + 1 - 2\gamma) \tag{7.9}$$

We next invoke a simple change of variables to get rid of the u parameter for the following analysis. Namely, define $\omega = \beta u$ and solve for the desired $q_{\hat{\Psi}}$ as follows:

$$\frac{q_{\hat{\Psi}}}{2u^2} \geq \max_{\omega} f(\gamma, \omega)$$

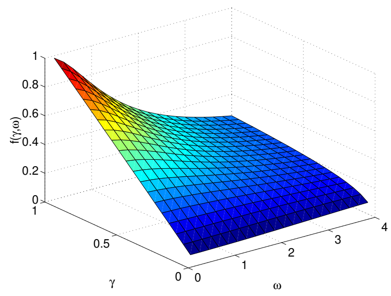


Figure 7.10: $f(\gamma, \omega)$.

Where above we have replaced the expression on the right for brevity with a two-dimensional function $f(\gamma, \omega)$. The 2d function is plotted in Figure 7.10 and is given by the following:

$$f(\gamma, \omega) = \frac{\log\left(\gamma \exp(\omega) + \gamma \exp(-\omega) + 1 - 2\gamma\right)}{\omega^2}$$

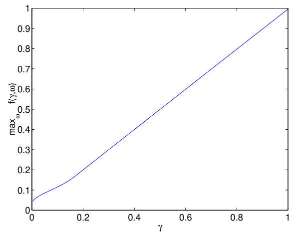


Figure 7.11: $\max_{\omega} f(\gamma, \omega)$.

Unfortunately, maximizing over ω cannot be done analytically. Instead, one possibility is to obtain the function $\max_{\omega} f(\gamma, \omega)$ numerically as show in Figure 7.11. Instead of dealing with $\max_{\omega} f(\gamma, \omega)$ directly (which may be awkward due to its non-analytic nature), we can work with analytic linear upper bounds upon it. For example, we can have various linear upper bounds of the form:

$$a\gamma + b \geq \max_{\omega} f(\gamma, \omega)$$

Figure 7.12 depicts the linear bounds on the $\max_{\omega} f(\gamma, \omega)$ function. Some values of a and b which provide numerically guaranteed upper bounds on the right hand side are listed in the Appendix in Table 10.3. Matlab code to derive the function relating a and b is also given in the Appendix in Section 10.3. We merely have to select one of those many valid settings for a and b to obtain an upper bound on $\max_{\omega} f(\gamma, \omega)$. Therefore, inserting this intermediate bound allows us to obtain the desired $q_{\hat{\Psi}}$ in terms of the (a, b) linear bound:

$$q_{\hat{\Psi}} \geq 2u^2 \max_{\omega} f(\gamma, \omega)$$

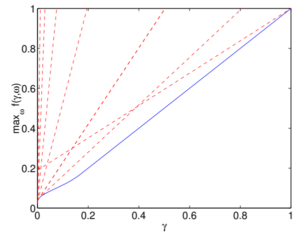


Figure 7.12: Linear Upper Bounds on the numerical $\max_\omega f(\gamma,\omega)$. Here, we show various settings of the upper bounds $a\gamma + b$ as we vary the (a, b) settings.

$$
\begin{aligned}
q_{\hat{\Psi}} &\geq 2u^2(a\gamma + b) \\
q_{\hat{\Psi}} &\geq 2\max_{mn} v_{mn}^2 \left(a\frac{\sum_{mn} h_{mn} v_{mn}^2}{\max_{mn} v_{mn}^2} + b \right) \\
q_{\hat{\Psi}} &\geq 2a\sum_{mn} h_{mn} v_{mn}^2 \quad + \quad 2b\max_{mn} v_{mn}^2
\end{aligned}
$$

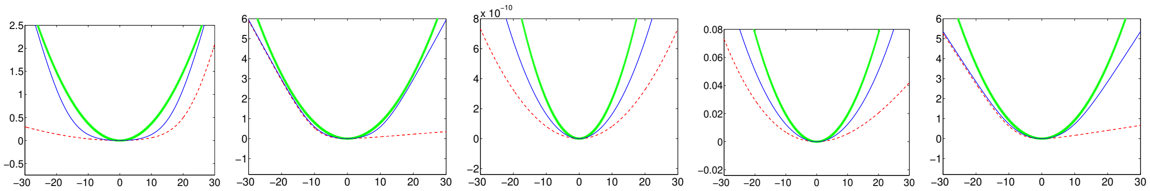


Figure 7.13: Quadratic Bound on Log Gibbs Partition Functions. The quadratic bound ('x' green line) is above the the simple intermediate upper bound (the solid blue line) which is above the Gibbs partition function (dashed red line).

In the above, we have expanded our parameters u and γ in terms of their definitions. Figure 7.13 depicts the desired resulting quadratic bounds from the formula for $q_{\hat{\Psi}}$ on the intermediate bounds as in Equation 7.9. We optimize over the possible choices (a, b) such that the $q_{\hat{\Psi}}$ value is minimized to generate the tightest bound. We have thus found a quadratic bound for the one dimensional case. The next step is to fold this bound into the original multidimensional formulation and guarantee it in the general full-dimensional bounding problem.

7.8.2 Scaling Up to the Multidimensional Formulation

We now have a quadratic bound that is guaranteed along a particular slice of the $L(\Psi)$ function when $\Psi = \beta\hat{\Psi}$. Let us rewrite this unidimensional bound in terms of our whitened data vectors $\mathcal{Z}_{mn}$. Furthermore, for computational efficiency we will also make sure that the computation of $q_{\hat{\Psi}}$ will only require simple inner products of these vectors. First recall the following definition:

$$
v_{mn} = \hat{\Psi}^T \left(\mathbf{Z}_{mn} - \sum_{ij} h_{ij} \mathbf{Z}_{ij} \right)
$$

Replacing that into the current bound on $q_{\hat{\Psi}}$ we get the following:

$$
\begin{aligned}
q_{\hat{\Psi}} &\geq 2a \sum_{mn} h_{mn} v_{mn}^2 \quad + \quad 2b \max_{mn} v_{mn}^2 \\
q_{\hat{\Psi}} &\geq 2a \sum_{mn} h_{mn} \left(\hat{\Psi}^T \left(\mathbf{Z}_{mn} - \sum_{ij} h_{ij} \mathbf{Z}_{ij} \right) \right)^2 \quad + \quad 2b \max_{mn} \left(\hat{\Psi}^T \left(\mathbf{Z}_{mn} - \sum_{ij} h_{ij} \mathbf{Z}_{ij} \right) \right)^2 \\
q_{\hat{\Psi}} &\geq 2a \sum_{mn} h_{mn} \left(\hat{\Psi}^T \mathbf{Z}_{mn} \right)^2 - 2a \left(\sum_{ij} h_{ij} \hat{\Psi}^T \mathbf{Z}_{ij} \right)^2 \quad + \quad 2b \max_{mn} \left(\hat{\Psi}^T \mathbf{Z}_{mn} - \hat{\Psi}^T \sum_{ij} h_{ij} \mathbf{Z}_{ij} \right)^2
\end{aligned}
$$

For the terms that depend on the b scaling, we shall merely invoke Jensen's inequality in a straightforward way on the quadratic expression. Again, this makes us get an intermediate upper bound on $q_{\hat{\Psi}}$ which satisfies our requirements even more strictly. More specifically, we have:

$$
\begin{aligned}
\left(\hat{\Psi}^T \mathbf{Z}_{mn} - \hat{\Psi}^T \sum_{ij} h_{ij} \mathbf{Z}_{ij} \right)^2 &= 4 \times \left(\frac{1}{2} \hat{\Psi}^T \mathbf{Z}_{mn} + \frac{1}{2} \left(-\hat{\Psi}^T \sum_{ij} h_{ij} \mathbf{Z}_{ij} \right) \right)^2 \\
&\leq 4 \times \frac{1}{2} \left(\hat{\Psi}^T \mathbf{Z}_{mn} \right)^2 \quad + \quad 4 \times \frac{1}{2} \left(\hat{\Psi}^T \sum_{ij} h_{ij} \mathbf{Z}_{ij} \right)^2
\end{aligned}
$$

Reinserting this bound into our current $q_{\hat{\Psi}}$ expressions, we obtain the following stricter condition on $q_{\hat{\Psi}}$:

$$
\begin{aligned}
q_{\hat{\Psi}} &\geq 2a \sum_{mn} h_{mn} \left(\hat{\Psi}^T \mathbf{Z}_{mn} \right)^2 - 2a \left(\sum_{ij} h_{ij} \hat{\Psi}^T \mathbf{Z}_{ij} \right)^2 \quad + \quad 2b \max_{mn} \left(2 \left(\hat{\Psi}^T \mathbf{Z}_{mn} \right)^2 \quad + \quad 2 \left(\hat{\Psi}^T \sum_{ij} h_{ij} \mathbf{Z}_{ij} \right)^2 \right) \\
q_{\hat{\Psi}} &\geq 2a \sum_{mn} h_{mn} \left(\hat{\Psi}^T \mathbf{Z}_{mn} \right)^2 - 2(a - 2b) \left(\sum_{ij} h_{ij} \hat{\Psi}^T \mathbf{Z}_{ij} \right)^2 \quad + \quad 4b \max_{mn} \left(\hat{\Psi}^T \mathbf{Z}_{mn} \right)^2
\end{aligned}
$$

We can easily guarantee that $a - 2b$ is always positive (which is evident from the allowable values in the tables), and thus we see that the middle term in the above expression is negative. We can therefore delete it and obtain an even stricter bound on $q_{\hat{\Psi}}$:

$$
q_{\hat{\Psi}} \quad \geq \quad 2a \sum_{mn} h_{mn} \left(\hat{\Psi}^T \mathbf{Z}_{mn} \right)^2 \quad + \quad 4b \max_{mn} \left(\hat{\Psi}^T \mathbf{Z}_{mn} \right)^2
$$

Now recall the following property from the definition of $\mathbf{Z}_{mn}$ in Equation 7.5. If we are dealing with a normalized unit-norm version of the Ψ vector (i.e. $\hat{\Psi}$), and dot product it with $\mathbf{Z}_{mn}$ which is padded with zeroes, this is equivalent to dot-producting the truncated version of $\mathcal{Z}_{mn}$ with a truncated version of of $\hat{\Psi}$, namely $\hat{\Psi}_m$:

$$
\hat{\Psi}^T \mathbf{Z}_{mn} \quad = \quad \hat{\Psi}_m^T \mathcal{Z}_{mn}
$$

In the above we have the $\hat{\Psi}_m$ vector simply be a vector of less than unit norm corresponding to the unit norm vector $\hat{\Psi}$ with all entries that are not in the m'th index position zeroed out. This allows us to rewrite the above as:

$$q_{\hat{\Psi}} \quad \geq \quad 2a \sum_{mn} h_{mn} \left(\hat{\Psi}_m^T \mathcal{Z}_{mn}\right)^2 \quad + \quad 4b \max_{mn} \left(\hat{\Psi}_m^T \mathcal{Z}_{mn}\right)^2$$

Any inner product can be written as the magnitudes of the vectors times the cosine of the angle between them, i.e. $\hat{\Psi}_m^T \mathcal{Z}_{mn} = \|\hat{\Psi}_m\| \, \|\mathcal{Z}_{mn}\| \cos(\theta)$. Therefore, we can find a stricter upper bound once again by noting the following:

$$\left(\hat{\Psi}_m^T \mathcal{Z}_{mn}\right)^2 \leq \hat{\Psi}_m^T \hat{\Psi}_m \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}$$

Therefore we can finally write the following condition on $q_{\hat{\Psi}}$ which will guarantee an upper bound:

$$q_{\hat{\Psi}} \quad := \quad 2a \sum_{mn} h_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad + \quad 4b \max_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}$$

This definition of $q_{\hat{\Psi}}$ will give us a guaranteed quadratic bound on our original log-partition function for *arbitrary* choice of direction $\hat{\Psi}$ that:

$$\frac{1}{2} q_{\hat{\Psi}} \beta^2 \quad \geq \quad \log \sum_m \sum_n h_{mn} \exp\left(\beta \hat{\Psi}^T \mathbf{U}_{mn}\right) \qquad \forall\, \beta$$

We can now plug in the definition of $q_{\hat{\Psi}}$ in that quadratic expression and guarantee that it holds for any β and any $\hat{\Psi}$:

$$\frac{1}{2}\left(2a \sum_{mn} h_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad + \quad 4b \max_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}\right) \beta^2 \quad \geq \quad \log \sum_m \sum_n h_{mn} \exp\left(\beta \hat{\Psi}^T \mathbf{U}_{mn}\right) \qquad \forall\, \beta$$

We should point out here, that we can vary the chosen a and b values for each $\hat{\Psi}$ direction, they need not be fixed throughout to guarantee the bounds. In other words, they can be made functions of the direction $\hat{\Psi}$ without violating the bound:

$$\left(a_{\hat{\Psi}} \sum_{mn} h_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad + \quad 2b_{\hat{\Psi}} \max_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}\right) \beta^2 \quad \geq \quad \log \sum_m \sum_n h_{mn} \exp\left(\beta \hat{\Psi}^T \mathbf{U}_{mn}\right) \qquad \forall\, \beta$$

We now return to the original problem, which was finding a quadratic upper bound on a multi-dimensional log Gibbs partition function, i.e. $H(\Psi) \geq L(\Psi)$ by computing valid r_m scalars that satisfy:

$$\frac{1}{2} \sum_m r_m \Psi_m^T \Psi_m \quad \geq \quad \log \sum_m \sum_n h_{mn} \exp\left(\Psi^T \mathbf{U}_{mn}\right) \qquad \forall\, \Psi$$

The above can be written in terms of $\beta\hat{\Psi}$ and then we can directly plug in the unidimensional bound we have derived:

$$\frac{1}{2} \sum_m r_m \beta \hat{\Psi}_m^T \beta \hat{\Psi}_m \quad \geq \quad \log \sum_m \sum_n h_{mn} \exp\left(\beta \hat{\Psi}^T \mathbf{U}_{mn}\right) \qquad \forall\, \hat{\Psi}, \beta$$

$$\frac{1}{2} \sum_m r_m \beta \hat{\Psi}_m^T \beta \hat{\Psi}_m \quad \geq \quad \left(a_{\hat{\Psi}} \sum_{mn} h_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad + \quad 2b_{\hat{\Psi}} \max_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}\right) \beta^2 \qquad \forall\, \hat{\Psi}, \beta$$

We can now divide out the β^2 (the $\beta = 0$ case is not relevant since we have tangential contact at $\beta = 0$) from both sides:

$$\frac{1}{2}\sum_m r_m \hat{\Psi}_m^T \hat{\Psi}_m \quad \geq \quad \left(a_{\hat{\Psi}} \sum_{mn} h_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad + \quad 2b_{\hat{\Psi}} \max_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \right) \qquad \forall \, \hat{\Psi}$$

By inspection, we can turn the above aggregate bound on the r_m into M individual bounds on isolated r_m. Satisfying these bounds on an individual basis will satisfy the aggregate bound:

$$\frac{1}{2} r_m \hat{\Psi}_m^T \hat{\Psi}_m \quad \geq \quad a_{\hat{\Psi}} \sum_n h_{mn} \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad + \quad 2b_{\hat{\Psi}} \max_n \hat{\Psi}_m^T \hat{\Psi}_m \; \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \qquad \forall \; m$$

Simplifying by dividing out the $\hat{\Psi}_m^T \hat{\Psi}_m$ yields:

$$r_m \quad \geq \quad 2a_{\hat{\Psi}} \sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad + \quad 4b_{\hat{\Psi}} \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \qquad \forall \; m$$

Where we are still free to pick an arbitrary $a_{\hat{\Psi}}, b_{\hat{\Psi}}$ pair for each m value. Thus, we should really index those as:

$$r_m \quad \geq \quad 2a_m \sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad + \quad 4b_m \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \qquad \forall \; m$$

The smallest value of r_m which still satisfies the above is the tightest bound. Therefore, we set r_m to the above to obtain, finally, a succinct formula for it:

$$r_m \quad = \quad 2a_m \sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad + \quad 4b_m \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \qquad \forall \; m$$

Expanding out the definition $w_m - w'_m = r_m$, we ultimately obtain:

$$w_m \quad = \quad 2a_m \sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \quad + \quad 4b_m \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} + w'_m$$

Where we have (from previous sections) the following definition [2]:

$$xsw'_m \quad = \quad \min w'_m \text{ such that } \frac{h_m}{w'_m}\left(\left.\frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} - X_m \right) + \left.\frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}\right|_{\tilde{\Theta}_m} \in \frac{\partial \mathcal{K}(\Theta_m)}{\partial \Theta_m}$$

And the $\mathcal{Z}_{mn}$ are defined as follows (as shown earlier):

$$\mathcal{Z}_{mn} \quad = \quad \mathcal{K}''(\tilde{\Theta}_m)^{-1/2}(X_{mn} - \mathcal{K}'(\tilde{\Theta}_m))$$

It should be noted that we can always increase the w_m terms and guarantee a bound because these provide a stricter bounding. Consequently, any of the intermediate terms in the computation of w_m which are too difficult to compute can be upper bounded if they result in an increase in the w_m and this will still guarantee a strict global bound.

[2] The value of w'_m depends on the nature of the constraints on the $\mathcal{K}_m$ function. In the Gaussian case, Θ is an arbitrary Euclidean vector and therefore w'_m is always zero. In the multinomial and other cases, it is trivial to compute w'_m with some simple algebra.

7.9 Analytically Avoiding Lookup Tables

The use of lookup tables is mathematically unappealing so we instead now provide a fully analytic treatment of the a_m and b_m tradeoff. Recall that we had achieved a fully analytic bounding process up until the point where we were trying to solve for $q_{\hat{\Psi}}$:

$$\frac{q_{\hat{\Psi}}}{2u^2} \geq \max_{\omega} \frac{\log\left(\gamma \exp(\omega) + \gamma \exp(-\omega) + 1 - 2\gamma\right)}{\omega^2}$$

This bounding thus required the maximization over ω of the following function:

$$f(\gamma, \omega) = \frac{\log\left(\gamma \exp(\omega) + \gamma \exp(-\omega) + 1 - 2\gamma\right)}{\omega^2}$$

It was deemed too difficult to exactly solve the max of $f(\gamma, \omega)$ and instead it was handled numerically. However, what we can do is find an upper bound on $\max_\omega f(\gamma, \omega)$ and thus provide an even stricter condition on $q_{\hat{\Psi}}$. Naturally, this bound will not be as tight as the one resulting from the a, b numerical linear bounds yet is fully analytic. First let us define the function $g(\gamma) := \max_\omega f(\gamma, \omega)$ which we have found too hard to compute analytically. We thus have:

$$q_{\hat{\Psi}} = 2u^2 g(\gamma)$$

An analytic expression for $g(\gamma)$ is too difficult, however note the following interesting property: [3]:

$$g(\gamma) = \gamma \qquad \forall \ \gamma \geq \frac{1}{6}$$

This is true, since the limit of $f(\gamma, \omega)$ as ω goes to zero (via l'Hopital's rule) is:

$$\lim_{\omega \to 0} \frac{\log(\gamma \exp(\omega) + \gamma \exp(-\omega) + 1 - 2\gamma)}{\omega^2} = \gamma$$

The maximum of $f(\gamma, \omega)$ occurs at $\omega \to 0$ whenever $\gamma \geq 1/6$. This is done by proving that:

$$\frac{\log(\gamma \exp(\omega) + \gamma \exp(-\omega) + 1 - 2\gamma)}{\omega^2} \leq \gamma$$

The proof of the last inequality is equivalent to proving:

$$\exp(\omega) + \exp(-\omega) - 2 \leq \frac{\exp(\gamma\omega^2) - 1}{\gamma}$$

Expanding all exp functions into power series in ω and comparing terms yields the result. This last inequality can also be used to show that if $0 < \gamma < 1/6$ then $g(\gamma) > \gamma$. Thus, we know that $g(\gamma) = \gamma$ for $\gamma \geq 1/6$. Recall that $g(\gamma)$ arose when we were trying to find a quadratic bound on the $\log(\gamma \exp(\omega) + \gamma \exp(-\omega) + 1 - 2\gamma)$. Thus, we had:

$$\frac{q_{\hat{\Psi}}}{2u^2}\omega^2 \geq \log(\gamma \exp(\omega) + \gamma \exp(-\omega) + 1 - 2\gamma)$$

[3] This property was noted and proved with the help of Dr. Michael Ulm, FB Mathematik, Universitaet Rostock.

Evidently, then, if $\gamma \geq 1/6$, we have:

$$q_{\hat{\Psi}} \geq 2u^2\gamma \qquad \forall \;\; \gamma \geq \frac{1}{6}$$

Now, let us look more closely at the interval when $\gamma \leq 1/6$. For simplicity We define $h(\gamma, \omega)$ as follows:

$$h(\gamma,\omega) \;\; = \;\; \log(\gamma \exp(\omega) + \gamma \exp(-\omega) + 1 - 2\gamma)$$

Replacing, we can rewrite our condition as follows:

$$\frac{q_{\hat{\Psi}}}{2u^2}\omega^2 \geq h(\gamma,\omega)$$

Let us now split the left hand side and the right hand side into two components:

$$\frac{q_1}{2u^2}\omega^2 + \frac{q_2}{2u^2}\omega^2 \geq h_1(\gamma,\omega) + h_2(\gamma,\omega) \tag{7.10}$$

Naturally, we have:

$$\begin{aligned} q_{\hat{\Psi}} &= q_1 + q_2 \\ h(\gamma,\omega) &= h_1(\gamma,\omega) + h_2(\gamma,\omega) \end{aligned}$$

Now, if we can guarantee the following individual inequalities we can guarantee a stricter version of the overall inequality in Equation 7.10.

$$\begin{aligned} \frac{q_1}{2u^2}\;\omega^2 &\geq h_1(\gamma,\omega) \\ \frac{q_2}{2u^2}\;\omega^2 &\geq h_2(\gamma,\omega) \end{aligned}$$

So far, he solution has not really developed any further through this breakdown. However, we will next specify the h_1 and h_2, and we shall be able to find a simple analytic solution for q_1 and q_2.

For $h_1(\gamma,\omega)$, we will use the following function:

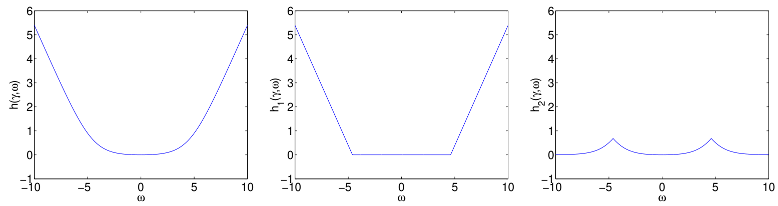
$$h_1(\gamma,\omega) \;\; = \;\; \frac{|\omega - \log(\gamma)| - (\omega - \log(\gamma))}{2} + \frac{|-\omega - \log(\gamma)| - (-\omega - \log(\gamma))}{2}$$

The $h_2(\gamma,\omega)$ is then naturally constrained to be:

$$h_2(\gamma,\omega) \;\; = \;\; h(\gamma,\omega) - h_1(\gamma,\omega)$$

Clearly $h_1(\gamma,\omega)$ and $h_2(\gamma,\omega)$ add to form $h(\gamma,\omega)$ so this breakdown is legitimate. Figure 7.14 depicts the different functions, $h(\gamma,\omega)$, $h_1(\gamma,\omega)$ and $h_2(\gamma,\omega)$. Now, it is relatively easy to upper bound the above $h_1(\gamma,\omega)$ and $h_2(\gamma,\omega)$ functions individually. Basically, $h_1(\gamma,\omega)$ is a symmetric function and we can deal with only one side at a time. When $\omega \geq 0$, we need only guarantee that the quadratic bound is greater than a line with slope=1 and an ω-intercept of $-\log(\gamma)$. Therefore, we have:

$$\frac{q_1}{2u^2}\;\omega^2 \;\; \geq \;\; \omega + \log(\gamma)$$



(a) $h(\gamma,\omega)$ (b) $h_1(\gamma,\omega)$ (c) $h_2(\gamma,\omega)$

Figure 7.14: Bounding Separate Components.

It is straightforward to show that:

$$\frac{1}{4\log(1/\gamma)}\ \omega^2 \ \ \geq \ \ \omega + \log(\gamma)$$

This also holds for the $\omega \leq 0$ case by symmetry. Therefore, we obtain q_1 as follows:

$$q_1 \ \ = \ \ \frac{2u^2}{4\log(1/\gamma)}$$

Now, we need to solve for q_2 to upper bound $h_2(\gamma,\omega)$. Again, by symmetry of $h(\gamma,\omega)$ and the quadratic upper bound, we need only consider the case when $\omega \geq 0$. We shall split the interval where we will verify this bound into two separate components: $0 \leq \omega \leq \log(1/\gamma)$ and $\log(1/\gamma) \leq \omega < \infty$.

For the first interval, $\log(1/\gamma) \leq \omega < \infty$, we see that $h_2(\gamma,\omega)$ simplifies as follows:

$$\begin{aligned}
\frac{q_2}{2u^2}\ \omega^2 \ &\geq \ h_2(\gamma,\omega) \qquad \log(1/\gamma) \leq \omega < \infty \\
\frac{q_2}{2u^2}\ \omega^2 \ &\geq \ h(\gamma,\omega) - (\omega - log(1/\gamma)) \qquad \log(1/\gamma) \leq \omega < \infty
\end{aligned}$$

Thus, we can find q_2 by maximizing as follows:

$$\begin{aligned}
\frac{q_2}{2u^2} \ &= \ \max_{\log(1/\gamma)\leq\omega<\infty} \frac{h(\gamma,\omega) - \omega - \log(\gamma)}{\omega^2} \\
\frac{q_2}{2u^2} \ &= \ \max_{\log(1/\gamma)\leq\omega<\infty} \frac{\log(\gamma\exp(\omega) + \gamma\exp(-\omega) + 1 - 2\gamma) - \omega - \log(\gamma)}{\omega^2}
\end{aligned}$$

It is clear in the above that the numerator $h(\gamma,\omega) - \omega - \log(\gamma)$ is monotonically decreasing (its gradient is always negative). Meanwhile, the denominator ω^2 is monotonically increasing. Therefore, to maximize the fraction, we want to keep ω as small as possible, thus we set it to $\omega = \log(1/\gamma)$. This gives us the following condition for q_2:

$$\frac{q_2}{2u^2} \ = \ \frac{\log(\gamma\exp(-\log(\gamma)) + \gamma\exp(\log(\gamma)) + 1 - 2\gamma) + \log(\gamma) - \log(\gamma)}{\log(\gamma)^2} \tag{7.11}$$

$$\frac{q_2}{2u^2} \ = \ \frac{\log(\gamma^2 - 2\gamma + 2)}{\log(\gamma)^2} \tag{7.12}$$

For the second interval, $0 \leq \omega \leq \log(1/\gamma)$, and there $h_1(\gamma,\omega)$ is simply zero. This permits $h_2(\gamma,\omega)$ to simplify to: $h_2(\gamma,\omega) = h(\gamma,\omega)$. Therefore, we need to upper bound as follows:

$$\frac{q_2}{2u^2}\ \omega^2 \ \ \geq \ \ h_2(\gamma,\omega) \qquad 0 \leq \omega \leq \log(1/\gamma)$$

$$
\begin{aligned}
\frac{q_2}{2u^2}\,\omega^2 &\geq h(\gamma,\omega) \qquad 0 \leq \omega \leq \log(1/\gamma) \\
\frac{q_2}{2u^2}\,\omega^2 &\geq \log(\gamma\exp(\omega)+\gamma\exp(-\omega)+1-2\gamma) \qquad 0 \leq \omega \leq \log(1/\gamma)
\end{aligned}
$$

By concavity of the logarithm function, we can upper bound $\log(z)$ by $z-1$. If we invoke this upper bound on the right hand side, we obtain an even stricter condition for q_2.

$$
\begin{aligned}
\frac{q_2}{2u^2}\,\omega^2 &\geq (\gamma\exp(\omega)+\gamma\exp(-\omega)+1-2\gamma)-1 \qquad 0 \leq \omega \leq \log(1/\gamma) \\
\frac{q_2}{2u^2} &= \gamma \max_{0\leq\omega\leq\log(1/\gamma)} \frac{\exp(\omega)+\exp(-\omega)-2}{\omega^2}
\end{aligned}
$$

The right hand side function being maximized over ω is monotonically increasing (its gradient is always positive). Therefore, to maximize it, we set ω as large as possible within the interval being considered (i.e. $\omega = \log(1/\gamma)$). Thus, to satisfy the bounding for this interval, we obtain the following condition on q_2:

$$
\frac{q_2}{2u^2} = \gamma\,\frac{\exp(-\log(\gamma))+\exp(\log(\gamma))-2}{\log(\gamma)^2} \tag{7.13}
$$

$$
\frac{q_2}{2u^2} = \frac{(\gamma-1)^2}{\log(\gamma)^2} \tag{7.14}
$$

$$
q_2 = 2u^2\,\frac{(\gamma-1)^2}{\log(\gamma)^2} \tag{7.15}
$$

Now we must aggregate both Equation 7.12 with Equation 7.15 to obtain an overall bound on q_2 for both intervals. It is clear that Equation 7.15 is always stricter and subsumes Equation 7.12 since the following is provably true (by simply invoking the $\log(z) \geq z-1$ rule):

$$
(\gamma-1)^2 \geq \log(\gamma^2-2\gamma+2)
$$

Therefore, we have the following values for q_1 and q_2 (when $\gamma \leq 1/6$):

$$
\begin{aligned}
q_1 &= \frac{2u^2}{4\log(1/\gamma)} \\
q_2 &= 2u^2\,\frac{(\gamma-1)^2}{\log(\gamma)^2}
\end{aligned}
$$

Therefore, we can now compute the $q_{\hat{\Psi}}$ value as their sum:

$$
\begin{aligned}
q_{\hat{\Psi}} &= q_1+q_2 \\
q_{\hat{\Psi}} &= 2u^2\left(\frac{1}{4\log(1/\gamma)}+\frac{(\gamma-1)^2}{\log(\gamma)^2}\right)
\end{aligned}
$$

This is only for the $\gamma \leq 1/6$ region. We can combine it with our value for the $\gamma \geq 1/6$ as follows:

$$
q_{\hat{\Psi}} = 2u^2 \begin{cases} \gamma & \gamma \geq 1/6 \\ \frac{1}{4\log(1/\gamma)}+\frac{(\gamma-1)^2}{\log(\gamma)^2} & \gamma \leq 1/6 \end{cases}
$$

For reasons that will become evident soon, the $l0$ discontinuity at $\gamma = 1/6$ is not desirable and prevents the overall concavity (with respect to the γ variable) of the right hand side. By checking the curvature of $\frac{1}{4\log(1/\gamma)}$ and $\frac{(\gamma-1)^2}{\log(\gamma)^2}$, we can see that these two functions are concave for $\gamma \leq 1/6$. However, the sudden jump to the linear function when $\gamma = 1/6$ prevents concavity. To maintain concavity, we use a linear upper bound for $\gamma \geq 1/6$ which keeps the $l0$ and $l1$ continuity. Thus, we will replace $q_{\hat{\Psi}}$ with a concavified version as follows:

$$q_{\hat{\Psi}} \quad = \quad 2u^2 \begin{cases} \tilde{m}\gamma + \tilde{b} & \gamma \geq 1/6 \\ \frac{1}{4\log(1/\gamma)} + \frac{(\gamma-1)^2}{\log(\gamma)^2} & \gamma \leq 1/6 \end{cases}$$

The gradient and the value at $\gamma < 1/6$ are:

$$\begin{aligned} \tilde{m} &= \left. \frac{1}{4\gamma\log(\gamma)^2} + \frac{2(\gamma-1)}{\log(\gamma)^2} - \frac{2(\gamma-1)^2}{\gamma\log(\gamma)^3} \right|_{\gamma=1/6} \\ \tilde{m} &= \frac{1}{4/6\log(6)^2} + \frac{-5/3}{\log(6)^2} + \frac{3(5/3)^2}{\log(6)^3} \\ \tilde{m} &\approx 1.397 \\ \tilde{b} &= \left(\frac{1}{4\log(6)} + \frac{25/36}{\log(6)^2}\right) - \frac{\tilde{m}}{6} \end{aligned}$$

In fact, to maintain concavity the value of $\tilde{m}$ can be any value less than the slope computed above. However, we also must make it upper bound the original value, i.e. $\tilde{m}\gamma + \tilde{b} \leq \gamma$ for $1/6 \leq \gamma \leq 1$. Therefore, we can merely use the value $\tilde{m} = 1$ (which would result in a $\tilde{b} > 1/6$ and thus an upper bound on the original linear γ function). Therefore, we obtain:

$$q_{\hat{\Psi}} \quad = \quad 2u^2 \begin{cases} \gamma + \frac{1}{4\log(6)} + \frac{25/36}{\log(6)^2} - 1/6 & \gamma \geq 1/6 \\ \frac{1}{4\log(1/\gamma)} + \frac{(\gamma-1)^2}{\log(\gamma)^2} & \gamma \leq 1/6 \end{cases}$$

In the next section we will justify why we went through the trouble to concavify the above function over γ. Returning to our original solution for $q_{\hat{\Psi}}$ we had:

$$q_{\hat{\Psi}} \quad \geq \quad 2u^2 g(\gamma)$$

Therefore, we effectively have derived a guaranteed concave upper bound on $g(\gamma)$. This analytic upper bound on $g(\gamma)$ is depicted in Figure 7.15 and is given by the following equation.

$$g(\gamma) \quad \leq \quad \begin{cases} \gamma + \frac{1}{4\log(6)} + \frac{25/36}{\log(6)^2} - 1/6 & \gamma \geq 1/6 \\ \frac{1}{4\log(1/\gamma)} + \frac{(\gamma-1)^2}{\log(\gamma)^2} & \gamma \leq 1/6 \end{cases}$$

7.10 A Final Appeal to Convex Duality

We shall now explain why we have gone through this procedure to obtain an analytic *concave* upper bound on the $g(\gamma)$ function. Let us define our new upper bound on the right hand side of the above expression as $G(\gamma)$, i.e. $g(\gamma) \leq G(\gamma)$. Figure 7.15 depicts the two functions. Also, we have already shown that $G(\gamma)$ is concave.

$$G(\gamma) \quad = \quad \begin{cases} \gamma + \frac{1}{4\log(6)} + \frac{25/36}{\log(6)^2} - 1/6 & \gamma \geq 1/6 \\ \frac{1}{4\log(1/\gamma)} + \frac{(\gamma-1)^2}{\log(\gamma)^2} & \gamma \leq 1/6 \end{cases}$$

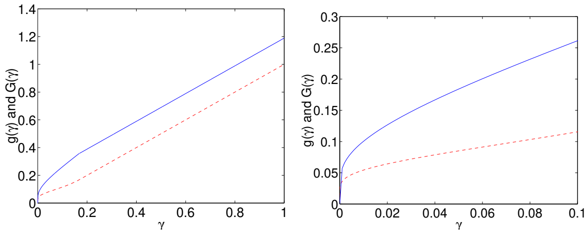


Figure 7.15: Bounding the Numerical $g(\gamma)$ (dashed red line) with Analytic $G(\gamma)$ (solid blue line).

Recall that we avoided working with $g(\gamma)$ directly by only considering upper bounds on it of the form $a\gamma + b \geq g(\gamma)$. Ultimately, our final solution to the derived bounds had the form below which involved these varying (a, b) terms:

$$w_m = 2a_m \sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} + 4b_m \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} + w'_m$$

The idea here is to now minimize the w_m quantities above with respect to all possible settings of the (a, b) term pairs to obtain as tight a bound as possible. By varying the tangential contact point γ of the linear (a, b) bounds we obtain a continuum of variational bound (a, b) pairs parameterized by a variable, say t, i.e. $(a(t), b(t))$. Alternatively, we may merely parameterize b as a function of a to avoid the additional dummy variable, therefore, we have a continuum of pairs $(a, b(a))$. We can now write the above more succinctly as:

$$w_m = \min_a 2a \sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} + 4b(a) \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} + w'_m$$

Manipulating further, we obtain:

$$w_m = 4 \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \left(\min_a \; a \frac{\sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}}{2 \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}} + b(a) \right) + w'_m$$

Now note that we had defined the a and b pairs to upper bound as follows:

$$a\gamma + b(a) \geq g(\gamma)$$

If instead, we satisfy the following stricter condition (i.e. since $G(\gamma) \geq g(\gamma)$), we have a valid alternative choice of for (a, b) pairs:

$$a\gamma + b(a) \geq G(\gamma)$$

Manipulating further, we note that following convex-duality form:

$$\begin{aligned} b(a) &\geq G(\gamma) - a\gamma \\ b(a) &= \max_\gamma G(\gamma) - a\gamma \end{aligned}$$

It is well known that for any concave function $G(\gamma)$, $b(a)$ becomes its (negated) convex dual by the above procedure. This is because $b(a)$ is formed by the epi-graph of linear bounds on top of the concave $G(\gamma)$ function. This property allows us to use the following standard convex-duality result (see the Appendix Section 10.2 for further clarification):

$$G(\gamma) \quad = \quad \min_a b(a) + a\gamma \tag{7.16}$$

Using the notation in the Appendix where we show the basic result relating a function f to its dual f^* , we can clearly see that the above holds if we define $f(\gamma) := G(\gamma)$ and $f^*(a) := -b(a)$.

Now, recall that we had the following minimization problem to obtain w_m:

$$w_m \quad = \quad 4 \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \; \left(\min_a \; a \frac{\sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}}{2 \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}} \; + \; b(a) \right) \quad + w'_m$$

In the above, we have seen that the $G()$ function arises from a similar minimization. We obtain exactly the epi-graph definition of $G()$ in Equation 7.16 if we simply define the term multiplying a in the above equation as γ:

$$\gamma \quad = \quad \frac{\sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}}{2 \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}}$$

We clearly see that the minimization over a can therefore be avoided giving us the following analytic result for the reverse-Jensen bound:

$$w_m \quad = \quad 4 \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn} \; G\left(\frac{\sum_n h_{mn} \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}}{2 \max_n \mathcal{Z}_{mn}^T \mathcal{Z}_{mn}} \right) \quad + w'_m$$

Thus, to compute the upper bounds, we need to merely solve for the max and the expected norms of $\mathcal{Z}_{mn}$ and then plug in their ratio into the $G()$ function and so on.

This concludes the derivation.

$\square$

Chapter 8

Imitative Learning

It is by imitation, far more than by precept, that we learn everything;

Edmund Burke, 1729-1797

We have discussed generative and discriminative learning and seen how the two can be theoretically fused into a powerful hybrid. We will now see these ideas come together in an applied sense as a combination of discriminative and generative tools will be used in an imitation system. Our motivation for imitative learning as a useful tool for learning interactive autonomous agents was already outlined in the introduction. This chapter will outline details of our implementation. The imitative architecture we will propose here uses perception to learn passively from a human teacher as he interacts with his environment. It then utilizes its resulting models to synthesize an interactive character that responds appropriately to perceived external stimulus.

To acquire data from the human teacher, we will use a generative perception model that automatically represents the human's visual and acoustic activity as well as the external stimulus presented to him. A generative model is appropriate so that we can provide a priori structure to the complicated perceptual domain as well as be able to sample from this model to create virtual animations of the teacher. This generative representation then feeds training data to a discriminative prediction system that learns to forecast the human's measurements in response to stimuli. A discriminative system here is favored to focus resources on the prediction task. We wish to optimize performance on our ability to specifically predict the output behavior of an agent *given* the stimulus of the external world.

In the next sections we begin by describing the architecture behind such an imitative system and briefly relate it to the Action-Reaction Learning (ARL) platform. Various key issues and limitations are brought up and motivate important generative and discriminative learning tools. Subsequently, we present the data collection method and hardware. A generative model is then described to model the perceptual domain and describes visual data as well as auditory data in the form of coefficient vectors. A discriminative (conditional) hidden Markov model using CEM is then used to learn to forecast a time series of such audio-visual coefficient vectors. Subsequently, we apply these tools to a large dataset of human interactions and generate a program that learns behavior. A qualitative as well as quantitative evaluation of the imitation learning is then given. We complete the chapter with a brief summary and open questions.

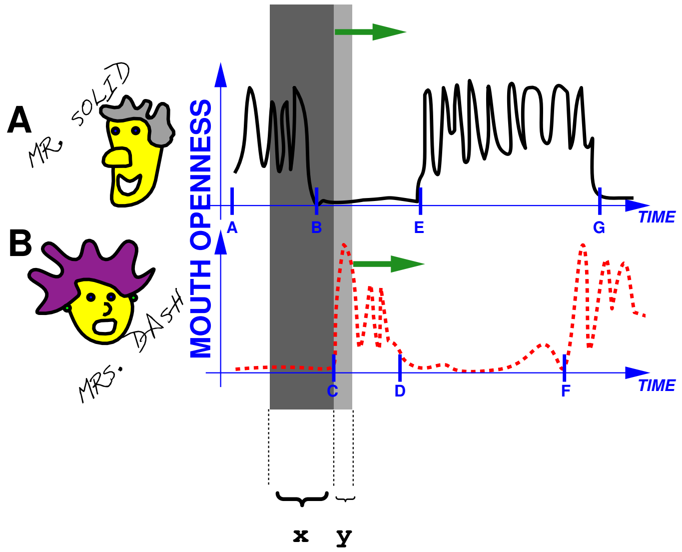


Figure 8.1: Dialog Interaction and Analysis Window

8.1 An Imitation Framework

The imitation framework we will describe learns an autonomous agent that is able to interact and respond appropriately to external stimulus from the world and participants within it. Given the ability to perceive real behavior in humans interacting in the world, we can collect data to learn a predictive model. In earlier work [97] Action-Reaction Learning described a system that learns the behavior of two agents while they are interacting. This can be seen as a more specific instance of imitation where instead of having a teacher interacting with the world, we only consider two teachers interacting with each other. In ARL, one teacher embodies the 'agent' we wish to learn while another teacher embodies the 'world state'. We now review the ARL framework in more detail.

Action-Reaction Learning involves temporal analysis of a multi-dimensional time series that represents the interaction between two agents. Figure 8.1 displays such a stream or series. Let us assume that the stream is being generated by a vision algorithm which measures the openness of the mouth. Two such algorithms are being run simultaneously on two different people. One person generates the dashed line and the other generates the solid line.

Now, imagine that these two individuals are engaged in a conversation. Let us also name them Mr. Solid (the fellow generating the solid line) and Mrs. Dash (the lady generating the dashed line). Initially (interval A-B on the time axis), Mr. Solid is talking while Mrs. Dash remains silent. He has an oscillatory mouth signal while she has a very low value on the openness of the mouth. Then, Mr. Solid says something shocking and pauses (B-C). Mrs. Dash then responds with a discrete 'oh, I see' (C-D). She too then pauses (D-E) and waits to see if Mr. Solid has more to say. He takes the initiative and continues to speak (E). However, Mr. Solid continues talking non-stop for just too long (E-G). So, Mrs. Dash feels the need to interrupt (F) with a counter-argument and simply starts talking. Mr. Solid notes that she has taken the floor and stops to hear her out.

What Action-Reaction Learning seeks to do is discover the coupling between the past interaction and the next immediate reaction of both participants. The system will be used to learn a model of the behavior of Mrs. Dash (and Mr. Solid) so that it can predict and imitate her idiosyncrasies. Thus, we will learn how Mrs. Dash reacts to the current context (i.e. the past few seconds of activity of the users which is akin to a world state). The process begins by sliding a window over the temporal interaction as in Figure 8.1. The window looks at a small piece of the interaction and the immediate reaction of the users. This window over the time series forms the short term or iconic memory of the interaction and it is highlighted with a dark rectangular patch. The consequent reaction of Mrs. Dash and Mr. Solid are highlighted with the lighter and smaller rectangular strip. The first strip will be treated as an input $\mathbf{x}$ and the second strip will be the subsequent future behavioral output they should generate ($\mathbf{y}$). To predict and imitate what either Mrs. Dash or Mr. Solid will do next, a system system must estimate the future mouth parameters they will produce (these stored in $\mathbf{y}$). As the windows slide across a training interaction between the humans, many such $(\mathbf{x}, \mathbf{y})$ pairs are generated and presented as training data to the system. The task of the learning algorithm is to learn from these pairs and form a model relating $\mathbf{x}$ and $\mathbf{y}$. Once learning has converged, it can then generate a predicted $\mathbf{y}^*$ sequence whenever it observes a past $\mathbf{x}$ sequence. This allows it to compute and play out the future actions of one of the users (i.e. Mrs. Dash) when only the past interaction of the participants is visible.

Thus, the learning algorithm should discover some mouth openness behavioral properties. For example, Mrs. Dash usually remains quiet (closed mouth) while Mr. Solid is talking. However, after Solid has talked and then stopped briefly, Mrs. Dash should respond with some oscillatory signal. In addition, if Mr. Solid has been talking continuously for a significant amount of time, it is more likely that Mrs. Dash will interrupt assertively. A simple learning algorithm could be used to detect similar $\mathbf{x}$ data in another situation and then predict the appropriate $\mathbf{y}$ response that seems to agree with the system's past learning experiences.

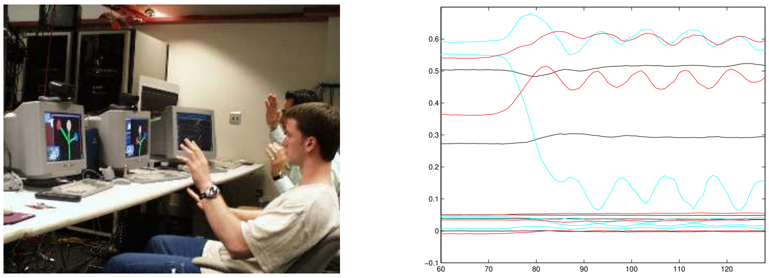
Note now that we are dealing with a somewhat supervised learning system because the data has been split into input $\mathbf{x}$ and output $\mathbf{y}$. The system is given a target goal: to predict $\mathbf{y}$ from $\mathbf{x}$. However, this process is done automatically without any manual data engineering. One only specifies a-priori a constant width for the sliding window that forms $\mathbf{x}$ and the width of the window of $\mathbf{y}$ (usually, the width will be 1 frame for $\mathbf{y}$ to conservatively forecast only a small step into the future). The system then operates in an unsupervised manner as it slides these windows across the data stream. Essentially, the learning uncovers a mapping between *past and future* to later generate its best possible prediction.

An interesting feature here (which we have stressed in the introduction chapter) is that the framework has a shared action and perception space. Therefore, synthesizing an action that is similar to a perceived one would merely involve copying the corresponding parameters. We don't need to uncover or use some complicated mapping. The AIM (Active Intermodal Mapping) problem that plagues many imitative learning approaches is effectively side-stepped. Therefore unraveling the complex mapping between a real human's activity and a robot or virtual agent's imitation of it is not necessary.

8.1.1 A Simple Example

We now present a simple example of the ARL framework in action. This demonstrates the feasibility of the approach in a particular domain but also permits us to later elucidate some weaknesses and motivate important extensions.

Here, the agent learns to imitate a teacher's head and hand gestures [97]. The world state is merely another person's head and hand coordinates which the teacher is interacting with. Figure 8.2(a) depicts the training scenario where two users are interacting with each other by performing simple gestures while a camera tracks their head and hands coordinates. Tracking is done merely by



(a) (b)

Figure 8.2: A Gestural Imitation Learning Framework. In (a) two users are being tracked in real-time to collect measurements of their head and hand positions. Each user sees a caricature of the other in real-time on his screen. In (b), a time series of the perceptual measurements from one user is shown. Each step in time represents the means and covariances of the 3 Gaussians that describe the head and hand positions.

modeling skin-colored[1] pixels in the image spatially as a mixture of three 2D Gaussians using EM. This gives us coordinates for the head and hands for both participants as a time series. One participant's parameters (actually the mean and covariance of the skin-shaped Gaussians) are shown in Figure 8.2(b). At each time step of the training data, the users' parameters over a window of the past few seconds is vectorized. This large dimensional space is compressed with PCA down to 60 dimensions and is used as input to predict the coordinates of the head/hands in the next frame (the output). The mapping from input to output is performed using a mixture of experts trained with maximum conditional likelihood (CEM) over a few minutes of interaction.

Once we have learned from the training data, the model can be used to synthesize behavior in response to new stimuli. By feeding back the prediction and looping it with real measurements off of a human, the interactive agent responds to novel gestures in real-time. Figure 8.3 depicts the online interaction process (after training) where a user is performing a tricky gesture and the virtual character model claps in response to it. Several simple gestures and their appropriate context are thus learned from data and can be re-synthesized in a stochastic real-time manner when a single user gestures towards the system.

8.1.2 Limitations

We now discuss three important limitations of the ARL framework and lead into extensions and tools that will resolve them. First, the lack of a higher order mapping makes it hard to acquire long term behavior. This is partly because we may not maintain the requisite fixed coordinate system of actions and perceptions. The use of PCA as a representation also prevents us from inserting important prior knowledge and structure in our model of the perception and temporal activity and thus restricts ARL's ability to generalize to novel behaviors. For example, simple yet nonlinear transformations of the training data may not be recognized by the model. Finally, ARL used conditional likelihood to focus on predicting the future from the past which is not necessarily where the discrimination power should be most appropriate. An alternative and more appropriate conditional criterion is predicting the teacher's actions (the one to be resynthesized) from another user's actions which will always be observable.

[1] The skin color distribution in RGB space itself is also modeled using EM and a mixture of 3D Gaussians.

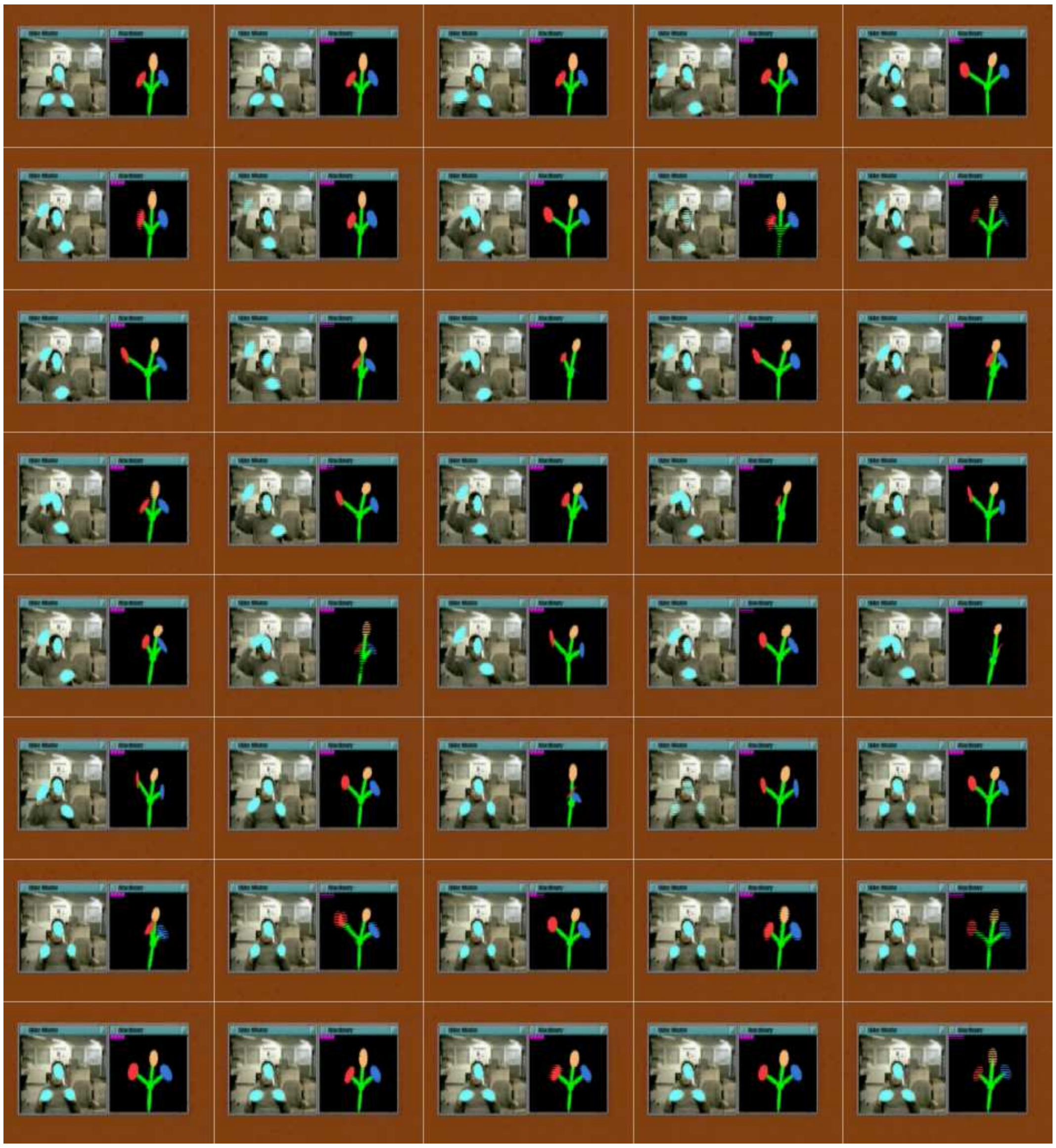


Figure 8.3: Online Interaction with the Learned Synthetic Agent. The user is being tracked in real-time while the agent synthesizes the most likely reaction to his activity. The synthesis is a real-time forecast and therefore varies according to slight variations in the human's gestures.

Consistent and Long Term Training Data

As we previously mentioned, ARL avoids the AIM problem and has a direct equivalence between action and perception. Thus we don't have a layer of abstraction which will recognize that a given gesture remains the same from different perspectives and viewing conditions. Therefore, we must take care that we maintain consistent representations and deal with changes in the coordinate system up front (i.e. if the cameras are moved, etc.). This is because the action and perception layer are not abstracted away for us by a higher order mapping. This often means that training data is short and scarce because it must be constrained to have the same consistent view point. In the setup above (Figure 8.2) we are relying on having a constant bird's eye view of the scene and the participants. A real organism that performs imitative learning does not have the luxury of a fixed view point of its teacher. Organisms are often mobile and perform long term (or lifelong) learning in a multi-perspective world. Although we won't show how to solve this problem explicitly, we will try to *alleviate* it by proposing imitative ARL learning on a wearable platform that is affixed to and follows a human teacher. This will let us extend the paradigm so that it can collect larger data sets for training the behavior model.

A Generalizable Behavior Representation

To render the learning tractable, ARL projects the window over the short term memory into a 60 dimensional approximation using PCA. This is reasonable since the time series data is made up of smooth trajectories. These will only get slightly smoothed from this approximation and will not lose too much detail (reconstruction accuracy is well above 90%). However, PCA may not be an appropriate representation since it does not capture any prior structure or knowledge we may have about the perceptual space or the temporal data. For example, a constant global translation of the head and hands should not change the meaning of a gesture. Furthermore, speed and phase variations in a gesture will also cause a radical change in a PCA representation even though the gesture itself may be the same. Unfortunately, these variations are not factored in the PCA representation and a subsequent predictive learning technique will have difficulty learning invariances to them and generalizing appropriately. Therefore, generative models are needed that can capture, or at least represent in a factored way, these types of variations up-front (be they temporal warpings, visual transformations or auditory variations).

Focusing on the Agent's Output Behavior

Another deficiency in ARL framework is that it is discriminative in an inappropriate sense: it predicts the future of both participants given the past. This regression model appropriately avoids wasting resources modeling the past when it has no predictive value for estimating the future (i.e. we optimize the conditional $p(future|past)$ and not the marginal $p(past)$). However, the past is not exactly given when the system has to interact with a single human being in an online synthesis scenario. The only data that is 'given' are the direct measurements from the external world and the single human who is triggering the system. Therefore, it would make more sense to try to form a conditional learning problem where the conditioning of the variables is broken into $p(agent|external\ world)$ where the external world may include other agents or other stimuli. While both measurements of the agent as well as the world are available during training, during testing (or online prediction) only the external stimulus will be available and it is up to the system to synthesize the agent's output measurements. Therefore, this form of conditioning is more appropriate.

8.2 Long Term Behavior Data Collection

To resolve the issue of maintaining a consistent perception of the teacher's behavior, we propose the use of a wearable computer system. This is a convenient way to collect a sizeable amount of data while the teacher engages in various natural activities and also preserves the regular conditions and point of view necessary for a non-AIM based imitative learning framework. In Figure 8.4 a user (or teacher) has a head mounted microphone and a camera mounted to a boom that perceives his face. This audio-video source is the perception space as well as the action space for the learner. Furthermore, a camera affixed to his glasses and a microphone that is aimed outwardly track the context (which is the perceived space for the external world). The wearable then collects data via a small computer that transcribes both channels of audio-video signals at roughly 100 megabytes per hour.

Figure 8.4: Two channel audio-visual imitation learning platform.

Other instances of wearable learning include work by Starner [180] and Clarkson [38]. However, only the contextual space was modeled and the teacher's action/perception data is not collected or coupled to the external context as in the above system. The platform above provides a consistent way to collect long-term data for the purposes of imitative learning without dealing with the problems of mapping from action to perception.

8.2.1 Data and Processing

The video that is captured arrives at 7Hz in both cameras and is stored as images of size 60 by 80 pixels in an 8-bit RGB representation. The images are first illumination-normalized by a histogram fitting operation. Then, the image is filtered using a Gaussian RGB mixture model of skin color to select only the skin-colored pixels from the images in both video streams. This focuses modeling resources on the wearable user's face and focuses the external stimuli on the head and hands of people in the scene (as well as occasional random skin-colored objects, unfortunately).

The audio being captured arrives at a 16kHz sampling rate in mono for each microphone (with 16-bit resolution on amplitude). The audio is processed by computing its sound energy and thresholding to avoid modeling insignificant background noise. Then we perform a fast Fourier transform over a Hamming window (with 50% overlap) and use magnitude values of the audio only (the phase information is discarded). Each resulting spectrogram is clipped to use only the lowest 200 frequencies of the audio signal (i.e. the highest represented frequency is about 6kHz). These spectrograms are generated at approximately 60Hz.

Figure 8.5: Wearable Interaction Data. Both the teacher and the world video are visible as well as the spectrograms for their audio signals.

Figure 8.5 portrays several frames of the user as he walks down a hallway and approaches an individual to begin conversation. The frames show both the user's face and his own eye's view (from the camera attached to his glasses). The spectrograms are shown interlaced between the corresponding video images. The higher frequencies are to the left and more recent vectors are at the bottom of each image. High intensities indicates high audio magnitude (the spectrograms are normalized for easy viewing).

8.2.2 The Task

Given this large data set, our task will be to train an imitative agent by learning a good predictive model of what the teacher will do given the external stimulus. The data set which spans several hours and hundreds of megabytes of images/spectrograms is initially split in half to form a training portion as well as a testing portion. Here, a piece of the training data is shown which covers approximately 10 seconds worth of images (both the teacher and the external view) as well as 10 seconds worth of spectrograms (from both microphones). The data is further split in half into a representation of the external world x (audio and video) and a representation of the agent y (audio and video). Therefore, we have $x = world$ and $y = agent$. We now have a standard regression formulation where we need to obtain y from x by learning from the training data.

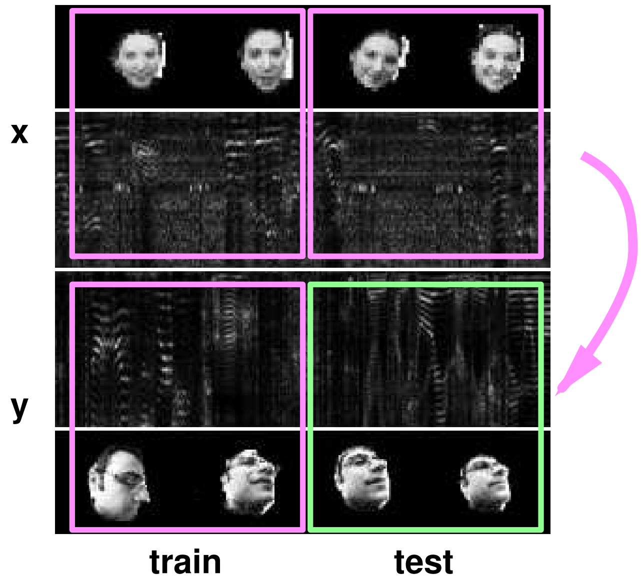


Figure 8.6: The Audio-Visual Prediction Task. The user is seen conversing with another participant in the external video source. Both his video signal and the participant are shown over a few frames as well as their audio signals (as spectrograms). Using the training data portion, we wish to learn a model that will map the outside world measurements x to the measurements y of the 'teacher' which the system will imitate.

8.3 Generative Modeling for Perception

As previous chapters argued, it would be naive to claim that the above raw sensory data we have captured (images and spectrograms) can be learned from directly without seeding the learning process with some structure. Although humans learn to see, learn to hear, and so forth, in a machine learning system, this unwieldy learning task must be seeded with some a priori structures, models and algorithms to make it tractable. Without a priori structure, data requirements and search space size grow rapidly and lead to inefficient convergence. Such is the case for the blank-slate approaches to behavioral model learning that Edelman proposes in the Darwin series of robots (Darwin III, IV, V) [49]. Therein, the direct mapping of color responses from a camera into a large unstructured neural network with connections to actuators does not seem practical and causes convergence problems.

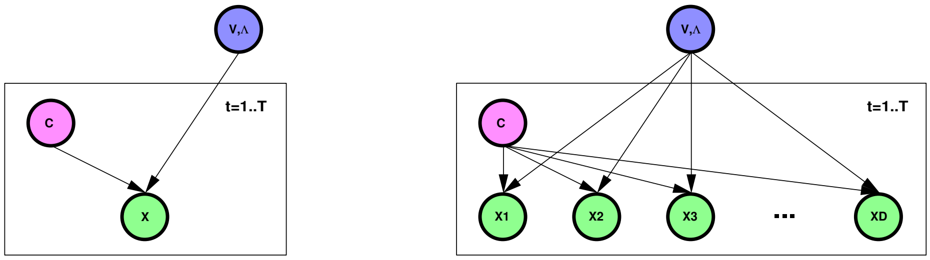
Of course, over-engineering the prior structure can be equally dangerous and there is a delicate balance between nature and nurture. Not only can over-engineered structures specify models that don't agree with the observed phenomena, providing too much extra knowledge and structure might also hurt. Extra knowledge and constraints can *reduce* tractability and computability. An algorithm which is guaranteed to find a global solution in polynomial time can be marred with local minima and poor gradient descent solutions when unusual additional constraints (which capture additional domain knowledge) are added to the system. In Bayesian networks, the specification of dependencies between random variable nodes is often restricted to a tree structure for propagation algorithms to converge and additional knowledge may create non-trees or cyclic links [149] [209]. We will thus propose a structured model that is not only intuitively plausible for the signals we will deal with but is also algorithmically feasible.

In this section, we describe a structured generative model which we will to use to handle the various perceptual signals being acquired by the hardware we proposed. We would like this model to quickly provide a useful and generalizable representation for the perceptual data that will be collected for imitation learning. Surprisingly, this generative model will be consistently applicable to both the image data and auditory data we will be dealing with. Although it may seem naive initially, this generative model will permit a mapping of high-dimensional audio and video into a compact representation which will greatly facilitate imitative learning.

Traditional ways to learn from and summarize complicated multi-variate data included, for example, principal components analysis (PCA)[20], factor analysis and multidimensional scaling [24]. Many such data summarization approaches can be cast in a generative model framework where *latent* values are estimated to compactly represent the data. Figure 8.8 depicts PCA as a generative Bayesian network.

Here, PCA's model (i.e. Θ) is the eigenvectors and eigenvalues (V, Λ respectively). The data is the X node (which can be split into each dimension as sub-nodes $X_1, \ldots, X_D$) while the coefficients c can be thought of as latent variables. The nodes in the 'box' drawing are replicated (T times for T instances of the data vectors). Given the eigenvectors, we are effectively forming a degenerate Gaussian model that can be sampled to produce the data. The latent variables constrain that Gaussian to a single mean value which, when sampled, will generate a spherical Gaussian over the data.

There are indeed many shortcomings with linear PCA as a representation which have encouraged variants such as Bayesian PCA [21], independent components analysis [14], auto-associator networks [20], and nonlinear embedding [183]. However, in applying these techniques to images, audio and time series, an important piece of knowledge is overlooked. All the aforementioned approaches assume the data they deal with can be rasterized directly into a vector form in a high-dimensional



(a) (b)

Figure 8.7: A Bayesian Network Description of PCA. The model Θ includes eigenvectors and eigenvalues V, Λ while the latent variables are the coefficients c. There are T instances of the data vector, X. The box acts as a replicator of the nodes T for all $t = 1..T$ instances of the training data. Figure (a) depicts the data vectors as single nodes which is equivalent to (b) which merely zooms in on each data vector by showing how it can be split into smaller tuples or its individual scalars $1..D$.

Euclidean space.

However, images, audio, and time series are not vectors. A single color image is not a single big vector but rather a *collection* of small vectors or tuples (pixels). Each of these tuples is vector of 5 values, (X, Y, R, G, B) which specify XY location and RGB color. Similarly, an audio spectrogram is not a vector but rather a collection of 2-tuples (A, F) where A is a band's amplitude and F is its frequency. Furthermore, a univariate time series is not a vector, it is a collection of 2-tuples (C, T) or coefficient value C and time value T. This permits us to generate an interesting variant of PCA where each datum in our data set (images, audio) is not a vector but rather a *collection* of vectors. This modification of the model is depicted in Figure 8.8 as a Bayesian network.

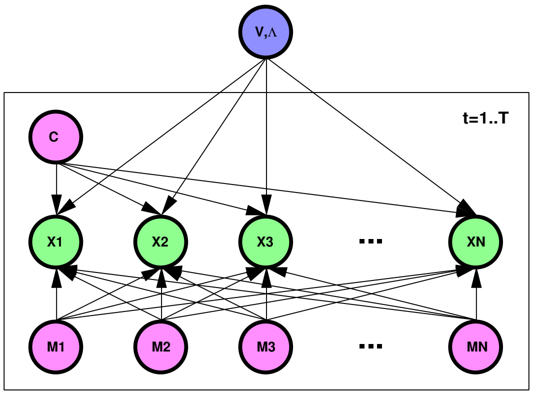


Figure 8.8: Bayesian Network Representing a PCA Variant for Collections of Vectors.

Here, we have split each data vector X (image, audio data, time series) into the subcomponent tuples $X_1, \ldots, X_N$ which agree with our prior knowledge about the data's structure. Since there is no explicit ordering on a collection of pixels or tuples, we can no longer put them into some fixed vector arrangement. Instead, multinomial variables $m_1, \ldots, m_N$ are introduced that assign an unknown discrete label to each subcomponent to *associate* it with a corresponding location or eigenvector subcomponent. Thus, the $m_1, \ldots, m_N$ compute a correspondence between the data and the model. Each m_i multinomial parameter vector can be thought of as a vector of positive mixture weights that sum to unity and effectively (in the hard case) picks a single destination assignment to an eigenvector component. We can stack the m_i vectors to form a large M-matrix which can also be constrained to sum to unity across columns. These 'matching' matrices are $N \times N$ and act as soft permutation matrices, re-sorting each data element's tuples *before* we compute the eigenspace. In regular PCA, these M matrices are locked to identity. Instead, we will allow them to be variable, doubly-stochastic matrices. Unlike regular matching procedures which match only a pair of images or a pair of exemplars, the matching in this generative model is done *simultaneously* over the whole data set. The M matrix elements can be appropriately repeated for each dimension in the tuple (i.e. in an image that is XYRGB, our tuples our 5 elements creating a $5N \times 5N$ matching matrix) to form a matrix that is the size of the x vector and we can image it transforming x as follows:

$$\hat{X} = MX$$

This transform will re-shuffle the tuples, forming a new data element $\hat{X}$ such that the resulting eigenspace will fit more accurately and with less reconstruction error. It turns out that if we also restrict the summation over M's columns to be unity, we get a doubly stochastic matrix which forces the eigenvectors and the data tuples to distribute in a soft one-to-one manner. This is equivalent to a softened case of the so-called *two-way assignment problem* [17]. This problem has an exact polynomial time solution called the *Auction Algorithm* [17]. However a statistical physics based approximation called the *Invisible Hand Algorithm* actually produces a more rapid result [112] [2]. Although a standard EM-propagation framework to the above problem could be feasible, we favored this alternative solution because of the doubly stochastic constraints and the faster convergence performance. The computation effectively iterates PCA computations interlaced with the invisible hand algorithm solutions until convergence (after a few dozen iterations). We compute the soft assignment matrices, then the coefficients, then the eigenspace iteratively, one at a time, while the other parameters are fixed. The solution seems to converge yet does so non-monotonically and is plagued by local minima. Finding a cleaner more global model and algorithm would be interesting avenues for future research. We now explicate how to apply this generative model to images and audio to finally form a multivariate time series of data.

8.3.1 A Generative Model on Images

We begin by collecting a small high-quality subset of images (a few hundred) of the agent and the external world after they have been skin-segmented (to obtain the faces). This data is then used to form a collection of (X, Y, I) tuples where I is the gray-scale intensity. Effectively, this representation treats the images and pixels as "point-clouds" where a collection of 3D points is considered instead of a continuous 2D intensity map. Approximately 2000 (X, Y, I) points are sampled from each image forming a speckled point-cloud representation of it. For each of these image sets in our data (agent and world), we then learn the latent coefficients, the doubly stochastic matrices as well as the eigenspace using the structured PCA model in Figure 8.8.

Figure 8.9 depicts the model's ability to reconstruct facial images of the teacher (after skin-color based segmentation) from a 20 dimensional representation (compare it with the reconstruction of PCA directly on these images). The advantage of the collection of vectors is that morphing in XY is just as easy as varying RGB quantities.

The reconstruction error for the variant is up to 2.5 orders of magnitude better than PCA for the same level of dimensionality reduction. Previous techniques in the literature also utilized optical

[2] As an interesting related side note, these same authors have also recently derived a statistical physics formulation for handling loopy Bayesian Networks.

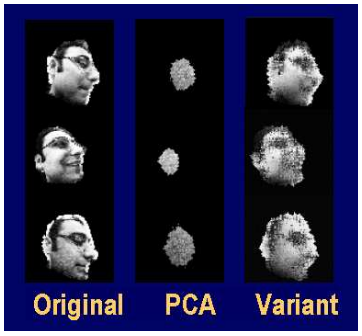
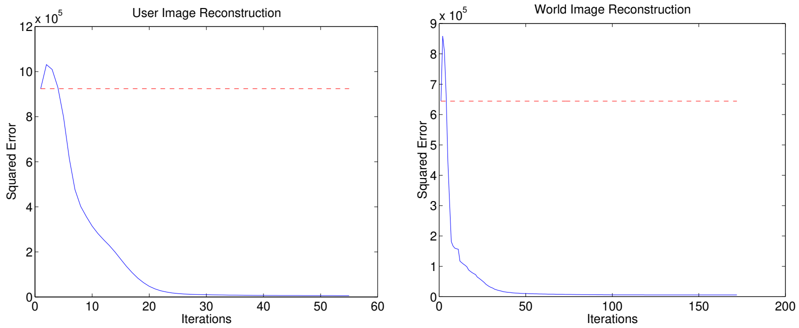


Figure 8.9: Reconstruction of Facial Images with PCA and Variant.

flow [18] or variations [143] of it for image matching and also produce better reconstruction than PCA.



(a) Agent Images (b) World Images

Figure 8.10: Reconstruction error for images of self (left) and world (right). 20 eigenvectors are used. The dashed red line depicts the reconstruction error of PCA while the solid blue line depicts the variant.

In Figure 8.10 we show the reconstruction accuracy of the variant compared to PCA where both use a projection down to 20 eigenvectors. The variant captures much more of the image structure by permitting pixel permutation. The whole data set is then processed with the agent's eigenspace where each agent image gets its own coefficients and matrix M. Each image is then summarized by the 20-dimensional coefficient vector that results. Similarly, all the world images are processed forming 20-dimensional coefficient vectors for each.

8.3.2 A Generative Model on Spectrograms

Similarly, we can apply the variant to spectrograms since these are not vectors either. In fact a spectrogram has structure since it is several 2-tuples of amplitude and frequency. These can be permuted to increase reconstruction accuracy over PCA as shown in Figure 8.11. Again, for 20 eigenvectors, reconstruction accuracy is over 2 orders of magnitude better for the variant. Note that we also factored out the magnitude (loudness) of the spectrogram from this representation and concatenate it to the coefficient vector. This produces a 21-dimensional vector that summarizes the

(200-dimensional) spectrogram data.

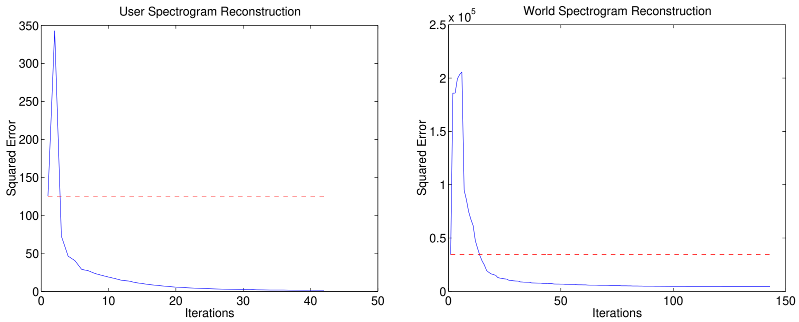


Figure 8.11: Reconstruction error for spectrograms of self (left) and world (right). 20 eigenvectors are used. The dashed red line depicts the reconstruction error of PCA while the solid blue line depicts the variant.

Spectrogram eigenspaces were learned for the agent's audio and for the world's audio from a small subset. Then, we compute the coefficient vectors for the whole data set of agent audio and world audio with their respective eigenspace to get a 21-dimensional coefficient vector for spectrogram at each time point in the sequential dataset.

8.4 Hidden Markov Models for Temporal Signals

The above representations generate a 20 or 21 dimensional vector for each frame of agent audio, agent video, world audio and world video. By processing the whole data set, we obtain four multidimensional time series of these coefficients. These are then time-aligned (with appropriate interpolation) and aggregated into a large 82-dimensional time series. Unlike the previous ARL formulation where PCA is used to represent a window over this time series, we shall instead describe the time series with a hidden Markov model. PCA does not have temporal invariance properties and therefore suffers from temporal misalignments in the data. It is well known that a hidden Markov model is well-suited to time series data and can effectively model time warpings and sequential variations. This arises by considering a hidden state variable and permitting summations of all possible state paths in the evaluation of the probability of a sequence. Figure 6.1 depicts the graphical model for a hidden Markov model however we shall make some simple modifications such that it more directly addresses the desired task at hand.

8.5 Conditional Hidden Markov Models for Imitation

Since we are interested in predicting the component of the time series (audio and video) that the agent would generate, we now have a discriminative prediction task, namely to predict y. This could be cast in the MED framework by learning a regression function derived from HMM likelihood measurements that would estimate the output with an epsilon-sensitive model. There are important problems with such an approach. Unlike a traditional regression setting, the output of the HMM model may exhibit some time-warpings and may not be perfectly aligned with the desired output even though it generates the appropriate behavior. Therefore, an epsilon-tube type of constraint may be inappropriate since it forces the outputs to be perfectly matched in time on a sample-by-sample basis. Furthermore, performing MED regression requires the resolution of many two-sided constraints on the output scalars. Since the training data has hundreds of thousands of samples as well as 20 output dimensions, the number of Lagrange multipliers would be in the millions making training very cumbersome. Therefore we will employ a maximum conditional likelihood as our surrogate discriminative framework to estimate the input-output HMM instead [16]. We will be discriminative in the conditional sense and utilize the CEM algorithm to estimate the HMM's parameters as elaborated (with Jensen and reverse-Jensen inequality derivations) in Chapter 6.

We begin by simplifying the time series learning task into two subproblems and actually learn two separate hidden Markov models conditionally. Recall that we had split the time series into half where x represents the signals that correspond the the world while y represents the signals that correspond to the agent. We can further split each of these into audio and video components where x_a is the audio component of the world signal while x_v is the video component. The y_a and the y_v are the agent's audio and video sequences respectively. We will learn one hidden Markov model such that it predicts $p(y_a|x_a, x_v)$ while we learn another to predict the agent's video, i.e. $p(y_v|x_a, x_v)$. This split is useful since it allows the hidden states in each Markov model to specialize into audio or video prediction and, more importantly, events in either domain occur at different time scales requiring a different transition model. Figure 8.12(a) depicts the HMM used to predict the agent's audio while Figure 8.12(b) depicts the HMM used to predict the agent's video. These form what is often called an input-output HMM structure [16] since the inputs x are related to outputs y through a hidden state s which evolves with Markovian dynamics.

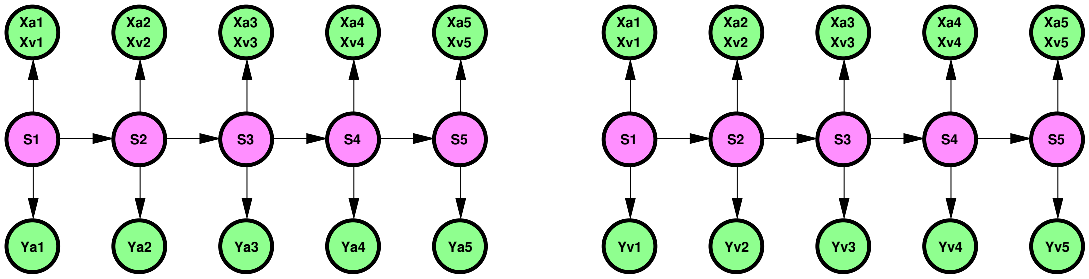
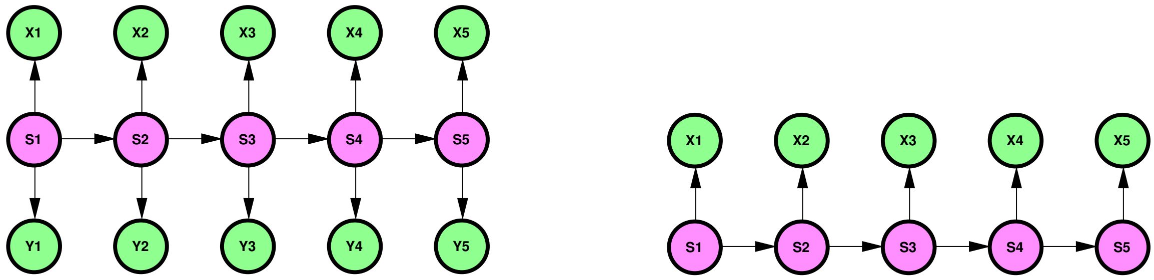


Figure 8.12: Audio and Video Prediction HMMs. In (a) the HMM is predicting the agent's audio signal from the external world's audio and video stimulus. In (b) the HMM is predicting the agent's video from the same input again.

We therefore train two input-output HMMs using CEM where we have 30 states for each and assume Gaussian emission models with diagonal covariance matrices. Therefore, we need to estimate the Gaussian means, covariances and state transition matrices for both these models. The first HMM has 21-dimensional Gaussian-emission models over y_a in the output and 41-dimensional Gaussian-emission models over x_a, x_v in the input. Meanwhile, the second HMM has 20-dimensional Gaussian-emission models over y_v in the output and 41-dimensional Gaussian-emission models over the x_a, x_v input.

Since we will estimate the HMMs using conditional likelihood, we will focus resources on predicting the agent's behavior *from* the input world stimulus. Therefore, we will maximize the joint log-likelihood of both the HMM over both x and y components of the time series (i.e. both inputs and outputs) *minus* the marginal likelihood of the HMM over the x component of the time series. This process is depicted in Figure 8.13. This permits us to focus on salient stimuli such as conversations with others since they results in a significant reaction from the agent. Meanwhile, episodes of the data where the agent is not expressing any interesting responses (i.e. walking alone in the hallway, background noise in the world, etc.) will be ignored even though they may contain much structure in the world-signal on its own. In a standard maximum likelihood scenario, these external world stimuli would get modeled and waste resources since they can be quite structured. However, their structures may not be important or useful in a task-related way and should not be modeled for their own sake unless they provide information about the output y. Furthermore, in using a conditional likelihood criterion, we become slightly more robust to model inaccuracies which maximum likelihood may be sensitive to. The behavior of the agent and the external world does not truly follow an HMM. Therefore, the guarantees of Bayesian and maximum likelihood estimation are lost. However, by imposing the task (agent prediction) on the system via conditional maximum likelihood, we can be less sensitive to some aspects of model-data mismatch.



$\log p(x, y) \quad - \quad \log p(x)$

Figure 8.13: Conditional HMM Estimation. Effectively, we maximize the joint $p(x, y)$ log-likelihood of the HMM over both x and y (inputs and outputs) minus the marginal $p(x)$ log-likelihood where the HMM has been summed over to obtain a marginal only over the x input.

The above optimization therefore requires us to maximize the conditional log-likelihood of the HMM parameters with respect to the time series of x_t, y_t vectors. This conditional log likelihood is simply $\log p(x, y) - \log p(x)$ yet these are latent distributions due to the state paths being hidden in an HMM. Therefore, we can write the conditional log-likelihood as: $\log \sum_s p(s, x, y) - \log \sum_s p(s, x)$ where we recognize the intractable log-sum and a negated log-sum forms as discussed in previous chapters. We utilize the Jensen inequality to lower bound the first term and the reverse-Jensen inequality to lower bound the second term. This then permits us to do a simple m-step to reestimate the HMM's parameters. This process is therefore the CEM loop for HMMs and is iterated until convergence of conditional likelihood.

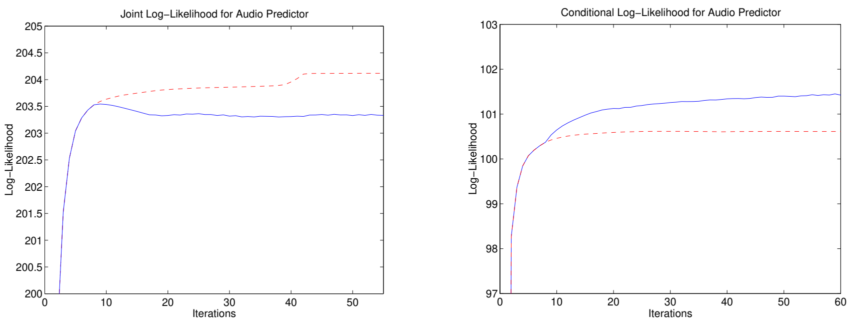
8.6 Experiments

The hidden Markov models are trained as in Chapter 6. The audio prediction HMM is first initialized with 5 iterations of EM and then converges with CEM to a maximum of conditional likelihood. The video prediction HMM is optimized only with CEM after a random initialization. Due to the large size of the dataset (the number of samples in the trellis is of the order of hundreds of thousands of samples), we could not compute the w_m parameters exactly for the reverse-Jensen inequality and instead used heuristics (described in Chapter 6) to make the optimization more efficient. Therefore, the monotonic convergence properties of CEM are compromised with these heuristics.

8.6.1 Training and Testing Likelihoods

The data set we have thus compiled is basically a 130,000 sample time series of multidimensional vectors (82-dimensional to be precise). We use the first 75000 samples for training while the last 55000 samples are used for testing. These each form one long continuous sequence and therefore, to train the HMMs, we have to deal with trellis sizes of up to 75000 samples. The forward-backward algorithm and its counter-part for the reverse-Jensen inequality need special consideration to operate on such a lengthy trellis window. We begin by scaling the α, β probabilities in the forward-backward algorithm appropriately to avoid numerical errors [156]. Furthermore, we avoid computing the reverse-Jensen bounds exactly since their computation scales poorly with large trellis sizes (in terms of number of samples and number of hidden states). Instead, we use the heuristic that $w_m = \tilde{w}_m$ where the reverse-Jensen bound's width parameter is copied from the width parameter of the regular Jensen bound (based on a symmetry type of argument).

Figure 8.14 depicts the convergence of the EM and the approximate CEM algorithm for the audio-prediction HMM. While EM converges monotonically to the maximum likelihood solution, it is inappropriate to optimize maximum likelihood in this setting since the world state (the x vectors) will always be measurable and need not be predicted or modeled on their own. Since we desire a conditional mapping from x to y, the conditional criterion is more suitable. Convergence for both algorithms required less than 30 minutes on a modest Pentium III machine.

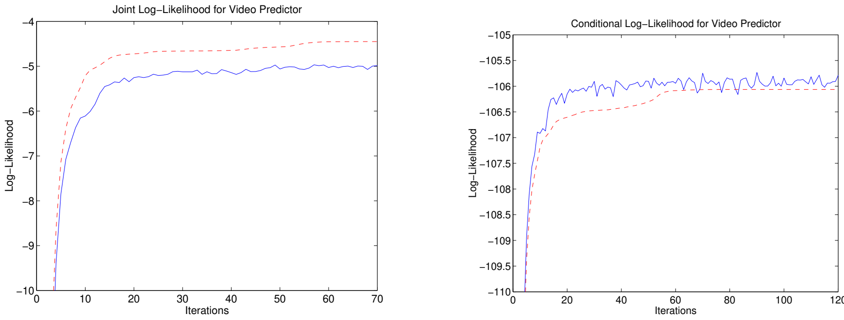


(a) (b)

Figure 8.14: Training Log-Likelihoods for Audio Prediction HMM. In (a) we show the joint log-likelihood per iteration and in (b) we show the conditional log-likelihood per iteration. The dashed red line is EM's resulting log-likelihoods while the continuous blue line is CEM's resulting log-likelihoods. Note the common starting point and the common first 5 EM-iterations for both algorithms. The use of heuristics to speed up the reverse-Jensen inequality for the HMM (trellis size here is T=75000) prevents monotonic convergence in CEM.

Figure 8.15 depicts a similar behavior of the learning algorithms on the video prediction HMM. Here the EM algorithm increases log-likelihood yet compromises conditional log-likelihood. The CEM counterpart focuses resources in the opposite direction. Training time for both algorithms required less than 30 minutes on a modest Pentium III machine.

The optimization of a surrogate conditional likelihood instead of ML is performed (i.e. CEM instead of EM) since we ultimately would like to optimize conditional likelihood on the test data. To evaluate the performance of our HMMs, we need to see how well they can estimate y from x. Therefore, it is unfair to evaluate performance on the test data with maximum likelihood. The x are always 'given' and therefore should not improve our score simply because we were able to predict them. Therefore, a predictive HMM that needs to generate y from x should be evaluated by a discriminative final



(a) (b)

Figure 8.15: Training Log-Likelihoods for Video Prediction HMM. In (a) we show the joint log-likelihood per iteration and in (b) we show the conditional log-likelihood per iteration. The dashed red line is EM's resulting log-likelihoods while the continuous blue line is CEM's resulting log-likelihoods. Note the common (random) starting point for both algorithms. The use of heuristics to speed up the reverse-Jensen inequality for the HMM (trellis size here is T=75000) prevents monotonic convergence in CEM.

criterion. However, unlike traditional classification problems where, for instance, classifier accuracy is a good performance metric, regression accuracy is not directly applicable to HMM forecasts and outputs. Regression accuracy is a natural and acceptable performance metric in a static function approximation problem. However, the HMM's outputs (just like its inputs) are sequences. These sequences can easily accommodate time-warping. Therefore, the HMM can generate time sequences with flexible time-warpings as well. It makes it inappropriate, consequently, to merely 'subtract' an HMM's regressed estimate from the testing data since the two may not be perfectly aligned. For example, consider the case where the output of our HMM was a sinusoid and it correctly matched the overall shape of the desired sinusoid in the testing output data. However, if a small phase shift were to arise, these two sinusoids would be misaligned and give rise to a large RMS error. It is unfair to penalize the HMM since it produced the desired overall behavior but was not in perfect alignment with the testing data. Thus, it is often inappropriate to use static regression evaluation techniques in an HMM setting. Instead, we will only report conditional likelihood accuracy on testing since other metrics do not capture the invariance in the predictions we would like to impose.

Log-Likelihood	EM	CEM
Joint under Training	204.12	203.33
Joint under Testing	202.55	202.22
Conditional under Training	100.61	101.50
Conditional under Testing	**99.61**	**100.58**

Table 8.1: Testing Log-Likelihoods for Audio Prediction HMM.

Table 8.6.1 depicts the testing log-likelihoods for the HMM on the agent's audio prediction task. We also show the training likelihoods for comparison. Both EM and CEM perform as expected, yielding good joint likelihood and conditional likelihood test scores respectively. However, CEM performs better during conditional likelihood testing (matching the performance EM had during its *training*, in fact) which is the desired outcome. Note, here, that the reported quantities are on a logarithmic scale and therefore, in terms of likelihood, the CEM solution is at least 2.5 times more likely.

Log-Likelihood	EM	CEM
Joint under Training	-4.45	-4.96
Joint under Testing	-21.12	-20.61
Conditional under Training	-106.06	-105.73
Conditional under Testing	**-122.46**	**-121.26**

Table 8.2: Testing Log-Likelihoods for Video Prediction HMM.

The situation is a little more unusual in the video prediction task shown in Table 8.6.1. Here, the testing log-likelihoods for the HMMs (and the training ones for that matter) are somewhat similar for both EM and CEM. Part of the problem could be that the video signals have a slower time scale than the audio ones so conditional learning may have less of a role. Furthermore, CEM's convergence was particularly poor here due to the speedup heuristics we are employing. CEM still performs better in terms of training and testing conditional likelihoods and, quite surprisingly, does slightly better on testing under *joint* likelihood as well.

While the above results provide a quantitative evaluation of the models, algorithms and the data fit, it is not clear that the HMM prediction systems are capable of synthesizing imitative behavior that qualitatively matches our expectations. In the next section we show the results of using the predictions to animate a synthetic character that mimics the agent teacher and comment anecdotally about the resulting behavior/performance.

8.7 Resynthesis Results

Given a generative model on the temporal data (i.e. the HMMs) as well as a generative model of the perceptual input (the structured variant of PCA), we can now synthesize agent reactions to test data where only the world stimulus is measured. This is done by first solving for the state distributions (i.e. trellis) using the measurements from the external stimuli alone. In other words, we use a marginalized hidden Markov model where the Gaussian emission models are only over the x component of the time series. The forward-backward algorithm then computes a distribution the hidden states in the HMM. Given the hidden states, it is straightforward to compute the expected value of the output vectors y_t at each time point by averaging the means of the Gaussian emissions weighted by their corresponding state assignment probability. This method is reminiscent of other HMM regression approaches where we map an input variable to an output by first solving for the hidden states. These include matching visemes and phonemes [27], adding stylistic variations to dynamics [28], coupling musical instrument expression to sound generation [173] as well as synthesizing facial expressions between interacting people [73]. However, the HMMs we have estimated used CEM and therefore their parameters where optimized specifically for this type of task. More formally, we have the following type of resynthesis (or regression):

$$\hat{y}_t = \sum_{m=1}^{M} \hat{\alpha}_t(m)\hat{\beta}_t(m)\vec{\mu}_y(m)$$

The above α, β values are the standard (normalized) forward-backward probabilities while the $\mu_y(m)$ is the 'y' vector component of the mean of the m'th Gaussian emission model (where $m = 1..M$ indexes the states of the HMM).

The hidden Markov models thus provide us with a time series of predicted vectors of audio and video coefficients for the agent. In other words, we have $\hat{y}_a$ and $\hat{y}_v$ for each time point. Given these coefficients, we can reconstruct the original signals (images and spectrograms) by simply multiplying the coefficients with the eigenvectors computed by the PCA variant. After some straightforward operations, it is possible to synthesize spectrograms and point-clouds. Unfortunately, the point-clouds are difficult to visualize and therefore, we merely find the closest nearest-neighbor in the training data to the vector $\hat{y}_v$ and render the corresponding image of the agent at that time point.

8.7.1 Resynthesis on Training Data

We now describe the predictions that the HMMs produce when applied to the training data. Here, we censor the true outputs y and only show the input signal x, namely the external world measurements. We begin by double-checking the HMM's performance on the training dataset. Since the HMMs were trained on these exemplars, this process does not test their ability to generalize and synthesize proper output behavior. Effectively, we are only evaluating the HMMs' ability to encode the training data and to repeat/mimic the behavior without extrapolation. What we can show is a compression argument since the HMM parameters effectively summarize the agent's y signals much more compactly than the original training data representation. Each of the hidden Markov models has roughly 30^2 parameters for the transition matrix and 30×60 parameters for the Gaussian mean and Gaussian (diagonal covariance) parameters. Therefore, with a total of 9000 scalar parameters, we are trying to summarize a 41-dimensional time series with approximately 75,000 samples. Thus, 3 million scalars are being captured using 9000 scalars. We obtain a compression level of over 3 orders of magnitude when we encode the data with the HMMs. In principle, though, this is a rough estimate since the accuracy or number of significant digits of the scalars (in the data and in the HMM) has not been accurately specified.

We next show the resynthesis for a few thousand samples as in Figure 8.16 which spans sample values 1733 to 3209 (the training data spans time indices 0-74999 while the testing spans 75000-130000). This figure was generated by taking individual frames from real-time (5Hz) movies we generated from the HMM predictions and the reconstructed audio and video coefficients (the movies are obviously easier to interpret than these figures yet unfortunately cannot be integrated into a document). In the figure, we portray, from left to right order, the world audio spectrograms, the agent's audio spectrograms, the 'cloud of points' PCA representation of the external world faces, the representation of the agent's face, then the raw video of the external world as well as the raw video of the agent. The raw video of the agent is computed from the point-cloud's representation from the PCA variant. We use the coefficient vector representing the cloud-point and find its nearest neighbor. This nearest neighbor match is found from all images in the training data set and their corresponding 20-dimensional vector representations. Efficient nearest neighbor searching is implementing using KD-trees. This lets us render the image that looks most like the point-cloud the generative model is recovering. It was crucial to find a nearest neighbor raw video match since the cloud of points is a confusing representation and causes excessive speckling. Note, also, that the point clouds have been translated and centered to the middle of the image.

The spectrograms in Figure 8.16 are also shown. Here, time for the spectrograms increases as we move from top to bottom in each image. Furthermore, as we move across the figure from image to image, time increases from top to bottom of the figure. Spectrograms can be inverted to play back sound however there are synthesis problems since the phase information is no longer present (we only model magnitudes). Although heuristics exist for reconstructing phase [75], we simplified the problem by using random phase values. This causes a significant amount of noise and clicking in the audio resynthesis but the audio is still discernible.

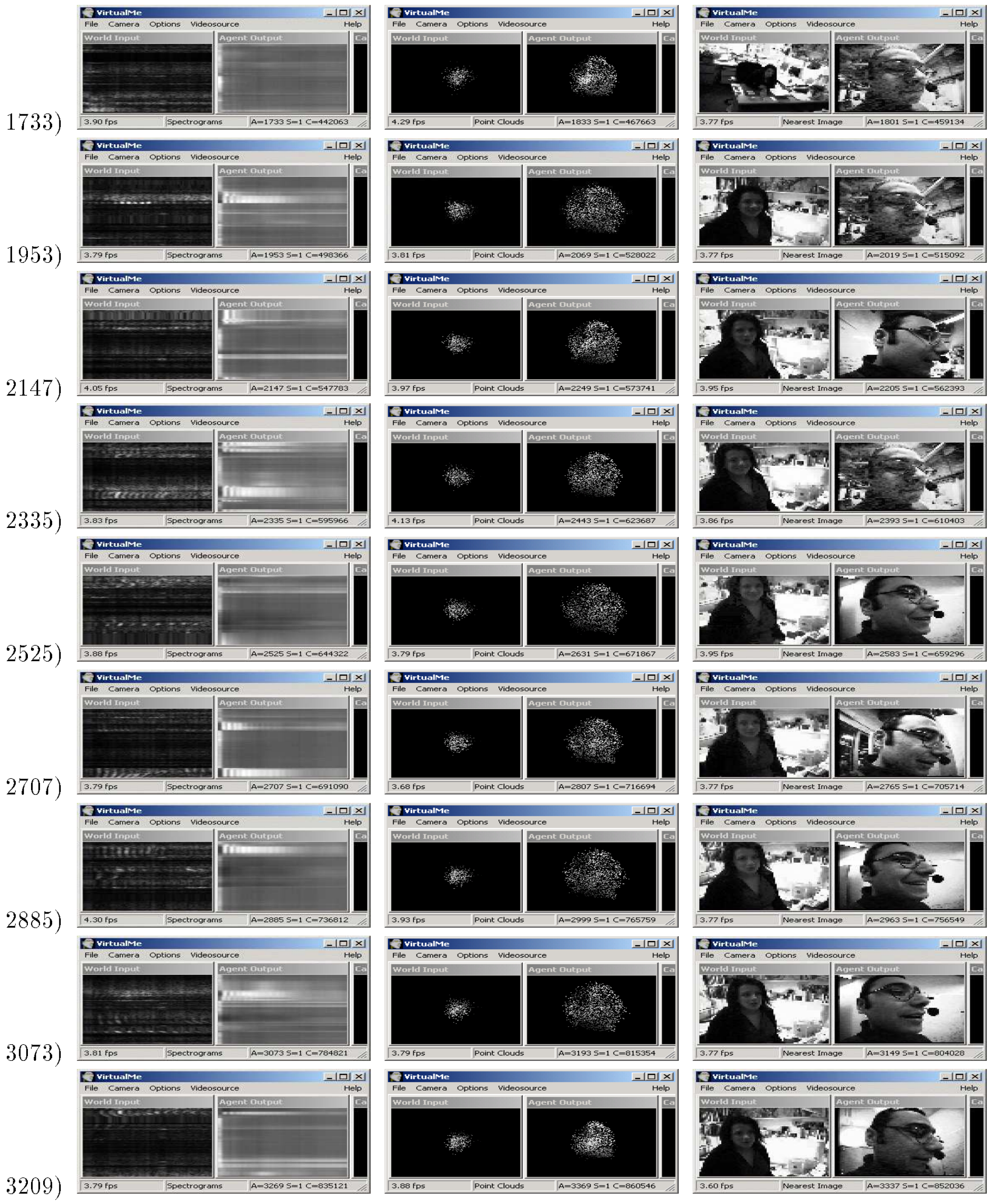


Figure 8.16: HMM Resynthesis on Training Data. Time is increasing from top to bottom at about 3 seconds per row. The figures are arranged from left to right as follows: the world spectrograms, the agent's spectrograms, the centered world visual point-cloud, the agent's visual point-cloud, the world's image data and the agent's image data reconstructed from the point cloud. Also, to the left of the figures is the time index for the time series.

Observing Figure 8.16, we see the synthetic agent initiating a conversation as he approaches an individual in the external world. The agent says "Hi" followed by the human saying "Hi". Then the agent says "how are you" which the human replies "fine and you". Most of the other agent articulations are mumbles that are difficult to decipher and generally sound like "I see", "hmm" and "yeah" except these are interleaved appropriately into the real human speech in the external world stream. The agent also says what sounds like "hi" and "how are you" at semantically inappropriate places in the conversation yet are interleaved appropriately in terms of timing with the real human's speech and audio energy. Furthermore, there is some head movement and visual cues of speech in the agent's synthesized video during the times when the agent is generating audio.

In Figure 8.17, we skip a few seconds and go to the end of the conversation the agent is having with this human and transition to the next person (time stamps 6265 to 7773). This is still data in the training set yet it is important to note that the agent maintains a similar interactive behavior even though there are many changes in audio/video signals as it transitions from one human to another. Nevertheless, both humans are female which provides some similarity at the spectrogram level. The agent responds to the real human audio with mumbled words that are again similar to the previous "Hi", "I See" and so forth. However, the audio energy and the timing are correctly interleaved and the agent remains quiet when it is not actively engaged by human conversation in the external world channel.

One important caveat here is that this form of resynthesis is not real-time and is not causal. The HMM has access to the whole sequence of external world stimuli to make its synthesized agent's audio/video. Therefore, it is possible to anticipate future events that would not be available in an online system. This lets us make more flexible predictions since the HMM can synthesize the output sequence in retrospect after obtaining measurements from the world channel that would otherwise follow the output. For example, when the agent approaches the individual in the external world, it says "Hi" first since the external individual begins speaking immediately thereafter and a new face initially appears from the noisy background. This form of anticipatory synthesized data would be more difficult to do in a real-time or in an online synthesis setting which would require the external world stimulus to trigger the system causally. In a sense, regressing all the agent's activity off-line using the HMMs is unfair since consistency can be achieved in the synthesized sequences without the usual causal constraints in real-time behavior synthesis. Furthermore, the agent's ability to synthesize audio that is semantically and/or meaningfully related to the external world triggers is not due to a linguistic understanding of the spectrograms. Rather, this is a byproduct of the HMM fitting or over-fitting to the training data and capturing an interaction with enough redundancy to repeat it but not to generalize it to novel situations. In fact, the agent also generates nonsensical responses (although they are acoustically well-timed) during this synthesis on training data which suggests that the compression achieved by the HMMs is discarding some information and summarizing some responses with other generic responses (i.e. forming a crude notion of a few prototype responses).

8.7.2 Resynthesis on Test Data

To test the imitative learning and the HMMs, we maintained 55,000 samples fully hidden from the training algorithms. These samples form a continuous sequence of several minutes of interactions (just as the samples used for the training sequence). Figure 8.18 depicts the resulting synthesized agent just as in the previous format. As the agent initially approaches the human, it remains quiet. Once they are within range and conversing, it interleaves rather mumbled "Hi", "I see", "How are you" and "I'm not sure" expressions with the audio of the human in the external world channel. Once again, there is some facial motion which appears to be most active when the agent is producing audio. The coupling is not as strong here as in the training case and there are no

Figure 8.17: HMM Resynthesis on Training Data Continued. Time is increasing from top to bottom at about 3 seconds per frame. The figures are arranged from left to right as follows: the world spectrograms, the agent's spectrograms, the centered world visual point-cloud, the agent's visual point-cloud, the world's image data and the agent's image data reconstructed from the point cloud. Also, to the left of the figures is the time index for the time series.

sensible responses to the external world triggers. The more meaningful responses in the training were, therefore, artifacts of over-fitting and the test shows that only a very superficial audio-visual interaction is occurring primarily being driven by the audio energy in the external world channel. Not only is there insufficient data to learn higher order behavioral patterns, it is also unlikely that a flat hidden Markov model can capture a meaningful notion hidden state. After all, it only has access to a few dozen hidden states and the simple spectrogram and visual features we are using. However, it is interesting that the model recovers some of the timing issues in the interactions and interjects audio that integrates smoothly into the conversation flow with "Umms", "Yeahs" and so forth. Such prosodic and textural interaction is typically difficult to design into structured synthetic conversational agents and speech recognition systems due to its behavioral as opposed to syntactical/semantic nature.

8.8 Discussion

We have thus seen two approaches to imitative learning. First, we discussed the Action-Reaction Learning approach which models the imitation as a time series prediction problem by learning a probabilistic model of future given past $p(future|past)$. Second, we have proposed the alternative imitative approach of learning where we predict the agent given the external world via: $p(agent|external\ world)$. Furthermore, instead of using PCA to represent audio-visual signals, we proposed a more structured variant that is well suited to handling images and spectrograms and has superior reconstruction accuracy. Also instead of using PCA to model temporal data, we proposed a more appropriate hidden Markov model framework which is able to handle time-varying signals and handle dynamic time warping. The HMM was estimated conditionally using CEM to obtain a more discriminative predictive distribution and therefore produces better forecasting than the standard EM approaches. This model was applied to a large data set of human interactions acquired with a wearable computer and used to then synthesize interactions from only external world stimulus. The system quantitatively performed better than EM and qualitatively generated interesting yet simple audio-visual responses to the external world channel triggers.

One important extension to the above is the transition to a real-time causal system. While the above HMMs benefited from having simultaneous access to all the external world channel (from sequence start to end), an online prediction setting would require an HMM to only receive observations in a causal stream from the external world channel while generating predictions itself. Therefore, we can compute the HMM's state trellis only on the available world measurements and only synthesize the agent's measurements in an incremental fashion. To make the forecasting more precise, it is important not to use the most recent estimates of the HMM since these may be less reliable than estimates that are bracketed by a small amount of world measurements in the near future. In other words, the synthesis of the agent could use a small lag, i.e. a few frames, to get better estimates as long as it is only a few milliseconds behind in its real-time responses to the user.

Another important caveat is that the acquisition of "behavior" here is being described in a very limited and constrained sense. The behavior does not involve any natural language processing or visual understanding and therefore is not based on any deep reaction to the stimulus but rather a superficial coupling between the audio and video of the two channels. It is clear that various synthetic systems that utilize speech recognition, natural language processing, specialized visual interfaces and manual animations generate more compelling synthetic interactions. However, the objective here is to recover these models from data. The learning process does uncover some slight facial motion responses as well as simple auditory reactions from real data. It is also able to synthesize them at somewhat appropriate times in response to simple acoustic and visual cues from the external world channel. The hidden Markov model is a manageable level of representation for recovering

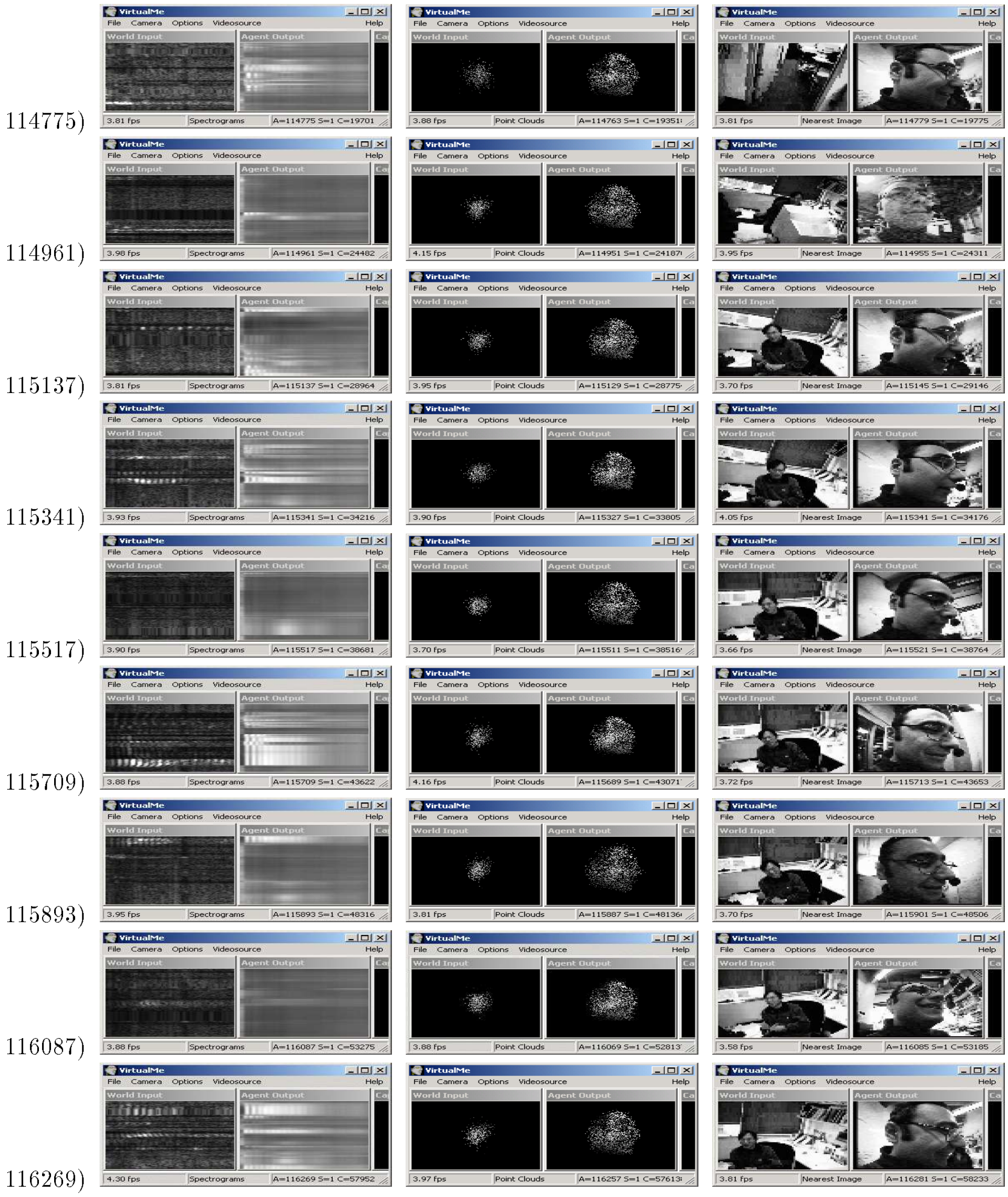


Figure 8.18: HMM Resynthesis on Testing Data. Time is increasing from top to bottom at about 3 seconds per row. The figures are arranged from left to right as follows: the world spectrograms, the agent's spectrograms, the centered world visual point-cloud, the agent's visual point-cloud, the world's image data and the agent's image data reconstructed from the point cloud. Also, to the left of the figures is the time index for the time series.

such simple couplings directly from data and does so reasonably in this setting. Furthermore, we have shown that a conditional estimate performs slightly better than a regular EM formulation since the objective is specifically to resynthesize agent measurements from external world triggers. Therefore, we have obtained a preliminary yet albeit simple implementation of the imitative learning paradigm we discussed. It was cast into the framework of generative models (a probabilistic variant of PCA as well as HMMs) which were further endowed with discriminative estimation (conditional likelihood) to focus on the task. The paradigm of imitative learning is a flexible one and was implemented here by-and-large without any manual user supervision. Therefore, simple imitative behavior can be acquired automatically merely by collecting data of a real human teacher's responses to the external world stimuli and attempting to mimic the agent's responses to the context using a statistical regression approach. The proposed paradigm is now amenable to more sophisticated generative models (i.e. hierarchical HMMs or stochastic grammars) as well as various specialized speech, vision and animation algorithms for more sophisticated results. These various augmentations can be cascaded into the overall imitation learning platform and hopefully will improve the realism of the synthetic interactions and make them more compelling.

Chapter 9

Conclusion

The thesis has motivated and situated two schools of thought: generative and discriminative learning. Both have deeply complementary advantages yet in their traditional incarnations also pose difficult incompatibilities. We have provided a common mathematical framework that unites the two and subsumes their strengths. The framework, based on maximum entropy, regularization theory and Jensen/reverse-Jensen inequalities provides a principled fusion of discriminative and generative learning. We can now span the rich and flexible space of generative models yet estimate them discriminatively to maximize performance on the tasks at hand. The probabilistic modeling resources are harnessed optimally by a discriminative criterion avoiding the intermediate sub-goal of estimating a good generator. The end result is better performance with the *same* models.

This framework was then applied to a real domain of imitation learning where a user's audio-visual interaction data is recorded with a wearable computer. Imitative learning is cast as a both discriminative and generative learning task. This data is processed and re-synthesized with generative perception while behavior is learned with discriminatively estimated generative models. The end result is a synthetic agent that learns stimulus-response interactive behavior autonomously in an unsupervised setting. The agent can then interactively respond to external stimulus and mimic the teacher's behavior in response to it. This provides an easy way for a naive user to create a synthetic character clone that demonstrates simple interactions.

9.1 Contributions

In this section, we enumerate and detail the various contributions of this thesis. These include conceptual, theoretical, practical and experimental types of contributions. The domains of relevance for these contributions include machine learning, machine perception, behavior modeling, statistics, mathematical inequalities, human-computer interaction and wearable computing.

- Motivating and Situating Generative, Conditional and Discriminative Learning

  The thesis provides motivation by referring to various works including the author's which depict the importance of generative learning in many applied domains. The author's own demonstrations in computer vision, human-computer interfaces, and wearable computing emphasize a probabilistic approach. In addition, various works in discriminative learning were also discussed which powerfully elucidate the deeply complementary advantages of both schools of thought (generative and discriminative). A taxonomy and formal treatment of various learning approaches was described across the range of generative, conditional and discriminative estimation as well as across various levels of model integration: local, prior and (Bayesian) averaging.

- Connecting Generative, Discriminative and Imitative Learning

  We provide several arguments for expressing imitative learning as both a generative and discriminative learning problem. The generative models are motivated by the need to re-synthesize actions or perceive behavior as well as seed the learning with priors and structures (i.e. Markov assumptions, representations, etc.). Discriminative learning is also made necessary by the emphasis on prediction accuracy as well as computational arguments about saliency and focus of attention. From a practical stand-point, the connection between imitation and discriminative/generative learning is also made feasible by avoiding the traditional active intra-modal mapping problems and considering a joint action-perception space. This then maps the problem directly into a time-series regression scenario.

- MED: A Combined Generative and Discriminative Framework

  Through the Maximum Entropy Discrimination (MED) formalism, this thesis provides a rigorous mathematical framework for spanning both the generative and discriminative schools of thought. Connections to regularization theory are made and the technique follows as a natural extension with interesting computational advantages. Discriminative support vector machines are subsumed as well as the whole range of generative exponential family models. The framework combines the flexibility of Bayesian modeling with discriminative estimation. Algorithms for estimating the exponential family, support vector machines, Gaussian models and multinomial models are portrayed and empirical results argue for their estimation using MED as opposed to a generative criterion. In addition to empirical verification, we further motivate MED by an appeal to three theoretical generalization guarantees that are based on well-established sparsity, VC-dimension and PAC-Bayes arguments.

- Extensions to the MED Framework

  We demonstrate the practical extensibility of the MED framework by discussing the metaphor of augmented distributions which permits us to cascade various estimation problems into the MED framework elegantly. We extend MED beyond the classification domain to the regression domain and subsume the support vector regression method as well as generative model based regression. Furthermore, discriminative feature selection in both regression and classification settings is presented and rendered tractable. Transduction in a classification and regression setting is also derived in closed-form. Empirical results justify these flexible extensions to standard discriminative classification. Various optimization techniques are also illustrated that make the framework competitive from a computational stand-point.

- CEM and Discriminative Mixture Models

  We present a rigorous mathematical framework to extend discriminative and conditional approaches to latent domains. This begins by recognizing the predominance of mixtures of the exponential family in the domain of generative learning. We elucidate the intractabilities that arise with latent models. Variational bounds are motivated as an efficient and principled way to resolve such intractabilities. A bound-based discriminative variant of the Expectation-Maximization, the so-called Conditional Expectation-Maximization is proposed which provides a tractable monotonically convergent algorithm for latent discriminative learning. The algorithm is based on the reverse-Jensen inequality which is shown to apply to all mixtures of the exponential family. The bounds are shown for mixtures of Gaussians, multinomials, Poisson, gamma, and exponential distributions. Monotonic convergence on conditional likelihood is guaranteed and shown to empirically outperform EM-based techniques. The bounds also render latent MED computations tractable. Furthermore, other important manipulations are elucidated which permit the bounds to apply to data sets, mixing proportions as well as other variations of mixture models.

- Discriminative Graphical Models

  The reverse-Jensen bounds are applicable to structured graphical models permitting them to be estimated discriminatively. However, the caveat is that intractable computation may arise. We demonstrated that efficient computation of the bounds is possible for some structured graphical models by deriving the reverse-Jensen inequality for an HMM. This clearly shows that the discrimination and bound techniques are potentially applicable to the case of structured graphical models such as Bayesian networks and thus permit our discriminative learning techniques to span a large portion of the generative modeling spectrum.

- Analytic Reversal of Jensen's Inequality

  An analytic reversal of Jensen's inequality was derived and globally guaranteed. The reversal goes beyond other Jensen reversals and Jensen converses in the literature to specifically target discriminative estimation problems. The reversal spans all mixtures of the exponential family and is tighter than curvature-based bounds. Other statistical and mathematical applications are also possible. The reversal permits dual sided bounds on many probabilistic quantities which otherwise only had a single bound from the standard Jensen inequality case.

- A Generative Model for PCA on Collections of Tuples

  We proposed a generative model that can be seen as an extension of principal components analysis (PCA) to collections of tuples. PCA applies to vectors which may not be the appropriate representation of topological data which requires the joint solution of the correspondence problem with the subspace estimation problem. We describe a generative model that introduces the correspondence solution as latent variables in PCA and can derive an iterative algorithm for estimating all relevant parameters. This model is readily applicable to images as well as other types of topographic signals and provides reconstruction that has orders of magnitude better squared error than PCA with the same level of coding/compression.

- A Wearable Platform for Behavior Acquisition

  A wearable platform for long-term behavior acquisition was developed. This apparatus is capable of collecting in real-time two channels of video and audio which capture the relevant aspects of the interaction of the user with his outside environment. The platform captures the user's reaction as well as the external world's stimulus in a consistent perceived frame of reference. This provides a digitized dataset that is well suited to imitation learning and agent behavior acquisition.

- Autonomous Imitative Learning for Interactive Agents

  An automatic framework was proposed for learning the interactions between an agent and the outside world, including other agents. The approach is fully unsupervised and results in an autonomous system which is able to discriminatively predict the behavior of an agent from the external stimulus. The perceptual signals are encoded with a generative model which permits straightforward resynthesis while the temporal behavior is learned with a conditionally estimated hidden Markov model. By simply observing the behavior being manifested, the system is able to learn the interactive model without any manual supervision and subsequently synthesize behavior automatically.

- Combining Perception, Learning and Behavior Acquisition

  We have demonstrated a real-time system that performs perception, learning and acquires and synthesizes real-time behavior. This closes the loop between generative perception, discriminative prediction and imitative learning we proposed and shows how a common discriminative and probabilistic framework can be extended throughout the multiple facets of a large system.

9.2 Future Theoretical Work

The tools developed in this thesis open many long-term questions as well as immediate directions for future work. This section speculates and postulates on the theoretical aspects of what was presented to challenge and stimulate ongoing efforts and ideas.

Having formed a joint generative-discriminative framework, it is now important to explore the continuum between the generative and discriminative solutions it can produce. As mentioned earlier, through a regularization parameter the MED framework can interpolate between a purely generative empirical Bayes model to a purely discriminative solution. What intuitions can be garnered about the appropriate level of regularization? Beyond regularization parameters, what other parameters in the framework (i.e. epsilon-insensitivity in regression, number of latent models, etc.) might have principled settings or may be estimable without brute-force cross-validation? Furthermore, what intuitions can we form about which models are most amenable to (and most likely to benefit from) discriminative estimation?

Another immediate problem is the presence of local minima in the CEM and MED frameworks when latent models are used. While MED effectively eschews the local minima problem for exponential family models and promises interesting convergence properties, global or pseudo-global solutions may be within reach for latent situations as well. Conventional deterministic annealing or regularization arguments are certainly possible avenues however a formal approach that specifically takes advantage of MED or the reverse-Jensen bounds may prove more appropriate.

The MED/CEM framework facilitates many important extensions which demonstrate and prove its flexibility. While transduction, feature selection, latent models, etc. have been explored, these may only be the proverbial tip of the iceberg and may open the flood-gates to other estimation scenarios. For instance, missing or corrupted data in the input space may be addressed with an appropriate prior and an augmented MED projection. Alternatively we may consider choosing other distributions for model priors, margin priors, bias priors, etc. to explore the effects these would have on the MED solutions. The proposed discriminative-generative frameworks also permit us to explore novel probabilistic models that would not necessarily be practical in a purely generative setting. This may include, for instance, un-normalizable generative models.

The reverse-Jensen approaches were demonstrated to be efficiently applicable to structured graphical models like the HMM. It is clear that the bounds can be applied to latent Bayesian networks yet can the efficient implementation derived for the HMM be translated into a more generic setting? For instance, is there a general junction-tree approach or generic recipe for computing the reverse-Jensen bounds on an arbitrary tree-structured latent Bayes net? Alternatively, we may consider other forms of structure beyond the independence properties of graphical models. For instance, latent estimation with models that not only have independency constraints but also have sufficiency and separability constraints or even causal structures could be investigated. Finally, various tightenings or alternative parameterizations of the bounds (under structured and unstructured models) may be possible by varying the derivation of the reverse-Jensen inequality leading to improved convergence as well as more efficient computation.

One critical future effort involves exploring other applications of the reverse-Jensen inequality. The derivation and its sub-components may have an impact in different situations other than discriminative learning. The bounds, for example, may be useful algorithmically to generate optimization routines, approximation methods, etc. for a wide range of domains. In addition, there may be other direct statistical applications. It is well known that Jensen is at the heart of many statistical inequalities which build up from it and therefore we may produce converses for these inequalities using the thesis' reverse-Jensen derivation. Similarly, more general mathematical applications and theoretical implications in other fields should be explored.

From a theoretical stand point, the relation of MED and reverse-Jensen bounds to other approaches such as Bayes point machines, relevance vector machines, boosting, exponential update rules, generalized additive models and so on should be elaborated. If deep connections can be made, some synergistic combinations may be feasible. Bridging multiple techniques and schools of thought was the predominant theme in this thesis. Therefore, continuing to find commonalities between learning frameworks is paramount to avoiding the many pitfalls of the field as well as harnessing its many strengths.

9.3 Future Applied Work

We now discuss future work that addresses the applied aspects of this thesis, namely the imitation learning and machine perception tools as well as novel applications of the machine learning algorithms we have developed.

The imitation learning platform we have described is capable of far more than a few hours of data acquisition. With a slightly 'ruggedized' version of the wearable, it may be possible to acquire several days worth of interaction data. Furthermore, a more adequate hardware platform would facilitate data acquisition due to the discomfort various individuals had while interacting with the user and the somewhat intrusive wearable apparatus. The longer-term and more natural data may lead to a more complex imitation learning where higher order behavior couplings could be modeled and more redundant patterns could be isolated. The possibility of performing the learning in an online setting would also permit continuous or developmental learning where the data acquisition process would not be separate from the model estimation stage and the behavior synthesis stage.

Admittedly, the implementation of imitative learning in this thesis was more generic and domain-independent than it had to be. The facial, auditory and temporal model had almost no manual engineering and were not domain specific. If a focused effort to form a parametric model of these types of data was made up-front, performance may be improved. For example, we may put stronger priors into the system and more sophisticated structures, effectively seeding the learning process with enough knowledge to converge to a more realistic autonomous agent. Therefore, various visual, auditory and temporal behavior models need to be explored including, in particular, hierarchical models which are potentially tractable in the discriminative frameworks we have outlined. Furthermore, more sophisticated features and representations of the interaction could be used. These include features from a speech recognition engine, contextual and ambient audio/video as well as more powerful facial modeling and tracking. Nevertheless, the excessive modularization and structuring of the problem may have its own disadvantages: it abstracts away and fixes the lower layers of processing in the task at hand and prevents a learning algorithm from exploring deeper relationships between the higher order behavior and the grounded sensory data.

However, mimicry of social interactive behavior is not the only domain of interest for the temporal interaction learning that was at the core of the imitation framework. Fundamentally, the system is discovering a discriminative higher order mapping between temporal processes. Learning to invert and re-synthesize such mappings has many applications ranging from control theory to music synthesis. Other applications can also be considered as we introduce novel sensors that go beyond the audio-visual platform that we investigated. The recent availability of practical and novel measurement devices such as accelerometers, physiological sensors, GPS and various other types of instrumentation permit us to collect a vast panorama of complex signals. While some may behave deterministically, predicting many of these will require modeling complex higher order behavior. This modeling process can be cast in the temporal/imitative learning platform we have discussed and may also benefit from discriminative estimation.

The generative/discriminative estimation framework (MED and CEM) have many other powerful applications on their own. One critical area is in the speech recognition domain. There, recent results have indicated that discriminative HMMs outperform many methods on difficult large-corpus recognition tasks (i.e. the switchboard dataset has been dominated by the Cambridge group's discriminative HMM estimation). Unfortunately, the discriminative variants used in the speech recognition community are strewn with heuristics, approximate bounds and local/gradient-based optimization. Furthermore, these are often more conditional than discriminative since the HMMs are not optimized to form a large-margin decision boundary for example. The MED and CEM machinery appears well suited to tackle these problems given that we have implemented a various tools for conditional and discriminative HMMs successfully in this framework. In fact, the list of machine learning application domains ranging from bioinformatics to web-page classification is simply too long to enumerate. Fortunately, this provides an endless array of challenging problems to explore and many potential clients for discriminative-generative learning.

Chapter 10

Appendix

This appendix provides various standard supplemental derivations and implementation details that help support the main thesis body.

10.1 Optimization in the MED Framework

At this point, we will discuss the various implementation details of the optimization of the $J(\lambda)$ objective function in the MED framework. The important feature is that $J(\lambda)$ is concave and therefore any procedure that locally increases it (monotonically or otherwise) will eventually converge to the global optimum. This will consistently give us the best setting of the Lagrange multipliers in the dual space optimization. Since consistent global convergence is guaranteed, we will instead focus on the speed of convergence and discuss multiple algorithms. Some natural optimization techniques in this setting include Newton-Raphson, gradient descent, line search and conjugate gradient descent. Unfortunately, these do not take advantage of some simple yet important decoupling properties in the objective function. This limitation is portrayed initially in our presentation of a simple constrained gradient descent approach. This then motivates the use of very fast axis-parallel approaches which benefit from the decoupling of the objective function which only requires 'local' computations. We finally propose an optimized variant of axis-parallel which learns how to transition between subsets of the variables to speed up the training process.

10.1.1 Constrained Gradient Ascent

One possible approach to maximizing $J(\lambda)$ is to compute the gradients with respect to the Lagrange multipliers and to take a small in their direction:

$$\lambda^{+} \;=\; \lambda^{-} + w \left.\frac{\partial J(\lambda)}{\partial \lambda}\right|_{\lambda^{-}}$$

Where λ^{+} are the next values of the Lagrange multipliers, λ^{-} are the previous ones and w denotes the step size. However, this form of optimization will disregard the constraints that λ are non-negative. This problem can be taken care of by reparameterizing them as follows:

$$\nu_t^2 \;=\; \lambda_t$$

We can use the surrogate variables ν which, when squared, form the λ vector. Therefore, we have:

$$\nu^+ = \nu^- + w \left.\frac{\partial J(\lambda)}{\partial \nu}\right|_{\nu^-}$$

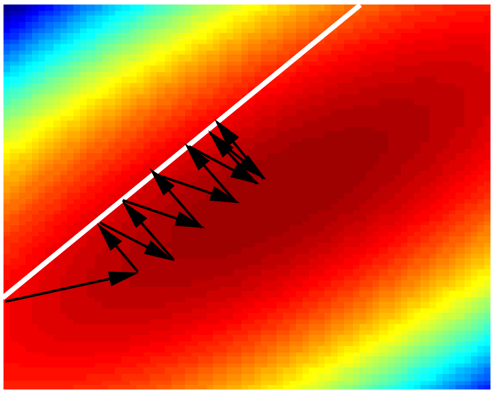


Figure 10.1: Constrained Gradient Ascent Optimization in the MED framework.

This maintains the non-negativity constraint on the λ vector as we perform gradient ascent. However, in problems where a non-informative bias is used, we also have the additional constraint: $\sum_t \lambda_t y_t = 0$. This can be resolved by projecting each step in the unconstrained gradient ascent back onto the plane $\sum_t \lambda_t y_t = 0$. However, since we are operating in ν-space, this planar constraint behaves as a quadratic constraint: $\sum_t \nu_t^2 y_t = 0$. Nevertheless, this projection is still solvable analytically.

In addition, to speed up the convergence, we allow the step size w to vary with each iteration. If the step results in an increase in the objective function $J(\lambda)$, then we take the step and also slightly increase w. If it doesn't result in an increase, we do not take the step and retry the gradient step with w scaled down by one half.

In practice, computing the gradients and the updated $J(\lambda)$ function is slow in many problems. This, compounded with the fact that we are constantly re-projecting onto the constraint surface, leads to slow convergence. One way to vastly improve the optimization process is to only consider updating a single λ_t variable at a time and only computing $J(\lambda)$ after that single perturbation. In many problems, this permits us to decouple the computations and effectively consider only a single datum at a time, speeding up each iteration considerably (i.e. by an order equal to the cardinality of the data set, T). This approach is elaborated in the following subsections.

10.1.2 Axis-Parallel Optimization

As discussed in the previous subsection, gradient ascent types of updates may not be efficient in the MED framework since each step requires computations of gradients and the new objective function over all the training data set. However, if we only consider updating a single Lagrange multiplier at a time, the computations only involve manipulation of a single data point in detail as well as some simple sufficient statistics that summarize the effects of the rest of the data. This principle, axis-parallel optimization, is similar to the notion of smallest possible working sets in [146] and Platt's sequential minimal optimization [154]. The difference here is that the working set is a single variable and we only optimize one dimension while all others are fixed.

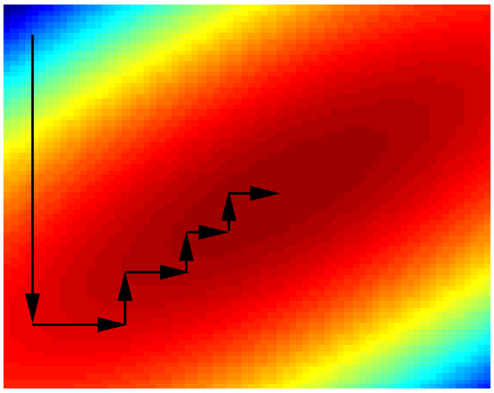


Figure 10.2: Axis Parallel Optimization in the MED framework.

Certainly, axis-parallel optimization has been around for a while and has its advantages and disadvantages. Other than computational efficiency, an additional advantage in MED is due to the overall concavity of the objective function. Thus, optimizing over a single variable at a time is guaranteed to increase the objective and iterating these axis optimizations will eventually converge to the global optimum. Figure 10.2 depicts the optimization in a toy 2D problem.

In certain cases, the update for a single axis can be computed analytically. Take for example the MED SVM kernel-based classification with a Gaussian prior of covariance σ^2 on bias (as opposed to a non-informative prior). We show here a resulting objective just as the one in Equation 3.11 where the constraint $\sum_t y_t \lambda_t = 0$ is no longer necessary:

$$J(\lambda) = \sum_t [\lambda_t + \log(1 - \lambda_t/c)] - \frac{\sigma^2}{2}\left(\sum_t y_t \lambda_t\right)^2 - \frac{1}{2}\sum_{t,t'} \lambda_t \lambda_{t'} y_t y_{t'} K(X_t, X_{t'})$$

The only constraints in effect in the objective function above is that the Lagrange multipliers are non-negative and upper bounded by c. A simple analytic update rule exists for maximizing one Lagrange multiplier, λ_i at a time. Holding all others fixed, taking derivatives with respect to λ_i and setting to zero yield a quadratic equation of the form:

$$\lambda_i \leftarrow \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$$

where we have the following scalars to specify the quadratic equation:

$$\begin{aligned}
A &= K(X_i, X_i) + \sigma^2 \\
B &= -1 - c\sigma^2 - cK(X_i, X_i) - y_i \sum_{t \neq i} K(X_i, X_t) y_t \lambda_t + \sigma^2 y_i \sum_{t \neq i} y_t \lambda_y \\
C &= -1 + c + c y_i \sum_{t \neq i} K(X_i, X_t) y_t \lambda_t - c\sigma^2 y_i \sum_{t \neq i} y_t \lambda_y
\end{aligned}$$

These two solutions to the quadratic equation are clamped so that $\lambda_i \in [0, c)$ and are then evaluated to see which one causes the greatest increase in the objective function. In certain cases, it is difficult to obtain an analytic update rule for a single Lagrange multiplier as above. We instead use Brent's method, which is a guaranteed 1D search method (which is more efficient than, eg. bisection search). This gives the maximum of the objective function for the single Lagrange multiplier numerically without too much computational overhead.

At this point, we will focus on how to choose the axes intelligently in the axis-parallel optimization. Typically, in axis-parallel, we iterate by randomly selecting one axis from the T possible choices (if the optimization of $J(\lambda)$ is T-dimensional) Eventually, the objective converges and we cease optimizing with a simple heuristic stopping criterion. Optimization is typically very fast and MED classification with hundreds of data points takes just a few seconds. We next discuss a more efficient strategy than random selection which can bring convergence improvements of about an order of magnitude (which can be important in large data set problems or high-dimensional feature selection).

10.1.3 Learning Axis Transitions

While the previous approach of randomly selecting an axis and maximizing it in isolation does produce a fast learning algorithm for MED, it can be made significantly faster by a smarter routine for axis selection. One strategy is to learn which axes are critical for producing a large improvement in our objective function $J(\lambda)$. This can be seen as a first order table or model which puts a scalar weight on each axis, measuring its expected contribution to the increase in the objective function. We could thus sample from this table as a distribution and update axes that are crucial to increasing $J(\lambda)$ more frequently than irrelevant axes. A natural extension to this T-element table is to consider a $T \times T$ matrix where columns corresponds to the last axis that was optimized and the rows correspond to the next axis to optimize. Each row of this stochastic matrix thus specifies a distribution over the choice of axes for the next iteration given the last candidate that was attempted. Effectively, we define a Markov transition matrix over axes. By identifying which axes are good followers of the current axis, we can sample more specifically from our list to get a greater expected improvement in the objective function.

More specifically, we compute the improvement in the objective function brought about by an axis optimization as $\Delta J(\lambda)$ as we go from an old value to a new one on an axis. Needless to say, all values of $\Delta J(\lambda)$ are non-negative (since each axis-parallel step is guaranteed to increase the objective). An additional problem is that the ΔJ values must be *discounted* since we expect large gains at the early stages followed by exponentially reducing gains in J as we near convergence. Therefore, we model the change of the time-varying values ΔJ_t as they arrive in an online manner over time. This is done by fitting an exponential model to the values of the form:

$$\Delta J_t \quad \approx \quad \alpha \exp(-\beta t)$$

This fitting of the parameterized curve can be done with a simple least squares criterion in an online way (i.e. we don't need to explicitly store the values of ΔJ_t). Figure 10.3 shows the fitting procedure to some values of ΔJ. Thus, we can now adjust the values of the ΔJ to obtain values which are appropriately discounted:

$$\tilde{\Delta} J_t \quad = \quad \Delta J_t - \alpha \exp(-\beta t)$$

This can now be seen as the current 'true' benefit of a given axis choice. In a greedy strategy, we pick the axis that generated largest $\tilde{\Delta} J_t$ from our current axis iteration. Thus, we can form a table of the $\tilde{\Delta} J_t$ with the expected value of an axis optimization given a current axis. At each iteration we select the axis which (given our current axis) has the highest value of $\tilde{\Delta} J_t$. We also still interleave random axis selections about 20% of the time to encourage exploration to fill up our table of axis-axis discounted objective function increments. In practice, we need not store all $T \times T$

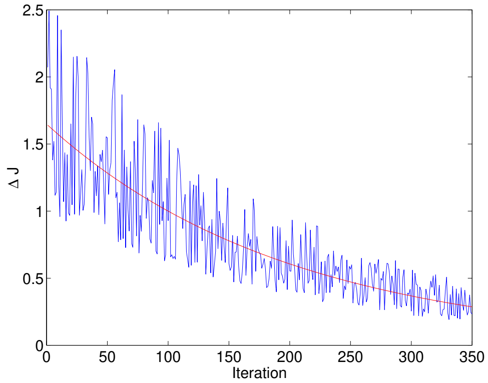


Figure 10.3: Approximating the decay rate in the change of the objective function.

axis-axis transition values but only the handful of transitions with the highest discounted $\tilde{\Delta} J$ values. Figure 10.4 depicts the approximately 10-fold increase in optimization speed that results from this axis choice strategy (here an MED linear regression with feature selection problem is depicted).

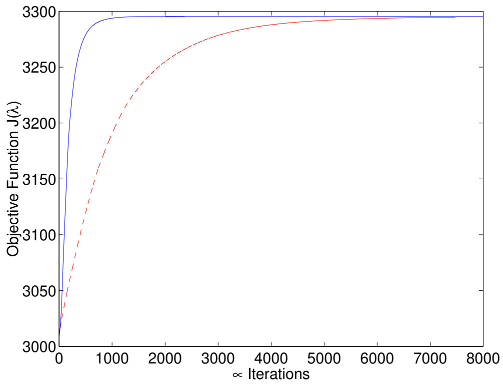


Figure 10.4: Axis-Parallel MED Maximization with Learned Axis Transition (solid line) and Random Transition (dashed line).

10.2 A Note on Convex Duality

We briefly review some aspects of convex duality that were used to develop Section 7.9. These are borrowed directly from a treatment of conjugacy and duality of convex functions (pp. 102-106) in Rockafeller [166] and mirror the development of Jaakkola [80]. A standard property of a closed convex set in $\Re^n$ is that it is the intersection of the closed half-spaces which contain it. This concept can be translated from convex sets to convex functions. As in sets, a closed proper convex function f on $\Re^n$ has an epigraph which is the intersection of the closed half-spaces in $\Re^{n+1}$ which contain it. These hyper-planes can be represented by linear functions on $\Re^{n+1}$ and we can define a closed convex function $f(x)$ as the point-wise supremum of the collection of all affine functions $h(x) = \lambda^T x - g(\lambda)$

such that $h(x) \leq f(x)$. Thus, we have:

$$\begin{aligned} h(x) &\leq f(x) \\ \lambda^T x - g(\lambda) &\leq f(x) \end{aligned}$$

By simply bringing $g(\lambda)$ in the above to the right hand side, we set up the simple inequality in Equation 10.1. The theory of conjugacy can be regarded as the theory of the "best" inequalities of this type:

$$\lambda^T x \leq f(x) + g(\lambda) \tag{10.1}$$

Let W denote the set of all pairs of functions $f(x), g(\lambda)$ for which the above inequality is valid. The "best" pairs of (f, g) in W for which the inequality cannot be tightened, i.e. those such that if $(f', g') \in W$, $f' \leq f$ and $g' \leq g$ then $f' = f$ and $g' = g$ occur when the functions are mutually conjugate. In other words, when $f = g^*$ and $f^* = g$. Any closed proper convex function f can undergo a conjugacy operation, $f \rightarrow f^*$ which produces its conjugate (or dual) function f^* in a symmetric one-to-one correspondence. The functions thus obey the following relation (by a trivial manipulation of Equation 10.1):

$$\begin{aligned} f^*(\lambda) &= \max_x \left\{ x^T \lambda - f(x) \right\} \\ f(x) &= \max_\lambda \left\{ x^T \lambda - f^*(\lambda) \right\} \end{aligned}$$

We can also replace the max or supremum operations in the above to obtain looser inequalities. Reinserting $f^*(\lambda) = g(\lambda)$ in Equation 10.1 gives us Fenchel's inequality for any closed convex function f and its conjugate f^*:

$$\lambda^T x \leq f(x) + f^*(\lambda)$$

Furthermore, it is easy to show that the dual of the dual function returns the original function, i.e. $f = (f^*)^*$. The conjugacy operation $f \rightarrow f^*$ is closely related to the Legendre transformation.

10.3 Numerical Procedures in the Reverse-Jensen Inequality

The Reverse-Jensen bound is computable analytically without lookup tables or numerical procedures. For completeness, we provide below a slightly tighter incarnation which involves maximizing a transcendental function that can only be done numerically. Recall that it was possible to use the values of (a, b) to specify the bounds numerically. These arose as parameters of linear upper bounds on the the function $f(\gamma, \omega)$ in Section 7.8. Recall that we needed to maximize $f(\gamma, \omega)$ over ω to obtain the desired function $g(\gamma) = \max_\omega f(\gamma, \omega)$. However, this is *merely* a one-dimensional function allowing us to solve for it numerically and write it down as a look-up table. It is easier to handle this function by considering linear upper bounds upon it parameterized as $a\gamma + b \geq g(\gamma)$. In Figure 10.5 the resulting function between a and b is plotted (on a logarithmic scale for better visualization). Furthermore, a detailed list of values is provided in Table 10.3 as well as the Matlab code for generating them numerically.

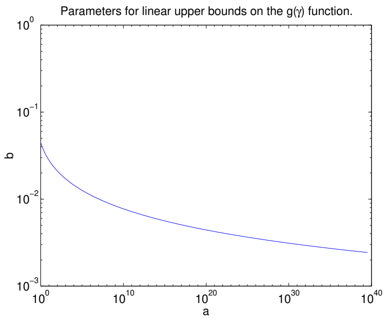


Figure 10.5: The function relating a and b values for linear upper bounds on the $g(\gamma)$ function.

a	b
1.00000e+00	4.47519e-02
6.19173e+00	3.01078e-02
3.83376e+01	2.34440e-02
2.37376e+02	1.94024e-02
1.46977e+03	1.66321e-02
9.10043e+03	1.46021e-02
5.63475e+04	1.30340e-02
3.48889e+05	1.17862e-02
2.16022e+06	1.07668e-02
1.33755e+07	9.91606e-03
8.28179e+07	9.19510e-03
5.12787e+08	8.57580e-03
3.17504e+09	8.03689e-03
1.96590e+10	7.56307e-03
1.21723e+11	7.14420e-03
7.53679e+11	6.77054e-03
4.66658e+12	6.43429e-03
2.88942e+13	6.13169e-03
1.78905e+14	5.85592e-03
1.10773e+15	5.60539e-03
6.85881e+15	5.37498e-03
4.24679e+16	5.16397e-03
2.62950e+17	4.96864e-03
1.62812e+18	4.78815e-03
1.00808e+19	4.62047e-03
6.24182e+19	4.46410e-03
3.86477e+20	4.31857e-03
2.39296e+21	4.18208e-03
1.48166e+22	4.05413e-03

```
% MATLAB CODE TO GENERATE THE LOOKUP TABLE


D=1000;
G=zeros(D,1);
Z=G;
W=G;
OPTI=zeros(15,1);
OPTI(2)=1e-6;
OPTI(14)=59000;

for j=1:D
  g    = (1.2)^(-D+j);
  s    = sprintf('-log((%e)*exp(x)+(%e)*exp(-x)-(%e)*2+1)/(x*x)',g,g,g);
  w    = fmin(s,1e-8,1000.0,OPTI);
  G(j) = g;
  Z(j) = log(g*exp(w)+g*exp(-w)+1-2*g)/(w*w);
end

A=zeros(D/2,1);
B=A;

for j=1:(D/2)
  a    = (1.1)^(j-3.2);
  b    = max(Z-G*a);
  A(j) = a;
  B(j) = b;
end
```

Table 10.1: List of a and b values for linear upper bounds on the $g(\gamma)$ function. Furthermore, Matlab code is provided next to the list to numerically generate the values of a and b.

Bibliography

[1] S. Amari. Information geometry of the EM and em algorithms for neural networks. *Neural Networks*, 8(9):1379–1408, 1989.

[2] M.A. Arbib and G. Rizzolatti. Neural expectations: a possible evolutionary path from manual skills to language. *Communication and Cognition*, 21:188–94, 1996.

[3] H. Attias. A variational baysian framework for graphical models. In *Advances in Neural Information Processing Systems 12*, 1999.

[4] A. Azarbayejani and A Pentland. Recursive estimation of motion, structure, and focal length. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6), 1995.

[5] A. Azarbayejani, C. Wren, and A. Pentland. Real-time 3-d tracking of the human body. In *Proceedings of IMAGE'COM 96*, May 1996.

[6] K. Azoury and M. Warmuth. Relative loss bounds for on-line density estimation with the exponential family of distributions. In *In Proc. 15th Conf. on Uncertainty in Artificial Intelligence*, 1999.

[7] N.I. Badler, C. Phillips, and B.L. Webber. *Simulating Humans: Computer Graphics, Animation and Control.* Oxford University Press, 1993.

[8] D. Barber and C. Williams. Gaussian processes for bayesian classification via hybrid monte carlo. In *Advances in Neural Information Processing Systems 9*, 1997.

[9] O. Barndorff-Nielsen. *Information and Exponential Families in Statistical Theory.* John Wiley & Sons, 1978.

[10] D.E. Barton. A class of distributions for which the maximum-likelihood estimator is unbiased and of minimum variance for all sample sizes. *Biometrika*, 43:200–202, 1956.

[11] J. Bates, B. Loyall, and S. Reilly. An architecture for action, emotion and social behaviour. Technical Report CMU-CS-92-144, School of Computer Science, Carnegie Mellon University, 1992.

[12] L.E. Baum. An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. *Inequalities*, 3:1–8, 1972.

[13] L.E. Baum and J.A. Egon. An inequality with applications to statistical estimation for probabilistic functions of a markov process and to a model for ecology. *Bull. Amer. Meteorol. Soc.*, 73:360–363, 1967.

[14] A.J. Bell and T.J. Sejnowski. An information maximisation approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

[15] A.J. Bell and T.J. Sejnowski. Learning the higher-order structure of a natural sound. *Network: Computation in Neural Systems*, 7, 1996.

[16] Y. Bengio and P. Frasconi. Input-output HMM's for sequence processing. *IEEE Transactions on Neural Networks*, 7(5):1231–1249, September 1996.

[17] D. Bertsekas and J. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Athena Scientific, 1997.

[18] D. Beymer and T. Poggio. Image representation for visual learning. *Science*, 272, 1996.

[19] A. Billard and G. Hayes. Robot's first steps, robot's first words. In *GALA'97 Conference*, Edinburgh, 1997.

[20] C. Bishop. *Neural Networks for Pattern Recognition.* Oxford Press, 1996.

[21] C. Bishop. Bayesian pca. In *Advances in Neural Information Processing Systems 11*, 1999.

[22] C. Bishop and M. Svens. Gtm: The generative topographic mapping. *Neural Computation*, 10(1):215–235, 1998.

[23] B. Blumberg, P. Todd, and P. Maes. No bad dogs: Ethological lessons for learning. In *From Animals To Animats, Proceedings of the Fourth International Conference on the Simulation of Adaptive Behavior*, 1996.

[24] I. Borg and P. Groenen. *Modern Multidimensional Scaling : Theory and Applications.* Springer Series in Statistics, 1997.

[25] G. Box and G. Tiao. *Bayesian Inference in Statistical Analysis.* John Wiley & Sons, 1992.

[26] M. Brand. Structure discovery via entropy minimization. In *Neural Information Processing Systems 11*, 1998.

[27] M. Brand. Voice puppetry. Technical Report 99-20, Mitsubishi Electric Research Labs, 1999.

[28] Matthew Brand and Aaron Hertzmann. Style machines. In Kurt Akeley, editor, *Siggraph 2000, Computer Graphics Proceedings*, pages 183–192. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

[29] C. Bregler. Learning and recognizing human dynamics in video sequences. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 1997.

[30] C. Bregler and S. Omohundro. Nonlinear manifold learning for visual speech recognition. In *ICCV 95*, 1995.

[31] R.A. Brooks, C. Breazeal, M. Marjanovic, B. Scassellati, and M. Williamson. *The Cog Project: Building a Humanoid Robot.* Lecture Notes in Computer Science: Springer, (in press).

[32] L.D. Brown. *Fundamentals of Statistical Exponential Families.* Institute of Mathematical Statistics, 1986.

[33] P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, J. Lafferty, R. Mercer, and P. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990.

[34] W. Buntine. Operations for learning with graphical models. *JAIR*, 2, Dec. 1994.

[35] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Knowledge Discovery and Data Mining*, 1998.

[36] J. Cassell. Embodied conversational agents: Representation and intelligence in user interface. *AI Magazine*, (in press).

[37] E. Charniak. *Statistical Language Learning.* MIT Press, Cambridge, MA, 1993.

[38] B. Clarkson and A. Pentland. Unsupervised clustering of ambulatory audio and video. In *ICASSP '99*, 1999.

[39] S. R. Cooke, B. Kitts, R. Sekuler, and M.J. Mataric. Delayed and real-time imitation of complex visual gestures. In *Proceedings of the International Conference on Vision, Recognition, Action: Neural Models of Mind and Machine*, 1997.

[40] T.M. Cover and J.A. Thomas. *Elements of Information Theory.* John Wiley & Sons, 1991.

[41] I. Csiszar and G. Tusnady. Information geometry and alternating minimization procedures. *Statistics and Decisions*, 1:205–237, 1985.

[42] D. DeCarlo and D. Metaxas. Deformable model-based shape and motion analysis from images using motion residual error. In *International Conference on Computer Vision*, 1998.

[43] S. Della Pietra, V. Della Pietra, and J. Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, 1997.

[44] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, B-39, 1977.

[45] T.G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[46] S.S. Dragomir, editor. *Journal of Inequalities in Pure and Applied Mathematics*, 2000.

[47] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* John Wiley & Sons, 1973.

[48] S. Dumais. Using svms for text categorization. *IEEE Intelligent Systems Magazine*, 13(4), 1998.

[49] G.M. Edelman, G.N. Reeke, W.E. all, G. Tononi, D. Williams, and O. Sporns. Synthetic neural modeling applied to a real-world artifact. In *Proc. Natl. Acad. Sci.*, volume 89, pages 7267–7271, 1992.

[50] D. Edwards and S. Lauritzen. The tm algorithm for maximising a conditional likelihood function. *To Appear in Biometrika*, 2001.

[51] L. Emering, R. Boulic, S. Balcisoy, and D. Thalmann. Real-time interactions with virtual agents driven by human action identification. In *First ACM Conf. on Autonomous Agents'97*, 1997.

[52] T. Evgeniou, M. Pontil, C. Papageorgiou, and T. Poggio. Image representations for object detection using kernel classifiers. In *Proceedings of Asian Conference on Computer Vision*, 2000.

[53] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. In *Advances in Computational Mathematics*, volume 13, 2000.

[54] O. Faugeras and T. Papadopoulo. A nonlinear method for estimating the projective geometry of 3 views. In *Sixth International Conference on Computer Vision*, pages 477–484, January 1998.

[55] A. Fern and R. Givan. Online ensemble learning: An empirical study. In *Proceedings of the Seventeenth International Conference on Machine Learning*. Morgan Kaufmann, 2000.

[56] W. Freeman and E. Pasztor. Markov networks for super-resolution. In *Proceedings of 34th Annual Conference on Information Sciences and Systems*, 2000.

[57] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on Machine Learning*, 1996.

[58] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[59] B. Frey and N. Jojic. Estimating mixture models of images and inferring spatial transformations using the em algorithm. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1999.

[60] J.D. Funge. *Making Them Behave: Cognitive Models for Computer Animation.* PhD thesis, University of Toronto, Graduate Department of Computer Science, 1997.

[61] A. Galata, N. Johnson, and D. Hogg. Variable-length markov models of behaviours. *Computer Vision and Image Processing*, 81(3), 2001.

[62] N. Gershenfeld and A. Weigend. *Time Series Prediction: Forecasting the Future and Understanding the Past.* Addison-Wesley, Santa Fe Institute Studies in the Sciences of Complexity, 1993.

[63] Z. Ghahramani and M. Beal. Variational inference for bayesian mixture of factor analysers. In *Advances in Neural Information Processing Systems 12*, 1999.

[64] Z. Ghahramani and M. Jordan. Learning from incomplete data. Technical Report MIT-AITR-1509, Massachusetts Institute of Technology, 1995.

[65] M. Gibbs. *Bayesian Gaussian Processes for Regression and Classification.* PhD thesis, University of Cambridge, 1997.

[66] F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7:219–269, 1995.

[67] A. Hartemink, D. Gifford, T. Jaakkola, and R. Young. Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. In *In Pacific Symposium on Biocomputing*, volume 6, pages 422–433, 2001.

[68] T. Hastie and R. Tibshirani. *Generalized Additive Models (Monographs on Statistics and Applied Probability Series, 43).* CRC Press, 1990.

[69] D. Heckerman, D. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1:49–75, 2000.

[70] D. Heckerman, C. Meek, and G. Cooper. A bayesian approach to causal discovery. Technical Report MSR-TR-97-05, Microsoft Research, 1997.

[71] B. Heisel, T. Poggio, and M. Pontil. Face detection in still gray images. Technical Report 1687, Massachusetts Institute of Technology, 2000. AI Memo.

[72] R. Herbrich and T. Graepel. Large scale bayes point machines. In *Advances in Neural Information System Processing 13*, 2001.

[73] D. Hogg. Synthetic interaction. http://www.comp.leeds.ac.uk/dch/interaction.htm, 2000. Web Page.

[74] D. Hogg, N. Johnson, R. Morris, D. Buesching, and A. Galata. *Visual Models of Interaction. Spatial Temporal Modelling and its Applications.* Leeds University Press, 1999.

[75] T. Horn. Image processing of speech with auditory magnitude spectrograms. *Acustica - Acta Acustica*, 84:175–177, 1998.

[76] D. Hosmer and S. Lemeshow. *Applied Logistic Regression.* Wiley & Sons, 1989.

[77] M. Iacobini, R.P. Woods, M. Brass, H. Bekkering, J.C. Mazziotta, and G. Rizzolatti. Cortical mechanisms of human imitation. *Science*, 286, 1999.

[78] S. Intille. *Visual Recognition of Multi-Agent Action.* PhD thesis, Massachusetts Institute of Technology, 1999.

[79] M. Isaard and A. Blake. A mixed-state condensation tracker with automatic model-switching. In *International Conference on Computer Vision 6*, 1998.

[80] T. Jaakkola. *Variational Methods for Inference and Estimation in Graphical Models.* PhD thesis, Massachusetts Institute of Technology, 1997.

[81] T. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. In *http://www.ai.mit.edu/-tommi/*, 1998.

[82] T. Jaakkola, M. Diekhans, and D. Haussler. Using the fisher kernel method to detect remote protein homologies. In *Proceedings of ISMB'99*. Copyright AAAI, 1999.

[83] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems 11*, 1998.

[84] T. Jaakkola and M. Jordan. Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10:25–37, 2000.

[85] T. Jaakkola, M. Meila, and T. Jebara. Maximum entropy discrimination. In *Advances in Neural Information Processing Systems 12*, 1999.

[86] E.T. Jaynes. Information theory and statistical mechanics. *Phys. Rev.*, 106:620–630, 1957.

[87] T. Jebara, A. Azarbayejani, and A. Pentland. 3d structure from 2d motion. *IEEE Signal Processing Magazine*, 16(3), 1999.

[88] T. Jebara, C. Eyster, J. Weaver, Starner T., and A. Pentland. Stochasticks: Augmenting the billiards experience with probabilistic vision and wearable computers. In *Proceedings of the International Symposium on Wearable Computers*, 1997.

[89] T. Jebara, Y. Ivanov, A. Rahimi, and A. Pentland. Tracking conversational context for machine mediation of human discourse. In *AAAI Fall 2000 Symposium - Socially Intelligent Agents*, 2000.

[90] T. Jebara and T. Jaakkola. Feature selection and dualities in maximum entropy discrimination. In *Uncertainty in Artifical Intelligence 16*, 2000.

[91] T. Jebara and A. Pentland. Parametrized structure from motion for 3d adaptive feedback tracking of faces. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[92] T. Jebara and A. Pentland. Maximum conditional likelihood via bound maximization and the cem algorithm. In *Advances in Neural Information Processing Systems 11*, 1998.

[93] T. Jebara and A. Pentland. Action reaction learning: Automatic visual analysis and synthesis of interactive behaviour. In *International Conference on Vision Systems*, 1999.

[94] T. Jebara and A. Pentland. On reversing jensen's inequality. In *Advances in Neural Information Processing Systems 13*, 2000.

[95] T. Jebara, K. Russel, and A. Pentland. Mixtures of eigenfeatures for real-time structure from texture. In *Proceedings of the International Conference on Computer Vision*, 1998.

[96] T. Jebara, B. Schiele, N. Oliver, and A. Pentland. Dynamic personal enhanced reality system. Technical Report 463, Massachusetts Institute of Technology - Vision and Modeling, Cambridge, MA, November 1998. Also appears in Proceedings of the 1998 Image Understanding Workshop.

[97] T.S. Jebara. Action reaction learning: Analysis and synthesis of human behavior. Master's thesis, MIT Media Laboratory, 1998. Vision and Modeling TR# 507.

[98] F.V. Jensen. *An Introduction to Bayesian Networks*. Springer, 1996.

[99] J.L.W.V. Jensen. Sur les fonctions convexes et les inegalites entre les valeurs moyennes. *Acta Math.*, 30:175–193, 1906.

[100] T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the 16th International Conference on Machine Learning*, 1999.

[101] N. Johnson, A. Galata, and D. Hogg. The acquisition and use of interaction behaviour models. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 1998.

[102] M. Jordan and C. Bishop. *Introduction to Graphical Models*. In progress, 2001.

[103] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. *Learning in Graphical Models*, chapter An introduction to variational methods for graphical models. Kluwer Academic, 1997.

[104] M.I. Jordan. *Learning in Graphical Models*. Kluwer Academic Publishers, 1998.

[105] M.I. Jordan and R.A. Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6:181–214, 1994.

[106] L. Kaelbling, A. Cassandra, and J. Kurian. Acting under uncertainty: Discrete bayesian models for mobile-robot navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1996.

[107] L.P. Kaelbling and M.L. Littman. Reinforcement learning: a survey. *Journal of Artificial Intelligence Research*, 4, 1996.

[108] T. Kailath. *Linear Systems*. Prentice-Hall, Englewood Cliffs, NJ, 1980.

[109] M. Kearns, Y. Mansour, Ng. A., and D. Ron. An experimental and theoretical comparison of model selection methods. *Machine Learning*, 27(1):7–50, 1997.

[110] J. Kivinen and M. Warmuth. Boosting as entropy projection. In *Proceedings of the 12th Annual Conference on Computational Learning Theory*, 1999.

[111] D. Koller and M. Sahami. Toward optimal feature selection. In *Proceedings of the 13th International Conference on Machine Learning*, 1996.

[112] J. Kosowsky and A. Yuille. The invisible hand algorithm: Solving the assignment problem with statistical physics. *Neural Networks*, 7:477–490, 1994.

[113] S. Kullback. *Information Theory and Statistics.* Dover Publications Inc., 1959.

[114] S. Kullback and R. Leibler. On information and sufficiency. *Ann. Math. Stat.*, 22:79–86, 1951.

[115] J. Lafferty, S. Della Pietra, and V. Della Pietra. Statistical learning algorithms based on bregman distances. In *Proceedings of the Canadian Workshop on Information Theory*, 1997.

[116] J. Langford, M. Seeger, and N. Megiddo. An improved predictive accuracy bound for averaging classifiers. In *Proceedings of the Eighth International Conference on Machine Learning*, 2001.

[117] S. Lauritzen. The em algorithm for graphical association models with missing data. *Comp. Stat. Data Anal.*, 19:191–201, 1995.

[118] S.L. Lauritzen. *Graphical Models.* Oxford Science Publications, 1996.

[119] R.D. Levine and M. Tribus, editors. *The Maximum Entropy Formalism.* MIT Press, 1979.

[120] D. MacKay. *Bayesian Methods for Adaptive Models.* PhD thesis, Caltech, 1991.

[121] J.R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics.* John Wiley & Sons, 1988.

[122] C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.

[123] M.J. Mataric. Sensory-motor primitives as a basis for imitation: Linking perception to action and biology to robotics. In C. Nehaniv and K. Dautenhahn, editors, *Imitation in Animals and Artifacts.* MIT Press, 1999.

[124] M.J. Mataric and M. Pomplun. Fixation behavior in observation and imitation of human movement. *Brain Res. Cogn. Brain Res.*, 7:191–202, 1998.

[125] D. McAllester. Pac-bayesian stochastic model selection. *Machine Learning Journal*, June 2001. To Appear.

[126] J. McClelland, A. Pentland, J. Weng, and I. Stockman, editors. *Workshop on Development and Learning*, Michigan State University, April 5-7th 2000. NSF and DARPA.

[127] P. McCullagh and J. Nelder. *Generalized Linear Models (The Monographs on Statistics and Applied Probability, Vol 37).* CRC Press, 1990.

[128] M. Meila and T. Jaakkola. Tractable bayesian learning of tree belief networks. In *Proceedings of the Sixteenth Annual Conference on Uncertainty in Artificial Intelligence.* Morgan Kaufmann, 2000.

[129] R. Meir. Empirical risk minimization versus maximum-likelihood estimation: A case study. *Neural Computation*, 7(1):144–157, 1995.

[130] A.N. Meltzoff and M.K. Moore. Imitation of facial and manual gestures by human neonates. *Science*, 198:74–78, 1977.

[131] A.N. Meltzoff and M.K. Moore. Newborn infants imitate adult facial gestures. *Child Dev.*, 54:702–709, 1983.

[132] A.N. Meltzoff and M.K. Moore. Infant's understanding of people and things: From body imitation to folk psychology. In J.L. Bermudez, A. Marcel, and N. Eilan, editors, *The Body and the Self*, pages 43–69. MIT Press, Cambridge, MA, 1995.

[133] A.N. Meltzoff and M.K. Moore. Explaining facial imitation: A theoretical model. *Early Development and Parenting*, 6:179–192, 1997.

[134] X.L. Meng and D.B. Rubin. Maximum likelihood estimation via the ecm algorithm: A general framework. *Biometrika*, 80(2), 1993.

[135] T. Minka. *A Family of Algorithms for Approximate Inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

[136] T.P. Minka. Inferring a gaussian distribution. www.media.mit.edu/∼tpminka/papers/minka-gaussian.ps.gz, 1998.

[137] B. Moghaddam, T. Jebara, and A. Pentland. Bayesian face recognition. *Pattern Recognition*, 33(11), 2000.

[138] B. Moghaddam and M-H. Yang. Gender classification with support vector machines. In *Proceedings of the 4th IEEE Int'll Conf. on Face and Gesture Recognition*, 2000.

[139] R. Morris and D. Hogg. Statistical models of object interaction. *International Journal of Computer Vision*, 37(2), 2000.

[140] K. Muller, A. Smola, A. Ratsch, B. Scholkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. In *Proc. of the Int. Conf. on Artificial Neural Networks*, 1997.

[141] Johnson N. and Hogg D. C. Learning the distribution of object trajectories for event recognition. *Image and Vision Computing*, 14(8), 1996.

[142] C. Nakajima, I. Norihiko, M. Pontil, and T. Poggio. Object recognition and detection by a combination of support vector machine and rotation invariant phase only correlation. In *Proceedings of International Conference on Pattern Recognition*, 2000.

[143] C. Nastar, B. Moghaddam, and A. Pentland. Generalized image matching: Statistical learning of physically-based deformations. In *Fourth European Conference on Computer Vision (ECCV'96)*, 1996.

[144] R.M. Neal and G.E. Hinton. A new view of the em algorithm that justifies incremental and other variants. *Submitted to Biometrika*, 1993.

[145] N. Oliver. *Towards Perceptual Intelligence: Statistical Modeling of Human Individual and Interactive Behaviors*. PhD thesis, Massachusetts Institute of Technology, 2000.

[146] E. Osuna, R. Freund, and F. Girosi. Improved training algorithm for support vector machines. In *Proc. IEEE Neural Networks in Signal Processing*, 1997.

[147] N. Oza and S. Russell. Online bagging and boosting. In *Eighth International Workshop on Artificial Intelligence and Statistics*, 2001.

[148] C. Papageorgiou and T. Poggio. Trainable pedestrian detection. In *Proceedings of International Conference on Image Processing*, 1999.

[149] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1997.

[150] J.E. Pecaric, F. Proschan, and Y.L. Tong. *Convex Functions, Partial Orderings, and Statistical Applications.* Academic Press, 1992.

[151] A. Pentland and A. Liu. Modeling and prediction of human behavior. In *IEEE Intelligent Vehicles 95*, 1995.

[152] A. Pfeffer. Sufficiency, separability and temporal probabilistic models. In *Uncertainty in Artificial Intelligence*, 2001.

[153] J. Piaget. *Play, dreams, and imitation in childhood.* Norton, New York, 1951.

[154] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schlkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning.* MIT Press, 1999.

[155] T. Poggio and F. Girosi. A sparse reprsentation for function approximation. *Neural Computation*, 10(6), 1998.

[156] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recogntion. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[157] V.S. Ramachandran. Mirror neurons and imitation learning as the driving force behind the "great leap forward" in human evolution. Essay, 2000.

[158] C. Rathinavelu and L. Deng. The trended hmm with discriminative training for phonetic classification. In *ICSLP*, 1996.

[159] C. Rathinavelu and L. Deng. Speech trajectory discrimination using the minimum classification error learning. In *IEEE Transactions on Speech and Audio Processing*, 1997.

[160] R. Reed, R. Marks, and R. Marks. *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks.* MIT Press, 1999.

[161] J. Rennie and R. Rifkin. Improving multiclass text classification with the support vector machine. Technical Report AIM-2001-026, Massachusetts Institute of Technology, 2001. AI Memo.

[162] B.D. Ripley. *From Statistics to Neural Networks*, chapter Flexible non-linear approaches to classification, pages 105–126. Springer, 1994.

[163] J Rissanen. Modelling by the shortest data description. *Automatica*, 14, 1978.

[164] G. Rizzolatti and M.A. Arbib. Language within our grasp. *Trends Neurosci.*, 21:188–94, 1998.

[165] G. Rizzolatti, L. Fadiga, B. Galles, and L. Fogassi. Premotor cortex and the recognition of motor actions. *Cognitive Brain Research*, 3, 1997.

[166] R. Rockafeller. *Convex Analysis.* Princeton University Press, 1972.

[167] Y. Rubinstein and T. Hastie. Discriminative vs informative learning. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, 1997.

[168] P. Rujan. Preceptron learning by playing billiards. *Neural Computation*, 9:99–122, 1997.

[169] S. Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3:233–242, 1999.

[170] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. In *Proc. 11th Annual Conference on Computational Learning Theory*, 1998.

[171] B. Schiele and J. Crowley. Probabilistic object recognition using multidimensional receptive field histograms. In *International Conference on Pattern Recognition*, 1996.

[172] B. Schiele and A. Waibel. Gaze tracking based on face color. In *International Workshop on Automatic Face and Gesture Recognition*, pages 344–349, 1995.

[173] B. Schoner. *Probabilistic Characterization and Synthesis of Complex Driven Systems*. PhD thesis, Massachusetts Institute of Technology, 2000.

[174] C.E. Shannon. A mathematical theory of communication. *Bell System Tech. J.*, 27, 1948.

[175] M. Shwe, B. Middleton, D. Heckerman, M. Henrion, E. Horvitz, H. Lehmann, and G. Cooper. Probabilistic diagnosis using a reformulation of the internist-1/qmr knowledge base: Part-i. *Methods of Information in Medicine*, 30:241–255, 1991.

[176] K. Sims. Evolving virtual creatures. In *Proceedings of SIGGRAPH '94*, volume 26, 1994.

[177] N. Slonim and N. Tishby. Agglomerative information bottleneck. In *Advances in Neural Information Processing Systems 12*, 2000.

[178] A. Smola and B. Scholkopf. A tutorial on support vector regression. Technical Report NC2-TR-1998-030, NeuroCOLT2 Technical Report Series, 1998.

[179] T. Starner and A. Pentland. Visual recognition of american sign language using hidden markov models. In *International Workshop on Automatic Face and Gesture Recognition*, 1995.

[180] T. Starner, B. Schiele, and A. Pentland. Visual contextual awareness in wearable computing. In *Intl. Symp. on Wearable Computers*, 1999.

[181] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[182] M. Szummer and T. Jaakkola. Kernel expansions with unlabeled examples. In *Advances in Neural Information Processing Systems 13*, 2000.

[183] J.B. Tenenbaum, V. De Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 2000.

[184] D. Terzopoulos, X. Tu, and Grzeszczukm R. Artificial fishes: Autonomous locomotion, perception, behavior, and learning in a simulated physical world. *Artificial Life*, 1(4):327–351, 1994.

[185] E.L. Thorndike. Animal intelligence. an experimental study of the associative process in animals. *Psychological Review, Monograph Supplements*, 2(4):109, 1898.

[186] S. Thrun. Probabilistic algorithms in robotics. Technical Report CMU-CS-00-126, Carnegie Mellon University, April 2000.

[187] S. Thrun and L.Y. Pratt. *Learning to Learn*. Kluwer Academic, 1997.

[188] M. Tipping. The relevance vector machine. In *Advances in Neural Information Processing Systems 12*, 1999.

[189] N. Tishby, W. Bialek, and F. Pereira. The information bottleneck method: Extracting relevant information from concurrent data. Technical report, NEC Research Institute, 1998.

[190] D. Todt and H. Hultsch. How songbirds deal with large amounts of serial information: retrieval rules suggest a hierarchical song memory. *Biological Cybernetics*, 79:487–500, 1998.

[191] M. Tomasello, A.C. Kruger, and H.H. Ranter. Cultural learning. *Behav. Brain Sci.*, 16:495–552, 1993.

[192] M. Tomasello, S. Savage-Rumbaugh, and A. Kruger. Imitative learning of actions on objects by children, chimpanzees, and enculturated chimpanzees. *Child Dev.*, 64:1688–1705, 1993.

[193] E. Uchibe, M. Asada, and K. Hosoda. State space construction for behaviour acquisition in multi agent environments with vision and action. In *Proceedings of the International Conference on Computer Vision*, 1998.

[194] N. Ueda and R. Nakano. Deterministic annealing variant of the em algorithm. In *Advances in Neural Information Processing Systems 7*, 1995.

[195] V. Vapnik. Principles of risk minimization for learning theory. In *Advances in Neural Information Processing Systems 4*, 1992.

[196] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[197] V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

[198] M. Warmuth, N. Cesa-Bianchi, and A. Krogh. Bounds on approximate steepest descent for likelihood maximization in exponential families. *IEEE Transaction on Information Theory*, 40(4):1215–1220, 1994.

[199] T. Watkin. Optimal learning with a neural network. *Europhysics Letters*, 21:871–876, 1993.

[200] J. Weizenbaum. Eliza - a computer program for the study of natural language communication between man and machine. *Communications of the Association for Computing Machinery*, 9, 1966.

[201] J. Weng, W.S. Hwang, Y. Zhang, and C.H. Evans. Developmental robots: Theory, method and experimental results. In *Proc. of the International Symposium on Humanoid Robots*, 1999.

[202] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for svms. In *Advances Neural Information Processing Systems 13*, 2000.

[203] C. Williams and C. Rasmussen. Gaussian processes for regression. In *Advances in Neural Information Processing Systems 8*, 1996.

[204] A. Wilson and A. Bobick. Recognition and interpretation of parametric gesture. In *International Conference on Computer Vision*, 1998.

[205] A. Witkin and M. Kass. Spacetime constraints. In *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 24, 1988.

[206] W. Wolberg and O. Mangasarian. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. In *Proceedings of the National Academy of Sciences*, volume 87, U.S.A., 1990.

[207] P. Woodland and D. Povey. Large scale discriminative training for speech recognition. In *Proceedings of the ASR 2000*, Paris, September 2000.

[208] C.-H. Yeang, S. Ramaswamy, P. Tamayo, S. Mukherjee, R. Rifkin, M. Angelo, M. Reich, E. Lander, J. Mesirov, and T. Golub. Molecular classification of multiple tumor types. *Intelligent Systems in Molecular Biology*, 1(1):1–7, 2001. Bioinformatics Discovery Note.

[209] J. Yedidia, W.T. Freeman, and Y. Weiss. Generalized belief propagation. In *Advances in Neural Information Processing Systems 13*, 2000.

[210] S. Yoon, R. Burke, B. Blumberg, and G. Schneider. Interactive training for synthetic characters. In *Proceedings of the AAAI*, 2000.

[211] J. Zhang. Selecting typical instances in instance-based learning. In *Proceedings of the Ninth International Machine Learning Conference*, 1992.