

**Incorporating Discourse and Syntactic Dependencies into Probabilistic
Models for Summarization of Multiparty Speech**

Michel Galley

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2007

©2007

Michel Galley

All Rights Reserved

ABSTRACT

Incorporating Discourse and Syntactic Dependencies into Probabilistic Models for Summarization of Multiparty Speech

Michel Galley

Automatic speech summarization can reduce the overhead of navigating through long streams of speech data by generating concise and readable textual summaries that capture the “aboutness” of speech recordings, such as meetings, talks, and lectures. In this thesis, we address the problem of summarizing meeting recordings, a task that faces many challenges not found with written texts and prepared speech. Informal style, speech errors, presence of many speakers, and apparent lack of coherent organization mean that the typical approaches used for text summarization are generally difficult to apply to speech. On the other hand, meetings provide a rich source of structural and pragmatic information that presents many research opportunities.

In this work, we illustrate how techniques that exploit automatically derived pragmatic and syntactic structures can help create summaries that are more on-target and readable than those produced by current state-of-the-art approaches. We present a conditional Markov random field model for summary sentence selection that exploits long-distance dependencies between pragmatically related utterances, such as between a question and its answer, or an offer and its acceptance. These dependencies are automatically inferred by a probabilistic model of interpersonal interaction, and are instrumental in achieving accuracies superior to competitive approaches. We also investigate the problem of rendering errorful and disfluent meeting utterances into sentences that are readable, pertinent, and concise. We present a trainable syntax-directed sentence compression system that automatically removes information-poor phrases and clauses from the syntax tree of each input sentence. A novel aspect of this sentence compression approach lies in its use of Markov synchronous context-free grammars, which provide a probabilistic decomposition of deletion rules into lexico-syntactic sub-structures more amenable to robust estimation. This sentence compression technique

was effectively applied to both speech and written texts, and produces sentences that were judged more grammatical than those generated by previous work.

Contents

1	Introduction	1
1.1	Approach Overview and Thesis Organization	3
1.2	Thesis Contributions	5
2	Corpora	8
2.1	The ICSI Meeting Corpus	8
2.1.1	Transcription and Meta-Data	9
2.1.2	Audio Channels and Acoustic Features	10
2.1.3	Dialog Act and Dialogic Pair Annotation	11
2.1.4	Agreement and Disagreement Annotation	12
2.1.5	Content Selection Annotation	12
2.1.6	Sentence Compression and Revision Annotation	13
2.2	NIST, CMU, and AMI Meeting Corpora	19
2.3	The Switchboard Corpus	20
2.4	The Ziff-Davis Corpus	20
3	Utterance Selection	22
3.1	Introduction	22
3.1.1	Notations and Preliminary Remarks	24
3.2	Dialogic Pairs	25
3.2.1	Linguistic Background	25
3.2.2	Automatic Identification	26
3.2.3	Features	28

3.2.4	Results	30
3.3	Conditional Random Fields	31
3.3.1	Parameter Estimation and Probabilistic Inference	33
3.4	Applications	34
3.4.1	Generic Utterance Selection	34
3.4.1.1	Motivation	35
3.4.1.2	Probabilistic Ranking	36
3.4.1.3	Features	38
3.4.1.4	Evaluation	41
3.4.1.5	Results	44
3.4.2	Agreement and Disagreement Classification for Summarization	50
3.4.2.1	Problem Statement	51
3.4.2.2	Features	51
3.4.2.3	Empirical Study	51
3.4.2.4	Classifier	54
3.4.2.5	Results	55
3.5	Conclusion	56
4	Sentence Compression	57
4.1	Preprocessing	62
4.1.1	Syntactic Parsing of Conversational Speech	63
4.2	Synchronous Grammars for Sentence Compression	65
4.2.1	Extracting Synchronous Grammars	66
4.2.2	A Generative Model	68
4.2.2.1	Lexicalized Head-Driven Markov Model	69
4.2.2.2	Conditioning Variables and Smoothing	73
4.2.3	Discriminative Framework	75
4.2.3.1	Features	76
4.2.4	Decoder	77

4.3	Evaluation Metrics	78
4.3.1	Word Error Rate	78
4.3.2	BLEU	79
4.4	Results	79
4.4.1	Written Texts	80
4.4.2	Conversational Speech	87
4.5	Related Work	91
4.5.1	Sentence Compression	91
4.5.2	Machine Translation	94
4.6	Conclusion	95
5	Integration and Error Analysis	96
5.1	Automatic Speech Recognition	96
5.2	Removing Edited Words from Disfluent Utterances	100
5.2.1	Distribution of Repairs	102
5.2.2	Baseline Approach	102
5.2.3	Improvements over Baseline	105
5.2.4	Results	107
5.3	Content Selection	108
5.3.1	Automatic Utterance Segmentation	109
5.3.2	Results	109
5.4	Utterance Compression	111
5.4.1	Evaluation	112
5.4.2	Results	114
6	Final Discussion	116
6.1	Contributions	116
6.1.1	Utterance Selection	116
6.1.2	Utterance Compression	117

6.1.3	Integration	118
6.2	Deliverables	118
6.3	Limitations and Future Work	119
A	Penn Treebank Tags	123
B	Annotation Guidelines for Sentence Compression	126
B.1	Overview	126
B.2	Summarization in Stages	129
C	Sample Outputs	137
C.1	Ziff-Davis	137
C.2	ICSI Meeting Corpus	142
	Bibliography	164

List of Figures

2.1	Sentence compression annotation interface.	17
3.1	A skip-chain CRF with edges between utterances of the same dialogic pair.	32
3.2	Summarization content units of the ICSI meeting corpus.	42
3.3	ROC curve comparing skip-chain CRF with linear-chain CRF.	48
3.4	Correlation plot between correct and approximate posteriors.	49
4.1	Full sentence and a summary sentence containing a minor rewording.	67
4.2	Syntax tree of the disfluent Switchboard sentence.	70
4.3	Motivating example for bi-lexical dependencies in sentence compression.	74
5.1	Illustration of the minimum edit distance dynamic programming algorithm to find the minimum-WER sentence in a lattice.	98
5.2	Lattice minimum word error as a function of the threshold on lattice density.	99
5.3	Structure of a self-repair.	101
5.4	A self-repair with an approximate copy.	106
5.5	ROC curve comparing different levels of automation in utterance selection.	110
5.6	Motivating example for compression word error rate (CWER).	112
B.1	The three panels of the revision interface.	127
B.2	The annotation pane.	128
B.3	Reparanda, interruption points, and repairs.	131
B.4	Self-repairs.	131

B.5	Filler words.	132
B.6	Hedges.	132
B.7	Summarization stages.	133
B.8	Summarization stages.	134
B.9	Insertions and substitutions.	135
B.10	Improve style and fluency in the revision annotation interface.	135
B.11	Removal of self-repairs, filler, and hedge words.	136

List of Tables

2.1	Excerpt of an anonymized transcription of the ICSI Meeting corpus. Dashes are used to represent various forms of disfluencies: (1) a dash at the end of a turn represents an abandoned utterance; (2) a dash appended to the end of a token represents a word fragment (e.g., “1-”); (3) a dash in the middle of a turn represents a short pause (as in, e.g., “But - General”) which often precedes a self-repair (Shriberg, 1994). . .	10
2.2	Statistics on the content selection annotation.	13
2.3	Sample abstractive summary of the ICSI meeting corpus.	14
2.4	Annotation stages in sentence compression.	16
2.5	Inter-labeler agreement in sentence compression on the development set of the ICSI meeting corpus.	18
2.6	Inter-labeler agreement in sentence compression on the test set of the ICSI meeting corpus.	18
3.1	Features for dialogic pair identification (ranking features).	29
3.2	Features for dialogic pair identification (classification features).	30
3.3	Results for dialogic pair identification.	30
3.4	Contingency table displaying ranking and classification predictions for dialogic act identification.	31
3.5	Snippet of a meeting displaying three dialogic pairs.	35
3.6	Contingency tables illustrating the statistical correlation arising from dialogic pairs.	36
3.7	Features for extractive summarization.	39

3.8	Some content selection features found helpful on the ICSI corpus.	45
3.9	Results for content selection by feature set.	46
3.10	F_1 scores for content selection on the test set of the ICSI meeting corpus with different graphical model structures.	47
3.11	F_1 scores for content selection on the test set of the ICSI meeting corpus with different ranking approaches.	48
3.12	Features for agreement and disagreement classification.	52
3.13	Empirical distributions for some agreement and disagreement dependencies (previous tag, consistency, and symmetrical). The empirical prior for the agreement tag is $p(\text{AGREE}) = .188$ and the disagreement tag is $p(\text{DISAGREE}) = .106$	53
3.14	Agreement/disagreement classification accuracies with 3 classes.	55
3.15	Agreement/disagreement classification accuracies with 4 classes.	56
4.1	Excerpt of an extractive summary of a meeting.	58
4.2	Parsing results on the Switchboard corpus.	64
4.3	Cumulative number of training sentence pairs and number of SCFG productions acquired from Ziff-Davis.	66
4.4	Word error rates on the Ziff-Davis development set for different Markov structures.	81
4.5	Word error rates on the Ziff-Davis development set for different head-modifier annotations.	82
4.6	Coverage analysis for the lexicalized Markov SCFG model.	83
4.7	Classification accuracies, precision, recall, and F_1 scores for the most frequently deleted categories.	84
4.8	Human evaluation on the Ziff-Davis test set.	85
4.9	Sentence compressions for the test set of the Ziff-Davis corpus.	86
4.10	Word error rates on the ICSI development set for different Markov structures.	87
4.11	Word error rates on the ICSI development set with different head-modifier annotations.	88
4.12	Results for discriminative training.	88

4.13	Sentence compressions for the test set of the ICSI meeting corpus (first 8 compressions).	90
4.14	Sentence compressions for the test set of the ICSI meeting corpus (sample errors).	91
5.1	Decoder and lattice word error rates on sentence compression test and development sets.	100
5.2	Distribution of repairs in Switchboard.	102
5.3	Word edit distance between interregnum and repair.	103
5.4	POS edit distance between interregnum and repair.	103
5.5	Features for disfluency removal.	104
5.6	Results for edited word detection on the development of the Switchboard corpus.	105
5.7	Results for edited word detection on the test set of the Switchboard corpus.	107
5.8	The four experimental setups for the evaluation of content selection.	108
5.9	F ₁ scores for content selection on the test set of the ICSI meeting corpus for varying degrees of automation.	110
5.10	The four experimental setups for the evaluation of sentence compression.	111
5.11	Compression word error rates (CWER) on the ICSI development set.	114
5.12	Compression word error rates of the generative model on the ICSI test set.	115
A.1	Clause-level tags of the Penn Treebank.	124
A.2	Phrase-level tags of the Penn Treebank.	124
A.3	Word-level tags of the Penn Treebank.	125
C.1	Sentence compressions for the Ziff-Davis test set.	137
C.2	Sentence compressions for the test set of the ICSI meeting corpus.	142

Acknowledgments

I am eternally grateful to my advisor, Kathy McKeown, for her thoughtful guidance and constant encouragement. She gave me ample freedom in pursuing my ideas, while helping me situate my work in a broader context. She shaped the way I write and do research at a very deep level. I would also like to thank the other members of my thesis committee—Michael Collins, Daniel Ellis, Julia Hirschberg, and Owen Rambow—for the remarkable breadth and depth of their feedback, and for many thought-provoking questions. I am particularly indebted to Julia Hirschberg for her sound advice and constructive criticism, and to Michael Collins for discussions that have greatly improved this work (especially in the machine learning aspects). It was also a great pleasure to have Owen Rambow and Daniel Ellis in my committee; this work benefited substantially from their expertise in computational models of syntax and in speech recognition.

I would like to express my appreciation and gratitude to all my friends and colleagues at Columbia for their help, ideas, and good work: Regina Barzilay, Hrvoje Benko, Fadi Biadsy, Sasha Blair-Goldensohn, Gabor Blasko, Pablo Duboue, Noémie Elhadad, David Elson, Frank Enos, Elena Filatova, Agustín Gravano, Luis Gravano, Nizar Habash, Gabriel Illouz, Martin Jansche, Min-Yen Kan, Judith Klavans, Alexander Koller, Jackson Liscombe, Sameer Maskey, Smaranda Muresan, Shmuel Popper, Dragomir Radev, Andrew Rosenberg, Barry Schiffman, Advait Siddharthan, Kapil Thadani, and Stephen Wan. Special thanks to my officemates, for both providing a sounding board for ideas and for many coffee and tea breaks: Alpa Jain, Amélie Marian, Ani Nenkova, Carl Sable, David Evans, and Panagiotis Ipeirotis.

Interactions with people outside Columbia have also influenced my research. I am particularly indebted to Kevin Knight and Daniel Marcu, who were not only thoughtful, insightful, and fun

mentors, but who also had great influence on the way I think as a scientist. My special thanks to Alexandros Potamianos and Eric Fosler-Lussier, my two first research supervisors, for their support and patience, and for inspiring me to grow as a researcher. I have also greatly enjoyed conversations with Hongyan Jing, Jeff Kuo, Imed Zitouni, Joseph Olive, Chin-Hui Lee at Bell Labs; with Deepak Ravichandran, Mark Hopkins, Steve DeNeeffe, Philipp Koehn, Franz Och, Jonathan Graehl, Alexander Fraser, Chin-Yew Lin, Marcello Federico, David Kauchak, Michael Fleischman, Lara Taylor, Liang Zhou, Ignacio Thayer, Radu Soricut, Ulf Hermjakob, Dragos Munteanu, and Madhur Ambastha at ISI; with Wei Wang and Greg Langmead at Language Weaver.

To my friends who helped me remain sane during my graduate school years (listed here in the order we first met): Laurent, Pascal, Alexandre, Larry, Connie, Andrew, Aurélie, Latifa, George, Min, Pablo, Smara, Noémie, Dave Evans, Daniel, Dimitris, Vassilis, Alkmini, Nadja, Deepak, Haruka, Dave Kauchak, and Liang. Thanks in particular to my friends Pablo and Smara, whose continued presence and support helped me find the strength to stay the course during difficult times. I am infinitely indebted to both of them.

Finally, this work would not have been possible without the love, support, and encouragement of my family. I dedicate this dissertation to them. To my parents Claire and Gérard for their constant support and understanding. To my wife Tomoko for keeping me happy. To Benno for fond and lasting memories—I miss you.

To my parents and my wife.

To Benno, in memoriam.

Chapter 1

Introduction

Text summarization is increasingly becoming a mature field, with well established methods for aggregating information sources such as news articles. Summarization enables users to reduce reading time and to digest large amounts of textual information. Information sources are often long and repetitive, and the production of generic summaries that synthesize relevant and important information offers many practical benefits, and it was shown to significantly increase effectiveness of real users in various fact-gathering tasks (McKeown et al., 2005).

While previous summarization techniques were mainly applied to newswire texts, this thesis addresses the problem of summarizing meeting recordings. We believe meetings are a particularly promising application area, since such social gatherings are widespread in all spheres of public life, e.g., in academia, businesses, and social organizations. Meetings are critical opportunities for group members to share information, learn from others, solve problems, and make decisions that move a group forward. However, as meetings generally require significant investments of time, it is important to plan, organize, and conduct meetings as efficiently as possible. In such a setting, any effective means to access and navigate previous meetings could constitute great time savings, and be instrumental in making future meetings more productive.

We envision many practical usages that can be drawn from the ability to identify relevant and important information in meeting transcripts. One possibility is to use a meeting browser that seamlessly integrates (e.g., with hyperlinks between sentences) short summaries, summaries of passages

corresponding to particular topics, and full transcripts. Summaries, or *minutes* of meetings aim to answer general questions such as: What was the purpose of the meeting? What matters were discussed? What problems were solved during the meeting? What is to be accomplished?

The use of document summarization techniques is particularly desirable when faced with speech documents such as meeting transcriptions, since these documents are particularly difficult to navigate. As opposed to their written counterparts, spoken documents lack structural elements like title, chapters, sections, and paragraphs that can ease the task of the reader. In most cases, transcripts are just long streams of words, and segmentation into turns often does not correspond to well-formed sentences. For meetings of one or more hours, the navigation of meeting transcripts of this nature can be really burdensome. Furthermore, because meetings are often informal and conversational, it is generally difficult to predict important information based on its relative position in a document, and simple document skimming strategies, such as reading a handful of lead sentences, rarely provides a good understanding of the document (as opposed to, e.g., newswire texts, in which the first paragraph often covers the main point of a story).

Speech summarization presents many research opportunities not found in the text summarization field. Additional information can be extracted from the speech signal, such as pitch, energy, and pausal duration. Meetings also present a rich source of structural and pragmatic information unique to multi-party speech; a main focus in our work examines how structural information derived from the verbal exchange between speakers can be exploited to improve a trainable summarizer.

While speech summarization presents new research opportunities, it also presents many difficulties. Speech summarization in general, and meeting summarization in particular faces various issues not found in written texts, and these make the automatic summarization task extremely challenging. Several factors make it difficult to produce coherent and fluent summaries: informal style, presence of many speakers, absence of punctuation and other recognizable elements of document structure, and often apparent lack of coherent organization. Furthermore, meetings are notoriously hard to recognize automatically (the word error rate on the meeting corpus (Janin et al., 2003) studied in this thesis is 37.7%). In many cases, this means that approaches used for text summarization are problematic with conversational speech. The most widely used approach to text summarization—sentence extraction—may be a relatively poor choice when applied to noisy input. Indeed, disfluencies, ungrammatical constructions, and other speech errors are frequent in spontaneous speech,

and make it difficult to directly re-use transcription utterances to create summaries. The task of automatically rendering disfluent and errorful utterances into utterances that are readable and concise is thus of great practical interest, and a significant portion of this thesis is dedicated to this problem. This problem is often termed *sentence compression*.

We now give an overall outline of the thesis and of our approach to meeting summarization, and then overview our main contributions.

1.1 Approach Overview and Thesis Organization

For our purposes, it is convenient to divide the meeting summarization problem into two subtasks that are largely independent: *content selection*, i.e., identifying a set of salient utterances that is a practical substitute for the entire meeting transcription; *sentence compression*, i.e., correcting the various nonfluencies typical of conversational speech, and operating below the sentence level to further remove unimportant lexical material. These two tasks are respectively addressed in Chapter 3 and Chapter 4. Prior to them, we describe the ICSI meeting corpus in Chapter 2, on which most of our experiments on content selection and sentence compression were applied. In the same chapter, we also describe related speech and text corpora that were used to train auxiliary resources—such as a syntactic parser for conversational speech—and to compare our approaches to other genres (such as written texts).

In Chapter 3, we address the problem of selecting relevant and important utterances to build a summary. Since an objective in summarization is to build summaries that are coherent and cohesive, it is important to account for certain pragmatic constraints in the process of building summaries, and avoid, for instance, building a summary that contains a question without its answer. Thus, a special interest in this chapter is to provide computational means of identifying long-distance dependencies between pragmatically related utterances, such as question-answer and offer-accept pairs, which are sometimes termed *dialogic pairs*. In conversations with many participants, utterances of a dialogic pair are often not adjacent, because of intervening utterances produced by other speakers, such as backchannels and unrelated threads of conversation. Chapter 3 presents an approach to automatically identifying such dialogic pairs, and describes their use in two content selection classification tasks. In both cases, our approach to content selection relies on probabilistic graphical models, in

which dialogic pairs provide means to establish long-distance probabilistic dependencies between utterances. In the first task, we treat content selection as a binary classification problem, where each utterance is classified either as summary or non-summary. In the second application, we go beyond this generic content selection scheme and address the problem of identifying agreements and disagreements, which in turn could be used to produce summaries of the main points of the meeting that were agreed or disagreed.

In Chapter 4, we investigate the problem of rendering errorful and disfluent meeting utterances into utterances that are readable and concise. Our goal is to remove disfluencies, content-poor phrases such as “you know” and “I mean”, and also to discard longer phrases and clauses that are both unimportant and syntactically optional. We take an approach to utterance compression that capitalizes primarily on syntactic structure, and more specifically synchronous context-free grammars extracted from parallel texts of document-abstract pairs. Our approach to compression is divided into three steps: (1) we identify and correct self-repairs in the input utterance; (2) we parse the remaining words; (3) we apply syntactic compression rules automatically derived from parallel texts to reduce the sentence. To generate better compressions, we also explore other models and information sources, such as n -gram language models and tf.idf scores. This chapter also presents results on compressing written texts, in order to better compare our approach to previous work (which was almost exclusively applied to texts).

In order to enable better error analyses, we decided in Chapters 3 and 4 to fully automate only those pre-processing steps that pertain to our summarization work (e.g., dialogic pair identification), and steps for which no human annotation is available (e.g., syntactic parsing), and assumed accurate inputs in two other cases (i.e., we used reference transcriptions instead of ASR output, and reference disfluency annotation instead of relying on automatic disfluency removal). However, Chapter 5 evaluates a fully automated and integrated system for content selection and sentence compression. We assess the overall scalability of our selection and compression models by applying them to automatic speech recognition (ASR) outputs, and ensure that all preprocessing steps (e.g., automatic disfluency removal, segment boundary detection, and induction of all features) are fully automatic. In the same chapter, we also analyze performance on mildly errorful transcriptions, with word error rates lower than those of current state-of-the-art ASR systems. This enables us to determine to what extent our sentence compression would scale assuming significant future advances in speech

recognition. Finally, Chapter 5 also describes our experiments on automatically identifying edited words within self-repairs, such as “why didn’t he?” in the sentence “why didn’t he? – why didn’t she stay home?”. Since our method for removing edited words using shallow features achieves state-of-the-art performance without any reliance on syntactic information, we always attempt to remove edited words as a pre-processing step to our syntax-based model for sentence compression.

This thesis contains three appendices: in Appendix A, we include a list of syntactic labels used to represent the syntactic deletion rules of Chapter 4, along with definitions and examples. We believe this appendix may be helpful to some readers not familiar with the Penn Treebank annotation (Marcus et al., 1994). In Appendix B, we provide the full annotation guidelines provided to annotators who created reference compressions of meeting utterances. In Appendix C, we provide sample summaries generated by the utterance compression component.

1.2 Thesis Contributions

In this thesis, we make advances in meeting summarization by exploiting structural properties specific to multi-party speech, and by extending text summarization techniques to make them generally applicable to a difficult genre of multi-party informal speech. Overall, this thesis illustrates how techniques that exploit automatically derived pragmatic and syntactic structures can help create summaries that are more on-target, coherent, and readable.

- **Utterance selection:** We present a conditional random field (CRF) model for content selection that explicitly represents long-distance dependencies between utterances of a meeting, in order to exploit the strong statistical correlation between dialogic pairs (e.g., question-answer). To identify these pairs, we present a discriminative model that identifies them with 85.3% accuracy. We show that these pairs are in turn helpful to the content selection task, and empirical evaluations show that CRF models with long-distance edges representing dialogic pairs clearly outperform linear-chain CRF models exploiting only local context. As an intermediate step in our approach, we evaluate different solutions to the class imbalance problem arising in this summarization task. For this, we evaluate different probabilistic methods to extract the k most probable utterances, such as maximum a posteriori (MAP) and sum-product inference, and show that the latter approach significantly outperforms the former one. In the

generic content selection task, our best model reaches 92.9% of human performance, and outperforms our best-performing linear-chain CRF model (1.4% absolute increase in F_1 -score). In the agreement/disagreement classification task, long-distance pragmatic dependencies encoded into discriminative models provide a 1.3% absolute increase in classification accuracy over our best non-relational model.

- **Utterance compression:** To alleviate data sparseness issues pertaining to many previous syntax models for sentence compression, we introduce Markov synchronous context-free grammars (SCFG) for sentence compression. These grammars offer a probabilistic decomposition of deletion rules into lexical-syntactic sub-structures more amenable to robust estimation. They also enable the incorporation of a flexible amount of lexical dependencies and tree internal annotation, which are critical to accurately determine whether a given syntactic phrase is optional (as “like any other product” in the sentence “they sold it like any other product”) or mandatory (as in “it looks like any other product”). We also present a novel approach for extracting synchronous context-free grammar (SCFG) compression rules from parallel corpora of documents and summaries that have been parsed. The approach exploits the minimum tree edit distance alignment between the syntax tree of the full sentence and the one of the summary sentence. This extraction approach significantly improves upon previous work, which only exploited 1.75% of a standard document-abstract corpus (Ziff-Davis). Since we are able to exploit 25% of the Ziff-Davis corpus, this tree-to-tree alignment method substantially benefits our lexicalized Markov SCFG models. Finally, we also present a discriminative training framework in which the parameters of different models—e.g., n -gram language models, tf.idf scores, sentence length, ASR confidence scores—are directly optimized for compression quality, either through maximum-BLEU training or minimum-WER training.
- **Correcting speech repairs:** We present a linear-chain CRF model for disfluency removal that achieves very competitive performance on a standard test set of the Switchboard corpus. Our classifier’s strength lies in its use of hundreds of features that are aimed at identifying the lexical repeats that are generally associated with self-repairs, i.e., it contains many predicative features that ask whether any pair of words within a window of 5-6 tokens are the same, have the same part-of-speech, or have the same suffix. While the performance of our classifier

(.794 *F*-score on edited-word retrieval, and an overall .979 accuracy) is slightly lower than the best performance on the same test set that is known to us (Johnson and Charniak, 2004), our method is much simpler to implement, and is computationally much more efficient.

- **Integration and evaluation:** We present a fully automated meeting summarization system for utterance selection and sentence compression, which is applicable to real automatic speech recognition output. In contrast to most other works in natural language processing applied to spoken documents, we provide error analyses on fully automatically recognized texts. We also carry out analyses on transcriptions with varying degrees of recognition errors, ranging from fully accurate transcriptions to real ASR outputs. We find that our approaches to summarization scale relatively well on ASR transcripts.

Chapter 2

Corpora

This chapter provides an overview of the various corpora used to train and test the summarization components described in this thesis. Our work in this thesis was primarily applied to the ICSI Meeting corpus (Janin et al., 2003), which is described in Section 2.1. We review other meeting corpora in Section 2.2, such as meetings from CMU (Waibel et al., 2001), NIST (NIST, 2004a), and AMI (Carletta et al., 2005). In Section 2.3, we present the Switchboard corpus (Godfrey et al., 1992), whose treebank and disfluency annotations are used to train and test a syntactic parser (Charniak, 2000) and a model to identify and correct repairs. In Section 2.4, we finally present the Ziff-Davis corpus, a corpus that has been extensively studied in previous work on sentence compression. Ziff-Davis comprises a large collection of technical articles and their corresponding abstractive summaries, and is used in this thesis to compare our approach to prior work.

2.1 The ICSI Meeting Corpus

The ICSI Meeting Corpus (Janin et al., 2003) is a corpus of 75 research meetings, representing about 72 hours of speech, that were recorded at the International Computer Science Institute (ICSI) at UC Berkeley between 2000 and 2002. These meetings are all “naturally-occurring”, in the sense that they would have taken place anyway, and their style is quite conversational and informal. The number of participants in these meetings ranged from 3 to 10, and averaged 6. The total number of unique speakers is 60, including a large number (26) of non-native English speakers.

These recordings consist primarily of regular meetings of three research groups at ICSI: (1) 15 *Even*

Deeper Understanding (EDU) meetings, concerned with natural language processing and neural theories of language; (2) 29 *Meeting Recorder* (MR) meetings, concerned with the ICSI corpus itself; (3) 23 *Robustness* (RO) meetings concerned with environmental issues in speech recognition. While a few speakers regularly appear in both MR and RO meetings, participants of EDU meetings were not involved in any other meetings of the corpus.¹

2.1.1 Transcription and Meta-Data

Human word-level orthographic transcriptions are available for all meetings of the ICSI corpus. All meetings were transcribed using a professional transcription service, and a senior transcriptionist later verified the entire transcription.

Transcriptions incorporate punctuation, word fragments (e.g., “l-”), restarts (“-”), and pauses (“[pause]”). The original transcription also contains contextual comments about various speech and non-speech events (laughs, coughs, door slams, lip smack, etc.), but for the purpose of our experiments, all comments except laughs (“[laugh]”) were ignored. This is motivated by the fact that speech recognition outputs generally do not contain most of these events. Table 2.1 provides an excerpt of a meeting.

All meetings were automatically recognized using the SRI-ICSI (Mirghafori et al., 2004) speech recognition system, a recognizer that was primarily trained on conversational telephone speech (CTS). We overview this recognizer in Section 5.1, before presenting results for content selection and sentence compression for different levels of transcription quality. In the case of sentence compression, we evaluate four different levels: (1) the transcript with punctuation and mixed case; (2) ASR-like transcripts, which are all lowercase, without punctuation, and without word fragments (3) ASR transcripts produced by the SRI-ICSI recognizer; (4) ASR oracle transcripts that are slightly less errorful than real ASR outputs.

A peculiar characteristic of the ICSI corpus is the presence of a “digits” section at the beginning or the end of almost each meeting. Participants were asked to read digits, in order to collect data for far-field signal speech recognition. Since these sections were not removed prior to summarization, the utterance selection component (Chapter 3) incorporates “digits” features to automatically identify and remove them (for instance, one feature counts the number of digits in each utterance).

¹Except in one case where a regular participant of MR and RO meetings attended the beginning of an EDU meeting.

Speaker	Transcription
me013	So that's about eight per minute [laugh] .
fe008	But a thousand events in twelve minutes - , that's -
mn005	Yeah , [pause] but -
mn005	Yeah .
fe016	But that can include taps .
me013	Uh . Yeah .
mn005	But - General .
fe008	Well , but a thousand taps in eight minutes is a l- in twelve minutes is a lot .

Table 2.1: Excerpt of an anonymized transcription of the ICSI Meeting corpus. Dashes are used to represent various forms of disfluencies: (1) a dash at the end of a turn represents an abandoned utterance; (2) a dash appended to the end of a token represents a word fragment (e.g., “l-”); (3) a dash in the middle of a turn represents a short pause (as in, e.g., “But - General”) which often precedes a self-repair (Shriberg, 1994).

We believe it to be appropriate to include such corpus-specific features, since they are only aimed at removing utterances that would normally not occur in a meeting. This is the only case of corpus-dependent processing in this thesis.

2.1.2 Audio Channels and Acoustic Features

All meetings took place in a conference room at ICSI. Meetings were recorded using simultaneously up to 10 head-worn microphones and four omni-directional PZM table microphones (Janin et al., 2003), and the waveform of each microphone was stored in a separate file (down-sampled to 16 kHz and reduced to 16 bits). Audio files range in length between 17 and 103 minutes.

In our experiments on content selection, feature extraction for a given speaker always relied on the recording of his or her head-worn microphone. In the rare cases when it was not available, we relied on the PZM microphone that is the closest to the center of the table (*D3* according to the seating chart provided with the ICSI corpus).

Participants sometimes exercised their rights to have portions of the audio recording deleted.

This happened, for instance, in segments containing utterances of people’s names. While speaker identifiers were anonymized, the occurrences of people’s names in the transcription and audio were only deleted at the participants’ explicit request. To satisfy such requests, the unwanted portions of the audio were “bleeped”, and corresponding segments of the transcriptions were deleted.

In this thesis, acoustic features were extracted using Praat (Boersma and Weenink, 2006), a program for speech analysis with scripting functionalities. For each segment, we extracted features such as minimum, maximum, and average F0 and energy (more details are available in Chapter 3). F0 contours were stylized using Praat’s built-in stylizer, and all features values were normalized by speaker. When Praat failed to produce a feature value, or when processing bleeps, missing features were set to the mean value of the given feature.

2.1.3 Dialog Act and Dialogic Pair Annotation

We used the ICSI Meeting Recorder Dialog act (MRDA) corpus (Shriberg et al., 2004) for our experiments on content selection. Dialog act (DA) labels describe the pragmatic function of utterances, and an utterance may be classified as a STATEMENT, as an OFFER, as a ACCEPT, etc. This auxiliary corpus consists of over 180,000 human-annotated dialog act labels, and inter-labeler agreement was considered substantial (Cohen’s κ statistic is .8). This corpus also contains the annotation of adjacency pairs, sometimes also termed *dialogic pairs* (DP). Dialogic pairs provide information about speaker interaction, and reflect the structure of conversations as paired utterances (e.g., APOLOGY-DOWNPLAY and OFFER-ACCEPT). In this corpus, each dialogic pair is represented as a pair of utterance indices. The labeling of dialogic pairs is used in our work to automatically determine the addressee of a given utterance (the DP identifier described in Subsection 3.2), and is instrumental in automatically determining the structure of probabilistic graphical models, as explained in Chapter 3. The dialog act and dialogic pair annotation for the ICSI Meeting corpus is described in more detail in (Shriberg et al., 2004; Dhillon et al., 2004).

Note that, while we sometimes refer to dialogic pairs using pairs dialog act labels (such as OFFER-ACCEPT), we actually do not attempt to predict these DA categories. As we will see in Section 3.2.2, we treat the problem of automatically identifying dialogic pairs as follows: given the index of utterance B , find the index of the preceding utterance A , so that (A,B) constitutes a dialogic pairs. To establish that the speaker of utterance B addresses the speaker of A , we do not

need to know that, for example, *A* is an offer and *B* is an accept (while such information would be useful, automatically identifying such tags is generally considered a very difficult problem—the best performance on the ICSI corpus that is known to us is 66% classification accuracy (Ji and Bilmes, 2005)).

2.1.4 Agreement and Disagreement Annotation

The work on agreement and disagreement identification described in Chapter 3 is meant to illustrate the use of our model of interpersonal interaction in other classification tasks related to summarization. Following (Hillard et al., 2003), our work relies on the annotation of the utterances of seven meetings with one of the four categories: AGREE, DISAGREE, BACKCHANNEL, and OTHER.² Backchannels (e.g., “uh-huh” and “okay”) were treated as a separate category, since they are generally used by listeners to indicate they are following along, while not necessarily indicating agreement.

As it was done (Hillard et al., 2003), these meetings were segmented into “spurts”, defined as periods of speech that have no pauses greater than 0.5 second, and each spurt was labeled with one of the four categories. Hillard et al.’s motivation for using spurt segmentation as a unit of analysis lies in the fact that it is automatic, and as opposed to other types of segmentation, it is particularly easy to produce automatically. The annotation collected by (Hillard et al., 2003) consists of 1795 hand-labeled spurts, to which 8094 automatically labeled spurts were added sometimes added to the training data.³ The proportion of class labels in the hand-labeled data is the following: 9.42% are agreements, 6.24% are disagreements, 22.8% are backchannels, and 61.6% are others. According to (Hillard et al., 2003), inter-labeler reliability estimated on 500 spurts with two labelers was considered fair, since the kappa coefficient (Cohen, 1960) was 0.63.

2.1.5 Content Selection Annotation

To train and evaluate the content selection system of Chapter 3, we used a corpus of extractive summaries produced at the University of Edinburgh (Murray et al., 2005b). Statistics about this corpus are displayed in Table 2.2. For each of the 75 meetings, human judges were asked to select

²Part of these annotated meetings were provided by the authors of (Hillard et al., 2003).

³Hillard used an unsupervised clustering strategy to generate these labels (Hillard et al., 2003).

	Train	Test
Meetings	69	6
Ref./meeting	1	3-5
Dialog act units	103,534	7,966
Length (% of words)	18.6%	19.1%
Length (% of DA units)	9.54%	9.70%

Table 2.2: Statistics on the content selection annotation: number of meetings per set, number of reference summaries per meeting, number of dialog act units, and average lengths expressed as percentages of words and DA units. Note that, if we take into account multiple references, the test set comprises a total of 28,519 DA units.

transcription utterances segmented by DA in order to build reference extractive summaries. This resulted in summaries whose length averaged 18.8% of the original documents in number of words, and 9.57% in number of DA units. No strict limit on length was imposed on the annotators. Inter-labeler agreement was measured using six meetings that were summarized by multiple coders, and resulted in a kappa score of .323. While this level of agreement is quite low, this situation is not uncommon to summarization, since there may be many good summaries for a given document; a main challenge lies in using evaluation schemes that properly account for this diversity.

The Edinburgh summarization corpus also contains 61 abstractive summaries, which are made of four sections: “abstract”, “decisions”, “problems”, and “progress and achievements”. It would potentially be interesting to use these abstracts in future research in abstractive summarization. However, these abstracts were not used in our work due to their relatively high abstraction level (see Table 2.3). Indeed, these summaries seldom re-use phrases of the original transcriptions, and it would be quite unreasonable to envision the automatic production of such summaries from noisy conversational transcripts.

2.1.6 Sentence Compression and Revision Annotation

In order to train a sentence compression system able to process meeting transcripts, we supervised the annotation of 2409 sentences drawn from 14 meetings of the ICSI corpus. Three native speakers of American English—a freelance magazine editor, a graduate student in physics, and an

ABSTRACT

The Berkeley Meeting Recorder group discussed research aims and corresponding concerns for future data collection. It was agreed that a substantial amount of meeting data is required from different domains, and comprising several speakers, to perform the types of discourse and acoustic analyses desired. Ongoing efforts by speaker mn005 to automatically detect regions of speaker overlap were considered. It was suggested that speaker mn005 focus on a small set of acoustic parameters, e.g. energy and harmonics-related features, to distinguish regions of overlap from those containing the speech of just one speaker. Disk space issues were discussed. And, finally, the problem of speaker anonymization was explored.

DECISIONS

Recordings must be of existing meetings that are conducted in English. Participants should ideally consist of professors and doctoral students, but no undergraduate students, who are willing to record their meetings at ICSI. The Meeting Recorder corpus should comprise data from a large number of speakers representing different domains. Attempts should also be made to optimize the speaker population for generating good language models. Speaker me011 will pursue volunteers from the Haas Business School to record their weekly meetings at ICSI. A tentative decision was made to offer participants a recorded version of their meeting on a cd rom once the transcript screening phase is complete for that meeting.

PROGRESS AND ACHIEVEMENTS

Research aims and corresponding concerns for future data collection were discussed. A student researcher will be working with speaker fe016 to investigate different strategies for automatically summarizing meetings, and identifying discussional 'hotspots'. Efforts by speaker mn005 are ongoing to detect regions of speaker overlap in the signal. A total of 230 regions of overlapping speech have been manually transcribed for a subset of meeting data. Supervised clustering and neural networks are being considered as means for classifying overlap. It was suggested that speaker mn005 focus on a small set of acoustic parameters, e.g. energy and harmonics-related features, to use the mixed signal to distinguish regions of overlap from those containing the speech of just one speaker. Future work may also involve focussing on additional signals, and using a Markov model to analyze acoustic parameters over larger time frames. Beam-forming was suggested as an alternate method of detecting overlapping speech. Efforts are ongoing to select an optimal method for anonymizing speakers. More disk space is gradually being made available for the storage of new Meeting Recorder data.

Table 2.3: Abstractive summary for one of the meetings of the ICSI meeting corpus (Bmr006). Each abstractive summary contains four sections: Abstract, Decisions, Progress and Achievements, and Problems (the last one is omitted here because of space constraints).

undergraduate student in journalism—were hired for the task. The work represented a total of 166 person-hours, i.e., each annotator spent on average 4 minutes per sentence.

The annotators were asked to remove disfluencies and unimportant information in order to create summary sentences that are concise and pertinent, while preserving as much of their grammaticality and original meaning as possible. For each meeting, annotators were given access to a human extractive summary taken from the set of human extractive summaries described in 2.1.5, and were asked to annotate sentences of the extractive summary in sequential order.⁴ In order to enable an understanding of each original sentence in its context, annotators always had access to the full transcription from which the extractive summary was drawn. As shown at the top of Figure 2.1, annotators were able to access four different panes: (1) the “annotation” pane, in which they are supposed to mark words for deletions; (2) the “transcription” pane, in which the current utterance is displayed in its original context in the full transcription; (3) the “extractive summary” pane, in which the current utterance is displayed within the extractive summary; (4) the “abstractive summary” pane, which contains compressions for all sentences that have already been processed by the same annotator.

As other summarization tasks, sentence compression is a subjective task, since annotators often do not agree on what is important or salient in texts. In order to maximize inter-labeler agreement, we developed annotation guidelines that provide specific instructions on what types of textual units should be discarded if deemed unnecessary. The annotation protocol was divided into multiple stages, where each stage corresponded to a specific type of editing (e.g., remove all hedges such as “I think” and “I believe”). Note that annotators always had the choice to retain any textual units they thought were important. For instance, the deletion of hedges was discouraged when the annotators felt that the mitigation or weakening effect of any given hedge is important to convey in the summary. The full annotation guidelines are provided in Appendix B, of which we now give a brief overview.

The seven annotation stages are listed in Table 2.4, which result in eight different levels of compression. Annotators were also specifically asked to ensure that the result of each annotation stage

⁴The 14 meetings annotated for sentence compression are the following: Bed004, Bed009, Bed010, Bed016, Bmr001, Bmr005, Bmr006, Bmr007, Bmr013, Bmr019, Bmr023, Bns002, Bro008, and Bro018. Bmr005 and Bed009 were annotated by two subjects, and were used respectively as development and test set.

Stage/Transcript	%ratio	Edit type	Edits
0	100	-	none (<i>full utterances</i>)
1	89.6	delete	remove repairs
2	82.9	delete	remove fillers and parentheticals
3	79.3	delete	remove hedges
4	64.2	delete	remove unimportant information (1)
5	53.1	delete	remove unimportant information (2)
6	54.0	free editing	make sentence grammatical
7	53.2	free editing	make sentence fluent and concise

Table 2.4: Annotation stages for the compression ICSI meeting sentences. Note that edits in the last column are cumulative, e.g., transcript-3 neither contains repairs, fillers, parentheticals, nor hedges. “ratio” refers to the ratio between the length of the transcription and the length of transcript-0 (note that length slightly increases in Stage 6, since the correction of grammar often require adding new words to the sentence).

yields a sentence that is as grammatical and consistent with the original meaning of the sentence as possible (since input sentences were not always grammatical, their compressions sometimes yielded ungrammatical sentences that could not be fixed before stage 6). In each of the first five stages, the annotator decided for each word whether or not it should be deleted. Once a word is removed in one of these five stages, it cannot be re-introduced in one of the subsequent stages.

Figure 2.1 displays how the process of removing words was implemented in an online graphical user interface: the annotator can select any range of words to be deleted, which then appear in red. These words are then marked for deletion and no longer appear in the subsequent stages.

In the first stage, annotators were asked to remove edited words in self-repairs, e.g., “why didn’t he?” in the utterance “why didn’t he? why didn’t she stay home?”. In the second stage, fillers (e.g., “um”, “uh”) and parentheticals (e.g., “I mean”) are removed. Stage 3 corresponds to hedges, such as “I think” and “I believe”. Stages 4 and 5 are admittedly the most subjective ones, since annotators are then asked to remove unimportant information. We decided to break down this summarization annotation into two, since we thought that multiple stages would provide us greater freedom to build reference summaries with different levels of compression. We defined these two annotation tasks as

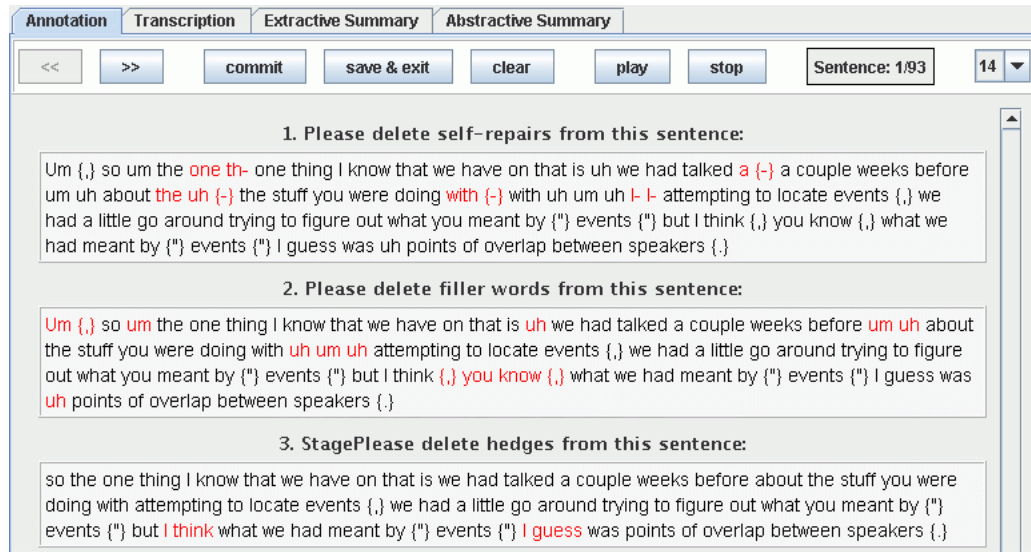


Figure 2.1: Sentence compression annotation interface.

follows:

- **Stage 4:** remove words or phrases that provide no or little factual information;
- **Stage 5:** remove any information that is not critical to the overall meeting summary.

We did not exploit the two last stages in our work, but nevertheless describe them here for completeness. Our decision to append revision stages after all compression stages was mostly driven by the fact that we are not dealing with arbitrary revisions, but only with deletions. In stage 6, annotators were asked to make each sentence grammatical, by allowing them to perform any kind of re-writing.⁵ In stage 7, annotators were given the freedom to perform any kind of editing as to make the end result as fluent, concise, and pertinent as possible. This stage effectively allows certain types of condensations not possible in other stages, e.g., rewriting “a few weeks ago” into “recently”.

We now analyze inter-labeler agreement on two meeting summaries (Bmr005 and Bed009) that were compressed by two different annotators, which are presented in Tables 2.5 and 2.6. These two annotators were the same in both cases. The tables analyze the level of agreement between

⁵While it may seem more sensible to put correction of grammar (stage 6) before all other stages, this would have meant that our sentence compression system would learn how to perform other corrections—such as removing unimportant information—from texts that are always grammatical, yet the input at run time often includes ungrammatical sentences.

	1→3	1→4	1→5
ratio of retained words (human-1)	.852	.713	.542
ratio of retained words (human-2)	.881	.640	.549
κ	.747	.525	.529
κ_{max}	.874	.833	.987

Table 2.5: Inter-labeler agreement in sentence compression on the development set of the ICSI meeting corpus (Bmr005). $x \rightarrow y$ refers to the transformation from transcript- x to transcript- y , which are defined in Table 2.4.

	1→3	1→4	1→5
ratio of retained words (human-1)	.895	.810	.632
ratio of retained words (human-2)	.919	.711	.575
κ	.695	.528	.555
κ_{max}	.851	.732	.881

Table 2.6: Inter-labeler agreement in sentence compression on the test set of the ICSI meeting corpus (Bed009).

different stages shown in Table 2.4, e.g., “1→ 5” refers to the transformation of a relatively fluent utterance (stage 1) into an utterance that has been significantly shortened (stage 5). In each case, we indicate how much text was retained by human-1 and human-2, and the level of inter-labeler agreement computed with Cohen’s κ (Cohen, 1960). κ scores respectively average .721, .526, .542 for the three tasks, which suggests that the level of agreement is moderate according to Landis and Koch (1977) (they consider .41 as the minimum that suggests moderate agreement, and .61 as a minimum to substantial agreement). Since there seems to be more agreement between stage 1 and 5 than between stage 1 and stage 4, the sentence compression task evaluated in Chapters 4 and 5 corresponds to the latter case (1→ 5).

In the sentence compression task, a relatively low κ score (e.g., .528) may not necessarily mean the two judges do not agree on what is important. In some cases, it might be due to a disagreement on how much to delete. To help give a sense of how to weight the two sources of disagreement, we also provide the maximum obtainable kappa score κ_{max} given the number of words that were selected by each annotator. For instance, the transformation from transcript-1 to transcript-4 yields $\kappa = .528$ while $\kappa_{max} = .732$. Since the difference between κ and κ_{max} is relatively small compared to the other cases, we might hypothesize that a major source of disagreement in this case was how much to reduce sentences.

2.2 NIST, CMU, and AMI Meeting Corpora

The increased level of interest in meeting processing and recent meeting recognition task evaluations (NIST, 2004b) spurred the emergence of additional meeting corpora. The CMU/ILS corpus (Waibel et al., 2001) contains 18 transcribed meetings that were either naturally occurring (research group meetings) or artificial (game playing tasks, military strategic exercises, discussions of general topics including news (Burger et al., 2002)). The NIST pilot corpus (NIST, 2004a) consists of 19 transcribed meetings, a mix of real meetings (staff regular meetings, technical presentations, planning events) and scenario-driven meetings (game playing, focus groups discussions about general subjects such as news events and movies).

The AMI corpus contains meetings recorded at IDIAP (McCowan et al., 2003) and the University of Edinburgh (Hain et al., 2005; Carletta, 2006) representing about 100 hours of meeting time.

Some are naturally occurring (research meetings), others are elicited, e.g. task-oriented meetings where participants play different roles in design teams (see (Carletta et al., 2005)), and informal discussions about freely chosen subjects (e.g. discussions about recent travels, last books and papers read, and movies).⁶ Rich transcription (RT) NIST evaluations (NIST, 2004b) rely on the ICSI, CMU, and NIST data, and define standard test sets for these corpora.

These meetings are not used in our work, due to the lack of annotation to either train or test a summarization system.

2.3 The Switchboard Corpus

Switchboard is a corpus of spontaneous conversations that comprises 2,430 telephone exchanges between unacquainted adults (Godfrey et al., 1992). From the corpus, 400 conversations have been annotated with disfluencies and syntactic structure (phrase-structure trees), and incorporated into the third release of the Penn Treebank (Marcus et al., 1994). We use this corpus to train a classifier that corrects self-repairs (Section 5.2) and a syntactic parser adapted to conversational speech (Section 4.1.1).

2.4 The Ziff-Davis Corpus

In Chapter 4, we not only analyze the performance of our sentence compression system on conversational speech, but also on written texts. Since previous work in sentence compression was mostly applied to written texts, we believe it is interesting to establish comparisons between the two genres.

To train the syntax-directed sentence compression model described in Chapter 4, we used the Ziff-Davis corpus (Harman and Liberman, 1993), a collection of newspaper articles announcing computer products. In our experiments, we used 24,985 document-abstract pairs drawn from the Ziff-Davis corpus.⁷ These abstracts contain 149,576 sentences, which represent about 3.3 million words. Full documents contain 1,407,049 sentences, which represent about 32.3 million words. Since our sentence compression component requires sentence pairs to be aligned in order to extract

⁶The current release is available from: <http://mmm.idiap.ch>

⁷The full corpus contains 39,060 documents. The 14,075 documents not used in our work did not contain any abstracts.

compression rules, we automatically aligned these abstract sentences to full document sentences. Since we are mostly interested in sentence pairs that do not involve a significant amount of rewording, a good criterion to match sentences between documents and abstracts is the minimum edit distance (Levenshtein, 1966). Specifically, each abstract sentence is aligned to the document sentence that minimizes the edit distance between the two.

While (Knight and Marcu, 2000) took a similar approach to identifying sentence pairs, their extraction of sentence pairs was limited to pairs that do not contain any insertion or substitution between from the full sentence to the abstract sentence. As a result, Knight and Marcu could only exploit 1,087 such paired sentences in the entire Ziff-Davis corpus, which represents a recall of only 1.75%.⁸ We go a step further by extracting sentence pairs that contain a mild amount of rewording, and allow any number of deletions and up to 7 word substitutions between the full sentence and the summary sentence (which provides a total of 17,019 sentence pairs, of which 16,787 are used for training). Our motivation in doing this is to address the general lack of appropriate training resources in sentence compression by incorporating training material that may not be as good as sentence-compression pairs, but that nevertheless can help collecting statistics of word and constituent deletions. In Section 4.2.1, we will present a method for extracting grammatical compression rules that takes significant advantage of these thousands of extra sentences.

⁸We refer here to the coverage of 1.75% reported by (Daumé III and Marcu, 2004). In fact, these 1,087 sentences represent only 0.72% of the 149,576 abstract sentences we were able to extract from the corpus. We assume this difference might be due to some discrepancies between our corpus pre-processing stages and the ones used by (Knight and Marcu, 2000).

Chapter 3

Incorporating Discourse Structure into Probabilistic Sequence Models for Utterance Selection

3.1 Introduction

The purpose of content selection is to reduce the input speech to a set of utterances that capture what is important in the source transcription. Since most meeting transcripts are lengthy—14,000 words on average—this can significantly reduce user reading time.

There has been a large body of work in content selection for text summarization (Mani and Maybury, 1999), with applications to many text domains, primarily newswire data. Several researchers have also addressed the problem of summarizing speech in domains such as broadcast news (Barzilay et al., 2000; Zechner and Waibel, 2000a; Hori et al., 2002; Maskey and Hirschberg, 2005) and voicemail (Koumpis and Renals, 2000; Koumpis and Renals, 2003b), but applications to meetings are relatively few (Murray et al., 2005a; Galley, 2006; Murray and Renals, 2006). We believe, however, that many factors make meeting summarization interesting. Meetings present a rich source of structural and pragmatic information that makes summarization of multiparty speech quite unique. Such information sources include time, duration, and identity of the speaker, and enable fine-grained pragmatic analyses to identify turn-taking events that are helpful cues to conversational

structure (Sacks, Harvey et al., 1974; Schegloff and Sacks, 1973).

A fundamental element of conversational structure is the adjacency pair (Schegloff, 1968), also known as *dialogic pair* (DP) (Harris, 2005), a concept drawn from the conversation analysis literature. DPs are pairs of utterances such as QUESTION-ANSWER or REQUEST-GRANT, in which the second utterance is said to be conditionally relevant to the first. Our analyses of patterns in the verbal exchange between participants found that dialogic pairs have particular relevance to summarization. As described in Subsection 3.4.1, we found that there is a strong statistical correlation between the two elements of a dialogic pair in summarization, and that one element of a pair (e.g., REQUEST) is unlikely to be included if the other element (e.g., GRANT) is not present in the summary. The motivation for using dialogic pairs in summarization is two-fold: (1) by accounting for the local structure of discourse, we expect to improve content selection accuracy by exploiting statistical dependencies between pragmatically-related utterances; (2) by penalizing the removal of an utterance pragmatically linked to a summary utterance, we hope to generate summaries that sound more coherent and natural (qualities that are however not accounted for in automatic evaluation metrics).

In this chapter, we investigate the question of how to incorporate such discourse-level dependencies into probabilistic classifiers for content selection. State sequence Markov models—such as hidden Markov models (HMM) (Rabiner, 1989) and conditional Markov models (Ratnaparkhi, 1996; McCallum et al., 2000; Lafferty et al., 2001)—have been highly successful in many classification tasks in speech and natural language processing. This is also the case in summarization, e.g., Conroy et al. (2004) built an HMM-based summarizer that consistently ranked among the top systems in recent Document Understanding Conference (DUC) evaluations. However, these sequence models only encode local (n -th order Markov) dependencies between utterances to be labeled. In multiparty speech, the two elements of a dialogic pair are generally arbitrarily distant. Hence, such models can only poorly account for dependencies underlying DPs in summarization, and are prone to overfitting with large Markov orders.

To address these limitations, we construct a *conditional Markov random field* (CRF) (Lafferty et al., 2001) whose structure not only reflects dependencies between the current utterance and its predecessor, but also explicitly represents dependencies between arbitrarily-distant pragmatically-related utterances, such as a QUESTION-ANSWER pair. In our work, we show that a CRF model

incorporating long-distance edges representing dialogic pairs achieves the highest accuracy in our experiments and reaches 92.9% of human performance.

In the second part of this chapter, we also investigate the use of dialogic pairs as a tool for structural prediction in more complex summarization tasks. In many applications and summarization settings, we may have prior knowledge and assumptions about the kind of information the user would want to retrieve, e.g., all statements that were agreed or disagreed by other speakers. To illustrate the benefit of a probabilistic model that explicitly represents dependencies among dialogic pairs, we address the problem of identifying agreements and disagreements, and show that these probabilistic models outperform competitive counterparts that only contain local structure (i.e., 1st order Markov structure).

This chapter is organized as follows. In Section 3.2, we provide linguistic background on dialogic pairs, then describe and evaluate an approach for automatically identifying such pairs. In Section 3.3, we present probabilistic graphical models exploiting the dependency structure between adjacent sentences and non-local discourse relations such as dialogic pairs. In Section 3.4, we present two sequence classification tasks that make use of dialogic pairs: content selection and agreement and disagreement classification.

3.1.1 Notations and Preliminary Remarks

In this chapter, we address sequence labeling tasks for multiparty speech such as meetings. Each sequence corresponds to a single meeting or conversation, and we assume we are provided with a segmentation $\mathbf{u} = (u_1 \dots u_T)$ into utterances, and that each utterance is associated with a given input \mathbf{x}_t and a label y_t we are attempting to predict. We define the following notation for the remaining part of this chapter:

- u_t identifies the t -th utterance of the conversation, where $1 \leq t \leq T$. u_t is defined as a quadruple $u_t = (t, s_t, e_t, p_t)$, where s_t and e_t respectively refer to the start and end time of u_t (both expressed in seconds), and $p_t \in \mathcal{P}$ is a nominal attribute that uniquely identifies one of the M speakers or participants in the conversation; for instance, $\mathcal{P} = \{\text{alice, bob, chris}\}$.
- $\mathbf{x}_t \in \mathbb{R}^n$ is an n -dimensional input vector corresponding to $u_t = (t, s_t, e_t, p_t)$, and that is extracted from the audio channel of participant p_t on the interval $[s_t, e_t]$. The input vector x_t

represents words, durations, and acoustic features such as pitch and energy, but also structural and duration features not directly observable from the channel (e.g., a predicate that is true if and only if $p_{t-1} = p_t$). Typically, many components x_t are simple boolean-value functions.

- $(b_i, a_i) \in \mathcal{D}$ identifies a dialogic pair, where $a_i \in \mathcal{A}$ and $b_i \in \mathcal{B}$ are respectively the indices the first and second element of the DP and satisfy $a_i < b_i$. The sets $\mathcal{A} \subset \{1 \dots T\}$ and $\mathcal{B} \subset \{1 \dots T\}$ are indices of all utterances that are first and second dialogic pair elements, respectively. Since no two DPs share the same second element (e.g., an answer generally does not address more than one question), the binary relation $\mathcal{D} = \{(b_i, a_i)\}_{i=1}^D$ is functional, and we can define the function $a : \mathcal{B} \rightarrow \mathcal{A}$ satisfying $a(b_i) = a_i$ for all $(b_i, a_i) \in \mathcal{D}$.
- y_t is the label to predict. In Section 3.4.1, $y_t \in \{+1, -1\}$ indicates whether or not u_t is part of the summary. In Section 3.4.2, $y_t \in \{\text{AGREE, DISAGREE, BACKCHANNEL, OTHER}\}$ specifies the pragmatic role of u_t .

The term *utterance* in this chapter is purposely ambiguous, since the notation presented here will be used interchangeably for different definitions of segmental units. In content selection (Section 3.4.1), utterances map to dialog act (DA) units (Allen and Core, 1997), because our utterance selection annotation is defined in terms of manually annotated DA segments. In agreement and disagreement classification (Section 3.4.2), the reference agreement and disagreement annotation (Hillard et al., 2003) is defined in terms of a silence-based segmentation (any pause of .5 seconds or more is treated as a boundary). However, nothing would prevent us from using different unit types, such as dialog turns, breath groups, and intonational phrases. In the chapter on system integration (Chapter 5), we will also rely on automatically-generated segments.

3.2 Dialogic Pairs

3.2.1 Linguistic Background

Adjacency pairs (Schegloff, 1968) or *dialogic pairs* (DP) (Harris, 2005) are fundamental units of conversational organization, in that they give coherence to the speech by identifying who is to speak and by defining a set of expected continuations in the conversation. A dialogic pair is said to consist of two parts (later referred to as *A* and *B*) that are ordered and produced by different

speakers. The first part makes the second one immediately relevant, as a question does with an answer, or a request does with a grant. Since most of the descriptive literature applies to two-party interaction, some authors further define dialogic pair as a pair of adjacent utterances (hence, the term *adjacency pair*). This requirement is problematic in many respects (Levinson, 1983): even when DPs are not directly adjacent, the same constraints between pairs and mechanisms for selecting the next speaker remain in place (e.g., the case of embedded question and answer pairs). The relaxation on a strict adjacency requirement is particularly important in interactions of multiple speakers, since other speakers have more opportunities to insert utterances between the two elements of the DP construction (e.g., interrupted, abandoned or ignored utterances; backchannels; DPs with multiple second elements, e.g., a question followed by answers of multiple speakers).¹ As noted previously, it is generally assumed that no two DPs share the same B part (e.g., no two questions were simultaneously answered by the same utterance), even though a few such rare cases emerged in the DP annotation.

The percentage of DPs labeled in our data that do not appear as adjacent utterances is 69.6%, which provides good reasons for explicitly modeling such dependencies in our graphical model structures. Since such non-adjacent cases are particularly frequent, the problem of identifying them automatically becomes non-trivial. The next subsection describes a model for DP identification that significantly outperforms a quite competitive baseline.

3.2.2 Automatic Identification

The task of dialogic pair identification can be quite naturally formulated as a *ranking* problem, otherwise also known in machine learning as ordinal regression (McCullagh, 1980; Cliff, 1996; Crammer and Singer, 2001). Ranking or re-ranking techniques have found many usages in human language technologies, including speech recognition (Ostendorf et al., 1991), information retrieval (Herbrich et al., 1998), syntactic parsing (Collins, 2000), machine translation (Och and Ney, 2002), and question answering (Ravichandran et al., 2003), where re-ranking algorithms are typically used to find an optimal hypothesis within an N -best list generated by a baseline system.

¹Since the very reason for explicitly incorporating these discourse dependencies into our graphical model structures lies in the fact that such sentence pairs are often *not* adjacent, we use the term *dialogic pair* instead of *adjacency pair* to avoid any confusion, even though the former term is relatively new and much less used in the descriptive literature.

The problem is to identify, for a given index b_t , the index or *rank* a_t , so that (b_t, a_t) identifies a dialogic pair. Since not every utterance is a second element of a dialogic pair, we must also determine whether the utterance of index b_t is member of \mathcal{B} . To cope with this possibility, we let the ranking model hypothesize a special null symbol \emptyset , so we define our ranking problem as follows: for any given utterance index t , we seek to determine the class $a_t \in \{\emptyset, 1, \dots, t-1\}$, so that $(t, a_t) \in \mathcal{D}$ if $t \in \mathcal{B}$, and $a_t = \emptyset$ otherwise.

We use log-linear models, more commonly known in NLP as maximum entropy models (Berger et al., 1996), to directly estimate the conditional probability $p(a_t|\mathbf{x}, t)$. It allows us to combine two types of feature functions, which we term *ranking features* and *classification features*. Ranking feature functions, as in (Och and Ney, 2002), can take non-zero values for any number of ordinal classes. For instance, a particularly helpful ranking feature for DP identification is one that returns the word-vector cosine similarity between the two utterances identified by index t and class label a_t , and capitalizing on the fact that the two utterances of a dialogic pair tend to have many words in common:

$$f_1(\mathbf{x}, t, a_t) = \begin{cases} \cos(x(t), x(a_t)) & \text{if } a_t \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$$

Conversely, classification features can take non-zero values for only one class, and represent characteristics of a particular class to predict. Many log-linear classifiers in NLP only rely on classification features, for instance in part-of-speech tagging (Ratnaparkhi, 1996), where a helpful feature to predict the class DT or determiner is one that returns 1 if the current word is *the*. In the case on DP identification, we use classification features to distinguish between the class \emptyset and others. A particularly helpful classification feature returns the number of words in the sentence if the class label is \emptyset , exploiting the fact that utterances that are not second elements tend to be longer than others:

$$f_2(\mathbf{x}, t, a_t) = \begin{cases} \text{length}(x(t)) & \text{if } a_t = \emptyset \\ 0 & \text{otherwise} \end{cases}$$

These two types of feature functions are then combined to define a single model, in which classification features compete against ranking features to increase the likelihood of the class \emptyset , and ranking features increase the likelihood of any given ordinal label to compete against all other classes. Given these J feature functions $f_j(\mathbf{x}, t, a_t)$ and J model parameters $\Lambda = (\lambda_1, \dots, \lambda_J)$, the

probability of $a_t \in \mathcal{A}_t = \{\emptyset, 1, \dots, t-1\}$ according to our model is

$$p_{\Lambda}(a_t|\mathbf{x}, t) = \frac{1}{Z_{\Lambda}(\mathbf{x}, t)} \exp\left(\sum_{j=1}^J \lambda_j f_j(\mathbf{x}, t, a_t)\right),$$

where the partition function $Z_{\Lambda}(\mathbf{x}, t)$ is defined as

$$Z_{\Lambda}(\mathbf{x}, t) = \sum_{a'_t \in \mathcal{A}_t} \exp\left(\sum_{j=1}^J \lambda_j f_j(\mathbf{x}, t, a'_t)\right).$$

To find the most probable class \hat{a}_t , we use the following decision rule:

$$\begin{aligned} \hat{a}_t &= \arg \max_{a'_t \in \mathcal{A}_t} \left\{ p_{\Lambda}(a'_t|\mathbf{x}, t) \right\} \\ &= \arg \max_{a'_t \in \mathcal{A}_t} \left\{ \sum_{j=1}^J \lambda_j f_j(\mathbf{x}, t, a'_t) \right\} \end{aligned}$$

Finally, to train the parameters Λ of the model, we use the generalized iterative scaling algorithm (Darroch and Ratcliff, 1972; Ratnaparkhi, 1998) as implemented in YASMET (Och, 2003). To reduce overfitting, we use a Gaussian prior of .3 (Chen and Rosenfeld, 1999). We use the same smoothing techniques for all log-linear models described later in this thesis.

3.2.3 Features

We now describe the features of the log-linear model described in the previous subsection. Ranking features, i.e., those that may take non-zero values for ordinal classes $\{1, 2, \dots\}$ are listed in Table 3.1. Note that each of these features characterizes a given utterance A (the utterance of index a_t) and B (utterance of index t) in order to establish whether or not they constitute a dialogic pair. This is in contrast with classification features explained later, which characterize B alone in order to determine if any property of B suggests that B is not part of a dialogic pair. Among ranking features, structural features encode some helpful information regarding ordering and overlap of utterances. For instance, a particularly helpful feature is the count the number of utterances between A and B, since this number is typically low.

Classification features, which may take non-zero values only for class \emptyset , are listed in Table 3.2. Classification features that are good predictors include structural features to determine whether the current utterance and its immediate predecessor are produced by the same speaker, which typically suggests the current utterance is not in a dialogic pair.

Structural features:

- number of speakers taking the floor between A and B
- number of utterances between A and B
- number of utterances of speaker B between A and B
- do A and B overlap?

Duration features: (time expressed in seconds)

- duration of A
- if A and B do not overlap: time separating A and B
- if they do overlap: duration of overlap
- seconds of overlap with any other speaker
- speech rate in A

Lexical features:

- number of words in A
- number of content words in A
- ratio of words of A (respectively B) that are also in B (respectively A)
- ratio of content words of A (respectively B) that are also in B (respectively A)
- cosine similarity between A and B
- number of n -grams present both in A and B (for n ranging from 2 to 4)
- first and last word of A
- number of instances in A of cue words listed in (Hirschberg and Litman, 1994)
- does A contain the first/last name of speaker B?
- does B contain the first/last name of speaker A?

Table 3.1: Features for dialogic pair identification (ranking features). *A* refers to the candidate utterance that may constitute the first element of a dialogic pair, and *B* identifies the current utterance.

<p><u>Structural features:</u></p> <ul style="list-style-type: none"> · number of consecutive utterances of the current speaker immediately before B · number of consecutive utterances of the current speaker immediately after B <p><u>Duration features:</u></p> <ul style="list-style-type: none"> · duration of B · speech rate in B <p><u>Lexical features:</u></p> <ul style="list-style-type: none"> · number of words in B · number of content words in B · first and last word of B · number of instances in B of cue words listed in (Hirschberg and Litman, 1994)

Table 3.2: Features for dialogic pair identification (classification features).

Feature sets	Accuracy
<i>Baseline</i>	73.8%
Lexical	79.4%
Structural and durational	83.6%
All	85.3%

Table 3.3: Results for dialogic pair identification.

Since lexical features cause the total number of features to be very large, we used forward feature selection (John et al., 1994) with cross-validation on the training data to remove many first-word and last-word features that occur infrequently and that are typically uninformative for the task at hand. Lexical features that remained after feature selection include mainly function words, in particular sentence-initial indicators of questions (“where”, “when”, etc.).

3.2.4 Results

We used the labeled dialogic pairs of 75 meetings described in Section 2.1.3. 60 were used for training, 10 for development, and 6 for testing instances. Note that we use here the same test set as in content selection used later in this chapter and described in Section 2.1.5. Table 3.3 displays the details of our results. Classification accuracy with all features is 85.27%, while the majority class

		PREDICTED		
		$\hat{a}_t \in \{1 \dots T\}$		
		$\hat{a}_t = a_t$	$\hat{a}_t \neq a_t$	$\hat{a}_t = \emptyset$
TRUE	$a_t \in \{1 \dots T\}$	1168	77	735
	$a_t = \emptyset$	—	681	7481

Table 3.4: Contingency table displaying ranking ($\hat{a}_t \in \{1 \dots T\}$) and classification ($\hat{a}_t = \emptyset$ or not) accuracies.

baseline, which always hypothesizes no dialogic pair (i.e., $a_t = \emptyset$), achieves 73.8%. We note that, while lexical features alone provide only 79.4% accuracy, they provide a significant return when added to other features (85.3%).

Table 3.4 displays the contingency table for the predictions of the model with all features. This table shows the details for three types of mispredictions. The model may incorrectly hypothesize that utterance B is not part of any dialogic pair (this happens in 735 cases). It may also hypothesize that it is in a DP while it isn't (681 cases). Finally, it may identify an incorrect first element A of the dialogic pair (only 77 cases).

3.3 Conditional Random Fields

We base our approach to classifying utterances for content selection on conditional Markov random fields (CRF) (Lafferty et al., 2001), which are undirected graphical models trained to maximize the conditional probability of the output \mathbf{y} given the input \mathbf{x} . CRFs have achieved empirical successes in various tasks in NLP, bioinformatics, and computer vision. While they have been primarily applied to sequence labeling tasks, recent applications of CRFs also include more complex structural prediction problems, such as classification on tree and graph structures (see (Sutton, 2006) for an overview). One type of CRF model that is particularly relevant to our problem is the *skip-chain CRF*, a model that extends linear-chain CRFs by taking into account probabilistic dependencies between distant entities (Sutton and McCallum, 2004; Finkel et al., 2005). This type of model has been used in named entity recognition—the task of classifying entities either a person, organization, etc.—to exploit the statistical dependencies between identical proper nouns in the input, which are

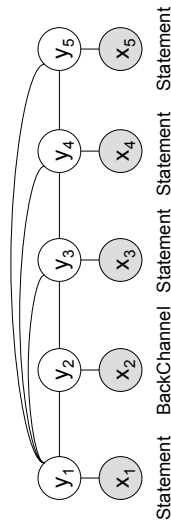


Figure 3.1: A skip-chain CRF with edges between utterances of the same dialogic pair.

often classified the same way (e.g., “Speaker Nancy Pelosi” and “Pelosi”).

Our approach to content selection uses a skip-chain CRF model. As shown in Figure 3.1, the graphical representation of a skip-chain CRF incorporates two types of edges: linear-chain edges, which connect adjacent utterances, and skip-chain edges, which connect pairs of utterances classified as dialogic pairs (in the pre-processing step described in Section 3.2.2). The model is defined using two potential functions Φ and Ψ , where Φ represents features defined over adjacent utterances (i.e., linear-chain edges) and Ψ represents features over dialogic pairs (i.e., skip edges). The probabilistic representation of the model is as follows:

$$p_{\Lambda}(\mathbf{y}|\mathbf{x}) \propto \prod_{t=1}^T \Phi(y_{t-1}, y_t, \mathbf{x}, t) \prod_{(b,a) \in \mathcal{D}} \Psi(y_a, y_b, \mathbf{x}, a, b) \quad (3.1)$$

Since skip-chain CRFs are implemented as log-linear models, each potential factorizes accord-

ing to a set of features f_j and g_j and weights $\Lambda = \{\{\lambda_k\}, \{\mu_k\}\}$:

$$\log \Phi(\mathbf{x}, y_{t-1}, y_t, t) = \sum_{j=1}^J \lambda_j f_j(\mathbf{x}, y_{t-1}, y_t, t)$$

$$\log \Psi(\mathbf{x}, y_a, y_b, a, b) = \sum_{j=1}^{J'} \mu_j g_j(\mathbf{x}, y_a, y_b, a, b)$$

Linear-chain feature functions f_j represent local dependencies that are consistent with a 1st-order Markov assumption. For instance, in the case of generic content selection, one such function is a predicate that is true if and only if the previous utterance is included in the summary, the current one is not included, and both are uttered by the same speaker:

$$f_1(\mathbf{x}, y_{t-1}, y_t, t) = \begin{cases} 1 & \text{if } y_{t-1} = +1 \text{ and } y_t = -1 \text{ and } p_{t-1} = p_t \\ 0 & \text{otherwise} \end{cases}$$

This feature allows us to effectively capture the tendency that the second consecutive utterance of the same speaker has less chances of being included than the first one. Given a set of dialogic pair indices $\mathcal{D} = \{(b, a)\}$, *skip-chain* feature functions g_j exploit dependencies between variables that are arbitrarily distant in the chain. For instance, the observation that OFFER-ACCEPT pairs, as other DPs, are often linked in summaries is encoded as a skip-chain feature predicate:

$$g_1(\mathbf{x}, y_a, y_b, a, b) = \begin{cases} 1 & \text{if } y_a = +1 \text{ and } y_b = +1 \text{ and } w_{t,1} = \text{"sure"} \\ 0 & \text{otherwise} \end{cases}$$

where $w_{t,1}$ represents the first word of utterance u_t (here, a “sure” identifying the acceptance).

Note that all these feature functions are automatically constructed from input vector \mathbf{x}_t . If n is the length of each input (or observation) vector \mathbf{x}_t and m is the number of classes, then the model contains $m^2 n$ linear-chain feature functions and $m^2 n$ skip-chain feature functions. Since $m = 2$ in our case, the total number of features is eight times the number of observations.

3.3.1 Parameter Estimation and Probabilistic Inference

We now discuss how to estimate the parameters Λ of a log-linear CRF model. We are given a training set $\mathcal{D} = (\mathbf{x}^{(i)}, \mathbf{y}^{(i)})_{i=1}^N$, and estimate model parameters Λ by optimizing the posterior probability $\log p(\Lambda | \mathcal{D})$.

2

While inference in highly connected graphs is generally intractable, the fact that long-distance edges or skip edges in our graphs rarely span more than 10 utterances makes it possible to train these models using exact probabilistic inference.³ However, this is often prohibitively expensive. In the case of generic content selection, our only two successful runs using the junction tree algorithm required six days to train. Hence, we relied on approximate inference with loopy belief propagation (unless otherwise noted).

3.4 Applications

We present two applications of sequence models with dialogic pair edges: in the first (Section 3.4.1), labels to predict are simple $\{+1, -1\}$ predictions for generic content selection. In Section 3.4.2, each label (e.g., AGREE) has a specific pragmatic function. Such labels could in turn be used to create more targeted summaries, such as one made of agreements, disagreements, and consensus.

3.4.1 Generic Utterance Selection

We now describe how probabilistic approaches described in Section 3.3 are being applied to content selection for meeting summarization. This subsection is organized as follows. We start by providing an empirical study that demonstrates that utterances linked together by dialogic pair dependencies have summary labels that are statistically highly correlated (Subsection 3.4.1.1). In Subsection 3.4.1.2, we then describe the class imbalance problem that arises in our sequence classification task, and we describe different solutions that are evaluated later in this section. In Subsection 3.4.1.4, we present and discuss summarization evaluation metrics. We present our results on this content selection task in Subsection 3.4.1.5.

²This is done using GRMM (Sutton, 2006), a tool for learning graphical models with arbitrary structure. GRMM implements several probabilistic inference engines for general graphs, such as the junction-tree algorithm (Lauritzen and Spiegelhalter, 1988; Jensen et al., 1990), loopy belief propagation (Murphy et al., 1999), and Gibbs sampling.

³Triangulated graphs produced here have clique sizes no larger than 7, which yield tables of size 128 (in the case of binary predictions).

Time	Speaker	DP	Transcript
1480.85-1493.91	1	A	are - are those d- delays adjustable? see a lot of people who actually build stuff with human computer interfaces understand that delay, and - and so when you - by the time you click it it'll be right on because it'll go back in time to put the -
1489.71-1489.94	2		<i>yeah.</i>
1493.95-1495.41	3	B	<i>yeah, uh, not in this case.</i>
1494.31-1495.83	2	B	<i>it could do that, couldn't it.</i>
1495.1-1497.07	4	B	we could program that pretty easily , couldn't we?

Table 3.5: Snippet of a meeting displaying three dialogic pairs, where a question (A) initiates three responses (B). Sentences in *italic* are not present in the reference summary. The skip-chain CRF structure corresponding to this snippet is displayed in Figure 3.1.

3.4.1.1 Motivation

Pragmatic influences can become quite complex in the case of dialogues or correspondences, especially when they involve multiple parties. In the case of summarization of conversational speech, Zechner (2002) found, for instance, that a simple technique consisting of linking together questions and answers in summaries—and thus preventing the selection of orphan questions or answers—significantly improved their readability according to various human summary evaluations. In email summarization (Rambow et al., 2004), Shrestha and McKeown (2004; 2007) obtained good performance in the automatic detection of question-answer pairs, which can help produce summaries that highlight or focus on the question and answer exchange. In a combined chat and email summarization task, a technique (Zhou and Hovy, 2005) for identifying dialogic pairs and appending any relevant responses to topic-initiating messages was instrumental in outperforming two competitive summarization baselines.

The need to model pragmatic influences, such as between a question and an answer, is also prevalent in meeting summarization. In fact, question-answer pairs are not the only discourse relations that we need to preserve in order to create coherent summaries, and, as we will see, most instances of DPs would need to be preserved together, either inside or outside the summary. Table 3.5 displays a DP construction with one statement (A part) and three respondents (B parts). This example illustrates that the number of turns between constituents of DPs is variable and thus difficult

Linear-chain edges	$y_t = +1$	$y_t = -1$
$y_{t-1} = +1$	529	7742
$y_{t-1} = -1$	7742	116040
Skip-chain edges	$y_b = +1$	$y_b = -1$
$y_a = +1$	6792	2191
$y_a = -1$	1479	121591

Table 3.6: Contingency tables: while the correlation between adjacent labels y_{t-1} and y_t is not significant ($\chi^2 = 2.3, p > .05$), empirical evidence clearly shows that y_a and y_b (where $(b, a) \in \mathcal{D}$) influence each other ($\chi^2 = 78948, p < .001$).

to model with standard sequence models. This example also illustrates helpful features discovered in this work. First, many speakers respond to A’s utterance, which is generally a strong indicator that the A utterance should be included. Secondly, while DPs are generally characterized in terms of pre-defined dialog acts, such as OFFER-ACCEPT, we found that the type of dialog act has much less importance than the existence of a speaker-address relation between each pair of utterances (DPs in the data represent a great variety of DA pairs, including many that are not characterized as DPs in the literature—e.g., STATEMENT-STATEMENT in the table). Since DAs seem to matter less than dialogic pairs, our aim is more to automatically identify and exploit such pairs rather than make any attempt to predict dialog act labels.

The intuition that the summarization labels (+1 or −1) are highly correlated with DPs is confirmed in Table 3.6. While contiguous labels y_{t-1} and y_t seem to seldom influence each other, the correlation between DP elements y_a and y_b is particularly strong, and they have a tendency to be either both included or both excluded. Note that the second table is not symmetric, because the data allows an A part to be linked to multiple B parts, but not vice-versa. While counts in Table 3.6 reflect human labels, we only use automatically predicted (a, b) pairs in the experiments of the remaining part of this section. To find these pairs automatically, we use the probabilistic ranker described in Section 3.2.2, which achieves an 85.3% accuracy (Galley et al., 2004b).

3.4.1.2 Probabilistic Ranking

We now discuss the problem of building a summary from the predictions of a probabilistic model such as skip-chain CRF. As we will see in Section 3.4.1.5, using the $\{-1, 1\}$ label predictions

of CRF models—i.e., finding the maximum a posteriori (MAP) assignment of the model—leads to significantly sub-optimal results, which is partially explained by the class imbalance problem. A highly skewed class distribution usually causes classifiers to predict the majority class disproportionately more often than the minority class. This is due to the fact that most classifiers are designed to maximize accuracy, either explicitly by minimizing Hamming loss (or 0/1 loss) or by optimizing objective functions related to the 0/1 loss (e.g., log-loss for log-linear models (Darroch and Ratcliff, 1972) and hinge-loss for support-vector classifiers (Vapnik, 1998)). It turns out that the classification task discussed here is severely affected by this class imbalance problem: while our training data contains 9.54% positive instances, our probabilistic models typically only classify about 3-5% of instances as positive.

This problem is actually very common, and existing literature presents several solutions to the class imbalance problem (Japkowicz, 2000), including upsampling the minority class (Ling and Li, 1998), downsampling the majority class (Kubat and Matwin, 1997), combining upsampling and downsampling (Chawla et al., 2002), and cost-sensitive learning (Domingos, 1999; Elkan, 2001; Gross et al., 2007; Sen and Getoor, 2006), most of which are not directly applicable to structural prediction problems like ours (where labels to be classified are not independent and identically-distributed).

A more fundamental issue is that it is generally not known in advance what should be the length of a summary. This length often reflects user’s preference and is generally hard to know in advance. While our training data contains about 9.5% of positive instances, we would like to be able to generate summaries of different lengths, e.g., 5%, 20% or 50% without having to train many different models tuned for different lengths. This is in contrast to most other classification tasks in NLP, e.g., part-of-speech tagging, for which class distributions do not dramatically change between training and testing.

A solution to these problems is to treat content selection as a ranking problem, i.e., use posterior probability estimates produced by the model to select the k most probable utterances in order to create a summary, where k is only known at run time. We compare three different approaches to selecting utterances:

- **sum-product ranking:** Use sum-product inference (Cowell et al., 1999), a general form of the forward-backward algorithm (Rabiner, 1989), to select the top- k utterances according to

$$p(y_t = 1|\mathbf{x}).$$

- **max-product ranking:** Use MAP inference (also known as max-product and max-propagation inference (Cowell et al., 1999)), a general form of the Viterbi algorithm (Viterbi, 1967), to select the top- k utterances according to the local probability estimate of $y_t = 1$ computed during MAP inference.⁴
- **max-product prediction:** Select $\hat{y} = \arg \max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})$. This is our baseline (non-ranking) approach, which simply uses the MAP assignment provided by the model in order to generate a summary.⁵

In our experiments, we will compare these three approaches to selecting summary utterances, and evaluate two inference engines, loopy belief propagation (BP) and the junction tree algorithm. Note that, in the case of loopy BP, sum-product and max-product inference is approximate. Loopy BP is the application of the standard BP algorithm (Pearl, 1988) to graphs with cycles. While BP is exact in the case of tree-based graphical models (e.g., chains), it is not only approximate in the case of general graphs, but also not guaranteed to converge. However, researchers have found that loopy BP generally converges for low-treewidth graphs (Murphy et al., 1999) (like ours), a finding that was confirmed in our case.

3.4.1.3 Features

We started our analyses with a large collection of features found to be good predictors in either speech (Inoue et al., 2004; Maskey and Hirschberg, 2005; Murray et al., 2005b) or text summarization (Mani and Maybury, 1999). Our goal is to build a very competitive feature set that capitalizes on recent advances in summarization of both genres. Table 3.7 lists some important features.

There is strong evidence that lexical cues such as “significant” and “great” are strong predictors in many summarization tasks (Edmundson, 1968). Such cues are admittedly quite genre specific,

⁴This probability can be written $p(y_t = 1|\mathbf{x}, \hat{\mathbf{y}}_{-t})$, where $\hat{\mathbf{y}}_{-t}$ denotes the MAP configuration without its t -th component.

⁵In the case of very short target lengths, it may happen that the produced summary is longer than the target. Since we can’t use any probabilities to break ties, we remove summary utterances—starting from the end of the document—until the constraint on length is satisfied.

<p><u>Lexical features:</u></p> <ul style="list-style-type: none"> · n-grams ($n \leq 3$) · number of words · number of digits · number of consecutive repeats <p><u>Information retrieval features:</u></p> <ul style="list-style-type: none"> · max/sum/mean frequency of all terms in u_t · max/sum/mean idf score · max/sum/mean tf.idf score · cosine similarity between word vector of u_t with centroid of the meeting · scores of LSA with 5, 10, 50, 100, 200, 300 dimensions <p><u>Acoustic features:</u></p> <ul style="list-style-type: none"> · seconds of silence before/during/after the turn · speech rate · min/max/mean/median/stddev/onset/outset f0 of utterance u_t, and of first and last word · min/max/mean/stddev energy · .05, .25, .5, .75, .95 quantiles of f0 and energy · pitch range · f0 mean absolute slope <p><u>Duration and structural features:</u></p> <ul style="list-style-type: none"> · duration of the previous/current/next utterance · relative position within meeting (i.e., index t) · relative position within speaker turn · large number of structural predicates, i.e. “is the previous utterance of the same speaker?” · number of DPs initiated with utterance u_t (u_t is the A part) <p><u>Discourse features:</u></p> <ul style="list-style-type: none"> · lexical cohesion score (for topic shifts) (Galley et al., 2003) · first and second word of utterance, if in cue word list · number of pronouns · number of fillers and fluency devices (e.g., “uh”, “um”) · number of backchannel and acknowledgment tokens (e.g., “uh-huh”, “ok”, “right”)
--

Table 3.7: Features for extractive summarization. Unless otherwise mentioned, we refer to features of utterance u_t whose label y_t we are trying to predict.

so we did not want to commit ourselves to any specific list, which may not carry over well to our specific speech domain; instead, we automatically selected a list of 200 n -grams ($n \leq 3$) based on information gain using 10-fold cross-validation on the training data.⁶ Other lexical features include: the number of digits mentioned,⁷ which is helpful for identifying sections of the meetings where participants collect data by recording digits; the number of repeats, which may indicate the kind of hesitations and disfluencies that negatively correlates with what is included in the summary.

The information retrieval feature set contains many features that are generally found helpful in summarization, in particular tf.idf and scores derived from centroid methods. In particular, we used the latent semantic analysis (LSA) features discussed in (Murray et al., 2005b), which attempt to determine sentence importance through singular value decomposition, and whose resulting singular values and singular vectors can be exploited to associate with each utterance a degree of relevance to one of the n LSA dimensions or “concepts” of the meetings. We used the same scoring mechanism as (Murray et al., 2005b), though we extracted features for many different n values.

Acoustic features extracted with Praat (Boersma and Weenink, 2006) were normalized by speaker, and include features derived from the F0 contour, energy, and duration. Structural features listed in Table 3.7 are those computed from the sequence model before decoding, e.g., the duration that separates the two elements of a DP. Finally, discourse features represent predictors that may substitute for DA labels (e.g., cue words). While DA tagging is not directly our concern, it is presumably helpful to capitalize on discourse characteristics of utterances involved in dialogic pairs, since different types of dialog acts may be unequally likely to appear in a summary.

⁶Specifically, we computed the information gain of the binary attribute X with respect to the binary class attribute Y . X is positive for a given utterance if and only if the given n -gram appears in the utterance, and Y determines whether or not the utterance is included in the summary. We scored each n -gram by information gain $I(Y; X) = H(Y) - H(Y|X)$ using 10-fold cross validation, and selected the 200 n -grams that had the highest average rank. This restriction to 200 n -grams prevents overfitting the data.

⁷As mentioned in Section 2.1.1, meetings of the ICSI corpus include artificial “digits” sections, in which meeting participants take long turns to read digits. We believe it is appropriate to include such corpus-specific features, since they are only aimed at removing utterances that would generally not appear in meetings that are fully naturally occurring.

3.4.1.4 Evaluation

Summarization evaluation is a difficult problem and there is no broad consensus on how to best perform this task. Two metrics have become quite popular in summarization, and particularly in multi-document summarization of news articles, namely ROUGE (Lin, 2004) and the Pyramid method (Nenkova and Passonneau, 2004b). ROUGE and Pyramid are techniques looking for content units of the hypothesis or *peer* summary that also appear in one or more reference or *model* summaries.

In the case of ROUGE, content units to match are represented as n -grams. In the Pyramid method, content units are represented with phrases or clauses that are considered by humans to have the same meaning (e.g., “called that non-overlap” and “cases where the transcribers said there was only one person talking here” in Figure 3.2 were deemed to represent the same content unit). Both Pyramid and ROUGE assess the quality of a peer by measuring the degree of overlap in content units between peer and summaries. Since peer summaries are typically built from different sources written by different authors, and that model summaries themselves are written by different abstractors, it is generally unlikely that a peer and a model summary will have more than a few sentences in common. In this context, it seems natural to judge the similarity between a peer summary and a set of models by attempting to match content units that appear in both.

Our situation is however quite different from multi-document summarization. In our case, all model summaries of a given document are utterance extracts of the very same input document, and the situation (common to multi-document summarization) that some propositional content is expressed in many different ways in the model summaries generally does not occur. In our own annotation of three meetings with summarization content units (SCU) as defined in (Nenkova and Passonneau, 2004a), we found that repetitions and reformulations of the same information are particularly infrequent, and that textual units that express the same content among model summaries are generally originating from the same document sentences, as can be seen in Figure 3.2. For instance, the SCUs *mention#4*, *mention#5*, *mention#8*, and *mention#9* are all expressed the same way in the model summaries of the meeting *Bmr019* (the fact that the expression in *mention#4* is repeated four times indicates that all four annotators agreed that “the next thing we had on the agenda was something about alignment” should be included in the summary). In our annotation of three test-set meetings (*Bmr019*, *Bmr005*, *Bed009*), we identified 621 different SCUs. 564 of them (90.8%) have only one wording (as, e.g., *mention#4*), 46 of them (7.4%) have two different wordings (as, e.g.,

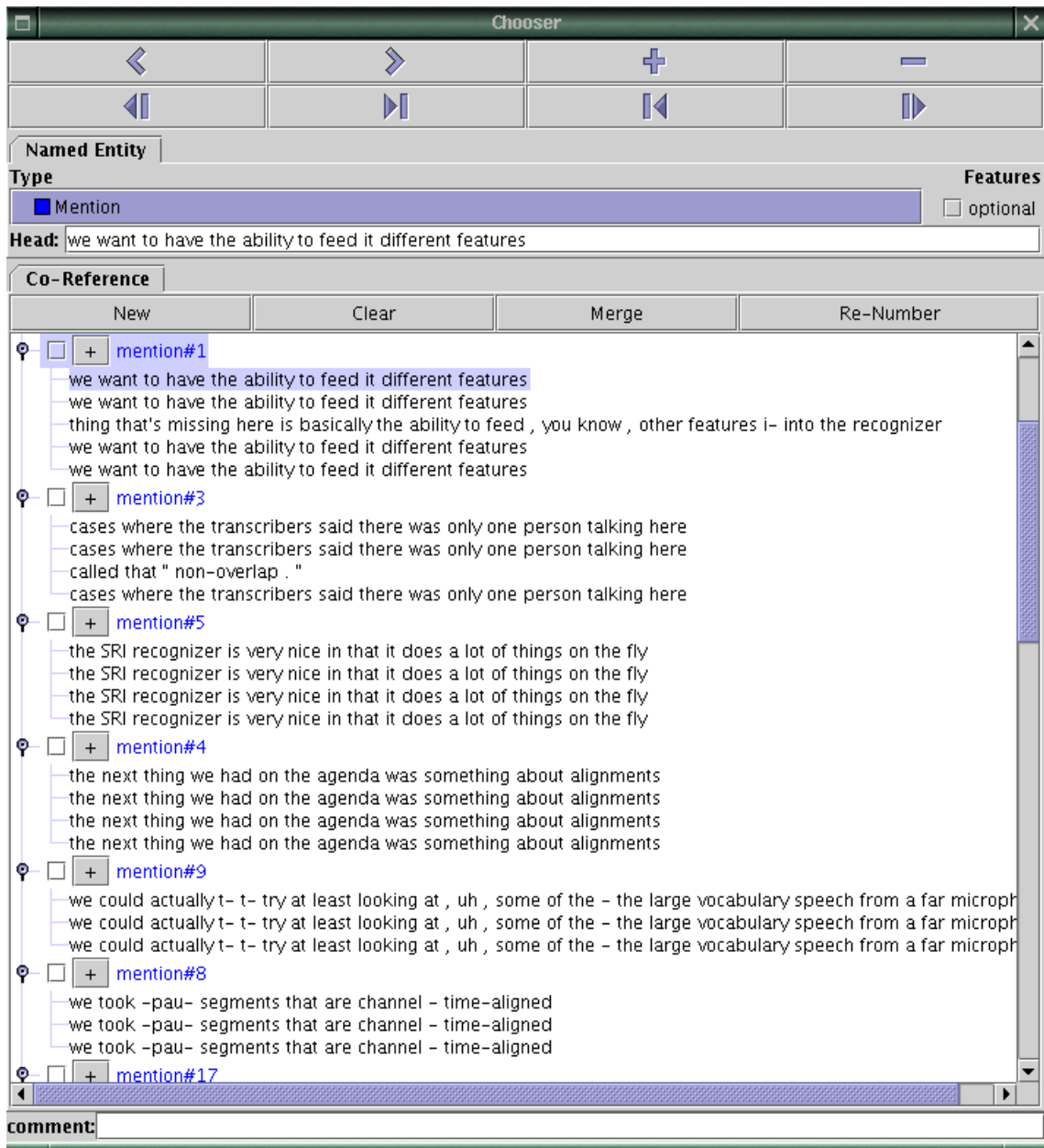


Figure 3.2: Summarization content units (SCU) of the ICSI meeting corpus (Bmr019). An SCU represents a grouping of textual units drawn from model summaries that express the same meaning. Since propositional content is seldom reformulated within the same meeting transcription, most SCUs are expressed with only one expression (e.g., the SCU *mention#5*).

mention#3), and 11 of them (1.7%) have three or more different wordings.⁸

Because of this lack of repetition and reformulation, we believe it is reasonable rely only exact matches to score peer summaries against model summaries, which allows us to sidestep the labor-intensive part of SCU annotation as defined in (Nenkova and Passonneau, 2004b). We thus use standard information retrieval metrics of precision, recall, and F_1 , where a peer sentence is deemed relevant if its exact match is included in the model summary. Since all test meetings provide multiple model summaries, precision and recall are averaged over all model-peer pairs, and F_1 is the weighted harmonic mean of average precision and average recall. Note finally that, prior to evaluating any peer summary, each summary (including model summaries) is converted from a sequence of utterance-level binary predictions to a sequence of word-level binary predictions, and that our evaluation is based on the latter representation. This is essentially to address the problem that the utterance segmentation used by the summarization system may differ from the utterance segmentation of model summaries. While, in this chapter, we slightly simplify the summarization problem by providing the reference utterance segmentation (i.e., DA segmentation) to the summarizer, Chapter 5 presents evaluations where the segmentation by utterances is fully automated. Since this segmentation is errorful and often differs from the reference DA segmentation, it is impossible to compare binary predictions at the utterance level, thus we need to assess binary predictions at the word level. We now precisely define the two evaluation metrics used in the two next subsections: F-measure and ROUGE.

F-measure In this evaluation, the peer summary H extracted from a given meeting is represented as a sequence of binary predictions (h_1, \dots, h_w) , where w is the number of words in the meeting, and where h_j indicates whether the j -th word of the meeting was selected by the underlying content selection system. Each model summary M_i (among a set of k model summaries $\{M_i\}_{i=1}^k$) is also represented as a binary sequence $(m_{i,1}, \dots, m_{i,w})$. We then define the average precision P of peer summary H as follows:

$$P = \frac{\sum_{i=1}^k \sum_{j=1}^w \llbracket h_j = +1 \rrbracket \llbracket h_j = m_{i,j} \rrbracket}{\sum_{i=1}^k \sum_{j=1}^w \llbracket h_j = +1 \rrbracket}$$

⁸Note that it never occurred that a given verbalization of an SCU would appear in different locations of the same meeting. In other words, if verbalizations are the same (as in *mention#4*), it means that all of them are drawn from the same utterance in the original transcript.

where $\llbracket x \rrbracket$ is a variant of the Kronecker delta function, which returns 1 if and only if the predicate x is true, and 0 otherwise. The average recall R can be computed as follows:

$$R = \frac{\sum_{i=1}^k \sum_{j=1}^w \llbracket h_j = +1 \rrbracket \llbracket h_j = m_{i,j} \rrbracket}{\sum_{i=1}^k \sum_{j=1}^w \llbracket m_{i,j} = +1 \rrbracket}$$

Note that, if the number k of model summaries is 1, the above expressions are respectively equal to standard precision and recall in information retrieval. F-measure is then be computed as follows:

$$F_1 = \frac{2 PR}{P + R}$$

Note also that average precision as computed here is proportional to summarization accuracy (Zechner and Waibel, 2000b), except that we do not divide average precision scores by the maximum achievable precision.

ROUGE ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is an evaluation metric (Lin, 2004) inspired by BLEU (see Section 4.3) that calculates n -gram overlaps between a peer summary and a set of model summaries. Given a peer summary H and a set of k model summaries $\{M_i\}_{i=1}^k$, ROUGE- n is defined as:

$$\text{ROUGE}_{-n} = \frac{\sum_{i=1}^k \sum_{g \in \text{gram}_n(M_i)} \text{count}(g, M_i, H)}{\sum_{i=1}^k \sum_{g \in \text{gram}_n(M_i)} \text{count}(g, M_i)}$$

where $\text{gram}_n(S)$ is the set of all n -grams contained in summary S (where S may be either a peer or model summary), $\text{count}(g, M)$ is the count of the n -gram g in model M , and $\text{count}(g, M, H)$ is the maximum between the number of occurrences of g in M and the number of occurrences in H .

3.4.1.5 Results

We follow (Murray et al., 2005b) in using the same six meetings as test data, since each of these meetings has multiple reference summaries. The remaining 69 meetings were used for training, which represent in total more than 103,000 training utterances (or DA units), of which 9,887 are positives (9.54%). The multi-reference test set contains more than 28,000 instances. More statistics regarding these transcripts, including inter-labeler agreements, are presented in Section 2.1.5.

The goal of a preliminary experiment was to devise a set of useful predictors from the initial full set of features. We performed feature selection by incrementally growing a log-linear model using

	FEATURES	F ₁
1	duration	.246
2	100-dimension LSA score	.268
3	duration of u_{t-1}	.275
4	duration between $u_{a(t)}$ and u_t (where $(t, a(t)) \in \mathcal{D}$)	.281
5	mean of IDF scores of content words in u_t	.284
6	relative position of u_t in the meeting	.286
7	number of DPs initiated at u_t	.288
8	duration of u_{t+1}	.288
9	number of fillers in u_t	.289
10	.25-quantile of energy	.290
11	number of lexical repeats	.292
12	lexical cohesion score produced by our topic segmenter (Galley et al., 2003)	.294
13	f0 mean of last word of utterance t	.294
14	50-dimension LSA score	.295
15	$p_t = p_{t+1}$? (true if same speaker)	.298
16	speech rate (words per second)	.302
17	“is that” occurs in u_t	.303
18	“for the” occurs in u_t	.303
19	$p_{t-1} = p_t$? (true if same speaker)	.305
20	“to try” occurs in u_t	.305
21	“meetings” occurs in u_t	.305
22	“<s> and” occurs in u_t	.306
23	“we have” occurs in u_t	.306
24	“new” occurs in u_t	.307
25	“<s> what” occurs in u_t	.307

Table 3.8: Some features found helpful on the ICSI corpus, listed in the order in which they were added during forward selection. The right column indicates the F₁-score of the classifier at that stage during feature selection. Unless otherwise noted, feature descriptions refer to the current utterance u_t .

FEATURE SET	F_1
lexical	36.7
IR	32.3
lexical + IR	38.7
acoustic	32.1
structural/durational	37.3
acoustic + structural/durational	37.4
all features	40.2
selected features (\mathcal{S})	41.0

Table 3.9: Results for content selection by feature set.

forward feature selection (John et al., 1994; Berger et al., 1996). Because of the imbalance between positive and negative samples, we found it more effective to select candidate features by gains in F_1 -measure through 5-fold cross validation on the entire training set. The increase in F_1 by adding new features to the model is displayed in Table 3.8; this greedy search resulted in a set \mathcal{S} of 217 predictors (i.e., \mathbf{x} has 217 dimensions).

We now analyze the performance of different sequence models on our test set. The target length for each summary was set to 10%, 18.6%, and 30% of the number of words of the full document. Note that 18.6% is the average (in terms of number of words) on the entire training data, and that the average on the test data is 19.1%. In Table 3.9, we use a 1st-order CRF model to compare our feature selection (\mathcal{S}) against all features and various categorical groupings. Overall, we notice lexical predictors and statistics derived from them (e.g., LSA features) represent the most helpful feature group, though all other features combined achieve a competitive performance.

Table 3.10 displays our main results comparing the different model structures. It compares a 1st-order linear-chain CRF with two skip-chain CRF models: one that incorporates edges between adjacent utterances (1st-order) and one that does not (0th-order). These three models were trained with loopy belief propagation, and sum-product probabilities were used to rank sentences. In order to assess the difficulty of content selection, we also provide human performance evaluation. Human performance is computed by cross-validating human summaries against other human summaries. The baseline is a lead summarizer, which adds the first words of the document until the constraint

	10%	18.6%	30%
1st-order skip-chain CRF	33.2	42.4	47.3
0th-order skip-chain CRF	32.7	41.5	45.5
1st-order linear-chain CRF	32.6	41.0	44.4
baseline	12.0	20.5	25.9
human	-	45.6	-
optimal	57.3	69.3	67.9

Table 3.10: F_1 scores for summaries of length 10%, 18.6% (break-even point), and 30% for different model structures.

on length has been satisfied. We see that the best model is the 1st-order skip-chain CRF model, which explicitly represents dialogic pair dependencies, and achieves 92.9% of human performance (at 18.6% compression). The difference between this best model and a standard linear-chain model is 1.4%. In the table, we also provide results for “optimal” summaries, whose F_1 scores are upper bounds for each column. Section 3.4.1.4 presents a method for computing these upper bounds.

Figure 3.3 displays ROC curves, which are typically used to evaluate ranking performance. It shows the 1st-order skip-chain CRF model and a standard linear-chain CRF. It appears that the former is particularly more effective in the mid-range in the tradeoff between true positive rate and false positive rate. The area under the curve (AUC) of the two models are respectively 0.760 and 0.744.

Table 3.11 displays performance for various sequence models trained using loopy belief propagation (BP). We tested three criteria for selecting summary utterances at runtime, sum-product inference and max-product inference (Subsection 3.4.1.2), and the classification approach that simply uses the maximum a posteriori assignment. We find that sum-product ranking provides significantly better results, which was confirmed in many other of our experiments.

We compare our results with loopy belief propagation against exact inference. We were able to train our best model (order-1 skip-chain CRF) model using the junction tree algorithm, with the exclusive access to a 4GB machine and a running time of almost a week. In comparison, experiments with loopy belief propagation presented in this section all took less than six hours on the same machine. Results with junction trees are almost indistinguishable from loopy BP in this case,

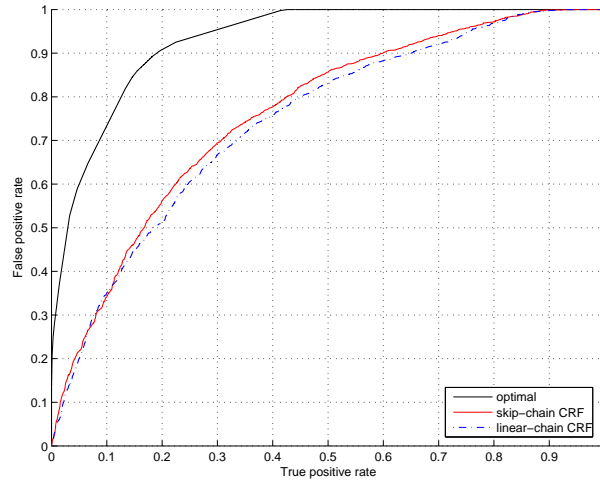


Figure 3.3: ROC curve comparing skip-chain CRF model with linear-chain CRF. “optimal” refers to the optimal summary, which may be obtained using the Pyramid method.

	10%	18.6%	30%
sum-product ranking	33.2	42.4	47.3
max-product ranking	31.4	40.9	45.4
max-product prediction	24.5	24.5	24.5
baseline	12.0	20.5	25.9
human	-	45.6	-
optimal	57.3	69.3	67.9

Table 3.11: F_1 scores for summaries of length 10%, 18.6% (break-even point), and 30% for different ranking approaches.

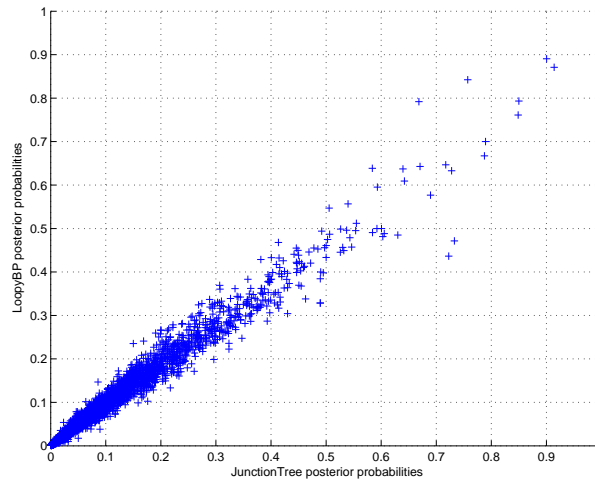


Figure 3.4: Correlation plot between the correct and approximate posteriors. Exact probabilistic inference (sum-product) is done with junction trees (JT) and approximate inference with loopy belief propagation (LoopyBP). This plot shows a correlation that is strong between JT and LoopyBP (Pearson coefficient: $r = .9985$).

and provided F_1 scores within a margin of .03% for the three summary sizes. The correlation plot between the predictions of the two models is displayed in Figure 3.4, and shows that the predictions of the two systems are highly correlated.

Finally, we performed a last experiment to compare our best system against Murray et al. (2005b), who used the same test data, but constrained summary sizes in terms of number of DA units instead of words. Since results in (Murray et al., 2005b) were reported on ROUGE scores computed against abstractive summaries (see Chapter 2), we use the same evaluation metric in our comparison. In their experiments, 10% of DAs had to be selected. Our system achieves .92 ROUGE-recall, .52 ROUGE-precision, and .65 ROUGE-F-score with the same length constraint. The discrepancy between recall and precision is largely due to the fact that generated summaries are on average much longer than model summaries (18.8% of the words vs. 3.4% of DAs), which explains why our precision is relatively low in this last evaluation. The best ROUGE-1 measure reported in (Murray et al., 2005b) is .69 recall, which is significantly lower than ours according to confidence intervals.

3.4.2 Agreement and Disagreement Classification for Summarization

One of the main features of meetings is the occurrence of agreements and disagreements among participants. Often meetings include long stretches of controversial discussion before some consensus decision is reached. Our goal is automated summarization of multi-participant meetings and we hypothesize that the ability to automatically identify agreement and disagreement between participants will help us in the summarization task. For example, a summary might resemble “minutes” of meetings with major decisions reached (consensus) along with highlighted points of the pros and cons for each decision. Towards this end, we now present a method to automatically classify utterances as agreement, disagreement, or neither.

Previous work in automatic identification of agreement/disagreement (Hillard et al., 2003) demonstrates that this is a feasible task when various textual, durational, and acoustic features are available.

We build on their approach by incorporating their most effective features, and show that we get an improvement in accuracy by treating agreement and disagreement classification as a sequential prediction problem. As for generic utterance selection, our approach first identifies dialogic pairs as explained in Section 3.2. This allows us to acquire, and subsequently process, knowledge about who speaks to whom. We hypothesize that pragmatic features that center around previous agreements between speakers in the dialog will influence the determination of agreement/disagreement. For example, if a speaker disagrees with another person once in the conversation, is he more likely to disagree with him again? In this section, we model context using conditional dynamic Bayesian networks (DBN) that explicitly represent these pragmatic dependencies.⁹ Our accuracy for classifying agreements and disagreements is 86.9%, which is a 4.9% improvement over (Hillard et al., 2003).

In this section, we first present the classification task, then describe the feature set, provide some empirical evidence justifying our choice of contextual features, and finally evaluate the classifier.

⁹Note that this work on agreement/disagreement classification (Galley et al., 2004b) was done much earlier than our work on generic content selection (Galley, 2006), and we did not experiment with skip-chain CRFs for agreement and disagreement classification.

3.4.2.1 Problem Statement

In our classification problem, each utterance y_t among the T utterances of a meeting must be assigned a tag $y_t \in \{\text{AGREE, DISAGREE, BACKCHANNEL, OTHER}\}$. To specify the speaker of the utterance (e.g. speaker B), the notation will sometimes be augmented to incorporate speaker identity, as with y_t^B . To designate the addressee of B (e.g. listener A), we will use the notation $y_t^{B \rightarrow A}$. For example, $y_t^{B \rightarrow A} = \text{AGREE}$ simply means that B agrees with A in the utterance of index t . This notation makes it obvious that we do not necessarily assume that agreements and disagreements are symmetric relations. We define:

$$\underset{Y \rightarrow X}{\text{pred}} (y_t^{B \rightarrow A})$$

as the tag of the most recent utterance before $y_t^{B \rightarrow A}$ that is produced by Y and addresses X. This definition will help our multi-party analyses of agreement and disagreement behaviors.

3.4.2.2 Features

Table 3.12 lists the features that were found most helpful at identifying agreements and disagreements. Regarding lexical features, we selected a list of lexical items we believed are instrumental in the expression of agreements and disagreements: agreement markers, e.g. “yes” and “right”, as listed in (Cohen, 2002), general cue phrases, e.g. “but” and “alright” (Hirschberg and Litman, 1994), and adjectives with positive or negative polarity (Hatzivassiloglou and McKeown, 1997). We incorporated a set of duration features that were described in the literature as good predictors of agreements: utterance length distinguishes agreement from disagreement, the latter tending to be longer since the speaker elaborates more on the reasons and circumstances of her disagreement than for an agreement (Cohen, 2002). Duration is also a good predictor of backchannels, since they tend to be quite short. Finally, a fair amount of silence and filled pauses is sometimes an indicator of disagreement, since it is a dispreferred response in most social contexts and can be associated with hesitation (Pomerantz, 1984).

3.4.2.3 Empirical Study

We first performed several empirical analyses in order to determine to what extent contextual information helps in discriminating between agreement and disagreement. By integrating the inter-

<p><u>Structural features:</u></p> <ul style="list-style-type: none">· is the previous/next utterance of the same speaker?· is the previous/next utterance involving the same B speaker? <p><u>Duration features:</u></p> <ul style="list-style-type: none">· duration of the utterance· seconds of overlap with any other speaker· seconds of silence during the utterance· speech rate in the utterance <p><u>Lexical features:</u></p> <ul style="list-style-type: none">· number of words in the utterance· number of content words in the utterance· perplexity of the utterance with respect to four language models, one for each class· first and last word of the utterance· number of instances of adjectives with positive polarity (Hatzivassiloglou and McKeown, 1997)· idem, with adjectives of negative polarity· number of instances in the utterance of each cue phrase and agreement/disagreement token listed in (Hirschberg and Litman, 1994; Cohen, 2002)
--

Table 3.12: Features for agreement and disagreement classification.

	$p(y_t y_{t-1})$	$p(y_t^{B \rightarrow A} \text{pred}_{B \rightarrow A}(y_t^{B \rightarrow A}))$	$p(y_t^{B \rightarrow A} \text{pred}_{A \rightarrow B}(y_t^{B \rightarrow A}))$
$p(\text{AGREE} \text{AGREE})$.213	.250	.175
$p(\text{DISAGREE} \text{AGREE})$.073	.107	.088
$p(\text{AGREE} \text{DISAGREE})$.139	.087	.234
$p(\text{DISAGREE} \text{DISAGREE})$.209	.261	.128

Table 3.13: Empirical distributions for some agreement and disagreement dependencies (previous tag, consistency, and symmetrical). The empirical prior for the agreement tag is $p(\text{AGREE}) = .188$ and the disagreement tag is $p(\text{DISAGREE}) = .106$.

pretation of the pragmatic function of an utterance into a wider context, we aim to detect cases of mismatch between a correct pragmatic interpretation and the surface form of the utterance, e.g., the case of weak or “empty” agreement, which has some properties of downright agreement (lexical items of positive polarity), but which is commonly considered to be a disagreement (Pomerantz, 1984).

While the actual classification problem incorporates four classes, the BACKCHANNEL class is ignored here to make the empirical study easier to interpret. We tested the validity of three types of long-distance dependencies, which make the following assumptions:

1. previous-tag dependency:

a tag y_t is influenced by its predecessor y_{t-1}

2. consistency dependency:

a tag $y_t^{B \rightarrow A}$ is influenced by $\text{pred}_{B \rightarrow A}(y_t^{B \rightarrow A})$, the most recent tag of the same speaker addressing the same listener; for example, it might be reasonable to assume that if speaker B disagrees with A, B is likely to disagree with A in his or her next speech addressing A.

3. symmetrical dependency:

a tag $y_t^{B \rightarrow A}$ is influenced by $\text{pred}_{A \rightarrow B}(y_t^{B \rightarrow A})$; the assumption is that B’s decision to agree or disagree with A may correlate with the polarity (agreement or disagreement) of what A said last to B.

Table 3.13 presents the results of our empirical evaluation of the three assumptions. For comparison, the prior distribution of classes is the following: 18.8% are agreements, 10.6% disagreements,

and 70.6% other. The table yields some interesting results, showing quite significant variations in empirical distributions when they are conditioned on various types of contextual information. We can see for example, that the proportion of agreements and disagreements (respectively 18.8% and 10.6%) changes to 13.9% and 20.9% respectively when we restrict the counts to utterances that are preceded by a DISAGREE. Similarly, that distribution changes to 21.3% and 7.3% when the previous tag is an AGREE. The change is even more noticeable between probabilities $p(y_t)$ and $p(y_t^{B \rightarrow A} | \text{pred}_{B \rightarrow A}(y_t^{B \rightarrow A}))$. In 26.1% of the cases where a given speaker B disagrees with A, he or she will continue to disagree in the next exchange involving the same speaker and the same listener. Similarly with the same probability distribution, a tendency to agree is confirmed in 25% of the cases. The results in the last column are quite different from the two preceding ones. While agreements in response to agreements ($p(\text{AGREE} | \text{AGREE}) = .175$) are slightly less probable than agreements without conditioning on any previous tag ($p(\text{AGREE}) = .188$), the probability of an agreement produced in response to a disagreement is quite high (with 23.4%), even higher than the proportion of agreements in the entire data (18.8%). This last result would arguably be quite different with more quarrelsome meeting participants.

The empirical distributions described here are admittedly dependent on the meeting genre and particularly speaker personalities. Nonetheless, we believe this model can as well be used to capture salient interactional patterns specific to meetings with different social dynamics.

3.4.2.4 Classifier

We use a conditional dynamic Bayesian network (DBN) model, which is many ways similar to maximum entropy Markov models (McCallum et al., 2000), except that it contains non-local dependencies. This conditional model has a directed graphical representation, as opposed to the undirected graphical models used in generic content selection.

More specifically, this DBN is parameterized as a log-linear model that estimates a probability of each label y_t given a set of observations \mathbf{x} and the three contextual tags \mathbf{d}_t described earlier, where:

$$\mathbf{d}_t = (y_{t-1}, \text{pred}_{B \rightarrow A}(y_t^{B \rightarrow A}), \text{pred}_{A \rightarrow B}(y_t^{B \rightarrow A}))$$

Given J feature functions $f_j(y_t, \mathbf{d}_t, \mathbf{x})$ and J model parameters $\Lambda = (\lambda_1, \dots, \lambda_J)$, the probability

Feature sets	Accuracy
(Hillard et al., 2003)	82%
Lexical	84.95%
Structural and durational	71.23%
All	85.62%
Conditional DBN	86.92%

Table 3.14: Agreement/disagreement classification accuracies with 3 classes: AGREE, DISAGREE, and OTHER.

defined by the model is:

$$p_{\Lambda}(y_t | \mathbf{d}_t, \mathbf{x}) \propto \exp \left(\sum_{j=1}^J \lambda_j f_j(y_t, \mathbf{d}_t, \mathbf{x}) \right)$$

We used the generalized iterative scaling algorithm (Darroch and Ratcliff, 1972; Ratnaparkhi, 1998) as implemented in YASMET (Och, 2003) to train this model. At test time, we search for the most probable sequence by performing a left-to-right decoding using a beam search. The algorithm is exactly the same as the one described in (Ratnaparkhi, 1996) to find the most probable part-of-speech sequence.

3.4.2.5 Results

We had 8,135 utterances available for training and testing, and performed two sets of experiments to evaluate the performance of our classifiers. In the first set of experiments, we reproduced the experimental setting of (Hillard et al., 2003), a three-way classification (BACKCHANNEL and OTHER are merged), and we used the same training set as (Hillard et al., 2003).

Performance is reported in Table 3.14. For the three first rows, we present the results of a log-linear model that does not incorporate any inter-sentential dependency, and the last row provides the accuracy of the conditional DBN classifier described in the previous subsection. In the second set of experiments, we aimed at reducing the expected variance of our experimental results and performed N -fold cross-validation in a four-way classification task. Table 3.15 summarizes the performance of our classifier with the different feature sets in this classification task, distinguishing the case where the three types of contextual labels (e.g., previous tag) are available during decoding from the case

Feature sets	Unstructured model	Conditional DBN
Lexical	82.62%	83.54%
Structural, durational	58.86%	62.10%
All	83.11%	84.07%

Table 3.15: Agreement/disagreement classification accuracies with 4 classes: AGREE, DISAGREE, BACKCHANNEL and OTHER.

where they are not.

First, the analysis of our results shows that with our three local feature sets only, we obtain substantially better results than (Hillard et al., 2003). This might be due to some additional features the latter work didn't exploit (e.g., structural features and adjective polarity), and to the fact that the learning algorithm used in our experiments might be more accurate than decision trees in the given task. Second, the table corroborates the findings of (Hillard et al., 2003) that lexical information represents the most helpful feature set. Finally, we observe that by incorporating label-dependency features representing pragmatic influences, we further improve the performance (about 1% in Table 3.15). This seems to indicate that modeling label dependencies in our classification problem is useful.

3.5 Conclusion

In this chapter, we presented a CRF model incorporating long-distance dependencies derived from the automatic analysis of participant interaction was shown to outperform linear-chain CRF models, despite the incorporation in all cases of the same competitive set of predictors resulting from cross-validated feature selection. Compared to a 1st-order CRF model, the skip-chain CRF model provides a 1.4% increase in F_1 -score. Our best performing system reaches 92.9% of human performance, and scales relatively well on automatic speech recognition output.

We have also shown how identification of dialogic pairs can help representing pragmatic dependencies between agreement and disagreement labels. Classifiers implementing graphical models with these dependencies yielded a substantial improvement (1.3% to reach a 86.9% accuracy in three-way classification).

Chapter 4

Syntax-Directed Sentence Compression

Approaches to summarization that rely exclusively on sentence extraction may perform reasonably well with written texts, but they often fail to produce effective summaries when applied to spoken documents, and, in particular, multi-party speech. As can be seen in Figure 4.1, sentences in conversational speech tend to be fragmentary and disfluent, and automatic speech recognition (ASR) introduces many errors. Lack of fluency and readability is particularly severe in meetings, because of their informal nature and high degree of overlapping speech. Furthermore, ungrammatical sentences occur relatively frequently since participants are generally not professional speakers, as opposed to other speech domains (e.g., broadcast news).

A sentence such as the one included in Figure 4.1 is clearly far too disfluent and verbose to be really useful for the purpose of summarization, which is to provide a quick overview of each meeting. To improve the readability of extractive summaries, we empower our summarization system with the ability to delete sub-sentential textual units, such as words, phrases, and clauses, a task that is generally termed *sentence compression* (Jing, 2000; Knight and Marcu, 2000; Daumé III and Marcu, 2002; Reizler et al., 2003; Dorr et al., 2003; Turner and Charniak, 2005).

While sentence compression has been primarily applied to written texts, we believe meeting transcriptions stand to benefit from sentence compression to a much greater extent than well-formed texts. In meetings, we have many opportunities to remove unnecessary material from transcriptions, e.g., by correcting self-repairs (Shriberg, 1994), deleting fillers (e.g., “uh” and “um”), parentheticals (e.g., “I mean”), hedges (e.g., “I think” and “kind of”), general extenders (e.g., “and so forth” and “or something like that”) (Overstreet, 1999), tag-questions (e.g., “isn’t it?”), and “semantically

<p>(Human) Well, I ju- I was just thinking, with reference to uh, things that have - that bear on the content or the status relations, would be the things - without being exhaustive, by any means, but just like I said, if there's a k- a certain topic that comes up in the meeting, and that knowing their relationship will clarify it, or [pause] if there's a certain dynamic that comes up - so, I mean, a person is asked a whole bunch of questions, more than you'd usually think they'd be asked, and it turns out it's because he's being prepared for a job interview or something like that, then it's useful to know that - that relationship.</p>
<p>(ASR) But I was just thinking with reference to of things that have that they're on the content or the status relations would be the things without being exhausted by any means you just like a say that there's a a certain topic it comes up in the meeting and that knowing their relationship clarified or if there's certain dynamic the comes up so I mean person is asked a whole bunch of questions more than you usually think they'd be asked it turns out is because it's being prepared for job interview or something like at that is useful to know that that relationship</p>

Table 4.1: A short excerpt of an extractive summary illustrating the inappropriateness of relying on sentence extraction alone in meeting summarization. The upper summary (human) is extracted from a human transcript, while the lower one contains the corresponding output of a state-of-the-art automatic speech recognition (ASR) system. This excerpt is taken from one of the 75 human extractive summaries created at University of Edinburgh (Carletta et al., 2003) (see Section 2 for more details).

empty” prefabricated phrases (e.g., “let’s face it” and “for some reason”) (Kjellmer, 1991; Nattinger and DeCarrico, 1992; Hughes, 1996). Furthermore, conversational speech contains a high incidence of markers of interpersonal dynamics, to signal such activities as taking the floor (e.g., “well” and “but”), holding the floor (e.g., “and what else”), verifying that the speaker is being understood (e.g., “you know what I mean?”), and starting a new topic (e.g., “so”). In many cases, such terms do not alter the propositional content of the sentences in which they appear, and may as well be deleted (EARS, 2003). As in sentence compression for texts, it is of course also desirable to go beyond small edits, and perform summarization at the subsentential level by removing the least informative clauses and phrases.

In this chapter, our goal is to render errorful and disfluent transcript utterances into concise sentences that are maximally readable, removing words or phrases that are not necessary in the generated output. Given the need to preserve grammaticality, most previous work moved away from purely word-based or string-based approaches¹ to instead employ syntactic knowledge to remove constituents from phrase-structure trees (Jing, 2000; Knight and Marcu, 2000; Dorr et al., 2003), a transformation that can often be expressed using *synchronous rewriting* and *synchronous grammars* (Lewis and Stearns, 1968; Aho and Ullman, 1969; Shieber and Schabes, 1990). Synchronous grammars are versatile formalisms for representing the recursive transformation of one language into another, and have found many usages in natural language processing, including natural language generation (Shieber and Schabes, 1990), machine translation (Abeillé et al., 1990; Wu, 1997; Yamada and Knight, 2001; Melamed et al., 2004; Galley et al., 2004a; Chiang, 2005), semantic parsing (Wong and Mooney, 2006), paraphrasing (Madnani et al., 2007), and theoretical syntax (Abeillé, 1994; Rambow and Satta, 1996).

Previous approaches to synchronous rewriting for sentence compression relied almost exclusively on *synchronous context-free grammars* (SCFG) (Lewis and Stearns, 1968; Aho and Ullman, 1969), which extend context-free grammars (CFG) in that their productions generate two right-hand sides (RHS) instead of one. Algorithms for extracting and weighting SCFG productions using parallel texts of document-abstract pairs have been proposed (Knight and Marcu, 2000; Turner and

¹Previous work that aimed to generate very short telegraphic sentences, such as headlines, is a notable exception (Witbrock and Mittal, 1999; Banko et al., 2000; Zajic et al., 2002), which arguably would benefit less from syntax-directed approaches than other compression tasks.

Charniak, 2005). However, while computationally attractive and linguistically plausible, these previous approaches to sentence compression based on synchronous context-free grammars suffer from two notable weaknesses:

- **Unsuitable syntax tree representation:** Compression rules are acquired from syntax trees such as Penn Treebank (PTB) trees (Marcus et al., 1994), whose labels are not specific enough to appropriately distinguish constituents that are syntactically or semantically optional. First, it is widely acknowledged that the category symbols of the PTB are too coarse to adequately render context-free productions independent of their contexts. Second, these labels mainly fit linguistic descriptive purposes, and are not meant to be optimal with regard to any particular constituent classification task such as in sentence compression. Third, as noted previously in syntactic parsing, many context-free rewrite rules in the PTB are seen only once, and relative frequency estimation on raw context-free productions leads to severe sparseness issues (Charniak, 1996; Collins, 1999; Klein and Manning, 2003). As we will see later in this chapter, this issue represents a major bottleneck in sentence compression, since relative frequency estimation on raw SCFG productions is even more brittle than for CFG productions, and this problem is exacerbated by the fact that sentence compression has only scarce training material available.
- **Unreasonable assumptions about summarization data:** While it is convenient to address the subsentential summarization task as sentence compression—i.e., requiring that the target sentence be a substring of the source—only few sentence pairs in summarization training corpora abide to this restriction. In practice, a large percentage of target sentences contain word substitutions and insertions, such as paraphrases and replacements of noun phrases with pronouns. In a large corpus commonly used in sentence compression (Ziff-Davis), summary sentences that are subsequences of full-document sentences account for only 1.75% of the entire summarization data. However, previous approaches to sentence compression discard any sentence pair that includes insertions and deletions, since the transformation from one sentence to the other cannot normally be represented in these models.

Our research objective in this chapter is to address these two serious limitations as we present a new statistical sentence compression model that offers solutions to both. Since previous syn-

chronous grammar approaches to sentence compression were applied exclusively to written texts, we capitalize on robust techniques to make our syntax-directed sentence compression approach applicable to noisy and disfluent input such as conversational speech. The main contributions of this chapter include:

- **Robust parameterization of compression probabilistic SCFG (Section 4.2.2):** To alleviate data sparseness issues pertaining to previous generative models for sentence compression (Knight and Marcu, 2000; Turner and Charniak, 2005), we present Markov SCFG grammars for sentence compression (Section 4.2.2). In the same way Markov grammars in syntactic parsing (Magerman, 1995; Collins, 1996) extend plain PCFG grammars (Booth and Thompson, 1973), Markov SCFG grammars extend probabilistic SCFG grammars by introducing a flexible number of conditional independences between right-hand side constituents, and thus yield more robust probability estimates. Markov SCFG provides several other benefits, including the ability to condition constituent deletions on a flexible amount of additional syntactic context (e.g., ancestors in parse trees). Assuming robust smoothing techniques are employed, they also enable us to incorporate lexical conditionings. Lexical information is especially useful to accurately determine whether a syntactic constituent (e.g., “like any other product”) is grammatically optional—as in “she sold it like any other product”—or mandatory—as in “it looks like any other product”. We show that our compression model is successfully applicable to both written texts and conversational speech, and that it significantly outperforms a state-of-the-art syntax model (Knight and Marcu, 2000) that does not incorporate Markovization and lexicalization, and produces sentences that were judged more grammatical.
- **Better modeling of revision bi-texts (Section 4.2.1):** Another part of our effort focuses on robust techniques for extracting SCFGs from parallel data. We improve upon (Knight and Marcu, 2000), who could only exploit 1.75% of a standard document-abstract corpus (Ziff-Davis) because of their restriction to sentence pairs that include no substitutions or insertions in target strings. To alleviate this restriction, we present a robust approach to tree-to-tree alignment between arbitrary document-abstract parallel corpora. By accounting for sentence pairs with substitutions in addition to deletions, we reach a retention rate of more than 25% of

the Ziff-Davis data, and show that lexicalized SCFG models greatly benefited from this extra training material.

- **Minimum error-rate training (Section 4.2.3):** We incorporate additional models and features to improve compression quality—such as n -gram language models built from written texts and speech data, tf.idf scores, sentence length features, and confidence scores provided by the speech recognizer—and tune the scaling factors of these different models as to directly optimize automatic evaluation metric scores, such as word error rate (WER).

This chapter is organized as follows. In Section 4.1, we present our experiments in producing syntax trees from speech utterances, which is prerequisite of our approach. In Section 4.2, we overview synchronous context-free grammars for sentence compression. In Section 4.2.1, we describe approaches to extracting synchronous grammars from parallel corpora. In Sections 4.2.2 and 4.2.3, we present generative and discriminative models for sentence compression, and in Section 4.2.4, we describe a decoder for syntax-directed sentence compression, and for producing document compressions that satisfy any pre-defined compression ratio. In Section 4.4, we present experimental results for these different models on various speech and text corpora. Finally, we relate our approach to previous work in Section 4.5 and conclude in Section 4.6.

4.1 Preprocessing

Our architecture for compressing conversational speech comprises three stages: first, the input sentence is stripped of *edited words* occurring in self repairs (e.g., “why didn’t he?” in the utterance “why didn’t he? why didn’t she stay home?”). Second, the remaining words are parsed to construct a phrase-structure tree using a standard statistical parser (Charniak, 2000) (for reasons explained later, it is often advantageous to extract an n -best list from the syntactic parser, and not just the best hypothesis). Finally, each phrase structure tree is trimmed down to a simpler tree using SCFG productions as presented later in this section.²

Our technique for identifying and removing edited words is presented in the next chapter. Our motivation for removing edited words before parsing arises from the fact that repairs generally

²Note that in the case of written texts, we parsed Ziff-Davis sentences using the parser described in (Charniak and Johnson, 2005).

decrease parsing accuracy, and that techniques that exploit shallow phenomena such as word repetitions achieve very good performance in identifying edited words (Charniak and Johnson, 2001). Since this chapter is primarily concerned with context-free formalisms for sentence compression, we will assume for the remainder of this section that input utterances do not contain edited words. In Section 5.2, we present a sequence model for disfluency removal that achieves performance that is close to state-of-the-art. We also evaluate the impact of automatic edited-word removal on compression performance, and show that its automation seldom decreases accuracies.

The remaining preprocessing step is syntactic parsing, which is presented in the next subsection.

4.1.1 Syntactic Parsing of Conversational Speech

Our parsing problem is to map any input sentence to a corresponding phrase-structure tree. In this work, we rely on a statistical parser that was trained on the Switchboard section (i.e., conversational speech) of the Penn treebank (Marcus et al., 1994). In training this parser on Switchboard, our assumption is that it should scale relatively well on meetings, because both genres are conversational and spontaneous in nature. For convenience to the readers who are not familiar with the Penn Treebank annotation, we included in Appendix A a list of all morpho-syntactic tags of the Penn Treebank along with descriptions and examples, since many of these tags appear in examples of this chapter. To train and test this parser, we removed all edited words from training and test treebank trees,³ and split our data into training, development, and test set following (Charniak and Johnson, 2001): training data consisted of all `sw[23]*.dps` files, development data consisted of all `sw4[5-9]*.dps` files and test consisted of all `sw4[0-1]*.dps` files.

In Table 4.2, we report our own experiments in parsing the test section of Switchboard corpus. As is generally the case in syntactic parsing, we evaluated parsing accuracy with the PARSEVAL measure (Black et al., 1991), which measures bracketing precision and recall. Precision and recall are defined as the number of correct constituents in the proposed parse respectively divided by the number of constituents in the proposed parse and the reference parse. Since our sentence compression models were evaluated with different inputs ranging from properly punctuated and cased human transcriptions to automatic speech recognition output, we trained two parsing models. The

³In the Switchboard annotation, constituents of consecutive edited words were assigned a special EDITED tag. Hence, it is possible to remove edited words without fundamentally changing parse tree structures.

TREEBANK FORMAT	≤ 100 words		≤ 40 words	
	LP	LR	LP	LR
No edited words (mixed case, punctuation)	88.24%	88.09%	89.21%	89.12%
No edited words (ASR-like)	87.76%	87.81%	88.64%	88.68%
Original treebank (mixed case, punctuation)	85.30%	84.17%	86.80%	86.66%

Table 4.2: Parsing results on Sections 40-41 of the Switchboard section of the Penn Treebank. **LP** and **LR** stand for labeled precision and recall. All experiments are performed with the parser described in (Charniak, 2000).

first one was trained with punctuation and mixed case, and the second one with ASR-like transcription (i.e., all tokens were lower-cased, and punctuation symbols were removed from parse trees). Overall, it appears from these results that lack of punctuation seldom affects parsing accuracy.

For comparison purposes, we also trained a model with the original treebank including its edited words, whose performance is shown in the last row of Table 4.2.⁴ This model is not used in our experiments on sentence compression, but it enables us to see that failures to remove edited words before parsing significantly decreases performance.

While results seem quite promising, we would also like to emphasize that any error the syntactic parser may produce does not necessarily imply a lower accuracy down the pipeline. The end product of our sentence compression system is always a sequence of words, and parse trees are akin to latent variables in our particular application. While incorrect parses make it more difficult to apply sensible compressions, many parsing errors do not affect compression evaluation. For instance, if a constituent is deleted, its internal structure—no matter how incorrect—cannot influence sentence compression evaluation since its yield does not appear in the output sentence; conversely, if a constituent is spared any deletion, its internal structure has no influence either, since all possible subtrees would have the same yield.

⁴We just removed empty elements **RM**, **RS**, and **IP**, which respectively indicate the start and end of a repair, and the location of the interruption point (Shriberg, 1994). As other empty elements, these tags cannot be identified by the parser used in our experiments (Charniak et al., 1998).

4.2 Synchronous Grammars for Sentence Compression

A significant body of previous work takes syntax-directed approaches to sentence compression, by typically assessing the merits of a phrase deletion based on its syntactic category and perhaps other elements of the tree structure surrounding the constituent.

One successful syntax-driven approach (Knight and Marcu, 2000, henceforth K&M) relies on synchronous context-free grammars (SCFG), otherwise also known as *pushdown transducers*, *syntax directed transduction grammars* and *syntax directed translation schemata* (Lewis and Stearns, 1968; Aho and Ullman, 1969). SCFGs can be defined as context-free grammars (CFGs) whose productions have two right-hand side strings instead of one, namely *source* and *target* right-hand side. In the case of sentence compression, we restrict the target side to be a sub-sequence of the source side (possibly identical), and we will call this restricted grammar a *deletion SCFG*. For instance, a deletion SCFG rule that removes an adverbial phrase (ADVP) between a noun phrase (NP) and a verb phrase (VP) may be written as follows:⁵

$$S \rightarrow \langle \text{NP ADVP VP}, \text{NP VP} \rangle$$

In a sentence compression framework similar to the one presented by K&M, we build SCFGs that are fully trainable from a corpus of document and reduced sentences. Such an approach comprises two subproblems: (1) transform tree pairs into synchronous grammar derivations; (2) based on these derivations, assign probabilities to deletion SCFG productions, and more generally, to compressions produced by such grammars. These two tasks are described respectively in Sections 4.2.1 and 4.2.2.

⁵Productions in SCFGs are generally required to define a one-to-one correspondence between right-hand side constituents, as in $S \rightarrow \langle \text{NP}_1 \text{ NP}_2 \text{ VP}_3, \text{NP}_1 \text{ NP}_2 \text{ VP}_3 \rangle$, where indices specify a recursive relationship between constituents (i.e., a linked pair such as $\langle \text{NP}_1, \text{NP}_1 \rangle$ must be rewritten together, and recursions through, e.g., $\langle \text{NP}_1, \text{NP}_2 \rangle$ are explicitly disallowed). However, as noted in (Chiang, 2006), a grammar with incompletely linked productions (such as the ones in K&M) are weakly equivalent to SCFG by adding productions with empty right-hand sides. In the case of deletion SCFGs, the transformed grammar only contains productions that are order-preserving, as in simple transduction grammars (Lewis and Stearns, 1968). The above ADVP deletion rule would be rewritten $S \rightarrow \langle \text{NP}_1 \overline{\text{ADVP}}_2 \text{ VP}_3, \text{NP}_1 \overline{\text{ADVP}}_2 \text{ VP}_3 \rangle$, where $\overline{\text{ADVP}}$ is a special non-terminal that can only yield an empty target language string, as in $\langle \overline{\text{ADVP}}, \overline{\text{ADVP}} \rangle \Rightarrow \langle \overline{\text{RB}}, \overline{\text{RB}} \rangle \Rightarrow \langle \textit{usually}, \epsilon \rangle$. For simplicity, we will omit these extra productions.

# of subs.	Sentences	SCFG	SCFG-adj
0	823	823	951
≤ 1	2674	2139	2460
≤ 2	5307	3315	3822
≤ 3	8384	4431	5164
≤ 4	11345	5495	6377
≤ 5	13836	6335	7340
≤ 6	15554	6934	8021
≤ 7	16787	7365	8516

Table 4.3: Cumulative number of training sentence pairs and number of SCFG productions acquired from Ziff-Davis (without and with adjunction decompositions), allowing 0, 1, or more substitutions between full and compressed sentences, and any number of deletions.

4.2.1 Extracting Synchronous Grammars

We now describe methods to train SCFG models from sentence pairs. Given a tree pair (\mathbf{f}, \mathbf{c}) , whose respective parses $(\pi_{\mathbf{f}}, \pi_{\mathbf{c}})$ were generated by a parser, the goal is to collect SCFG production counts required by relative frequency estimates. The approach taken by K&M is to analyze both trees and count the occurrence of a deletion SCFG rule $l \rightarrow \langle \alpha_f, \alpha_c \rangle$ whenever two nodes n_f and n_c in the two trees are “deemed to correspond”, i.e., categories of n_f and n_c are the same, the CFG production α_c rooted at n_c is a sub-sequence of the production α_f rooted at n_f , and each pair of matching right-hand constituents in α_f and α_c yields a string pair whose target side is a subset of the source side. This leads to a quite restricted number of different productions on the subset of the Ziff-Davis corpus that comprises only sentence-compression pairs: 823 different productions were extracted, 593 of which appear only once. Thus, this first approach has serious limitations; the assumption that sentence compression appropriately models human abstractive data is particularly problematic. This considerably limits the amount of training data that can be exploited in Ziff-Davis, and this makes it very difficult to train lexicalized models.

An approach to slightly loosen this assumption is to consider document-abstract sentence pairs in which the condensed version contains one or more substitutions or insertions. Consider for example the tree pair in Figure 4.1: the two sentences are syntactically very close, but the substitution

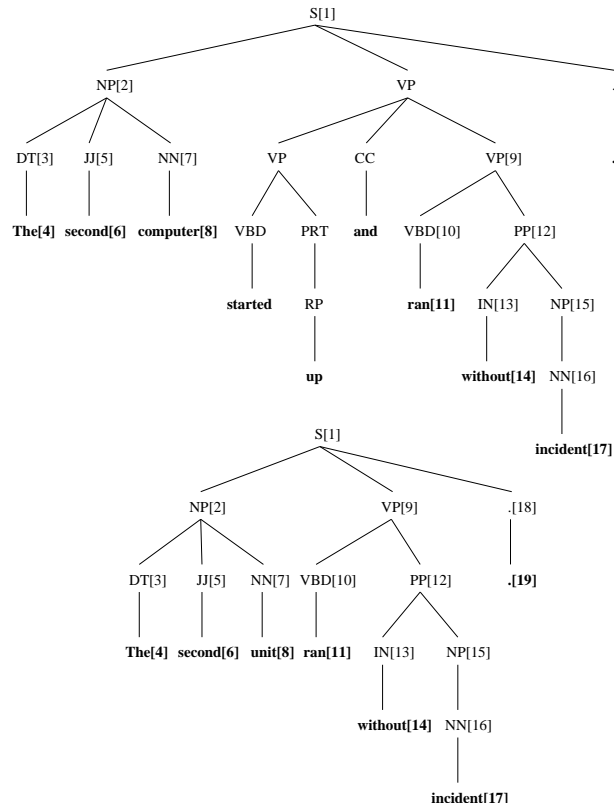


Figure 4.1: A full sentence tree (top) and a summary sentence tree (bottom) with a minor rewording between the two. While the latter sentence is not a compression of the former, it can still be used to gather statistics to train a sentence compression system, e.g., to learn the reduction of a VP coordination.

of “computer” with “unit” makes this sentence pair unusable in the framework presented in K&M. Arguably, there should be ways to exploit abstract sentences that are slightly reworded in addition to being compressed. To use sentence pairs with insertions and substitutions, we must find a way to align tree pairs in order to identify SCFG productions. More specifically, we need to define a constituent alignment between the paired abstract and document sentences, which determine how the two trees are synchronized in a derivation. Obtaining this alignment is no trivial matter as the number of non-deleting edits increases. To address this problem, we synchronized tree pairs by finding the constituent alignment that minimizes the edit distance between the two trees (Tai, 1979),⁶

⁶The edit distance between two trees is analogous to the edit distance between two strings. Any word and non-terminal insertion, substitution, and deletion has a cost of one. Note that node insertions and deletions may increase

i.e., minimize the number of terminal and non-terminal insertions, substitutions and deletions. This minimization problem is known to be NP hard, so we used an approximation algorithm (Zhang and Shasha, 1989) that runs in polynomial time. While criteria other than minimum tree edit distance may be effective, we found—after manual inspections of alignments between sentences with less than five non-deleting edits—that this method generally produces good alignments.⁷ Once a constituent alignment is available, it is then trivial to identify all deletion SCFG rules available in the synchronized tree pair, e.g., $NP \rightarrow \langle DT JJ NN, DT JJ NN \rangle$ in the Figure 4.1, which displays the tree-to-tree alignment produced by the algorithm described in (Zhang and Shasha, 1989).

We also exploited more general tree productions known as synchronous tree substitution grammar (STSG) rules. For instance, the STSG rule rooted at S can be decomposed into two SCFG productions if we allow unary rules such as $VP \rightarrow VP$ to be freely added to the compressed tree. More specifically, we decompose any STSG rule that has in its target (compressed) RHS a single context free production, and that contains in its source (full) RHS a single context free production adjoined with any number of tree adjoining grammar (TAG) auxiliary trees (Joshi et al., 1975). The figure displays an example where such a decomposition is applicable with an initial tree $S \rightarrow NP VP$ and an adjoined auxiliary tree $VP \rightarrow VP CC VP$.⁸ We found this approach quite effective, since many common compressions, such as coordination reductions or the removal of any NP within an NP, are generally missed by the extraction procedure of K&M. Statistics regarding the number of rules extracted from Ziff-Davis are displayed in Table 4.3.

4.2.2 A Generative Model

The overall goal of a sentence compression system is to transform a given input sentence f into a concise and grammatical sentence $c \in C$, which is a sub-sequence of f . Similarly to K&M

or reduce the depth of the tree. For instance, transforming the tree $(NP (NN authorization))$ into $(NP (NP (NN authorization)) (PP [...]))$ yields a cost of 1 for adding an NP node (plus the cost of inserting the PP subtree).

⁷Other methods to align trees have been used in machine translation, but the fact that we are aligning trees of the same language makes minimum-edit-distance tree alignments an effective solution in the case of sentence compression.

⁸To determine whether a given one-level tree is an auxiliary, we simply check the following properties: all its leaves but one (the “foot node”) must be nodes attached to deleted subtrees (e.g., VP and CC in the figure), and the foot node (VP[9]) must have the same syntactic category as the root node.

and many successful syntactic parsers (Collins, 1999; Charniak, 2000; Klein and Manning, 2003), our sentence compression system is *generative*, and attempts to find the optimal compression $\hat{\mathbf{c}}$ by estimating the following function:⁹

$$\hat{\mathbf{c}} = \arg \max_{\mathbf{c} \in \mathbf{C}} \left\{ p(\mathbf{c}|\mathbf{f}) \right\} = \arg \max_{\mathbf{c} \in \mathbf{C}} \left\{ p(\mathbf{f}, \mathbf{c}) \right\} \quad (4.1)$$

If $\tau(\mathbf{f}, \mathbf{c})$ is the set of all synchronized tree pairs that yield (\mathbf{f}, \mathbf{c}) according to some underlying SCFG, we can estimate the probability of the sentence pair using:

$$p(\mathbf{f}, \mathbf{c}) = \sum_{(\pi_{\mathbf{f}}, \pi_{\mathbf{c}}) \in \tau(\mathbf{f}, \mathbf{c})} P(\pi_{\mathbf{f}}, \pi_{\mathbf{c}}) \quad (4.2)$$

We note that, in practice (and as in K&M), Equation 4.2 is often approximated by restricting $\tau(\mathbf{f}, \mathbf{c})$ to a unique full tree $\hat{\pi}_{\mathbf{f}}$, the best hypothesis of an off-the-shelf syntactic parser. This implies that each possible compression \mathbf{c} is the target-side yield of at most one SCFG derivation.

As in standard PCFG history-based models, the probability of the entire structure (Equation 4.2) factorizes into probabilities of grammar productions. If θ is a derivation $\theta = r^1 \circ \dots \circ r^j \circ \dots \circ r^J$, where r^j denotes the SCFG rule $l^j \rightarrow \langle \alpha_f^j, \alpha_c^j \rangle$, we get:

$$p(\pi_{\mathbf{f}}, \pi_{\mathbf{c}}) = \prod_{j=1}^J p(\alpha_f^j, \alpha_c^j | l^j) \quad (4.3)$$

The question we will now address is how to estimate the probability $p(\alpha_f^j, \alpha_c^j | l^j)$ of each SCFG production.

4.2.2.1 Lexicalized Head-Driven Markov Model

A main issue in our approach is to reliably estimate productions of deletion SCFGs. In a sentence compression framework as the one presented by K&M, we use aligned trees of the form of the Penn Treebank (PTB) (Marcus et al., 1994) to acquire and score SCFG productions. However, the use of the PTB structure faces many challenges also encountered in probabilistic parsing.

Firstly, PTB tree structures are relatively flat, particularly within noun phrases. For instance, adjective phrases (ADJP)—which are generally good candidates for deletions—appear in 90 different

⁹In their noisy-channel approach, K&M further break down $p(\mathbf{c}, \mathbf{f})$ into $p(\mathbf{f}|\mathbf{c}) \cdot p(\mathbf{c})$, which we refrain from doing for reasons that will become obvious later.

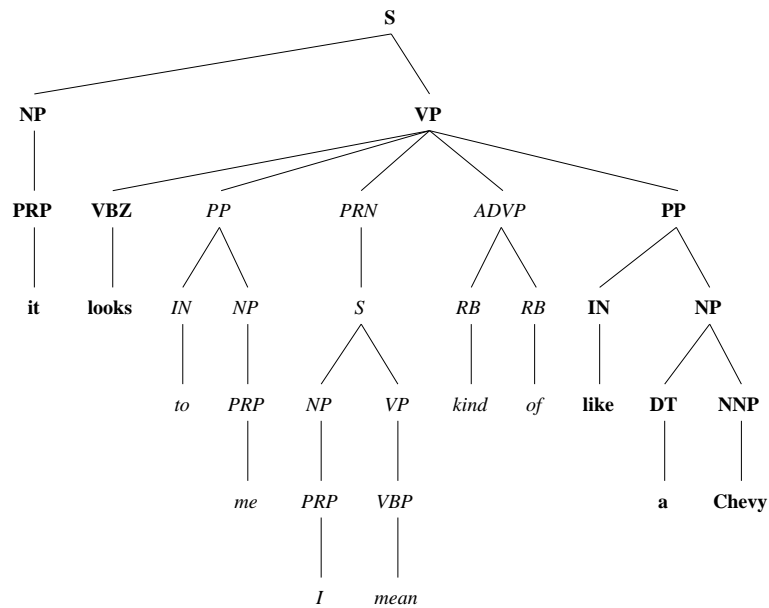


Figure 4.2: Syntax tree of the disfluent Switchboard sentence “I mean, it looks to me, I mean, kind of like a Chevy”, which is compressed into “it looks like a Chevy”. Constituents and subtrees that must be deleted appear in *italic*.

NP-rooted SCFG productions in Ziff-Davis, 61 of which appear only once, e.g., $NP \rightarrow \langle DT\ ADJP\ JJ\ NN\ NN, DT\ JJ\ NN\ NN \rangle$. While it may seem advantageous to maintain many constituents within the same domain of locality of an SCFG production, as we may hope to exploit its large syntactic context to condition deletions more accurately, the sparsity of such productions make them poor candidates for relative frequency estimation, especially in a task with limited quantities of training material. Indeed, the Ziff-Davis corpus (of 1,087 tree pairs) contains only 951 SCFG productions, 593 of which appearing only once.

Secondly, syntactic categories in the PTB are particularly coarse grained, and lead to many incorrect context-free assumptions. Some important distinctions, such as between arguments and adjuncts, are beyond the scope of the PTB annotation, and it is often difficult to determine out of context whether a given constituent can safely be deleted from a right-hand side.

One first type of annotation that can effectively be added to each syntactic category is its lexical head and its part-of-speech (POS), following work in syntactic parsing.¹⁰ This type of annotation is

¹⁰The head is the most important word of a constituent and the one that determines its syntactic category. For instance,

particular beneficial in the case of prepositional phrases (PP), which may be either complements or adjuncts. As in the case of Figure 4.2, knowing that the PP headed by “like” appears in a VP headed by “looks” helps us to determine that the PP is a mandatory complement of this verb, and that it should presumably not be deleted. Conversely, the PP headed by “to” modifying the same verb is an adjunct, and can safely be deleted if unimportant. Also, as discussed in (Collins, 1996), POS annotation can be useful as a means of backing off to more frequently occurring head-modifier POS occurrences (e.g., VBZ-IN) when specific bilexical co-occurrences are sparsely seen (e.g., “looks-like”). At a lower level, lexicalization is clearly desirable for pre-terminals. Indeed, current SCFG models such as K&M have no direct way of preventing highly improbable single word removals, such as deletions of adverbs “never” or “nowhere”, which may turn a negative statement into a positive one.¹¹

A second type of annotation that can be added to syntactic categories is the so-called *parent annotation* (Johnson, 1998), which was effectively used in syntactic parsing to break unreasonable context-free assumptions. For instance, a PP with a VP parent is marked as PP[^]VP. It is reasonable to assume that constituents deep inside a PP have more chances to be removed than otherwise expected, and one may seek to increase the amount of vertical context that is available for conditioning each constituent deletion.

To achieve the above desiderata for better SCFG probability estimates—i.e., reduce the amount of sister annotation within each SCFG production, by conditioning deletions on a context smaller than an entire right-hand side, and at the same time increase the amount of ancestor and descendent annotation through parent (or ancestor) annotation and lexicalization—we follow the approach of (Magerman, 1995; Collins, 1996; Charniak, 2000; Klein and Manning, 2003), i.e., factorize n -ary grammar productions into products of n right-hand side probabilities, a technique sometimes called grammar *Markovization* (Klein and Manning, 2003).

Markovization is generally head-driven, i.e., reflects a decomposition centered around the head

the head of the noun phrase “the same thing” is the noun “thing”. Head-finding rules used in our work are taken from (Collins, 1999, Appendix A).

¹¹K&M incorporate lexical probabilities through n -gram models. However, since n -grams cannot naturally model parallel texts, they are thus unlikely to penalize a compression from “they never arrived” to “they arrived”.

of each CFG production:

$$l \rightarrow \Delta L^m \dots L^1 H R^1 \dots R^n \Delta \quad (4.4)$$

where H is the head, L^1, \dots, L^m are the left modifiers, R^1, \dots, R^n are right modifiers, and Δ is a termination symbol needed for accurate probability estimations (e.g., to capture the fact that some constituents are more likely than others to be the right-most constituent); for simplicity, we will ignore Δ in later discussions. For a given SCFG production $l \rightarrow \langle \alpha_f, \alpha_c \rangle$, we ask, given the source RHS α_f that is assumed given (e.g., provided by a syntactic parser), which of its RHS elements are also present in α_c . That is, we write:

$$p(\alpha_c | \alpha_f, l) = p(k_l^m, \dots, k_l^1, k_h, k_r^1, \dots, k_r^n | \alpha_f, l) \quad (4.5)$$

where k_h, k_l^i, k_r^j ('k' for keep) are binary variables that are true if and only if constituents H, L_i, R^j (respectively) of the source RHS α_f are present in the target side α_c . Note that the conditional probability in Equation 4.5 enables us to estimate Equation 4.3, since $p(\alpha_f, \alpha_c | l) = p(\alpha_c | \alpha_f, l) \cdot p(\alpha_f | l)$. We can rely on a state-of-the-art probabilistic parser to effectively compute either $p(\alpha_f | l)$ or the probability of the entire tree π_f , and need not worry about estimating this factor. In the case of sentence compression from the one-best hypothesis of the parser, we can ignore $p(\alpha_f | l)$ altogether, since π_f is the same for all compressions.

We can rewrite Equation 4.5 exactly using a head-driven infinite-horizon Markovization, which is a simple application of the chain rule:

$$\begin{aligned} p(\alpha_c | \alpha_f, l) &= p(k_h | \alpha_f, l) \\ &\cdot \prod_{i=1 \dots m} p(k_l^i | k_l^1, \dots, k_l^{i-1}, k_h, \alpha_f, l) \\ &\cdot \prod_{i=1 \dots n} p(k_r^i | k_r^1, \dots, k_r^{i-1}, k_h, \Lambda, \alpha_f, l) \end{aligned} \quad (4.6)$$

where $\Lambda = (k_l^1, \dots, k_l^m)$ is a term needed by the chain rule. One key issue is to make linguistically plausible assumptions to determine which conditioning variables in the terms should be ignored. Following our discussion in the first part of this section, we may start by making an order- s Markov approximation centered around the head, i.e., we condition each binary variable (e.g., k_r^i) on a context of up to s sister constituents between the current constituent and the head (e.g., (R^{i-s}, \dots, R^i)). In order to incorporate bilinear dependencies between the head and each modifier, we also condition all modifier probabilities on head variables H (and k_h). These assumptions are overall quite

similar to the ones made in Markovized parsing models. If we assume that all other conditioning variables in Equation 4.7 are irrelevant, we write:

$$\begin{aligned}
 p(\alpha_c | \alpha_f, l) &= p_h(k_h | H, l) \\
 &\cdot \prod_{i=1 \dots m} p_l(k_l^i | L^{i-s}, \dots, L^i, k_l^{i-s}, \dots, k_l^{i-1}, H, k_h, l) \\
 &\cdot \prod_{i=1 \dots n} p_r(k_r^i | R^{i-s}, \dots, R^i, k_r^{i-s}, \dots, k_r^{i-1}, H, k_h, l)
 \end{aligned}
 \tag{4.7}$$

Note that it is important to condition predictions on both constituent histories (R^{i-s}, \dots, R^i) and non-deletion histories $(k_r^{i-s}, \dots, k_r^{i-1})$; otherwise we would be unable to perform deletions that must operate jointly, as in the production $S \rightarrow \langle \text{ADVP COMMA NP VP, NP VP} \rangle$ (in which the ADVP should not be deleted without the comma). Without binary histories, we often observed superfluous punctuation symbols and dangling coordinate conjunctions appearing in our outputs.

We now describe in details the kind of lexico-syntactic information used to represent the head constituent H and its modifiers L^i and R^i .

4.2.2.2 Conditioning Variables and Smoothing

We define four meta parameters in our model that determine how much sister, vertical, lexical, and POS annotation is used in maximum likelihood estimation. The question of how much of this annotation is needed for effective sentence is evaluated in details in Section 4.4 (for written texts and accurate speech transcripts) and Section 5.4.2 (for ASR output).

As explained previously, the parameter s conditions predictions on a context of up to s constituents between the head and the modifier. For example, if $s = 0$, the probability of retaining the PP headed by “like” in Figure 4.3 is $p_r(1 | \text{PP}, \dots)$. If $s = 2$, then it is $p_r(1 | \text{PRN, ADVP, PP}, \dots)$. The parameter v works the same way as s , but for vertical annotation. If $v = 1$, the probability is written $p_r(1 | \text{S, PP}, \dots)$.

The parameter l controls how much lexical information is incorporated into the model. If $l = 0$, the model is fully unlexicalized. If $l = 1$, the probability of retaining any constituent is additionally conditioned on its head word, e.g., $p(1 | \text{like}, \dots)$. For modifiers, a value $l = 2$ enables the probability of the modifier to be conditioned on two lexical items, one drawn from the head and the other from the modifier. In this case, the probability of retaining the PP is $p(1 | \text{looks, like}, \dots)$. The parameter t

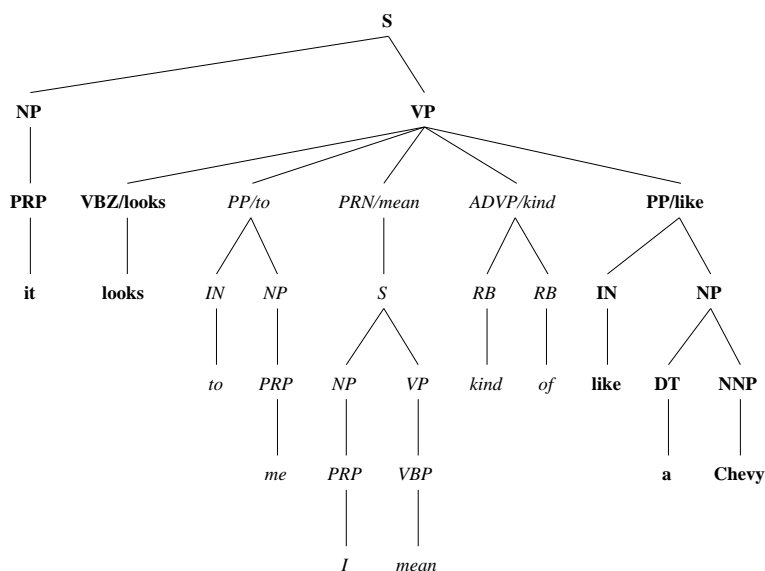


Figure 4.3: To determine that a constituent like “like a Chevy” must be retained in this sentence, it is effective to condition this decision on the pair of head words “looks-like”. Note that it is important to condition on both words, since it would be fine to delete a PP headed by “like” in a different context (e.g., “they arrived like that”).

for POS annotation works the same way as l , but it applies to the POS of head words. For instance, if $t = 2$, the probability is written $p(1|VBZ, IN, \dots)$. The motivation for using POS labels is that it provides a means to back-off to more robust statistics when a word or word pair is rarely seen (Collins, 1996; Eisner, 1996).

Since this probability model constrains a relatively large amount of conditioning variables, maximum likelihood estimation is likely to be plagued by sparsity issues if not appropriately smoothed. Smoothing is crucial for lexicalized models, and particularly for those incorporating bi-lexical head-modifier dependencies. In our work, we rely on Witten-Bell interpolation.¹² Generalized linear interpolation as defined by Bell et al. (Bell et al., 1990) is defined recursively as

$$p_{\text{WB}}(k|v_{1:k}) = \lambda_{v_{1:k}} p_{\text{ML}}(k|v_{1:k}) + (1 - \lambda_{v_{1:k}}) p_{\text{WB}}(k|v_{1:k-1}) \quad (4.8)$$

where $k \in \{1, 0\}$ is our prediction and $v_{1:k}$ is a vector of all conditioning variables in the model.

¹²Note that we relied on the SRI language modeling (SRILM) (Stolcke, 2002) toolkit library. After experimenting with many smoothing techniques implemented in SRILM, we found that Witten-Bell interpolation was the one that worked best with our models.

We used the following order of deletion in the Witten-Bell interpolation of the probability of retaining a head constituent H according to $p_h(k_h^i | \dots)$: lexical head, POS of the lexical head, vertical annotation, and syntactic category. The order of deletion goes from first to last, i.e., the lexical head is deleted first. For the probabilities $p_l(k_l^i | \dots)$ of retaining a constituent L that modifies the head constituent H , we remove in order: lexical head of H , lexical head of L , POS of the lexical head of H , POS of the lexical head of L , sister annotation (which indicates whether or not sisters were deleted or not), k_h (which indicates whether or not the head is deleted), syntactic category of the head, syntactic category of the modifier. We experimented with different orderings, and found this ordering works quite well in practice, and was used in all experiments reported in this paper. However, in future work, we would like to experiment with more powerful back-off techniques, such as generalized parallel backoff (Bilmes and Kirchhoff, 2003) to have more flexibility in combining these conditioning variables.

4.2.3 Discriminative Framework

While it has often been found that generative models perform no worse than discriminative models when the amount of training data is very scarce (Ng and Jordan, 2002; Johnson, 2001), we are nevertheless interested in applying a discriminative framework that can provide a natural means to incorporate various information sources aside from SCFG structures, e.g., n -gram language models and tf.idf scores.¹³

Following many successful applications in other NLP applications, in particular in statistical machine translation (SMT) (Och and Ney, 2003; Tillmann and Zhang, 2006; Liang et al., 2006), we take a minimum error rate training approach to optimizing the parameters of a log-linear model

$$p(\mathbf{c}|\mathbf{f}) \propto \exp\left(\sum_{j=1}^J \lambda_j h_j(\mathbf{c}, \mathbf{f})\right) \quad (4.9)$$

where $h_j(\mathbf{f}, \mathbf{c})$ are features described in the next subsection. In SMT, model parameters are typically optimized on BLEU scores (see Section 4.3), but we experiment with both maximum BLEU and minimum word error rate (WER) training. To optimize model parameters, we resort to the downhill

¹³We could have followed an independent path by constructing a discriminative equivalent to the generative model of Section 4.2.2, e.g., by maximizing the conditional log-likelihood (MCLE) of the training data as done for PCFGs in (Johnson, 2001).

simplex algorithm (Nelder and Mead, 1965) to iteratively compute changes in BLEU or WER by running the decoder on a development set of 264 meeting utterances. In our case, the simplex algorithm typically required between 50 to 200 iterations to converge (generally in less than 24 hours).

4.2.3.1 Features

We now describe the features included in the log-linear model of Equation 4.9. Note that the discriminative framework described here is not applied to the Ziff-Davis corpus, since our primary goal is to perform meeting summarization.

- $p_{\text{SCFG}}(\mathbf{f}, \mathbf{c})$: log-probability of the Markov SCFG model.¹⁴
- $p_{\text{LM}}(\mathbf{c})$: log-probability of a trigram mixture language model trained from many different sources, including all compressed utterances of our ICSI training set (21K words), Broadcast News (BN) transcripts included in the TDT (Graff, 1997) (12M words), and the Brown corpus (1M words).
- $\text{sum-idf}(\mathbf{c})$: sum of inverse document frequency (IDF) scores of all words in \mathbf{c} . IDF scores were collected from 92 million documents taken from Reuters stories, Broadcast News, and the Brown corpus.
- ratio of words that are retained in the compression ($|\mathbf{c}|/|\mathbf{f}|$).
- confidence score (only used in Chapter 5): in this work, we take word posterior probabilities provided with the ASR system described in Section 5.1.

Note that trigram log-probabilities are normalized by the length of the sentence, since they would otherwise always favor very short compressions. IDF scores were also normalized, since they would otherwise always prefer longer sentences.

¹⁴Note that, as in the generative model, $p_{\text{SCFG}}(\mathbf{f}, \mathbf{c})$ equals $\sum_{(\pi_{\mathbf{f}}, \pi_{\mathbf{c}}) \in \tau(\mathbf{f}, \mathbf{c})} p(\pi_{\mathbf{f}}, \pi_{\mathbf{c}})$. Each term $p(\pi_{\mathbf{f}}, \pi_{\mathbf{c}})$ is decomposed as $p(\pi_{\mathbf{c}}|\pi_{\mathbf{f}})p(\pi_{\mathbf{f}})$, where the first term is computed by the SCFG deletion model and where the second is provided by the syntactic parser. A difference with the generative model is that we add a scaling factor λ_j for PCFG probabilities, which is also directly optimized for either WER or BLEU: $p_{\text{SCFG}}(\mathbf{f}, \mathbf{c}) = \sum_{(\pi_{\mathbf{f}}, \pi_{\mathbf{c}}) \in \tau(\mathbf{f}, \mathbf{c})} p(\pi_{\mathbf{c}}|\pi_{\mathbf{f}})p(\pi_{\mathbf{f}})^{\lambda_j}$.

4.2.4 Decoder

The goal of the decoder is to find the most likely compression $\hat{\mathbf{c}}$ of the model. As seen in Equations 4.1 and 4.2, it can be computed by maximizing:

$$\hat{\mathbf{c}} = \arg \max_{\mathbf{c} \in \mathbf{C}} \left\{ \sum_{(\pi_{\mathbf{f}}, \pi_{\mathbf{c}}) \in \tau(\mathbf{f}, \mathbf{c})} p(\pi_{\mathbf{f}}, \pi_{\mathbf{c}}) \right\} \quad (4.10)$$

Finding the maximum-probability yield it difficult in general, because the above equation contains a summation over all possible tree structures that yield (\mathbf{f}, \mathbf{c}) . Another issue is that the SCFG model makes no attempt to model the probabilities $p(\mathbf{f})$ or $p(\pi_{\mathbf{f}})$. We thus approximate $\hat{\mathbf{c}}$ the following way: we first use a syntactic parser to parse the input sentence \mathbf{f} and generate a moderately large N -best list $(\pi_{\mathbf{f}}^1, \dots, \pi_{\mathbf{f}}^N)$. Second, for each tree $\pi_{\mathbf{f}}^i$, we search for the M most probable compressions $(\pi_{\mathbf{c}}^{i,1}, \dots, \pi_{\mathbf{c}}^{i,M})$ according to the estimate of $p(\pi_{\mathbf{c}} | \pi_{\mathbf{f}}^i)$ provided by the SCFG model. If we set M and N to be large enough, we can effectively cover most of the probability mass of Equation 4.10. This approximation yields

$$\hat{\mathbf{c}} = \arg \max_{\mathbf{c} \in \mathbf{C}} \left\{ \sum_{i=1}^N p(\pi_{\mathbf{f}}^i) \sum_{\pi_{\mathbf{c}} \in (\pi_{\mathbf{c}}^{i,1}, \dots, \pi_{\mathbf{c}}^{i,M})} p(\pi_{\mathbf{c}} | \pi_{\mathbf{f}}^i) \right\} \quad (4.11)$$

To find the M most probable derivations that maximize $p(\pi_{\mathbf{c}} | \pi_{\mathbf{f}}^i)$ for a given parse tree $\pi_{\mathbf{f}}^i$, we use a bottom-up beam search decoder¹⁵ that maintains at each node only the M most probable configurations, which are obtained by enumerating and sorting the derivations accessible from each child. Let r be the rank of the grammar (i.e., maximum number of symbol in any of its right-hand sides), then the overall time complexity of such an M -best decoding algorithm is $O(rM^r \log M)$ (Huang and Chiang, 2005). Note that, as opposed to syntactic parsing, the search space is relatively small in our case, since constituent brackets are fixed for each run of the decoder. Even though we could have implemented algorithms with much lower asymptotic complexities (Huang and Chiang, 2005), we are generally able to produce very large M -best lists in relatively small amounts of time.

As in the case for summaries produced in Chapter 3, it is often desirable to constrain the length of the output produced by our sentence compression system. In general, it seems more sensible to impose a length constraint at the document level rather than at the sentence level. First, this is

¹⁵Note that we specifically do not use dynamic programming because n -gram language model scores are incrementally computed during decoding as it is generally done in syntax-based machine translation decoding (Wu, 1996).

generally how summarization evaluations are performed, and any hard constraint on the length of a specific sentence often has undesirable effects (e.g., it may produce ungrammatical output if no phrase is grammatically optional).

In our evaluations in this chapter, we will assume that a target length L is provided at run time. For a given sequence of input sentences $\mathbf{f}_{1:S} = (\mathbf{f}_1, \dots, \mathbf{f}_S)$, the sequence of compressions $\mathbf{c}_{1:S} = (\mathbf{c}_1, \dots, \mathbf{c}_S)$ hypothesized by the decoder must satisfy

$$\sum_{i=1}^S |\mathbf{c}_i| = L \quad (4.12)$$

where $|\mathbf{c}_i|$ is the length of sentence \mathbf{c}_i . To ensure that this constraint is met, we resort to the following approximation. First, we assume all compressions are independent and identically-distributed (iid), which allows us to write:¹⁶

$$p(\mathbf{c}_{1:S}|\mathbf{f}_{1:S}) = \prod_{i=1}^S p(\mathbf{c}_i|\mathbf{f}_i) \quad (4.13)$$

Second, we use the list of k -best yields produced by our decoded ($k \leq MN$) for each input sentence to find a sequence of compressions that satisfy the constraint on length. This search is trivial to implement using dynamic programming on a table T of size $S \times L$, where $T[i, j]$ represents the most likely sequence of hypothesis compressions $\mathbf{c}_{1:i}$ that satisfy $\sum_{m=1}^i |\mathbf{c}_m| = j$.

4.3 Evaluation Metrics

Previous automatic evaluations for sentence compression evaluation results relied primarily on word error rate (WER) and BLEU (Papineni et al., 2002). We now provide an brief overview of these two metrics.

4.3.1 Word Error Rate

Word error rate is widely used in speech recognition to assess recognition quality, and has also been used in some prior work on sentence compression (Clarke and Lapata, 2006b). Word error rate is computed using the edit distance $e^{(i)}$ between the recognizer output and the reference transcript of

¹⁶Identifying dependencies between constituents in different sentences is of course beyond the scope of syntactic models described in this chapter.

sentence i , and is defined as

$$\text{WER} = \frac{\sum_i e^{(i)}}{\sum_i n^{(i)}}, \quad (4.14)$$

where $n^{(i)}$ is the length of the reference sentence i . The edit distance, or Levenshtein distance (Levenshtein, 1966), is the minimum number of insertions, substitutions, and deletions required to transform one sentence into the other. In natural language generation and sentence compression, WER is sometimes expressed as an accuracy measure and is termed *simple string accuracy* (SSA) (Bangalore et al., 2000; Clarke and Lapata, 2006b): $\text{SSA} = 1 - \text{WER}$.

4.3.2 BLEU

BLEU (BiLingual Evaluation Understudy) (Papineni et al., 2002) is an evaluation metric commonly used to assess the quality of machine translation outputs, and is currently the most popular metric in machine translation. Several studies have reported high correlation with human judgments of translation quality (Papineni et al., 2002; Coughlin, 2001).

In our case, BLEU measures sentence compression accuracy according n -grams ($1 \leq n \leq N$) found both in the hypothesis compression and any of the references. Specifically, it measure the geometric mean of n -gram precisions and is multiplied by a factor BP that penalizes hypotheses that are shorter than any reference compression:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^N \frac{\log p_n}{N}\right) \quad (4.15)$$

Here, p_n denotes the precision of n -grams in the hypothesis compression, and N is the largest n -gram size used for evaluation. As it is often the case in machine translation, we use $N = 4$.

Note that a related metric, ROUGE (Lin, 2004), is sometimes used to assess summarization quality. Since ROUGE is mainly intended to be used to assess summarization quality at the document level rather at the sentence level, we do not report results with ROUGE.

4.4 Results

This section analyzes performance of the generative and discriminative models presented earlier on written and spoken texts. While summarization of conversational speech is the main focus of this thesis, it is nevertheless interesting to analyze their application to other genres such as written text.

First, our sentence compression system is fully data-driven and thus it is in principle not restricted to any domain, genre, or language. Second, linguistic resources, such as treebanks and summarization annotation, are available in much larger quantities for written texts; an analysis of the learning curve on written texts can give some insight into how much data would be needed to achieve reasonable performance in other genres. Third, most of the previous work in sentence compression was applied to the Ziff-Davis corpus, and it thus seems natural to provide a comparison against a state-of-the-art system on such data.

Note that, in all our experiments, we set the size of N -best lists produced by the syntactic parser to be $N = 10$. This generally yields much superior results than just using the first parse, which is the approach taken by K&M.

4.4.1 Written Texts

All experiments on written texts were performed on the Ziff-Davis corpus, and reported using WER. Since WER and BLEU scores are often correlated, we only report BLEU scores when it is deemed interesting.

For our empirical evaluations on Ziff-Davis, we use the entire training set of 16,787 sentence pairs 2.4, which may contain any number of deletions and up to seven substitutions. We use the abbreviation $ZD-i$ to refer to the version of the corpus that contains all the sentence pairs that have a number of substitutions smaller or equal to i . After parsing all these sentences with the parser described in (Charniak and Johnson, 2005, release of November 2005), we apply the SCFG rule extraction method described in Section 4.2.1. In addition to this, 200 sentence pairs (with no substitutions or deletions) are reserved as our development test, and we use the same 32 sentence pairs that were selected by K&M for testing.

In our first evaluation, we experimented with different horizontal and vertical Markovizations (Table 4.4). First, it appears that vertical annotation is helpful. It provides reductions in WER from 47.9% ($v = 0, s = 2$) to 46.8% for ($v = 1, s = 2$), but higher orders ($v > 1$) do not provide systematic increases of performance. Sister annotation is much more effective than vertical annotation, and provides, for instance, an improvement from 52.7% WER ($v = 1, s = 0$) to 46.8% ($v = 1, s = 2$). Manual examinations of compression outputs confirmed this analysis: without sister annotation, deletions of punctuation symbols and function words (determiners, coordinate conjunctions, etc.)

Vertical Order	Horizontal Order			
	$s = 0$	$s = 1$	$s = 2$	$s = 3$
$v = 0$	52.2	48.5	47.9	48.5
$v = 1$	52.7	48	46.8	47.7
$v = 2$	51.8	48.5	47.5	47.8
$v = 3$	50.2	47	47.2	48.5

Table 4.4: Word error rates (%WER) on the Ziff-Davis development set for different Markov structures. These models do not incorporate any lexical or POS information ($l = 0, t = 0$). The relative WER reduction from ($v = 0, s = 0$) to ($v = 1, s = 2$) is 11.5% (5.4% absolute). The baseline (unsplit SCFG) achieves only 62.4% WER.

are often inaccurate, and compressions clearly lack fluency. This annotation is also helpful for phrasal deletions; for instance, we found that PPs are deleted in 31.4% of the cases in Ziff-Davis if they do not immediately follow the head constituent, but this percentage drops to 11.1% for PPs that immediately follow the head. It seems, however, that increasing sister annotation beyond $s > 2$ to provide no benefits.

In our second evaluation reported in Table 4.5, we assessed the usefulness of lexical and POS annotation (setting s and v to 0). As mentioned earlier, the meta-parameter l controls the degree of lexicalization, e.g., models with $l = 2$ incorporate head-modifier bi-lexicalization (e.g., “looks-like”). The same principle applies to t , e.g., models with $t = 2$ incorporate head-modifier POS pairs (e.g., VBZ-IN). The column ZD-7 displays results for the entire training set of 16,787 sentence pairs, which includes substitutions. ZD-0 refers to its subset of sentence-compression pairs that K&M were restricted to with their model. The overall best model in the table is one that uses pairs of words and pairs of POS tags (44.9% WER). It is important to notice that the incorporation of lexical information without adding the ability to back-off to POS labels is seriously hurting performance. For instance, if we do not include the POS label of the head (i.e., VBZ in the case of our “looks-like” example), we observe a change from 44.9% ($l = 2, t = 2$) to 48.3% in WER. We also observe that POS tags provide substantial benefits in the case of uni-lexical and unlexicalized models. Note that lexical conditioning also helps in the case where the training data is relatively small (ZD-0), though differences are less significant, and bilexical dependencies actually hurt performance. The

Lexicalization and POS	ZD-7	ZD-0
<i>unlexicalized models:</i>		
$l = 0, t = 1$	49.8	49.5
$l = 0, t = 2$	48.2	50.6
<i>uni-lexical models:</i>		
$l = 1, t = 1$	47.2	49.0
$l = 1, t = 2$	45.8	47.2
<i>bi-lexical models:</i>		
$l = 2, t = 1$	48.3	48.9
$l = 2, t = 2$	44.9	47.8
Baseline (unsplit SCFG)	62.4	

Table 4.5: Word error rates (%WER) on the Ziff-Davis development set for different head-modifier annotations (other meta parameters set to $s = 0$ and $v = 0$). In ZD-7, the larger training set with substitutions is used for training.

best model for ZD-0 is one that conditions deletions on two POS tags and just one head word.

Since many head-modifier word pairs are sparsely seen, it is interesting to analyze how often contexts of POS tags and head words observed at test time were seen in the training data. We note again that all models discussed in this chapter are smoothed with Witten-Bell interpolation, thus maximum likelihood estimates of the largest contexts are always interpolated with less specific ones as to make maximum likelihood estimates more robust. Table 4.6 displays the percentage of times the maximum context of each model is unseen in the training data. It effectively shows that contexts of pairs of head words are available 39.7% of the cases. We note some differences between this analysis and studies that have been done in syntactic parsing to assess the usefulness of bi-lexical dependencies. Bikel (2004) found that the maximum context of a bilexical parsing model (Collins, 1999) is only available in 1.49% of the cases. In the case of the bi-lexical parsing model, words seen less than 5 times in the training data were mapped to “unknown”, which makes this 1.49% not directly comparable to ours (in our case, we did not remove rare words, since we assumed that our use of Witten-Bell interpolation would by itself robustly handle statistics for rare words). We hypothesize another reason that may explain this difference: a syntactic parser needs

Lexicalization and POS	ZD-7	ZD-0
<i>unlexicalized models:</i>		
$l = 0, t = 1$	1.9	7.3
$l = 0, t = 2$	2.5	8.5
<i>uni-lexical models:</i>		
$l = 1, t = 1$	38.6	74.9
$l = 1, t = 2$	39.9	75.9
<i>bi-lexical models:</i>		
$l = 2, t = 1$	60.5	84.3
$l = 2, t = 2$	60.3	84.1

Table 4.6: Model coverage analysis: percentage of time the maximum context of POS and head words, which depends on l and t , is unknown at test time.

to process an extremely large search space, and is bound to hypothesize a large number of (often incorrect) head-modifier word pairs that the sentence compression model never sees in the parser’s N -best list.

It is also interesting to analyze classification accuracy at the level the model performs predictions, i.e., at the constituent level. Table 4.7 provides constituent-level classification accuracies, as well as precision and recall scores with respect to deletions (i.e, recall is 1 if the classifier manages to identify all deleted constituents). Note that this analysis is performed on machine-generated parses, and that performance measures here are only helpful for error analysis and should presumably not be used for evaluation. We find that model performs best with parentheticals (PRN), coordinating conjunctions (CC), adverbials, and S clauses. On the other hand, it has problems spotting adjective, noun phrase, and verb phrase deletions, which are infrequent in the training data (note, however, that classification accuracy is high for these three constituents).

In our search for the best model on the development set, we experimented with different combinations of Markovizations and lexical dependencies, and finally settled with a model incorporating bi-lexical dependencies. While it appears that lexical, POS, sister, and vertical annotation independently provide significant improvements, the addition of sister and vertical annotation does not provide any improvement over $l = 2, t = 2$ in terms of automatic evaluation. However, we found

Cat.	Accuracy	Precision	Recall	F ₁
COMMA	.798	.813	.946	.875
PRN	.615	.7	.778	.737
CC	.811	.781	.641	.704
ADVP	.523	.556	.625	.588
S	.852	.545	.571	.558
PP	.852	.71	.415	.524
ADJP	.864	.75	.375	.5
RB	.583	.364	.571	.444
SBAR	.727	.333	.615	.432
NP	.908	.667	.301	.415
VP	.934	.333	.25	.286
JJ	.857	.5	.05	.091

Table 4.7: Classification accuracies, precision, recall, and F₁ scores for the most frequently deleted categories. Since positive instances constitute the majority class, precision and recall are computed here with respect to negative predictions (i.e., deletions). Note that the table contains all categories that were deleted at least 5 times in the reference annotation of the development set.

Models	Grammar	Content	Len(%)
NoisyC	4.37 ± 0.5	3.87 ± 1.2	70.4
Markov	4.68 ± 0.4	4.22 ± 0.4	62.7
Human	4.95 ± 0.1	4.43 ± 0.3	53.3

Table 4.8: Human evaluation on the Ziff-Davis test set. Note that results reported in this table pertain to a slightly different version of the Markov SCFG model (Galley and McKeown, 2007) (minor fixes have been made, which provide some improvement on the development set).

that one level of sister and parent annotation positively impacted output quality, hence our final model includes both ($s = 1, v = 1, l = 2$).

We finally took the best configuration selected above, and evaluated our model against the noisy-channel model of K&M on the 32 test sentences selected by them. We performed a human evaluation against the output produced by Knight and Marcu’s original implementation of their noisy channel model (Table 4.8). For this human evaluation, we relied on six native speakers of American English who assessed grammaticality and content (importance) with scores from 1 to 5, using instructions as described in K&M. Compressions produced by humans, the Markov SCFG model, and K&M’s model were scored by each subject, and were provided to them in random order. Both types of evaluations favored the Markov model against the noisy channel model.

Table 4.9 shows several outputs of our system (Markov) that significantly differed from the output of the noisy channel model (NoisyC), which confirms our impression that Markovized models can produce quite grammatical output. Our compression for the first sentence underlines one of the advantages of constituent-based classifiers, which have the ability to delete a very long phrase at once (here, a PP). The three next sentences display some advantages of our approach over the K&M model: the latter model seems to perform deletions with too little lexico-syntactic information, and accidentally removes certain modifiers that are sometimes, but not always, good candidates for deletions (e.g., ADJP in Sentence 2, PP in sentences 3 and 4). On the other hand, our model keeps these constituent intact. Finally, the fifth example is one of the three cases (among the 32 sentences) where our model produced a sentence we judged clearly ungrammatical. After inspection, we found that the parser assigned a particularly errorful tree to this input, which may partially explain this ungrammatical output.

<i>Input</i>	<i>Many debugging features, including user-defined break points and variable-watching and message-watching windows, have been added.</i>
NoisyC	Many debugging features, including user-defined points and variable-watching and message-watching windows, have been added.
Markov	Many debugging features have been added.
Human	Many debugging features have been added.
<i>Input</i>	<i>The chemical etching process used for glare protection is effective and will help if your office has the fluorescent-light overkill that 's typical in offices.</i>
NoisyC	The process used for glare protection is and will help if your office has the overkill
Markov	The chemical etching process used for glare protection is effective.
Human	Glare protection is effective.
<i>Input</i>	<i>The utilities will be bundled with Quickdex II in a \$90 package called super quickdex, which is expected to ship in late summer.</i>
NoisyC	The utilities will be bundled
Markov	The utilities will be bundled with Quickdex II.
Human	The utilities will be bundled with Quickdex II.
<i>Input</i>	<i>The discounted package for the SparcServer 470 is priced at \$89,900, down from the regular \$107,795.</i>
NoisyC	The package for the 470 is priced
Markov	The discounted package for the SparcServer 470 is at \$89,900.
Human	The SparcServer 470 is priced at \$89,900, down from the regular \$107,795.
<i>Input</i>	<i>Prices range from \$5,000 for a microvax 2000 to \$179,000 for the vax 8000 or higher series.</i>
NoisyC	Prices range from \$5,000 for a 2000 to \$179,000 for the vax 8000 or higher series.
Markov	Prices range from \$5,000 for a microvax for the vax.
Human	Prices range from \$5,000 to \$179,000.
<i>Input</i>	<i>The source code, which is available for C, Fortran, Ada or VHDL, can be compiled and executed on the same system or ported to other target platforms.</i>
NoisyC	The source code, can be compiled and executed on the system or ported to other target platforms.
Markov	The source code can be compiled and executed on the same system or ported to other target platforms.
Human	The source code is available for C, Fortran, Ada or VHDL.
<i>Input</i>	<i>The DOS batch language has major limitations, of course.</i>
NoisyC	The language has major limitations, of course.
Markov	The DOS batch language has major limitations.
Human	The DOS batch language has major limitations.
<i>Input</i>	<i>Arborscan is reliable and worked accurately in testing, but it produces very large DXF files.</i>
NoisyC	Arborscan is but it produces large DXF files.
Markov	Arborscan is reliable and worked accurately.
Human	Arborscan produces very large DXF files.
<i>Input</i>	<i>Although it bears the IBM name, it synthesizer circuitry is identical to that of a Yamaha FB-01, a popular eight-voice synthesizer module.</i>
NoisyC	It synthesizer circuitry is identical to that of a Yamaha FB-01, a eight-voice synthesizer module.
Markov	Synthesizer circuitry is identical to that of a Yamaha FB-01.
Human	The synthesizer circuitry is identical to a Yamaha FB-01 eight-voice synthesizer module.

Table 4.9: Sentence compressions for the test set of the Ziff-Davis corpus.

Vertical Order	Horizontal Order			
	$s = 0$	$s = 1$	$s = 2$	$s = 3$
$v = 0$	52.4	52.5	53.2	53.6
$v = 1$	52.9	52.4	53.1	53.5
$v = 2$	52.5	52.8	52.1	53.0
$v = 3$	52.2	51.7	52.9	52.5

Table 4.10: Word error rates (%WER) on the ICSI development set for different Markov structures. Relative reduction of WER between the model without vertical and sister annotation ($v = 0, s = 0$) and the best other model ($v = 3, s = 1$) is 1.44% (.75% relative).

4.4.2 Conversational Speech

In the experiments described in this section, we use the ICSI sentence compression corpus described in Section 2.1.6. We use 1,393 utterance pairs for training, and respectively use 264 utterance pairs (7,412 annotated words) for development and 169 utterances pairs (3,710 annotated words) for testing. More statistics regarding these two sets are provided in Tables 2.5 and 2.6.

We provide again an analysis of the effect of sister and parent annotation in parse trees (Table 4.10). We find again that this kind of annotation helps the constituent classifier, and that vertical annotation provides more prediction power than sister annotation. The benefit of this annotation is less pronounced than for written texts. Indeed, while the relative decrease of error reported in the previous section was 11.5%, we only notice a 1.44% relative WER decrease with meeting utterances.

A positive finding in Table 4.11 is that uni-lexical and bi-lexical models provide noticeable improvements over unlexicalized models, despite the small amount of training material. The column ‘%unseen’ suggests that lexical probabilities are used relatively often. What may come as a surprise is that only 56.5% of the contexts of head-modifier word pairs are unknown at test time, while as many as 60.3% are unknown in the case of Ziff-Davis (for the equivalent model in Table 4.6). This finding is quite striking, since models for Ziff-Davis are trained from more than 10 times the same amount of data. We believe this to be mostly due to a genre difference. Spoken mode is characterized by low lexical density, and tendency for simpler and less abstract vocabulary (Hughes, 1996), which might render lexicalized models more effective with spoken genres than with written texts.

Lexicalization and POS	%WER	%unseen
<i>unlexicalized models:</i>		
$l = 0, t = 0$	52.4	3.21
$l = 0, t = 1$	52.7	6.36
$l = 0, t = 2$	51.9	8.19
<i>uni-lexical models:</i>		
$l = 1, t = 0$	49.1	41.2
$l = 1, t = 1$	50.3	41.5
$l = 1, t = 2$	49.4	43.4
<i>bi-lexical models:</i>		
$l = 2, t = 0$	49	56.4
$l = 2, t = 1$	48.4	56.4
$l = 2, t = 2$	47.8	56.5
Baseline (unsplit SCFG)	63.1	-
Human	43.7	-

Table 4.11: Word error rates (%WER) on the ICSI development set for different head-modifier annotations (other meta parameters were set to $s = 0$ and $v = 0$). %seen represents the percentage of time the maximum context of POS and head words, which depends on l and t , is unknown at test time.

MODEL	%WER		%BLEU	
	devel	test	devel	test
Generative (SCFG)	47.84	43.16	65.44	68.10
Discriminative (minimum-WER)	47.04	42.82	66.74	68.71
Discriminative (maximum-BLEU)	48.68	47.03	67.77	70.67
Human	43.7	37.1	-	-

Table 4.12: Results for discriminative training with four features: SCFG compression model, n -gram language model, tf.idf score, and sentence length.

Results with discriminative training are displayed in Table 4.12. We trained two models by optimizing either WER or BLEU on our development corpus. It appears that the model trained for WER outperforms the generative model with both metrics and with both development and test sets. On the other hand, optimizing BLEU provides expected improvements in BLEU with both development and test sets, but it slightly worsens (i.e., increases) WER for both sets.

To illustrate these outputs, we include in Table 4.13 the first eight compressions generated by the discriminative model trained to minimize WER, along with input sentences and human compressions. Appendix C contains its compressions for the entire test set. Summary utterances have an average of 12.28 words per utterance, which represents 56 percent of the average number of words in the input sentences. In general, we found that this sentence compression system is able to remove accurately the various disfluencies, parenthetical phrases (“you know”), and information-poor phrases (e.g., “in this case” in the last example of the table). For longer deletions, it deletes mostly adverbial phrases and prepositional phrases, such as the temporal prepositional phrase “while he is changing the grammars to english” in example 4 for the table.

We performed an error analysis on the entire summary available in the appendix. Poor sentence generation is caused by three types of errors: incorrect input, incorrect parse, and incorrect compression. First, input sentences are not always well-formed grammatical sentences, as this can be seen in the first compression of Table 4.14 (these compressions are also taken from the appendix). The input sentence “well from my understanding of what the people at phillips were originally trying to do doesn’t quite fit” is not grammatical, and as a result, its compression “from my understanding of what people at phillips were trying to do” does not contain a main clause. Second, the quality of the input parses plays an important role in sentence compression, and incorrect attachments often yield compressions that are either syntactically or semantically incorrect. For instance, while the compression for the fifth input sentence in Table 4.13 seems grammatical, it distorts the meaning of the input sentence, mainly because the noun phrase “the syntactic tree that is underneath in the syntactic structure” was not attached to the verb “produce” as it should in the parse tree. Third, compression errors are caused by different factors, including wrong tree alignments and incorrect parses in the training data. Some compression errors also seem to stem from data sparseness issues. In the second example of Table 4.14, our system incorrectly deletes the argument “which of those it is” in “to add which those it is”. If the system had much more data available, it would presumably have

<i>Input</i>	ok well , so ralf and tilman are here .
Markov	ralf and tilman are here .
Human	ralf and tilman are here .
<i>Input</i>	i thought two things uh we 'll introduce ourselves and what we do .
Markov	i thought two things we 'll introduce ourselves .
Human	we 'll introduce ourselves .
<i>Input</i>	and um we already talked with andreas , thilo and david and some lines of code were already written today and almost tested and just going to say we have um again the recognizer to parser thing where we 're working on and that should be no problem and then that can be sort of developed uh as needed when we enter the tourism domain .
Markov	we talked with andreas thilo and david and lines of code were written today and tested .
Human	we already talked with andreas , thilo and david some lines of code were written today and almost tested we have the recognizer to parser that can be developed when we enter the tourism domain .
<i>Input</i>	and um there one of our diligent workers has to sort of volunteer to look over tilman 's shoulder while he is changing the grammars to english because we face two ways .
Markov	there one of our diligent workers has to volunteer to look over tilman 's shoulder .
Human	one of our workers has to look over tilman 's shoulder while he is changing the grammars to english .
<i>Input</i>	either we do a syllable concatenating um grammar for the english generation which is sort of starting from scratch and doing it the easy way , or we simply adopt the ah um more in-depth um style that is implemented in the german system and um are then able not only to produce strings but also the syntactic tree that is underneath in the syntactic structure which is the way we decided we were going to go because a , it 's easier in the beginning and um it does require some knowledge of those grammars and some linguistic background .
Markov	we do a syllable concatenating grammar for the english generation but also the syntactic tree that is underneath in the syntactic structure .
Human	either we do a syllable concatenating grammar for the english generation , or we adopt the more in-depth style in the german system and are then able not only to produce strings but also the syntactic tree it 's easier in the beginning and it does require knowledge of those grammars and linguistic background .
<i>Input</i>	johno , are you going to have some time to do that uh with these guys ?
Markov	johno are you going to have some time to do that with these guys ?
Human	johno , are you going to have time to do that ?
<i>Input</i>	the ultimate goal is that before they leave we can run through the entire system input through output on at least one or two sample things .
Markov	the ultimate goal is before they leave we can run through the entire system input through output .
Human	the goal is that we can run the entire system on one or two sample things .
<i>Input</i>	and by virtue of doing that then in this case johno will have acquired the knowledge of how to extend it .
Markov	johno will have acquired the knowledge of how to extend it .
Human	then johno will have acquired the knowledge of how to extend it .

Table 4.13: Sentence compressions for the test set of the ICSI meeting corpus (first 8 compressions).

<i>Input</i>	well from my understanding of what the people at phillips were originally trying to do does n't seem to quite fit into smartkom currently so what they 're really doing right now is only selecting among the alternatives , the hypotheses that they 're given enriched by the domain knowledge and the um discourse modeler and so on .
Markov	from my understanding of what the people at phillips were trying to do does n't seem to fit into smartkom currently .
Human	what the people at phillips were originally trying does n't fit into smartkom currently so they 're selecting alternatives , hypotheses enriched by domain knowledge and the discourse modeler .
<i>Input</i>	there are only three possibilities and what one would like is for this uh , knowledge modeling module to add which of those it is and give it to the planner .
Markov	there are only three possibilities and what one would like is for this knowledge modeling module to add and give it to the planner .
Human	one would like this knowledge modeling module to add which it is and give it to the planner .
<i>Input</i>	also of course it would be really nice to know what the plans are , in addition to what 's sort of already in code .
Markov	it would be nice to know what the plans are , in addition .
Human	it would be nice to know the plans , in addition to what 's already in code .

Table 4.14: Sentence compressions for the test set of the ICSI meeting corpus (sample errors).

learned that a clause headed by “which” and appearing as the first right modifier of the verb “add” is most certainly grammatically required. However, because the head-modifier pair “add-which” does not appear in the training data, it backs off to just considering the word “which” to determine whether or not to delete this clause. In this case, the backoff is unfavorable, since clauses headed by “which” are often deleted in the training data and are therefore deleted in most of the cases at test time.¹⁷ It sometimes also occurs that the distinction between grammatically optional and mandatory is difficult to capture with the kind of structural features discussed in this chapter. For example, in the last example of Table 4.14, the compression of “in addition to what 's already in code” into “in addition” seems a poor compression in that particular sentence. However, it would have been fine to delete the PP headed by “to” and modifying “addition” in a sentence such as “in addition to this, it would be nice to know the plans.”

4.5 Related Work

4.5.1 Sentence Compression

A large body of work addresses the problem of revision at the sentence level. Sentence compression techniques have been applied to single-document summarization (Jing and McKeown, 2000; Jing,

¹⁷Given the constraints imposed on length during decoding, SBARs headed by “which” are deleted in 71% of the cases.

2000; Knight and Marcu, 2000; Daumé III and Marcu, 2002; Reizler et al., 2003; Turner and Charniak, 2005), headline generation (Witbrock and Mittal, 1999; Banko et al., 2000; Zajic et al., 2002; Dorr et al., 2003) text compaction for small displays (Corston-Oliver, 2001), voicemail summarization (Koumpis and Renals, 2003a), text simplification (Chandrasekar et al., 1996; Carroll et al., 1998), information retrieval (Corston-Oliver and Dolan, 1999), audio scanning services for the blind (Grefenstette, 1998), and closed captioning (Linke-Ellis, 1999; Robert-Ribes et al., 1999).

A variety of techniques have been used for sentence compression: hand-crafted rules derived from a study of document-abstract pairs (Jing, 2000; Jing and McKeown, 2000; Dorr et al., 2003), generative synchronous grammars automatically learned from document-abstract pairs (Knight and Marcu, 2000; Daumé III and Marcu, 2002; Turner and Charniak, 2005), and various word or constituent classifiers such as hidden Markov models (Witbrock and Mittal, 1999; Banko et al., 2000; Zajic et al., 2002; Koumpis and Renals, 2003a), decision trees (Knight and Marcu, 2000), maximum-entropy models (Reizler et al., 2003), support vector classifiers (Daumé III and Marcu, 2004; Nguyen et al., 2005), and other large-margin techniques (McDonald, 2006). Some techniques relied on unsupervised learning (Hori, 2002; Clarke and Lapata, 2006a), in the sense that they did not exploit any parallel data. It is interesting to notice that, aside from (Witbrock and Mittal, 1999; Banko et al., 2000; Zajic et al., 2002; Koumpis and Renals, 2003a) compression techniques generally capitalize on syntactic knowledge. Their case is particular in the sense that their techniques have mainly been used to produce messages for small displays (Koumpis and Renals, 2003a) or headlines, tasks for which grammatical knowledge is admittedly less helpful. With headlines, generated sentences are typically very short, e.g., (Banko et al., 2000) produces 5-6 words per sentence on average, and the production of ungrammatical sentences is less likely and generally not particularly harmful to their understanding.

It may appear to some readers that our work shares many similarities with the speech summarizer of Hori (2002). As in (Hori, 2002), our sentence compression system incorporates a syntax model, an n -gram language model, and various information sources such as tf.idf scores and confidence scores from the speech recognizer. However, as opposed to most other syntax-directed approaches to sentence compression, the syntactic model of Hori is trained in an unsupervised fashion with the inside-outside algorithm (Lari and Young, 1990), and does not exploit treebanks now available for many languages (including Japanese (Kawata and Bartels, 2000) and English, the two

languages Hori experimented with). Furthermore, all models in (Hori, 2002) are monolingual and none are trained on parallel texts. Hence, a poor compression of the sentence “they almost missed the train” into “they missed the train” wouldn’t be penalized, since the target sentence is linguistically well formed and would expectedly rank high according to dependency models and n -gram language models properly trained on in-domain data. Their use of dependency trees is primarily meant to penalize compressions that remove heads (e.g., “missed”) without their modifiers (e.g., “they”), as in “they almost the train”, and it has no means to proscribe deletions that are semantically undesirable (e.g., “they missed the train”).

Besides (Hori, 2002), other approaches (Kikuchi et al., 2003; Chatain et al., 2006b; Turner and Charniak, 2005; Chatain et al., 2006a; Clarke and Lapata, 2006a) also rely exclusively on unsupervised methods.¹⁸ While it is desirable to address the sparseness issue by exploiting the abundance of non-parallel data in an unsupervised manner, our interest in our thesis was more to make supervised learning more effective by exploiting corpora of unrestricted document-abstract pairs, which are generally much more available than annotated sentence-compression pairs.

Other applications to utterance revision have primarily focused on making transcript utterances more fluent and readable, though they did not attempt significant reduction in sentence lengths. Zechner’s (2002) revision operations were essentially restricted to disfluency removal. More recently, the DARPA EARS program (Effective, Affordable, Reusable Speech-to-Text) aimed to achieve better transcript readability, by removing non-content words such as filled pauses, discourse markers, and disfluent speech. We consider our work to be somewhat orthogonal and complementary to previous research done under the auspices of the EARS program, since relevant research in EARS has primarily focused on metadata annotation and extraction (e.g., (Liu et al., 2004; Kim et al., 2004; Liu et al., 2005)) and has generally not directly targeted improvement in end-to-end readability. (Gibson et al., 2004) is the only work in EARS known to us directly addressing questions about readability.

¹⁸To be more precise, we specifically mean that none of these methods use any labeled parallel data. However, they often rely on *supervised* learning from non-parallel texts, e.g., build generative language models using n -gram or CFG production frequencies.

4.5.2 Machine Translation

Since the sentence compression task can easily be expressed as a translation problem, it is natural to compare our work to machine translation approaches. Previous work in statistical machine translation (SMT) can be broadly divided into three categories: word-based models (Brown et al., 1990), phrase-based models (Och et al., 1999; Koehn et al., 2003; Och and Ney, 2004), and syntax-based models (Wu, 1997; Yamada and Knight, 2001; Chiang, 2005).

In word-based machine translation, the IBM models (Brown et al., 1990) incorporate word-for-word translations, various “distortion” models for re-ordering words, word “fertility” models that account for insertions and deletions. Of these three model types, only the fertility model is directly relevant to sentence compression. While our syntax-based model does more than just estimating probabilities at the word level (i.e., it does so at the constituent level), it is interesting to contrast the fertility model of Brown et al. with probabilities of deleting words (i.e., leaf nodes) in our SCFG model. As it is the case with our model, the fertility model is trained with the maximum likelihood estimator (specifically, with maximum-likelihood re-estimation as in the expectation-maximization algorithm (Dempster et al., 1977)), thus we would expect that the deletion probabilities derived from fertility models would not have any particular advantage or disadvantage over our lexicalized word-level deletion probabilities,¹⁹ except that the SCFG model can use additional information such as POS tags and parent annotation to better classify infrequent words.

The alignment template approach to SMT first described in (Och et al., 1999) learns phrase-to-phrase translations, in which a phrase may be any consecutive sequence of words. Phrase-based models are among the best performing ones in SMT. While phrase-based models would become very appealing if training data in sentence compression abounded as in machine translation, it would seem difficult to apply phrase-based approaches to sentence compression with just a few thousand sentence pairs as training material.

Syntax-directed translation was originally applied to compilers (Irons, 1961), and formalized into grammars (Lewis and Stearns, 1968; Aho and Ullman, 1969) that are now often termed synchronous context-free grammars in NLP. Synchronous tree rewriting systems, such as tree substitution grammars (STSG) and tree adjoining grammars (STAG), extend SCFG grammars in that their

¹⁹We admit the comparison between the two models is quite difficult, in particular because any systematic differences in word alignment could lead to completely different probability estimates.

productions generate pairs of trees instead of pairs of strings, and have recently been applied to SMT (Poutsma, 1998; Eisner, 2003; Galley et al., 2004a; Galley et al., 2006; Cowan et al., 2006). As argued in (Joshi, 1987) the *extended domain of locality* of tree rewriting systems is useful in generation—and by extension in machine translation—because they offer effective means of localizing syntactic properties such as argument structure, word order, and morphological marking (e.g., agreement). However, many of these benefits bear little significance to the sentence compression problem, i.e., word order and morphology generally remain the same in the source and target sentence. We nevertheless believe there is a practical benefit of localizing argument structure, since dependencies between binary predictions (i.e., to delete or not) often do not appear within the same domain of locality of SCFG productions.

4.6 Conclusion

In this chapter, we presented a sentence compression system based on synchronous context-free grammars (SCFG), for which we defined a head-driven Markovization formulation. This Markovization enabled us to incorporate lexical conditioning variables into our models. We empirically evaluated different Markov structures, and obtained a best system that generates particularly grammatical sentences according to a human evaluation. This approach was found to scale well on conversational sentences drawn from meeting transcripts, and was found only 9.38% worse (relative difference in %WER) than human performance. A discriminative model trained to minimize word error rate (WER) and combining in particular SCFG probabilities, tf.idf scores, and a trigram language model provided a further improvement.

Chapter 5

Integration and Error Analysis

This chapter presents two fully integrated and automated systems for content selection and utterance compression. It provides a comprehensive evaluation of the utterance selection and compression systems described in Chapters 3 and 4, for which we assume all inputs and information sources are produced by automated means such as automatic speech recognition (ASR) and automatic disfluency removal.

This chapter also describes the development of a classifier that identifies edited words, i.e., words in a self-repair that are subsequently corrected (e.g., “why didn’t he?” in the utterance “why didn’t he? why didn’t she stay home?”) or abandoned. We present a linear-chain conditional Markov random field (CRF) classifier that achieves an accuracy of 97.9% and an F_1 score 79.4% on a standard test set of the Switchboard corpus. While slightly less effective than the best performing system to date evaluated on the same data (Johnson and Charniak, 2004), it is relatively simple to implement and fast, which is in contrast with the $O(n^5)$ running time of the tree adjoining grammar approach of (Johnson and Charniak, 2004). This edited-word detector is used as a pre-processing step prior to syntactic parsing.

5.1 Automatic Speech Recognition

Our main motivation in this chapter is to assess the scalability of our approaches on real ASR output. The use of ASR transcripts is expectedly more problematic in sentence compression than in utterance selection. Thus, for the evaluation of the compression component, we attempt to further

analyze how our sentence compression system scales with word error rates (WER) on the continuum between perfect transcriptions and the ASR output of the CTS recognizer described previously. The rationale is that such evaluations would allow us to reach WERs that better reflect current state-of-the-art performance (Stolcke et al., 2007), and also give us a better sense of how our summarization components would behave on more accurate ASR transcripts that may be available in the near future.

To perform evaluation on ASR transcripts, we used the output of the SRI-ICSI speech recognition system (Mirghafori et al., 2004), which was run on the entire ICSI meeting corpus. As opposed to the experimental setup described in (Mirghafori et al., 2004), the version of the SRI-ICSI recognizer that produced a full recognition of the ICSI meeting did not exploit any meeting data to train acoustic or language models. We could not use a more performant and recent recognition system, since most current state-of-the-art meeting recognition systems (such as (Stolcke et al., 2007)) use almost the entire ICSI meeting corpus as training data, making them unsuitable in any experiment—like ours—that uses ICSI meetings for testing.¹

As a result of this constraint, the speech recognition system used in our experiments was trained only on conversational telephone speech (CTS) data. It has its acoustic model trained on 350 hours of CTS such as Switchboard (Godfrey et al., 1992), and its mixture language model was trained from 4 million words of Switchboard transcripts, 150 million words of Broadcast News, and 191M words chosen from the web (Bulyko et al., 2003).

To get ASR transcripts with lower word error rates than real ASR outputs, we use word lattices² produced by the SRI-ICSI decoder, and search for the minimum-WER path in each lattice. A nice property of this method is that these paths are within the search space of the speech recognizer, and, as opposed to other approaches to produce oracle transcripts, they do not suffer from unreasonable artifacts such as containing out-of-vocabulary words.

The method we used to identify the minimum WER path in a lattice is a simple extension of the standard dynamic programming algorithm commonly used to find the minimum edit distance alignment between two strings (Levenshtein, 1966). The algorithm to find the minimum distance

¹Ideally, we would have liked to train the recognizer on the ICSI, NIST, and CMU meeting data, but leave out the few meetings that comprise our development and test sets. Since we do not have direct access to the SRI-ICSI recognizer, this wish is effectively difficult to satisfy.

²We thank Andreas Stolcke for preparing these word lattices and making them available to us.

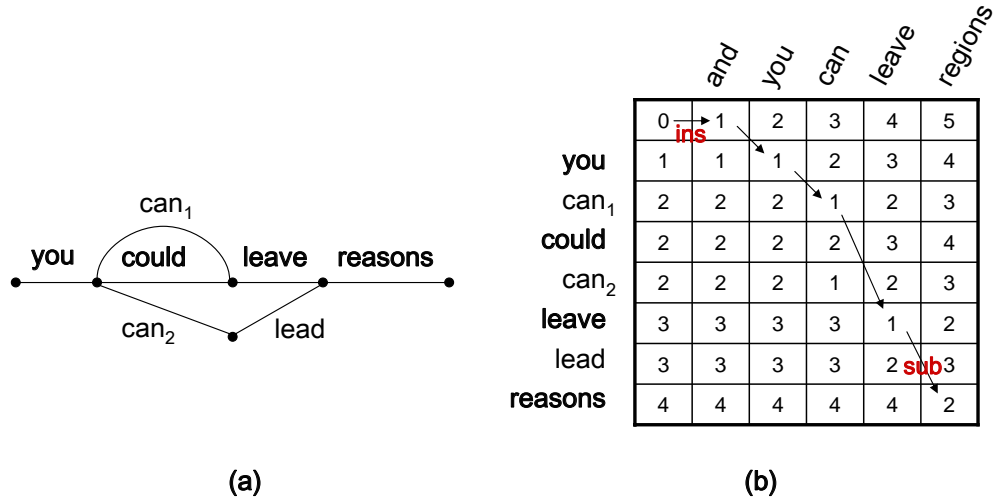


Figure 5.1: Illustration of the minimum edit distance (MED) dynamic programming algorithm to find the minimum WER in the lattice shown in (a) given a reference transcription “and you can leave regions”. While the edit distance between the highest-probability path (shown is bold) is 3, the MED algorithm has a cost of 2.

between a reference transcript and any path in the lattice is illustrated in Figure 5.1. When normalized by the length of the reference string, this measure is often called *lattice WER* (Weng et al., 1998; Stolcke, 2002). The algorithm operates as follows: given a reference sentence $w_{1:n} = (w_1, \dots, w_n)$ and a word lattice represented as a topologically sorted graph $G = (V, E)$, we build a table T of $n + 1$ rows and $|E| + 1$ columns. $T[i, j]$ represents the minimum cost of aligning $w_{1:i}$ with any path between the start vertex v_1 and edge e_i . If we define $\pi(j)$ to be the set of indices of all edges adjacent to e_j and preceding it, the value of $T[i, j]$ can then computed recursively using:

$$T[i, j] = \min \begin{cases} \min_{k \in \pi(j)} T[i - 1, k] + \text{subst-cost}(w_i, e_j) \\ \min_{k \in \pi(j)} T[i, k] + 1 \\ T[i - 1, j] + 1 \end{cases}$$

Note that it often happens that the minimum WER path in the lattice is not unique, in which case we use word posterior probabilities to break ties. To get oracle transcripts with WER worse (i.e., higher) than lattice WER, we simply prune lattices with a threshold on lattice density before extracting the path with minimum WER. In speech recognition, lattice density is defined as number

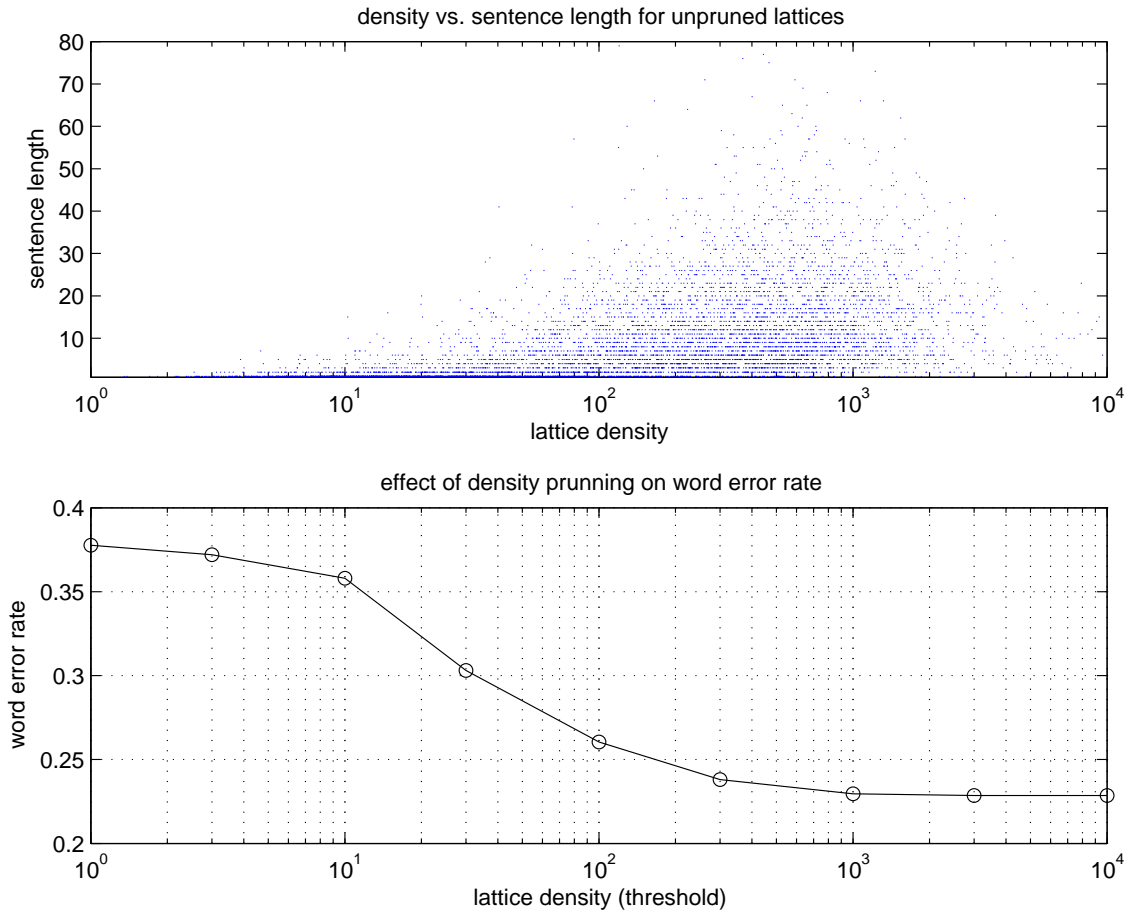


Figure 5.2: The upper scatterplot displays the relationship between lattice density and reference length for each unpruned lattice. The lower graph displays the increase of lattice WER as a result of pruning lattices with a threshold on density.

of words in the lattice divided by the number of words in the reference.

Figure 5.2 displays the result of varying the amount of pruning to get WERs that become progressively close to real decoder WER. The upper figure analyzes unpruned lattices (about 85,000), and shows the correlation between the length of the reference sentence and lattice density, which is 403 edges per word on average. The lower figure displays the effect of pruning lattices by imposing a threshold on lattice densities, and expresses the relationship between density *threshold* and WER.

Table 5.1 shows word error rates for the speech recognizer used in our experiments (SRI-04) (Mirghafori et al., 2004), which displays an overall WER is 37.77%. The overall WER of our oracle

SYSTEM	Hours	ICSI-all	ICSI-devel	ICSI-test
SRI-ICSI-04 decoder WER	350 (CTS)	37.77	32.66	31.57
SRI-ICSI-04 lattice WER	350 (CTS)	22.85	19.30	16.53

Table 5.1: Decoder and lattice word error rates (%WER) of the speech recognizer used in our experiments. ICSI-all refers to entire corpus (75 meetings). ICSI-devel and ICSI-test respectively refer to the development and test set of our experimental setup for sentence compression.

transcripts is 22.85%. As we pointed out earlier, the SRI-ICSI recognizer used in our experiments could not be trained on any meeting data. We would of course expect a significant reduction of WER with in-domain data. Furthermore, Stolcke et al. (2007) report that the SRI-ICSI recognizer of 2007 provides relative WER reduction of 24% (from 33.6% to 27.1%) on a NIST test set compared to the system of 2004, which may hint that the 22.85% WER of our oracle transcripts may be relatively close to current state-of-the-art performance.

5.2 Removing Edited Words from Disfluent Utterances

As mentioned in Chapter 4, we choose to identify and remove edited words during pre-processing, and leave the identification of other disfluencies, such as filled pauses (e.g., “uh”, “um”), explicit editing terms (e.g., “I mean”), and parentheticals for later stages. This is consistent with previous work (Charniak and Johnson, 2001) and enables a more direct comparison with their results.

There seems to be little evidence supporting the idea that treating syntax-directed sentence compression and disfluency removal as a joint problem is beneficial. First, repairs almost inevitably decrease parsing accuracies. Second, previous attempts to use CFG models to treat syntactic parsing and edit detection as a joint problem have not come close to state-of-the-art performance (Harper et al., 2005; Hale et al., 2006). Third, major clues in detecting speech repairs that are instrumental in achieving high accuracies (as shown later in this section) are relatively shallow phenomena, such as repeated words and part-of-speech tokens. Finally, a more troublesome issue is that context-free grammars—as our SCFG models—cannot reasonably account for the structure of repairs (Johnson and Charniak, 2004).

As analyzed by Shriberg (1994), a self-repair consists of four parts as seen in Figure 5.3: the

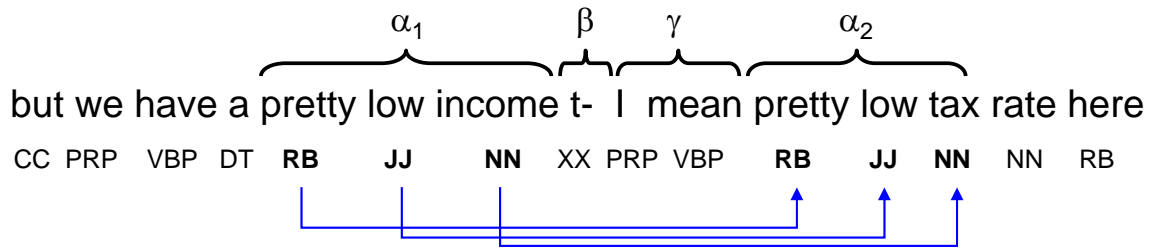


Figure 5.3: A self-repair comprises four parts: the reparandum (α_1), an interruption point (β), an optional editing term (γ), and the repair (α_2). Cross-serial dependencies between α_1 and α_2 make context-free grammars unsuitable recognizers.

reparandum, which is the speech production to be repaired; an *interruption point* or free final, which signals the point of interruption and may be associated with a word fragment; an optional *interregnum* which contains one or more editing terms such as “I mean”; the *repair*, which is the speech production that corrects the reparandum. The relationship between reparandum and repair is particular in sense that the repair is an approximate copy of the reparandum. These so-called cross-serial dependencies cannot be recognized by context-free grammars, and require instead the use of more powerful formalisms such as tree adjoining grammars (TAG) (Joshi et al., 1975; Shieber, 1988). This was the motivating factor for the development a TAG-based edit-word detector (Johnson and Charniak, 2004).

Previous approaches to identifying repairs have used textual and syntactic information (Bear et al., 1992; Core and Schubert, 1999; Charniak and Johnson, 2001; Johnson and Charniak, 2004), prosody (Nakatani and Hirschberg, 1993; Shriberg et al., 1997; Stolcke et al., 1998), and language models (Heeman and Allen, 1999; Johnson and Charniak, 2004). A central idea to some of these approaches is that repairs are often identified by two sequences of repeated words (Bear et al., 1992; Charniak and Johnson, 2001). For instance, Charniak and Johnson (2001) present a two-stage approach that tries to identify repeats of part-of-speech sequences, and then use various lexical and POS features pertaining to repeats and individual words in order to classify each word as either edited or not. Since Charniak and Johnson’s approach is relatively simple and provides good results on the Switchboard corpus—a corpus also made of informal conversations—we decided to build on their approach.

REPAIR TYPE	PERCENTAGE
$a_1 a_2$	73.03
$a_1 F a_2$	6.49
a_1	5.00
$a_1 P a_2$	4.18
$d_1 d_2$	3.85

Table 5.2: Distribution of repairs in Switchboard. F stands for filler (e.g., “um”, “uh”), and P for parenthetical (e.g., “I mean”). a_1 and a_2 are respectively the reparandum and the repair. d_1 and d_2 corresponds to the repetition of a discourse marker (e.g., “and - and”).

5.2.1 Distribution of Repairs

The 41,233 repairs annotated on the Switchboard corpus are distributed as shown in Table 5.2. a_1 and a_2 are respectively the reparandum and the repair, which may or may not constitute an exact copy. As we can see in the table, most repairs occur without any intervening strings (73%). There is also a small percentage of cases (5%) where the interregnum is abandoned (i.e., a_2 is missing). For cases where both a_1 and a_2 are present, we analyzed the distribution of edit distances (Levenshtein, 1966) between a_1 and a_2 . Table 5.3 displays edit distances in terms of words (i.e., the number of word insertions, substitutions, and deletions), and Table 5.4 relies on edit distances in terms of POS tags. In the latter case, we allow a word to be substituted with another word if the two share the same POS tag. As we can see, exact repeats between a_1 and a_2 represent only 57.2% of the cases. Even when we match a_1 and a_2 with their POS labels, cases of exact repeats (of POS sequences) still represent only 65.5%. The approach of Charniak and Johnson relies on the assumption that a_1 and a_2 are identical POS sequences, but this assumption often does not hold. As we will see, our approach tries to improve upon their limitations by defining features that can recognize any partial match between subsequences of a_1 and a_2 (e.g., identical substrings of words, POS labels, and suffixes).

5.2.2 Baseline Approach

In this subsection, we first describe the approach of (Charniak and Johnson, 2001, henceforth, C&J) to edited word detection, and then present and evaluate some simple changes that yield significant

EDIT DISTANCE	PERCENTAGE
0	57.2
1	27.1
2	8.9
3	1.3

Table 5.3: Word edit distance between a_1 and a_2 .

EDIT DISTANCE	PERCENTAGE
0	65.5
1	21.6
2	7.1
3	2.8

Table 5.4: POS edit distance between a_1 and a_2 .

improvements on the Switchboard corpus. From our reimplementation of their approach to our best model, we obtain a relative error reduction of 23.6% on the development set.

The split of the Switchboard data is the same as for the syntactic parser in Chapter 4 and consistent with C&J: training data consisted of all `sw[23]*.dps` files, development data consisted of all `sw4[5-9]*.dps` files and test consisted of all `sw4[0-1]*.dps` files. For evaluation purposes, we follow the approach of (Johnson and Charniak, 2004) by removing all word fragments and punctuation symbols from the data, as to make our edited term detector more effectively applicable to ASR transcripts.³ Hence, the presentation of C&J is slightly simplified, since all features that depend on word fragments and punctuation are ignored in our case.

C&J’s method is broken down into two steps. The first step is to identify so-called “rough copies” in the input strings, which are represented as substrings of the form $\alpha_1\beta\gamma\alpha_2$. Rough copies, such as the one depicted in Figure 5.3, must satisfy:

1. The strings of POS tags of source α_1 and copy α_2 are identical.
2. The *free final* β consists of zero or more words such as “and”, “or”, “actually”, “so”, etc.

³While the experiments in (Charniak and Johnson, 2001) are reported with word fragments and punctuation, (Johnson and Charniak, 2004) provides the results of running the earlier work under this new evaluation condition.

VARIABLES	EXPLANATION
W_0	Current word
$T_{-1}, T_0, T_1, T_2, T_f$	POS tags
N_m	Number of words in common between α_1 and α_2
N_u	Number of words in α_1 that do not appear in α_2
N_i	Number of words in interregnum
N_l	Number of words to left edge of source
N_r	Number of words to right edge of source
C_t	True if followed by identical POS tag
C_w	True if followed by identical word
T_i	POS tag of the word following the interregnum

Table 5.5: Features to identify edited words within disfluencies.

3. The *interregnum* γ consists of zero or more strings such as “uh”, “you know”, “I guess”, “I mean”, etc.

The search for such a pattern is repeated multiple times until no more matches can be found, and longest matches are extracted first. Overlaps between rough copies are explicitly disallowed.

The second step is treated as a word-level binary classification problem, in which many of the conditioning variables are defined in terms of features pertaining to rough copies. The classifier presented in C&J is a linear model discriminatively trained to minimize BoostLoss (Collins, 2000). In our case, we instead minimize LogLoss as in so-called maximum entropy models (Berger et al., 1996). All our models are smoothed using a Gaussian prior (Chen and Rosenfeld, 1999) of .3, which was manually tuned on our development set.

Features presented in C&J are summarized in Table 5.5. All integer-valued features are bounded between 0 and 4 and converted to binary features. Features pertaining to each word include the current word W_0 , a fixed window of POS labels T_{-1}, \dots, T_2 , and two flags C_t and C_w that are set to true if the current word is followed by the same word or by the same POS, respectively. Features pertaining to rough copies are assigned to each word within the copy, and each word that is not within a rough copy has all rough-copy features set to NULL. Rough copy features attempt to quantify the similarity between α_1 and α_2 , and include some features of the interregnum and the

	%Precision	%Recall	F ₁
baseline	92.1	61.9	74.1
1st-order CRF	91.8	64.6	75.8
relaxed matched of $\beta\gamma$ (free final and interregnum)	88.0	67.7	76.6
word-copy features	90.9	71.8	80.2

Table 5.6: Results for edited word detection on the development of the Switchboard corpus.

free final. Note that the subscript f , as in T_f , is used to refer to the first word of the free final.

5.2.3 Improvements over Baseline

We make several changes to improve upon C&J following a partially manual hill-climb on the development set and inspection of errors produced on that set. We believe the development to be large enough (51,123 instances) that there is no noticeable variance in our experiments. The performance after each step of our manual hill-climb is displayed in Table 5.6.

The first row of the table corresponds to our reimplementaion of the C&J approach, which is our baseline. First, we hypothesized that a sequence model such as a 1st-order CRF would yield some error reduction, since edited terms generally occur contiguously. The error reduction over the baseline, which is equivalent to a 0th-order CRF model, is 1.7% (6.56% relative).

Second, we also noticed that the baseline has a particularly low recall, presumably because criteria for identifying rough copies are rather strict and seriously restrain what words may appear between α_1 and α_2 (i.e., $\beta\gamma$). In order to improve the recall of rough copy detection, we loosen the criteria to match rough copies as to allow $\beta\gamma$ to be any sequence of predefined length I . To ensure that this relaxation does not impact precision too adversely, we enable this relaxed search criterion only after we have found all rough copies satisfying the more constrained conditions mentioned in Section 5.2.2. We found the value of $I = 6$ to be optimal on the development set. This modification provides .8% error reduction (3.3% relative), and increases recall by 3.1%.

Third, we obtained our most significant improvement on this data using a large set of features that attempt to identify individual word repeats, POS repeats, and suffix repeats (word-copy features). As we can see in Figure 5.4, the requirement that α_1 and α_2 be the same sequence of words or POS tags is often unrealistic. Word insertions, deletions, and substitutions are common, and many

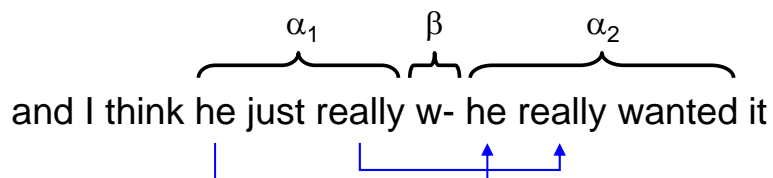


Figure 5.4: Copies α_1 and α_2 in a repair often contain insertions, deletions, substitutions. Hence, the assumption that α_1 and α_2 are the same POS sequences is often incorrect.

rough copies are consequently missed. A more robust approach is to allow the classifier to identify repeats by other means than our rule-based rough-copy matcher. For each word pair (w_i, w_{i+k}) within a window of size K , we define feature predicates that determine whether w_i and w_{i+k} are: (1) the same word; (2) words with the same POS label; (3) words with the same suffix. This change alone provides a large error reduction of 3.6% (15.4% relative).

We finally analyze the complexity of our approach to edit detection. Decoding the most likely sequence using a linear-chain model is linear in the length n of the utterance (Viterbi, 1967). Finding the maximal “rough copy” within the input sequence is a particular instance of the longest common substring (LCS) problem,⁴ which is solved in $O(n^2)$ using dynamic programming (Gusfield, 1997), and multiple passes to incrementally find non-overlapping copies of POS sequences takes $O(n^3)$ time. However, this asymptotic bound seems rarely met in practice and our decoder takes only about 70-80 seconds on average to decode the entire development set (5895 utterances) of the Switchboard corpus. LCS can be improved to run in time linear in the sequence using generalized suffix trees (McCreight, 1976), bringing time complexity for decoding each sentence down to $O(n^2)$. For comparison, the TAG-based approach of (Johnson and Charniak, 2004) runs in time $O(n^5)$, and thus requires some approximations to be computationally tractable. In practice, Johnson and Charniak only compute analyses on windows of 12 words or less.

SYSTEM	LM	%Precision	%Recall	%F ₁	%Accuracy
(Charniak and Johnson, 2001)	none	95.1	63.1	75.9	97.7
(Johnson and Charniak, 2004)	bigram	77.6	73.6	75.6	97.3
(Johnson and Charniak, 2004)	trigram	77.4	76.3	76.8	97.4
(Johnson and Charniak, 2004)	parser	82.0	77.8	79.7	97.8
This thesis work	none	91.5	70.1	79.4	97.9

Table 5.7: Results for edited word detection on the test set of Switchboard corpus (sections 40-41). Results from previous work are directly taken from (Johnson and Charniak, 2004). The LM column indicates what type of language model was used to score target strings. Note that accuracies were not reported in (Johnson and Charniak, 2004), but could be derived from precision and recall scores using the total number of genuine positives and negatives in the test data (3511 and 58012, respectively).

5.2.4 Results

Results on the test set of the Switchboard corpus (Sections 40-41) are displayed in Table 5.7. While performance of the best TAG-based model of (Johnson and Charniak, 2004) ($F_1 = .797$) is slightly superior to our best model ($F_1 = .794$), we note that ability to rescore hypothesized target strings with a syntax-based language model (Charniak, 2001) plays a very important role, and that no such feature is yet available in our model. In fact, our approach does not make use of any language model, and is thus unable to detect that the removal of several consecutive edited words generates unlikely trigrams in the target sentence. Furthermore, as opposed to the TAG-based approach, our method is unable to identify approximate repairs. It would certainly be possible to improve the repeat matching technique of our approach to deal with edit distances larger than 0, e.g., by drawing from previous work in biological sequence modeling. Schmidt (1998), for instance, presents a $O(n^2 \log n)$ algorithm for finding all (locally optimal) approximate repeats within a string of size n . An improved ability to identify rough copies is likely to provide a more reasonable balance between recall (70.1%) and precision (91.5%) and improve our overall F_1 score.

⁴The longest common substring problem is not to be confused with the longest common subsequence problem (Cormen et al., 2001). The fact that we are looking for two identical substrings within the same string, and not within two or more strings as it is typically the case, actually reduces search time.

	dialogic pairs	transcript	utterance segmentation
auto-DP	automatic	manual	manual
ref	<i>manual</i>	manual	manual
auto-DP-ASR	automatic	automatic	manual
auto-DP-ASR-seg	automatic	automatic	automatic

Table 5.8: The four experimental setups for the evaluation of content selection. The first one, auto-DP, is the same as in Chapter 3.

5.3 Content Selection

We now turn to the evaluation of our content selection component. We provide here results for four experimental conditions listed in Table 5.8. The first experimental setting of the table (*auto-DP*) is the same as in Chapter 3. We felt it was interesting to compare accuracies against a system that uses reference dialogic pair annotation (*ref*), since Chapter 3 only evaluates systems with automatically identified DPs.

Dialogic pair identification follows the method described in Section 3.2, which relies only on structural, textual, and durational information, and achieves 85.3% accuracy. In order to adapt the DP identifier to ASR transcripts, we trained two models: one with accurate transcripts and punctuation⁵ and the second one with ASR transcripts. Reference transcripts are the same as in Chapter 3, i.e., down-cased and with punctuation symbols.

We used the output of the SRI-ICSI-04 decoder described in Section 5.1 as our ASR transcripts. Since content selection didn’t seem to be seriously affected by the quality of the transcripts, we only performed evaluations with real decoder output and didn’t use any oracle ASR transcripts for the evaluation of content selection.

Each experimental setup among the three first rows required the extraction of an entirely new set of textual and information retrieval observations. Experiments with automatic segmentation (auto-DP-ASR-seg) required an entirely different set of observations to be recomputed again, since any change of segmentation affects what is observed from both the ASR transcripts and speech signal.

⁵We did not evaluate the effect of punctuation, but it would seem reasonable to assume they can provide helpful cues, e.g., question marks may help identifying DPs in the presence of question-answer pairs.

5.3.1 Automatic Utterance Segmentation

One potentially problematic aspect of this evaluation is that summarization annotation in (Carletta et al., 2003) is segmented by dialog acts, a type of segmentation that is notoriously hard to obtain automatically. Kolar et al. (2007) achieve an error rate of 46.6% in DA segmentation on meeting data, and most previous research in DA classification ignores the segmentation problem altogether.

Since we consider the DA segmentation problem not to be essential to summarization, we instead relied on a silence-based segmentation for the **auto-DP-ASR-seg** evaluation setting. In this evaluation setting, an utterance is defined as a period of speech by one speaker that has no pauses greater than t second (we arbitrarily set $t = 1$). This type of segmental unit, sometimes termed *spurt* (Shriberg et al., 2001), is often used to test real automatic conditions in various classification tasks.

Note that this definition of utterance with $t = 1$ is of course applied to both training and testing data. The fact that utterance segmentation differs from reference summarization annotation is not an issue. As explained in Section 3.4.1.4, reference utterance segmentation is not used at all in content selection annotation, since evaluation is performed at the word-level.

5.3.2 Results

As explained in Chapter 3, we run for each evaluation setting the sum-product loopy belief propagation (BP) algorithm and use posterior probabilities to select the most probable utterances to form a summary that satisfies a length constraint. We use the same 1st-order skip-chain CRF model that was found to work best on accurate transcripts.

Table 5.9 displays results for the four evaluation settings with different summary target lengths, and Figure 5.5 displays ROC for the four systems. Area under the curves (AUC) for the four experimental conditions are .76 (**ref**), .757 (**auto-DP**), .70 (**auto-DP-ASR**), and .66 (**auto-DP-ASR-seg**). As we noted in Chapter 3, the fact that we are evaluating each peer summary against multiple models makes it impossible in general to reach the maximum possible score, so we also provide F_1 scores for optimal summaries, which can be constructed following the Pyramid method (Nenkova and Passonneau, 2004b). As we did previously, results are provided with different thresholds on summary lengths, including the break-even point where the number of positive predictions matches the number of genuine positives in the test data.

	10%	18.6%	30%
ref	33.5	42.6	46.5
auto-DP	33.2	42.4	47.3
auto-DP-ASR	28.5	37.8	44.0
auto-DP-ASR-seg	24.4	32.8	38.4
baseline	12.0	20.5	25.9
human	-	45.6	-
optimal	57.3	69.3	67.9

Table 5.9: F_1 scores for summaries of length 10%, 18.6% (break-even point), and 30%, for varying degrees of transcription quality.

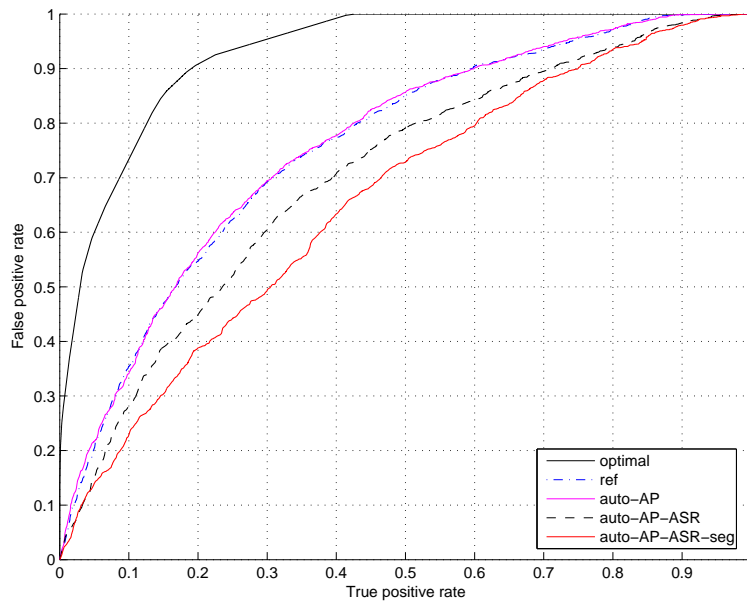


Figure 5.5: ROC curve comparing different levels of automation. “optimal” refers to the optimal summary, which may be obtained using the Pyramid method (Nenkova and Passonneau, 2004b).

	text-like	disfluency removal	%WER-devel	%WER-test
ref	yes	manual	0	0
no-punc	no	automatic	0	0
oracle-ASR	no	automatic	19.30	16.53
real-ASR	no	automatic	32.66	31.57

Table 5.10: The four experimental setups for the evaluation of sentence compression. “text-like” indicates whether the transcription contains punctuation and mixed case.

In these results, we find that automatic dialogic pair identification seldom affects performance, and that performance with automatic DPs is about the same as manually labeled dialogic pairs. However, decrease of performance between manual transcripts (auto-DP) and ASR outputs (auto-DP-ASR) is more pronounced. We hypothesize that degradation may be primarily due to the fact that our feature set incorporates a large number of lexical and cue-word features. We only performed feature selection on accurate transcripts, and it appears that many helpful features selected in this process are missing in ASR inputs (e.g., all features that depend on punctuation become ineffective). A feature selection applied instead to ASR transcripts would yield a different set of features, which presumably would capitalized on observations that do not so much depend on accurate transcripts (e.g., acoustic features).

5.4 Utterance Compression

In this section, we evaluate sentence compression on four types of transcripts, which we overview in Table 5.10. The two first experimental setups (**ref** and **no-punc**) aim to identify any empirical difference between taking as input (1) a relatively fluent textual transcription with punctuation, mixed case, and no edited words and (2) an ASR-like disfluent transcription whose edited words must be detected automatically. Transcriptions for the two last experimental setups (oracle-ASR and real-ASR) correspond respectively to minimum-WER paths in lattices and actual recognition outputs. We trained two sentence compression systems to handle respectively the original manual transcripts—i.e., with punctuation, mixed case, and word fragments—and ASR-like transcripts. To train and test the two models, we used two different syntactic parser models, which are described

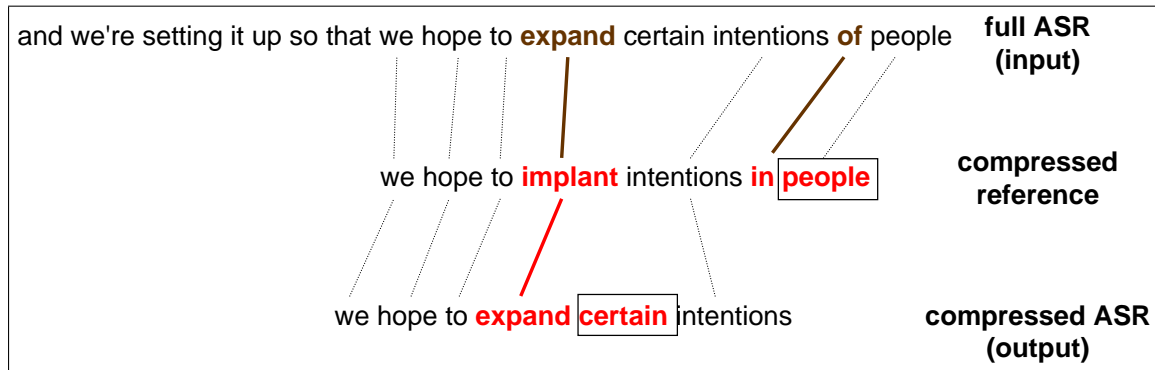


Figure 5.6: Motivating example for compression word error rate (CWER). WER and CWER are essentially the same, except that errors made by the sentence compression system are discounted if the speech recognizer makes the same error. For instance, both recognizer and compression system incorrectly include “expand” in their output, thus this error does not count towards CWER. Errors appearing in rectangles are not caused directly by the speech recognizer, and thus are considered as errors in the CWER computation.

in Section 4.1.1. Both were trained on a version of the Switchboard treebank whose edited words have been removed. For the second one, punctuation, case information, and word fragments were also removed.

We evaluate sentence compression with the approach presented in Chapter 4, and experiment both with the generative and discriminative model. Unless disfluencies have already been removed (as in **ref**), we automatically remove disfluencies with the method presented in Section 5.2. We then parse the remaining words with the appropriate syntactic parsing model overviewed in Section 4.1.1, which was trained both with and without punctuation.

5.4.1 Evaluation

Since input transcripts are errorful and differ from the ones used by human judges to perform sentence compression annotation, it is generally not possible for the sentence compression system to obtain a perfect match with the reference compression. Since the output of the sentence compression system may contain inserted and substituted words that were misrecognized by the ASR system in the first place, we need to adapt the WER metric to properly distinguish between errors that are the sole product of the recognizer and those caused directly by the sentence compression system.

Figure 5.6 displays an example of a sentence compression that contains both a word that was originally misrecognized by the ASR system (“expand”) and errors that are caused by the sentence compression system (“certain” should have been deleted and “people” is missing). The sentence compression system should admittedly not be penalized for inserting “expand”, since the word was incorrect in the ASR. To account for this difference, we define compression word error rate (CWER) to enable a more sensible evaluation of compression results. CWER naturally extends WER, in that CWER equals WER when the input provided to the sentence compression system contains no errors.

In CWER, we compare a reference compression c with a hypothesized compression \hat{c} of the ASR output. As in WER, we compute the minimum edit distance (MED) alignment between the two sentences (Levenshtein, 1966), which determines a count e of word insertions, word substitutions, and word deletions. The difference with WER is that we additionally compute the MED alignment between the full ASR sentence f and the reference compression c , in order to identify any words in the reference compression that do not align to any word in the ASR transcript (i.e., words the sentence compression system has absolutely no means of getting correct). From these two alignments, we then compute an error adjustment a , which is the count of all words c_i in the reference compression that satisfy both following conditions:

- c_i is an insertion or substitution error in the MED alignment between f and c (“implant”, “in”, and “people” in Figure 5.6).
- c_i is an insertion or substitution error in the MED alignment between \hat{c} and c (“implant”, “in”, and “people” in Figure 5.6).

Compression word error rate is then computed as follows:⁶

$$\text{CWER} = \frac{\sum_i (e^{(i)} - a^{(i)})}{\sum_i (n^{(i)} - a^{(i)})} \quad (5.1)$$

$n^{(i)}$, $e^{(i)}$, and $a^{(i)}$ are respectively the length, number of errors, and error adjustment of sentence i . Note that ASR errors are discounted in both numerator and denominator (without discounting errors in the denominator, CWER would unreasonably get close to zero for extremely high ASR word error

⁶Note that none of the terms $(n^{(i)} - a^{(i)})$ are negative. The only case where CWER becomes ill-defined is when no words of the reference correctly align to any word of the ASR sentences, which is highly unlikely.

	ref	no-punc	oracle-ASR	real-ASR
Lexicalization and POS	(WER=0)	(WER=0)	(WER=19.3%)	(WER=32.6%)
<i>unlexicalized models:</i>				
$l = 0, t = 1$	52.4	55.2	70.0	77.8
$l = 0, t = 2$	51.9	53.6	69.0	76.4
<i>uni-lexical models:</i>				
$l = 1, t = 1$	50.3	50.8	68.2	73.1
$l = 1, t = 2$	49.4	49.8	66.1	71.8
<i>bi-lexical models:</i>				
$l = 2, t = 1$	48.4	50.6	66.1	73.3
$l = 2, t = 2$	47.8	49.2	65.6	73.4
Baseline (unsplit SCFG)	63.1	64.9	76.4	86.4
Human	43.7	-	-	-

Table 5.11: Compression word error rates (CWER) on the ICSI development set. The best model for each type of transcript is shown in bold.

rates). In the example of Figure 5.6, $CWER = (4 - 2)/(7 - 2) = .4$, while the standard WER computed between \hat{c} and c is about .57.

5.4.2 Results

All results in this section are reported using CWER, since we do not have a meaningful way of adapting the BLEU metric to the compression task as we did with WER.

The results of running the SCFG compression model on different levels of transcription quality is shown in Table 5.11. Since errorful transcriptions are presumably likely to have a negative effect on any lexicalized model, we provide the same type of analysis of lexical and POS annotation as in Section 4.4. We find that lexicalization still provides large improvements over unlexicalized models, even in the case of real ASR transcripts. Relative WER reduction provided by the best lexicalized model over the best unlexicalized model is 5.5%, 4.4%, and 6.4% for **no-punc**, **oracle-ASR**, and **real-ASR**, respectively. Relative WER reduction provided by the best bi-lexical model over the best uni-lexical is 3.25% for **no-punc**, and 0.76% for **oracle-ASR**. However, bilexicalization was found

MODEL	ref (WER=0)	no-punc (WER=0)	oracle-ASR (WER=16.6%)	real-ASR (WER=31.6%)
Markov SCFG	43.2	45.4	61.6	67.9
Discriminative model (minimum-WER)	42.8	43.9	59.7	66.7
Baseline (unsplit SCFG)	53.4	56.6	69.8	76.9
Human	37.1	-	-	-

Table 5.12: Compression word error rates of the generative model on the ICSI test set.

to decrease performance in the case of real ASR transcripts.

Table 5.12 compares our generative and discriminative models for the four experimental conditions. As explained in Chapter 4, the parameters of a log-linear model incorporating five features—SCFG score, language model score, compression length, tf.idf score, and ASR confidence scores—which are optimized on the development corpus using the downhill simplex algorithm (Nelder and Mead, 1965). Simplex searches for the four experimental conditions took between 67 and 127 iterations to converge. Discriminative models are particularly more effective than the generative ones when tested on errorful transcriptions. Indeed, relative error reductions from generative to discriminative are respectively .94%, 3.4%, 3.2%, and 1.8%. One may hypothesize that the syntax-directed model—which requires accurate parses to be the most effective—loses some of its effectiveness as simpler models such as n -gram language models and tf.idf are presumably less affected by errorful inputs.

Chapter 6

Final Discussion

This thesis is concerned with the transformation of meeting transcriptions into short textual summaries that are maximally pertinent and readable, in the hope of improving effectiveness in meeting information access. The problem of finding important information in meetings and generating fluent summaries is treated as a structural prediction problem. At the document level, we select utterances using both local and long-range intersentential dependencies, such as between pragmatically related sentence pairs. At the sentence level, we select subsets of words that represent important information, while attempting to preserve both grammaticality and meaning.

6.1 Contributions

We make several significant advances in content selection for summarization of multi-party speech, and provide some contributions to sentence compression that were applicable to genres as diverse as conversational speech utterances and technical articles of the Ziff-Davis corpus. Contributions of the three main chapters of this thesis are summarized in the next subsections.

6.1.1 Utterance Selection

We presented a skip-chain conditional Markov random field (CRF) model for summary sentence selection that exploits long-distance dependencies between pragmatically related sentences, such as between a question and its answer, or an offer and its acceptance. These dependencies are automatically inferred by a probabilistic model of interpersonal interaction, and are instrumental in achieving

accuracies superior to competitive approaches. Compared to a standard 1st-order CRF model, the skip-chain CRF model provides a 1.4% increase in F_1 -score. Our best performing system reaches 92.9% of human performance.

6.1.2 Utterance Compression

Robust parameterization of synchronous context-free grammars: To alleviate data sparseness issues pertaining to previous generative models for sentence compression, we presented Markov synchronous context-free grammars (SCFG) for sentence compression. Markov SCFG grammars extend probabilistic SCFG grammars by introducing a flexible number of conditional independences between right-hand side constituents, and thus yield more robust probability estimates. Markov SCFG provides several other benefits, including the ability to condition constituent deletions on a flexible amount of additional syntactic context—such as parents and siblings in the tree—and lexical and POS annotation. We show that our compression model is successfully applicable to both written texts and conversational speech, and that it significantly outperforms a syntax-directed model that does not incorporate Markovization and lexicalization: 39% relative reduction of WER on written texts (from 62.4% to 44.9%), and 32% relative WER reduction (from 63.1% to 47.8%) on meeting utterances. In a human evaluation, our approach was found to produce sentences that are more grammatical than those of previous work (Knight and Marcu, 2000).

Better modeling of revision bi-texts. Another part of our effort focused on robust techniques for extracting SCFGs from parallel data. We improve upon (Knight and Marcu, 2000), who could only exploit 1.75% of a standard document-abstract corpus (Ziff-Davis) because of their restriction to sentence pairs that include no substitutions or insertions in target strings. To alleviate this restriction, we present a robust approach to tree-to-tree alignment between arbitrary document-abstract parallel corpora. By accounting for sentence pairs with substitutions in addition to deletions, we reach a retention rate of more than 25% of the Ziff-Davis data, and show that lexicalized SCFG models greatly benefited from this extra training material. This incorporation of this additional training material yields a relative WER reduction of 6.46% (2.9% absolute).

Minimum error-rate training. We incorporate additional models and features to improve compression quality—such as n -gram language models built from written texts and speech data, tf.idf scores, sentence length features, and confidence scores provided by the speech recognizer—and tune the scaling factors of these different models to optimize word error rate (WER) and BLEU on a development corpus. The incorporation of these other knowledge sources reduces WER down to 42.8%, while human performance is at 37.1%.

6.1.3 Integration

Integration and evaluation. We present fully integrated and automated systems for utterance selection and sentence compression, which are applicable to real automatic speech recognition output. We carried out evaluations on transcriptions with varying degrees of recognition errors, ranging from fully accurate transcriptions to real ASR outputs. We find that lexico-syntactic features that are novel to sentence compression—such as head-modifier lexical dependencies—scale well on ASR transcripts.

Correcting speech-repairs. We present a linear-chain CRF model for disfluency removal that almost achieves state-of-the-art performance on a common test set of the Switchboard corpus. It achieves .794 for F_1 -score on edited-word retrieval, and an overall 97.9% accuracy. While it reaches a slightly lower F_1 -score than (Johnson and Charniak, 2004), it has a substantially lower asymptotic running time.

6.2 Deliverables

Our work provides several deliverables that are accessible to the research community. Most resources are or will be available at: <http://www1.cs.columbia.edu/~galley/tools.html>.

- **Sentence compression system:** we make available a sentence compression system that incorporates most of the features described in Chapter 4 in one single executable. This program can be used to extract compression k -best lists, and to train Markov SCFG models with varying degrees of lexical and syntactic conditioning on parsed parallel texts. Provided that the

user will acquire additional resources such as SRILM n -gram language models and counts for tf.idf scores, this program can be used to discriminatively train model parameters with either maximum-BLEU training or minimum-WER training. This implementation also contains an implementation of the noisy channel model described in (Knight and Marcu, 2000), and the unsupervised SCFG extraction approach presented in (Turner and Charniak, 2005). While we cannot incorporate any of the parallel corpora described in this thesis, this tool however provides model files that performed best on these corpora, and thus can be used to compress sentences drawn from written texts and conversational speech. This tool is freely available to anyone who intends to use it for research and educational purposes.

- **ICSI sentence compression corpus:** we supervised three human judges who manually shortened a total of 2409 sentences drawn from the extractive summaries of 14 meetings. Sentences were shortened in seven stages, which represent training resources that can be used to train compression systems for a wide range of compression rates.
- **Correcting speech repairs:** we provide a tool to remove edited words from self-repairs in conversational speech, which incorporates all the features described in Section 5.2. This tool contains a model file that performed best on the Switchboard corpus. This program requires MALLET (McCallum, 2002) to train and test linear-chain CRF models.

6.3 Limitations and Future Work

We will now mention some limitations of our approach, and enumerate some issues and ideas that we believe merit future research. We focused our efforts on two relatively well defined problems—content selection and sentence compression—and left aside some features that would be desirable in a meeting summarization system. For instance, our summarization system does not attempt to model substitutions, and is thus unable to perform revisions such as paraphrasing and changes in referring expressions. Revisions other than deletions are often needed to make summaries more coherent and cohesive. For instance, (Nenkova, 2006) demonstrates the need for reference expression generation in order to improve the coherence of summaries of newswire texts. The scarce availability of training resources makes it computationally very challenging to extend our supervised model to revisions that require any substitutions. We believe it would be more reasonable to treat correc-

tions currently not covered by our approach either during pre-processing or post-processing (as it is done in (Nenkova, 2006)), since the relatively small number of revisions that require substitutions would probably not justify a dramatic increase of complexity in our syntax-based model.

Another drawback is that we spent more effort on devising methods and features that work well on accurate transcriptions and reference utterance segmentations, and spent comparatively less time attempting to fine tune the components in the fully automated settings of Chapter 5. Our goal in Chapter 5 was more to provide some results that can be used as initial points of reference for future research, but we believe that more experiments may yield better results. Common wisdom in machine learning dictates that training and testing data should have similar statistical properties, but we were not always able to abide to this rule, since it would have required a large amount of extra experiments. For practical reasons, we trained models on accurate transcriptions before testing them on ASR outputs, but this probably yielded suboptimal results.

Another limitation of our work arises from some general independence assumptions we formulated between the various summarization components. While it is computationally appealing to address summarization as a two-step approach—separating content selection from sentence revision—it is obvious that the underlying assumption that the two tasks are independent is often incorrect. The dependency between the two problems is bidirectional: the deletion of a sentence is likely to render a subsequent one incomprehensible, e.g., because a referring expression (e.g., “it”) might lose its referent (e.g., “your approach”). Conversely, the removal of a syntactic constituent may remove some text of the sentence that elicited its selection in the first place (e.g., an answer to a question). As mentioned earlier, one approach to address this limitation would be to correct any inconsistent predictions in content selection and sentence compression as a post-processing step. A computationally more challenging approach that would reflect a new trend and interest in NLP (Sutton et al., 2006) is to move away from pipeline approaches that make chains of independent decisions, and move towards systems that make multiple decisions jointly (e.g., perform selection and compression jointly). While it would seem computationally intractable to extend our utterance selection and compression approaches into an exact joint inference problem, we would be interested in exploring several approximate global inference approaches such as reranking (e.g., first extract a compression N -best list from each utterance and then use an intersentential model, such as a skip-chain CRF, that attempts to enforce pragmatic and referential constraints between the compressions

of different sentences).

We did not particularly study the impact of different utterance or sentence segmentations on the summarization task, but we believe it would merit some investigation. Since the ICSI summarization annotation was defined in terms of dialog act (DA) units, most of our experiments are reported on this kind of segmentation. We did not address the DA segmentation problem in our thesis, because we felt it was not particularly crucial to summarization and also because DA segmentation is difficult to obtain automatically. We would ideally want to explore better approaches than our use of pausal duration to segment utterances (Chapter 5), and it may be useful to have human labelers map DA summarization to a segmentation that is more realistic to automate, e.g., sentence segmentation. A segmentation by sentence is certainly preferable to the reader and the representation that is admittedly the best for the utterance compression component. It would also be interesting to investigate classification approaches that perform classification and segmentation jointly, such as semi-Markov CRFs (Sarawagi and Cohen, 2004).

We feel we could have more thoroughly exploited acoustic and durational information in this thesis. Our work in utterance selection explored a large set of pitch, energy, and durational features, but our dialogic act classifier uses no acoustic features, and our work on disfluency removal and sentence compression uses neither acoustic nor durational features. Previous research has shown (Shriberg et al., 1997; Stolcke et al., 1998) that acoustic and durational features are sometimes helpful to disfluency removal. It would also be interesting to study the use of pitch and energy features in sentence compression. While the relative slowness of our current minimum-WER training framework does not allow us to incorporate a large number of features, we would like to experiment with other discriminative learning techniques.

We could also have used learning algorithms that have been found superior to those used in this thesis, such as max-margin Markov networks (Taskar et al., 2003), kernel CRFs (Lafferty et al., 2004), and more effective loss functions (Collins et al., 2002). In the case of sentence compression, our work puts much more emphasis on feature analysis than on efficient optimization, and relies on a statistical framework that strives for careful feature selection and combination. However, we would like to extend our work to a more powerful discriminative framework, and in particular study the discriminative training of the parameters of the SCFG compression model (e.g., by maximum conditional likelihood estimation as it is sometimes done with CFG models (Johnson, 2001)).

We would like to improve our work by exploiting large amounts of unlabeled data, and more specifically rely on semi-supervised techniques such as co-training and the Yarowsky algorithm (Yarowsky, 1995; Blum and Mitchell, 1998; Abney, 2004). Some researchers in sentence compression have taken a fully unsupervised approach (Hori, 2002; Kikuchi et al., 2003; Chatain et al., 2006b; Chatain et al., 2006a; Clarke and Lapata, 2006a). However, we would argue against ignoring altogether exploitable parallel training data on the grounds that it is sparsely available. Another interesting idea is to perform domain adaptation (Gildea, 2001; McClosky et al., 2006), which attempts to make effective use of out-of-domain labeled data and in-domain unlabeled data to improve performance on a given domain. This was successfully applied to syntactic parsing, and it may provide some means to effectively use the sentence compression annotation of the Ziff-Davis corpus to improve performance on meetings.

Finally, we would like to point out that summaries as the ones discussed in this thesis need not be the end products of our speech summarization system, and we envision usages of our summarization system that are more elaborate than producing raw texts. Summaries may be integrated into a graphical user interface (GUI), such as hypertext documents in a web browser or a dedicated application, and provide smooth integration with other information gathering functionalities, such as query-based retrieval. We believe it is critical to provide direct links from summary utterances (either compressed or not) to utterances shown in context in the original document. One key benefit of an interactive GUI lies in its ability to better cope with classification errors, since some GUI control element or hyperlink would enable users to quickly access sentences or phrases that were inappropriately deleted. We believe an interface that would enable a seamless navigation between summaries and documents—such as NEWSBLASTER (Evans et al., 2004)—would greatly enhance the experience of meeting browsing and retrieval, perhaps the same way it was found to enhance the productivity of real users gathering information from news articles (McKeown et al., 2005).

Appendix A

Penn Treebank Tags

This appendix describes the most common clause-level, phrase-level, and word-level tags of the Penn Treebank corpus (Marcus et al., 1994). Note that this information is taken from (Bies, 1995), where one can find an exhaustive description of each tag.

Tag	Definition
S	Declarative, imperative, or infinitive clause
SINV	Inverted declarative sentence
SBAR	Clause introduced by a subordinating conjunction.
SBARQ	Question introduced by a <i>wh</i> -word
SQ	Inverted yes/no question or main clause of a <i>wh</i> -question

Table A.1: Clause-level tags of the Penn Treebank.

Tag	Definition	Example
VP	Verb phrase	-
NP	Noun phrase	-
PP	Prepositional phrase	-
ADJP	Adjective phrase	“so many”
ADVP	Adverb phrase	“kind of”
CONJP	Conjunction phrase	“and then”
FRAG	Fragment	“no” (as an isolated word)
INTJ	Interjection	“uh”
PRN	Parenthetical	“you know”
PRT	Particle	“up” as in “keep up”
WHAVP	<i>Wh</i> -adverb phrase	“how much”
WHNP	<i>Wh</i> -noun phrase	“what kind”

Table A.2: Phrase-level tags of the Penn Treebank. Examples are taken from Switchboard.

Tag	Definition
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
RB	Adverb
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
IN	Preposition or subordinating conjunction
MD	Modal
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP\$	Possessive pronoun
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
TO	to
UH	Interjection

Table A.3: Word-level tags of the Penn Treebank.

Appendix B

Annotation Guidelines for Sentence

Compression

This appendix contains the annotation guidelines provided to three human annotators who performed sentence compression annotation on meeting extracts of the ICSI meeting corpus (see Section 2.1.6 for more details).

B.1 Overview

The ultimate goal of the annotation introduced here is to support the production of meeting summaries that are maximally readable and pertinent, i.e., summaries that contain only important information and that are stripped from all disfluencies.

Data to be annotated is a set of spontaneous and informal meetings recorded in an academic institution and primarily concerned with computer science research (specifically: artificial intelligence, human language understanding, and networking research). Meetings are 60 minutes long on average, and involve a number of participants ranging from 3 to 10. Participants are not professional speakers, and given the conversational nature of this data, annotators will notice many hesitations and speech errors in transcriptions. Several participants are non-native English speakers, hence transcriptions contain numerous ungrammatical sentences.

Annotators are invited to revise and condense pre-existing meeting *extracts* in order to produce fluent and concise summaries. Each meeting extract is itself a selection of the most important

Annotation	Transcription	Extractive Summary	Abstractive Summary
			not listening and - so who is - he - in this context [laughter] ?
843.28	846.14	fe016	Yeah , there's a lot of pronoun - I believe it .
844.22	844.66	mn017	OK .
844.98	855.73	mn017	So I was just realizing - we've You guys have been talking about " he " um for at least uh , I don't know , three - three four minutes without ever mentioning the person's name again .
855.97	863.21	fe016	Yeah . Actually to make it worse , uh , Morgan uses " you " and " you " - pau- with

Figure B.1: The transcription, the unedited (“extractive”) summary, and the edited (“abstractive”) summary are accessible in three different panels.

sentences of a full meeting *transcription*. Extracts are 2,000 words long on average, while transcriptions are about 20,000 words long. Since extracted sentences were not altered in any way by the human annotators who selected them, they tend to be difficult to read, and generally contain many unimportant and removable phrases. In order to produce readable and pertinent meeting *abstracts*, annotators are expected to edit each extracted sentence in stages, and are prompted for specific operations to perform in each stage. Stages include, for instance, the deletion of filler words (e.g., “um” or “uh”), and the correction of ungrammatical constructions. A detailed description of each stage is provided in the second section of this document.

Annotators have access to the word-level transcription and audio recording of each utterance, and are able to navigate forward and backward in each meeting. Before starting to edit any meeting extract, they are expected to **read the entire extract** in order to gain an overall understanding of the meeting. If any sentence of the extract is ambiguous, they should refer to the full meeting transcription—where that sentence will appear in its full original context—in order to gain a better insight into the conversation. As shown in Figure B.1, annotators can access three documents through three different panels at any point in time: the full meeting transcription (“Transcription”), the meeting extract (“Extractive Summary”), and the meeting abstract built so far (“Abstractive Summary”). Each panel contains four columns: start time, end time, anonymized speaker identifier (e.g., “mn017”), and transcription.

A click on the “Annotation” pane lets annotators access the different edition stages. When annotators are done with an edition stage, they should click on “commit” to be taken to the next stage (see Figure B.2). Only deletion operations are possible in the first five stages: to delete words or phrases, annotators simply need to click on any word or select any range of words. As shown in Figure B.2, words marked for deletion appear in red, and disappear from the sentence in subsequent

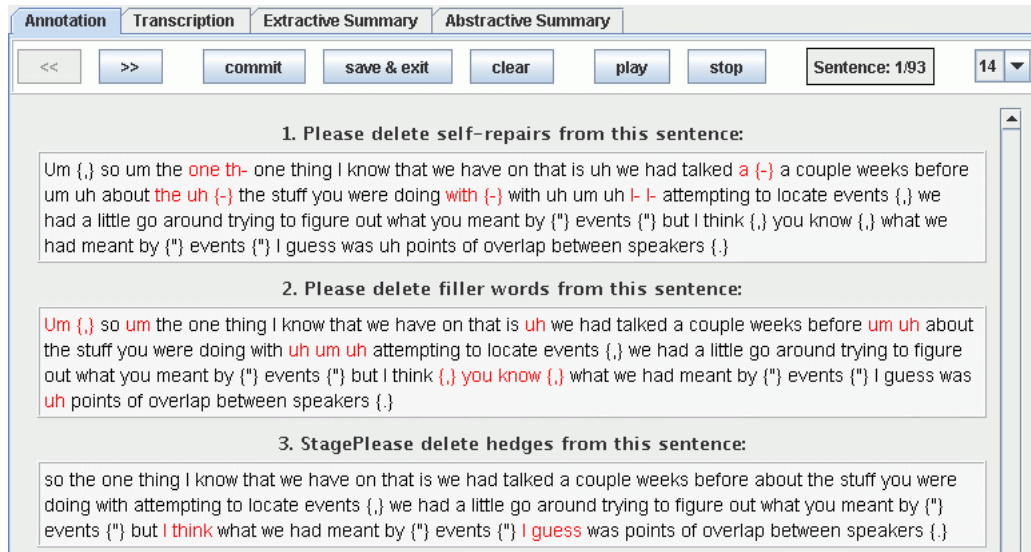


Figure B.2: The annotation pane (punctuation marks appear surrounded by brackets, e.g. “{.}”, in order to facilitate their selection.)

stages.

The upper part of the annotation pane contains seven buttons, and a box that lets users change font size. Besides “commit” explained previously, the “save & quit” button lets annotators store all edits to file and exit the annotation interface. Note that revisions are regularly saved without any user intervention, and this button only ensures that the very last changes are also saved. Annotators should always click this button to stop or interrupt any annotation session. By doing so, they ensure that all their edits are again available as they return to the annotation page. A click on “clear” lets users clear all edits pertaining to the current extract sentence, and step back to the first stage. Note that it is also possible to step back to previous stages without resorting to the “clear” button: for instance, if an annotator reaches Stage 3 as in Figure B.2, but clicks on the word “speakers” (last word of the sentence) in the panel corresponding to Stage 1, the deletion of the word will not only update Stage 1 (i.e., the word “speakers” will appear in red), but also propagate through Stages 2 and 3 (i.e., the word will disappear in these stages). Similarly, a new click on the same word in Stage 1 will blacken the word in Stage 1, and make “speakers” appear again in Stages 2 to 3.

The “play” and “stop” buttons respectively allow users to start and stop the audio recording of the current sentence of the extract. Finally, once annotators have completed all edit stages of a given

sentence, they simply need to click on the forward button (“>>”) to step to the next one (similarly, the “<<” button allows them to step back and modify their previous edits). Once the revision of part or all sentences is completed, annotators should read in the last panel (“Abstractive Summary”) the abstract they produced, and make sure that the main points of the meeting are present in the summary, and that this summary does not contain any superfluous information. If needed, they should modify their annotation accordingly. In summary, the creation of a meeting abstract should proceed as follows:

1. Read the entire meeting extract (panel “Extractive Summary”), and refer to the full transcription if needed (“Transcription”).
2. Revise sentences in order, one at a time (“Annotation”).
3. Check the quality of produced abstract (“Abstractive Summary”), and change revisions if needed (“Annotation”).
4. Click on “save & exit” to temporarily suspend or end the annotation of a meeting.

Annotators should note one final important point regarding the different edits stages: while the decision of what should or should not be altered in the sentence is quite subjective—i.e., there may be different reasonable ways of performing the requested edits, especially when it comes to determining what information is important—annotators should ensure as best as they can that their edits *do not make the sentence ungrammatical, or cause the revised sentence to bear a meaning that is significantly different from the unedited sentence*. When annotators encounter an extracted sentence that is originally ungrammatical, they should leave ungrammatical construction unchanged until they are specifically asked (in Stage 6) to edit the sentence to make it grammatically correct.

The remaining part of these annotation guidelines describes the seven edition stages in details.

B.2 Summarization in Stages

Summarization must be performed in seven stages. Annotators should *make sure that each revision operation is performed at the correct stage*, e.g., they should avoid removing disfluencies when asked to correct ungrammatical constructions. Note that there may be cases where a given input

sentence doesn't require revisions in all stages. For instance, if the sentence doesn't contain any self-repair (Stage 1), the annotator should simply click "commit" to proceed to the next stage without performing any edit.

Stage 1: Self-Repairs

By self-repairs, we refer to portions of speech in which a speaker's utterance is not complete and fluent, i.e., the speaker corrected or altered the utterance, or abandoned it entirely. Self-repairs may contain four parts, as it is the case in "I think one thing - I mean , one of the things ...". First, the *reparandum* is the portion of speech that later gets corrected or abandoned (here, "one thing"). Second, the *interruption point* is the point where the speaker breaks off, which may appear as an hyphen ("-") in the transcription. Third, an optional *editing phrase* , which consists of an overt statement (e.g., "I mean", "err", "rather") recognizing the existence of a disfluency. Finally, the correction consists of the portion of the utterance that contains what the speaker intended to say ("one of the things"). The three former parts are generally considered removable, and annotators should remove them if deemed unnecessary, ensuring if possible that the resulting edited sentence is grammatical. In the example, edits would result in a sentence starting with "I think one of the things ...". Note that grammaticality is a helpful cue to determine how to distinguish removable repair words from other words: in particular, "I think" in the example above is not part of the reparandum, since the correction "one the the things" is a continuation of "I think", hence the two first words should not be deleted.

It is sometimes quite difficult to distinguish repair words from other words only based on the transcription, and annotators are encouraged to listen to the audio recording if they are not sure.

In the example of Figure B.3, the annotator determined that "one th-", "a {-}", "the uh {-}", "with {-}", and "I- I-" are removable in the given sentence. Note that interruption points also sometimes appear at the end of incomplete words (e.g., "th-"). Finally, the second example in Figure B.3 contains an example of a self-repair containing an editing phrase ("I mean"), which generally should also be deleted.

Phrases to delete in a self-repair may be arbitrarily long. Figure B.4 contains an example where an entire clause is abandoned: "so that the meeting meetings will probably ultimately". Self-repairs may also come in a sequence, e.g. the abandoned clause mentioned before is followed by three

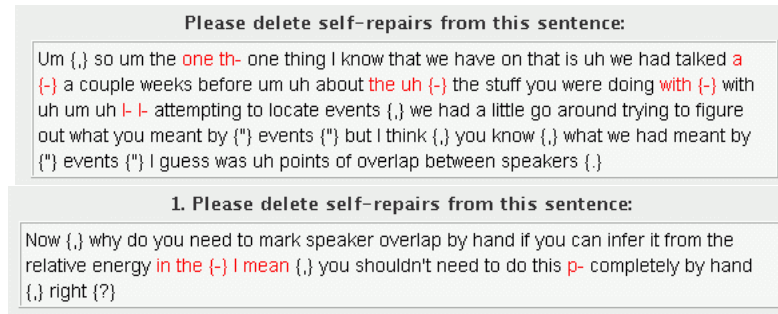


Figure B.3: Reparanda, interruption points, and repairs. The second example contains an editing phrase (“I mean”).

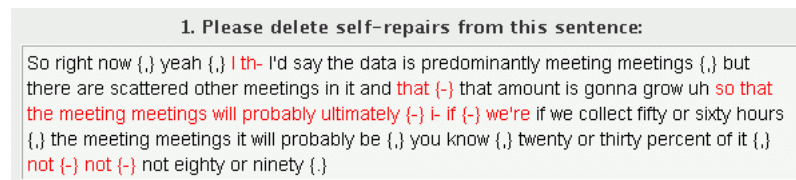


Figure B.4: Self-repairs may be arbitrarily long.

other abandoned phrases (“i-”, “if {-}” and “we’re-”).

After clicking on “commit”, the annotator is shown the same utterance in a new panel, but words that appeared in red are no longer present.

Stage 2: Filler Words

At this stage, annotators should remove filler words, which generally add no meaning to the speaker’s utterance. These constructions are words or phrases such as “um”, “uh”, “mmm”, “eh”, “ah”, “I mean”, “okay”, “so”, “you know”, “basically”, “well”, “but”, “like” that unskilled or unprepared speakers may utter, often unconsciously, while thinking what to say next and sometimes to maintain the floor. Note that, for certain filler words or phrases (“I mean”, “okay”, “so”, etc.), it must be determined from context whether these words are indeed fillers. For instance, while many occurrences of “I mean” in conversational speech are fillers, “mean” in “I say what I mean” has its literal meaning and is not a filler word. In some cases, it may be helpful to listen to the audio recording to make the distinction. Annotators should remove all filler words that do not alter the meaning of the summary. In the example of Figure B.5, an annotator determined that occurrences of “um”, “uh”,

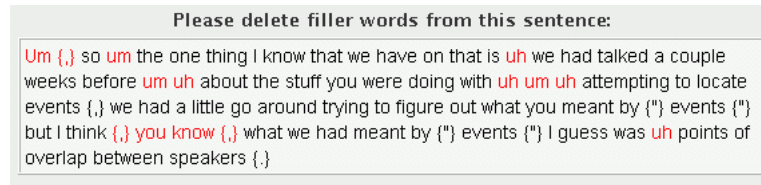


Figure B.5: filler words, such as “um”, “uh”, and “you know”.

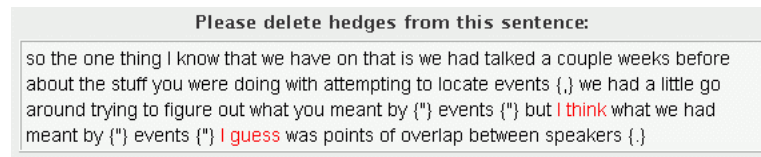


Figure B.6: hedges “I think” and “you know”.

and “you know” are fillers that add no meaning to the sentence.

Stage 3: Hedges

Hedges are words or phrases such as “I think”, “I guess”, “maybe”, “perhaps”, “kind of”, “sort of”, “I don’t know”, “somewhat”, “more or less”, “pretty much” and “rather” that mitigate or weaken the certainty of a statement. Often present in conversational speech, they generally mark politeness or deference, and seldom affect the meaning of the utterance, and are admittedly good candidates for removal. If an annotator thinks that the mitigating factor introduced by a hedge is particularly significant, e.g. when speakers are particularly uncertain about their statements, the annotator may decide to retain the hedge in the sentence. In Figure B.6, an annotator decided that hedges (“I think” and “I guess”) can safely be removed without significant change in meaning.

Stages 4 and 5: Unimportant Information

Any word or phrase that is deemed optional may be deleted in Stages 4 or 5. In Stage 4, annotators should only delete words or phrases that they are absolutely sure should not be included in the summary (because they add no or little factual information to the sentence, or for other reasons). For example, “guys” in “you guys”, “so I was just realizing”, “the other thing that occurred to me”, etc. add no useful information to the sentences of Figure B.7, thus these phrases were deleted by annotators.

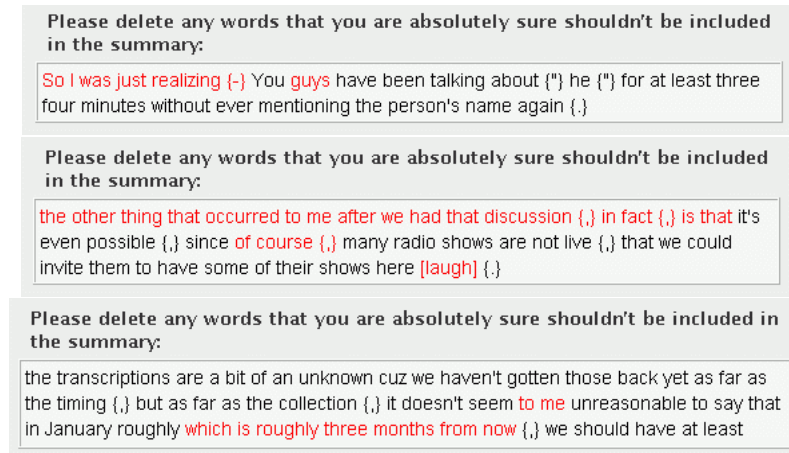


Figure B.7: Phrases that add no factual or useful information should be deleted.

Annotators can also assume that basic facts about meeting, such as meeting time and location, are known from context. Thus, the clause “which is roughly three months from now” of Figure B.7 may also be deleted, since it adds no extra information to “in January”. In general, annotators may delete information that is assumed to be general world knowledge, and thus doesn’t need to be repeated in the summary. For example, it is appropriate to compress “U.S. President George Bush” into “President George Bush”, “President Bush”, “George Bush”, and even “Bush”, if they think the person referred to is identifiable from context.

In Stage 5, annotators should delete any words or phrases that carry information that is not critical to the summary, i.e., that are not needed in order to gain an overall understanding of the meeting. In Figure B.8, an annotator determined that the fact that “radio shows are not live” has no overall significance in the meeting, and should not appear in a concise summary of the meeting. In the second example of the figure, one annotator found that knowing that the amount of data “is gonna grow” is redundant with respect to the remaining part of the sentence, and that the finality (“meeting meetings will probably be twenty or thirty percent”) is in this case more important than words in red. In the third example, one annotator found that it is important to know that “we’d have three hundred examples” than to know how it will be possible to collect that many examples (i.e., “by just getting the speakers overlaps ...”).

The two last stages (6 and 7) allow free text editing, i.e., the user can insert, delete, and substitute any word as in any text editor. Annotators should avoid large-scale rewordings unless they have

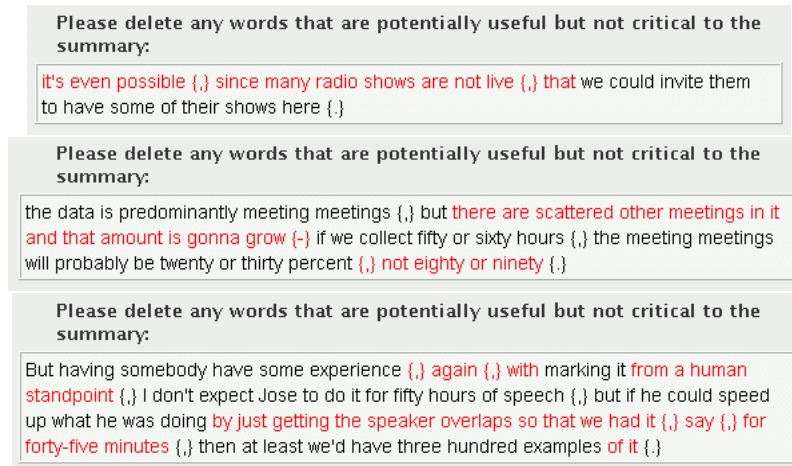


Figure B.8: Factual information may still be deleted if it is not particularly important.

good reasons to perform them. The number of word substitutions and insertions should remain relatively low if possible.

Note that once the user has reached Stage 6, all subsequent modifications performed in Stages 1-5 will not affect Stages 6 and 7 (i.e., clicking on a word in Stage 5 won't insert or remove any word in Stage 6). Hence, the user should make sure that she or he is satisfied with all edits in Stages 1-5 before committing Stage 5.

Stage 6: Ungrammatical Constructions

Annotator are asked here to edit the text and remove any ungrammatical constructions, which may force them to substitute or insert words (in addition to deleting words). They should strive to minimize the number of word editions while trying to preserve the same meaning. For instance, the utterance "I am wondering what do you think." should preferably be changed into "I am wondering what you think." instead of "what do you think?".

Stage 7: Fluency and Paraphrasing

Finally, annotators are given the opportunity to insert and substitute words in the sentence in order to further improve its fluency and conciseness. Certain condensations not allowed in previous stages are possible here, e.g., replacing a long phrase by an equivalent but shorter expression (e.g., "in a near future" may be replaced by "soon"). In Figure B.9, "the person's" can safely be replaced by

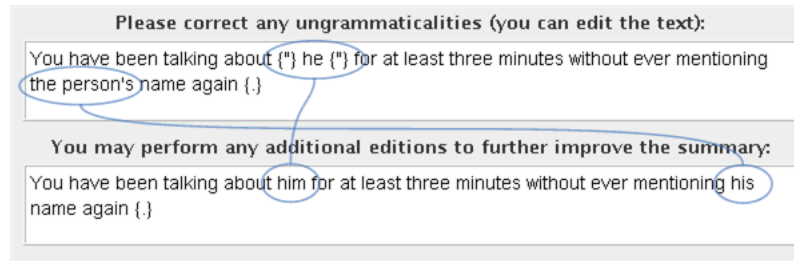


Figure B.9: Any deletion, substitution, and insertion may be performed in the last stage.



Figure B.10: improve style and fluency.

“his” without loss of information (it is known from context that the speaker refers to a male person).

Stage 7 can also be used to perform stylistic correction and improve fluency, e.g., remove a double negation, as in Figure B.10.

Stage 7 is the appropriate place to perform revisions that should normally take place in Stages 1-5, but that cannot be performed in those stages because they would require word insertions and substitutions. For instance, the phrase “I don’t think it is going to be” in Figure B.11 contains a hedge, but one cannot removed it without word substitutions (as in “it is not going to be”). Note that replacing the original phrase with “it is going to be” is incorrect, since it completely changes the meaning of the sentence. In this case, the annotator should leave the hedge unchanged in Stage 3, and remove it in Stage 7.

Please delete any words that you are absolutely sure shouldn't be included in the summary:

But at any rate {}, once they get that sorted out {}, they're making cassettes there {}, then they're handing it to someone who is doing it {}, and I don't think it's going to be that much more of a deal for them to do thirty hours then to do one hour {}. It's not going to be thirty

Figure B.11: if any removal of self-repair words (Stage 1), filler words (Stage 2), hedges (Stage 3), etc., also requires word insertions or substitutions to preserve the same meaning, one should perform suitable revisions in Stage 7 instead of Stages 1-5.

Appendix C

Sample Outputs

This appendix provides a set of sentence compressions for test sets of the Ziff-Davis corpus and the ICSI Meeting corpus.

C.1 Ziff-Davis

We provide here sentence compressions generated by the generative model described in Section 4.2.2 and by human abstractors. The 32 sentences represent the entire test set selected by (Knight and Marcu, 2000).

Table C.1: Sentence compressions for the Ziff-Davis test set.

ID	Transcription
<i>Input</i>	like facelift , much of atm 's screen performance depends on the underlying application .
Human	much of atm 's performance depends on the underlying application .
Markov	much of atm 's screen performance depends on the underlying application .
<i>Input</i>	another slight downside is that envelopes must be fed manually .
Human	envelopes must be fed manually .
Markov	another slight downside is that envelopes must be fed manually .
<i>Input</i>	finally , another advantage of broadband is distance .
Human	another advantage is distance .
Markov	another advantage of broadband is distance .
<i>Input</i>	prices range from \$5,000 for a microvax 2000 to \$179,000 for the vax 8000 or higher series .

ID	Transcription
Human	prices range from \$5,000 to \$179,000 .
Markov	prices range from \$5,000 for a microvax for the vax .
<i>Input</i>	with three exabyte drives , it is priced at \$17,850 .
Human	with three exabyte drives it is priced at \$17,850 .
Markov	it is priced at \$17,850 .
<i>Input</i>	the chemical etching process used for glare protection is effective and will help if your office has the fluorescent-light overkill that 's typical in offices .
Human	glare protection is effective .
Markov	the chemical etching process used for glare protection is effective .
<i>Input</i>	apparel makers use them to design clothes and to quickly produce and deliver the best-selling garments .
Human	apparel makers use them to design clothes .
Markov	apparel makers use to design clothes and to quickly produce and deliver the best-selling garments .
<i>Input</i>	replacement boards , such as those offered by intel , quadram and sota , require removal of the 8088 or 8086 processor .
Human	replacement boards require removal of the 8088 or 8086 processor .
Markov	replacement boards , such as those offered by intel require removal of the 8088 or 8086 processor .
<i>Input</i>	* 68000 sweden ab of uppsala , sweden , introduced the teleserve , an integrated answering machine and voice-message handler that links a macintosh to touch-tone phones .
Human	68000 sweden ab introduced the teleserve integrated answering machine and voice-message handler .
Markov	* 68000 introduced the teleserve to touch-tone phones .
<i>Input</i>	microsoft alone has lost one-third of its market value .
Human	microsoft lost one-third of its market value .
Markov	microsoft has lost one-third of its market value .
<i>Input</i>	the sas screen is divided into three sections : one for writing programs , one for the system 's response as it executes the program , and a third for output tables and charts .
Human	the sas screen is divided into three sections .
Markov	the sas screen is divided into three sections : one for writing programs , one for the system 's response as it executes the program , and a third for output tables and charts .

ID	Transcription
<i>Input</i>	also , trackstar supports only the critical path method (cpm) of project scheduling .
Human	trackstar supports the critical path method of project scheduling .
Markov	trackstar supports the critical path method .
<i>Input</i>	finally , the integrity of database must be a part of the database definition , enforced by the environment of the database-management system – not separately enforced by each application .
Human	integrity must be a part of the database definition , enforced by the environment of the database-management system .
Markov	the integrity of database must be a part of the database definition , enforced by the environment of the database-management system enforced by each application .
<i>Input</i>	the scamp module , designed and built by unisys and based on an intel process , contains the entire 48-bit a-series processor .
Human	the scamp module contains the entire 48-bit a-series processor .
Markov	the scamp module contains the entire 48-bit a-series processor .
<i>Input</i>	the first new product , atf prototype , is a line of digital postscript typefaces that will be sold in packages of up to six fonts .
Human	atf prototype is a line of digital postscript typefaces that will be sold in packages of up to six fonts .
Markov	the product is a line of digital postscript typefaces that will be sold in packages of up to six fonts .
<i>Input</i>	although it bears the ibm name , its synthesizer circuitry is identical to that of a yamaha fb-01 , a popular eight-voice synthesizer module .
Human	the synthesizer circuitry is identical to a yamaha fb-01 eight-voice synthesizer module .
Markov	synthesizer circuitry is identical to that of a yamaha fb-01 .
<i>Input</i>	the cad-based system can generate 1.49 million different drawing and specification combinations for a 56- by 92-foot building , pritchard said .
Human	the system can generate 1.49 million different drawing and specification combinations for a 56- by 92-foot building .
Markov	the cad-based system can generate 1.49 million different drawing and specification combinations for a 56- by 92-foot building .
<i>Input</i>	working in the score is not an intuitive process ; it takes a lot of practice .
Human	working in the score is not intuitive .
Markov	it takes a lot of practice .

ID	Transcription
<i>Input</i>	the faster transfer rate is made possible by an mti-proprietary data buffering algorithm that off-loads lock-manager functions from the q-bus host , raimondi said .
Human	the algorithm off-loads lock-manager functions from the q-bus host .
Markov	the faster transfer rate is made possible by an mti-proprietary data buffering algorithm that off-loads lock-manager functions from the q-bus host .
<i>Input</i>	the utilities will be bundled with quickdex ii in a \$90 package called super quickdex , which is expected to ship in late summer .
Human	the utilities will be bundled with quickdex ii .
Markov	the utilities will be bundled with quickdex ii .
<i>Input</i>	it implements the common cryptographic architecture and offers a comprehensive set of security products that allow users to implement end-to-end secure systems with ibm components .
Human	it implements the common cryptographic architecture and offers a comprehensive set of security products that allow users to implement end-to-end secure systems with ibm components .
Markov	it implements the common cryptographic architecture .
<i>Input</i>	actual hardware and maintenance costs have decreased 40 percent over the years , while the number of users supported has increased by over 300 percent since 1983 .
Human	actual hardware and maintenance costs have decreased .
Markov	actual hardware and maintenance costs have decreased 40 percent over the years .
<i>Input</i>	beyond that basic level , the operations of the three products vary widely .
Human	the operations of the three products vary widely .
Markov	the operations of the three products vary widely .
<i>Input</i>	establishing external (to the application) reference files allows controlled and timely access to all reference data (master files) , regardless of location or application , and enhance data integrity .
Human	external reference files enhance data integrity .
Markov	establishing external reference files allows controlled and timely access to all reference data and enhance data integrity .
<i>Input</i>	the discounted package for the sparcserver 470 is priced at \$89,900 , down from the regular \$107,795 .
Human	the sparcserver 470 is priced at \$89,900 , down from the regular \$107,795 .

ID	Transcription
Markov	the discounted package for the sparcserver 470 is at \$89,900 .
<i>Input</i>	the source code , which is available for c , fortran , ada or vhdl , can be compiled and executed on the same system or ported to other target platforms .
Human	the source code is available for c , fortran , ada or vhdl .
Markov	the source code can be compiled and executed on the same system or ported to other target platforms .
<i>Input</i>	arborscan is reliable and worked accurately in testing , but it produces very large dxf files .
Human	arborscan produces very large dxf files .
Markov	arborscan is reliable and worked accurately .
<i>Input</i>	examples for editors are applicable to awk patterns , grep and egrep .
Human	examples are applicable to awk patterns , grep and egrep .
Markov	examples for editors are applicable to awk patterns .
<i>Input</i>	many debugging features , including user-defined break points and variable-watching and message-watching windows , have been added .
Human	many debugging features have been added .
Markov	many debugging features have been added .
<i>Input</i>	standard fonts include courier , line printer (a sans serif face) , times roman and prestige elite .
Human	standard fonts include courier , line printer , times roman and prestige elite .
Markov	standard fonts include courier , line printer , times roman and prestige elite .
<i>Input</i>	the dos batch language has major limitations , of course .
Human	the dos batch language has major limitations .
Markov	the dos batch language has major limitations .
<i>Input</i>	when it runs the merge , the word processor will look for headings in the first line of the document to match those it finds in the form letter .
Human	the word processor will look for headings in the first line of the document to match those it finds in the form letter .
Markov	the word processor will look for headings in the first line of the document to match those it finds in the form letter .

C.2 ICSI Meeting Corpus

We provide here sentence compressions generated by the minimum-WER discriminative model (“hyp”) described in Section 4.4.2 and by a human annotator.

Table C.2: Sentence compressions for the test set of the ICSI meeting corpus.

ID	Transcription
<i>Input</i>	ok well , so ralf and tilman are here .
Markov	ralf and tilman are here .
Human	ralf and tilman are here .
<i>Input</i>	i thought two things uh we ’ll introduce ourselves and what we do .
Markov	i thought two things we ’ll introduce ourselves .
Human	we ’ll introduce ourselves .
<i>Input</i>	and um we already talked with andreas , thilo and david and some lines of code were already written today and almost tested and just going to say we have um again the recognizer to parser thing where we ’re working on and that should be no problem and then that can be sort of developed uh as needed when we enter the tourism domain .
Markov	we talked with andreas thilo and david and lines of code were written today and tested .
Human	we already talked with andreas , thilo and david some lines of code were written today and almost tested we have the recognizer to parser that can be developed when we enter the tourism domain .
<i>Input</i>	and um there one of our diligent workers has to sort of volunteer to look over tilman ’s shoulder while he is changing the grammars to english because we face two ways .
Markov	there one of our diligent workers has to volunteer to look over tilman ’s shoulder .
Human	one of our workers has to look over tilman ’s shoulder while he is changing the grammars to english .
<i>Input</i>	either we do a syllable concatenating um grammar for the english generation which is sort of starting from scratch and doing it the easy way , or we simply adopt the ah um more in-depth um style that is implemented in the german system and um are then able not only to produce strings but also the syntactic tree that is underneath in the syntactic structure which is the way we decided we were going to go because a , it ’s easier in the beginning and um it does require some knowledge of those grammars and some linguistic background .

ID	Transcription
Markov	we do a syllable concatenating grammar for the english generation but also the syntactic tree that is underneath in the syntactic structure .
Human	either we do a syllable concatenating grammar for the english generation , or we adopt the more in-depth style in the german system and are then able not only to produce strings but also the syntactic tree it 's easier in the beginning and it does require knowledge of those grammars and linguistic background .
<i>Input</i>	johno , are you going to have some time to do that uh with these guys ?
Markov	johno are you going to have some time to do that with these guys ?
Human	johno , are you going to have time to do that ?
<i>Input</i>	the ultimate goal is that before they leave we can run through the entire system input through output on at least one or two sample things .
Markov	the ultimate goal is before they leave we can run through the entire system input through output .
Human	the goal is that we can run the entire system on one or two sample things .
<i>Input</i>	and by virtue of doing that then in this case johno will have acquired the knowledge of how to extend it .
Markov	johno will have acquired the knowledge of how to extend it .
Human	then johno will have acquired the knowledge of how to extend it .
<i>Input</i>	when needed , if needed , when wanted and so forth .
Markov	when needed if needed when wanted .
Human	when needed .
<i>Input</i>	and um also um ralf has hooked up with david and you 're going to continue either all through tonight or tomorrow on whatever to get the er parser interface working .
Markov	and ralf has hooked up with david and you 're going to continue either through tonight .
Human	ralf has hooked up with david and you 're going to get the parser interface working .
<i>Input</i>	ok , so sort of one branch is to get us caught up on what 's going on .
Markov	one branch is to get us caught up on what 's going on .
Human	get us caught up on what 's going on .
<i>Input</i>	also of course it would be really nice to know what the plans are , in addition to what 's sort of already in code .
Markov	it would be nice to know what the plans are , in addition .
Human	it would be nice to know the plans , in addition to what 's already in code .

ID	Transcription
<i>Input</i>	was there uh a time when we were set up to do that ?
Markov	was there a time when we were set up to do that ?
Human	was there a time to do that ?
<i>Input</i>	ok , so we 'll find a time later in the week to uh get together and talk about your understanding of what smartkom plans are .
Markov	we 'll find a time to get together and talk about your understanding of what smartkom plans are .
Human	we 'll find a time to talk about smartkom plans .
<i>Input</i>	thursday around eleven ?
Markov	thursday around eleven ?
Human	thursday around eleven ?
<i>Input</i>	ok so facing to what we 've sort of been doing here
Markov	facing to what we 've been doing here
Human	what we 've been doing
<i>Input</i>	um well for one thing we 're also using this room to collect data .
Markov	we 're using this room to collect data .
Human	we 're using this room to collect data .
<i>Input</i>	no not meeting data but sort ah our version of a wizard experiment not like the ones in munich but pretty close to it .
Markov	no meeting data but sort our version of a wizard experiment .
Human	not meeting data but our version of a wizard experiment not like the ones in munich but close .
<i>Input</i>	the major difference to the munich ones is that we do it via the telephone even though all the recording is done here and so it 's sort of a computer call system that gives you tourist information tells you how to get places .
Markov	the major difference to the munich ones is we do it via the telephone though all the recording is done and so .
Human	the major difference to the munich ones is that we do it via the telephone even though all the recording is done here it 's a computer call system that gives you tourist information .
<i>Input</i>	and it breaks halfway through the experiment and a human operator comes on .
Markov	it breaks through the experiment and a human operator comes on .
Human	it breaks halfway through and a human operator comes on .

ID	Transcription
<i>Input</i>	and we 're setting it up so that we hope to implant certain intentions in people .
Markov	we 're setting it up so we hope to implant certain intentions in people .
Human	we hope to implant certain intentions .
<i>Input</i>	but what we actually observed in heidelberg is that most people when they want to go there they actually do n't want to enter , because it 's not really interesting .
Markov	is that most people when they want to go there they do n't want to enter .
Human	we observed in heidelberg that most people do n't want to enter because it 's not interesting .
<i>Input</i>	what let 's say a simple parse from an utterance wo n't really give us is what the person actually wants .
Markov	what wo n't give us is what the person wants .
Human	a simple parse from an utterance wo n't give us what the person wants .
<i>Input</i>	there are all kinds of decisions that we have identified in terms of getting to places and in terms of finding information about things .
Markov	there are all kinds of decisions that we have identified in terms of getting to places .
Human	there are decisions in terms of getting places and finding information .
<i>Input</i>	and then we 've identified more or less the extra-linguistic parameters that may play a role .
Markov	we 've identified more or less the extra-linguistic parameters .
Human	we 've identified the extra-linguistic parameters .
<i>Input</i>	and we also want to look closely on the linguistic information what we can get from the utterance .
Markov	we want to look closely on the linguistic information what we can get from the utterance .
Human	we also want to look closely on the linguistic information .
<i>Input</i>	and um so the idea is to construct uh um suitable interfaces and a belief-net for a module that actually tries to guess what the underlying intention was .
Markov	the idea is to construct suitable interfaces .
Human	the idea is to construct suitable interfaces and a belief-net for a module that tries to guess the underlying intention .
<i>Input</i>	and then enrich or augment the m-three-l structures with what more it sort of got out of that utterance .
Markov	enrich or augment the m-three-l structures with what .

ID	Transcription
Human	and then enrich the m-three-l structures with what it got out of that utterance .
<i>Input</i>	you know , that person does n't want to enter .
Markov	that person does n't want to enter .
Human	that person does n't want to enter .
<i>Input</i>	these bits of additional information are going to be embedded into the m-three-l structure in an sort of subfield that we have reserved .
Markov	these bits of additional information are going to be embedded into the m-three-l structure in of subfield we have reserved .
Human	these bits of additional information are going to be embedded into the m-three-l structure in an subfield that we have reserved .
<i>Input</i>	if the action planner does something with it , great .
Markov	if the action planner does something with it .
Human	if the action planner does something with it
<i>Input</i>	i mean ultimately if you can offer that information , somebody 's going to do something with it sooner or later .
Markov	if you can offer that information somebody 's going to do something with it .
Human	if you can offer that information , somebody 's going to do something with it .
<i>Input</i>	so that 's a functionality that does n't exist yet to do that dynamically , but if we can offer it that distinction , maybe somebody will go ahead and implement it .
Markov	that 's a functionality that does n't exist yet to do that .
Human	that 's a functionality that does n't exist yet , but if we offer that distinction , maybe somebody will implement it .
<i>Input</i>	well the obvious one would be if you envision this as a module within smartkom , where exactly would that sit ?
Markov	where exactly would that sit ?
Human	as a module within smartkom , where would that sit ?
<i>Input</i>	um so far i 've thought of it as sort of adding it onto the modeler knowledge module .
Markov	i 've as adding it onto the modeler knowledge module .
Human	adding it onto the modeler knowledge module .
<i>Input</i>	so this is one that already adds additional information but it could sit anywhere in the attention-recognition i mean basically this is what attention-recognition literally sort of can
Markov	this is one that but it could sit anywhere in the attention-recognition

ID	Transcription
Human	this already adds additional information but it could sit anywhere in the attention-recognition
<i>Input</i>	well it 's supposed to do .
Markov	it 's supposed to do .
Human	it 's supposed to do .
<i>Input</i>	that 's what it should do .
Markov	that 's what it should do .
Human	that 's what it should do .
<i>Input</i>	well from my understanding of what the people at phillips were originally trying to do does n't seem to quite fit into smartkom currently so what they 're really doing right now is only selecting among the alternatives , the hypotheses that they 're given enriched by the domain knowledge and the um discourse modeler and so on .
Markov	from my understanding of what the people at phillips were trying to do does n't seem to fit into smartkom currently .
Human	what the people at phillips were originally trying does n't fit into smartkom currently so they 're selecting alternatives , hypotheses enriched by domain knowledge and the discourse modeler .
<i>Input</i>	so if this is additional information that could be merged in by them .
Markov	if this is additional information could be merged in by them .
Human	this is additional information that could be merged in by them .
<i>Input</i>	and then it would be available to action planning and others .
Markov	it would be available to action planning and others .
Human	and then it would be available to action planning .
<i>Input</i>	so well , let me expand on that a little bit from the point of view of the generation .
Markov	let me expand on that from the point of view of the generation .
Human	let me expand on that from the point of view of the generation .
<i>Input</i>	so the idea is that we 've actually got this all laid out and we could show it to you um robert did n't bring it today
Markov	is we 've got this all laid out robert did n't bring it today
Human	we 've got this all laid out robert did n't bring it today
<i>Input</i>	there 's a first cut at a belief-net that is n't fully uh instantiated , and in particular some of the combination rules and ways of getting the conditional probabilities are n't there .

ID	Transcription
Markov	there 's a first cut at a belief-net that is n't instantiated .
Human	there 's a belief-net that is n't fully instantiated , some combination rules and ways of getting conditional probabilities are n't there .
<i>Input</i>	so one of the decisions is what we call this ave thing .
Markov	one of the decisions is what we call this ave thing .
Human	one decisions is what we call this ave thing .
<i>Input</i>	do you want to um access , view or enter a thing .
Markov	do you want to access view or enter a thing .
Human	do you want to access , view or enter .
<i>Input</i>	so that 's a discrete decision .
Markov	that 's a discrete decision .
Human	that 's a discrete decision .
<i>Input</i>	there are only three possibilities and what one would like is for this uh , knowledge modeling module to add which of those it is and give it to the planner .
Markov	there are only three possibilities and what one would like is for this knowledge modeling module to add and give it to the planner .
Human	one would like this knowledge modeling module to add which it is and give it to the planner .
<i>Input</i>	but , uh the current design suggests that if it seems to be an important decision and if the belief-net is equivocal so that it does n't say that one of these is much more probable than the other , then an option is to go back and ask for the information you want .
Markov	the current design suggests that if to be an important decision you want .
Human	the current design suggests that if it seems to be an important decision and if the belief-net is equivocal , then an option is to ask for the information you want .
<i>Input</i>	now there are two ways one can imagine doing that .
Markov	there are two ways one can imagine doing that .
Human	there are two ways one can imagine doing that .
<i>Input</i>	but for a full system , then one might very well formulate a query , give it to the dialogue planner and say this , you know are you planning to enter ?
Markov	for a full system one formulate a query , give it to the dialogue planner say this ?
Human	for a full system , one might formulate a query , give it to the dialogue planner
<i>Input</i>	so under that model then , there would be a loop in which this thing would formulate a query , presumably give it to you .

ID	Transcription
Markov	there would be a loop in which would formulate a query presumably give it to you .
Human	there would be a loop which would formulate a query , give it to you .
<i>Input</i>	that would get expressed and then hopefully you know , you 'd get an answer back .
Markov	that would get expressed you 'd get an answer back .
Human	that would get expressed and then , you 'd get an answer back .
<i>Input</i>	we probably wo n't do this early on , because the current focus is more on the decision making and stuff like that .
Markov	we wo n't do this because the current focus is more .
Human	we probably wo n't do this early on , because the current focus is on the decision making .
<i>Input</i>	so my suggestion then is that you um look into the currently ongoing discussion about how the action plans are supposed to look like .
Markov	is that you look into the ongoing discussion about how the action plans are supposed to look like .
Human	my suggestion then is that you look into the ongoing discussion about the action plans .
<i>Input</i>	and they 're currently in the process of agreeing on an xm l-ification of how of network a like something um state-transition dialogues would proceed .
Markov	they 're in the process of agreeing on an xm l-ification of how state-transition dialogues would proceed .
Human	they 're agreeing on an xm l-ification of how network a state-transition dialogues would proceed .
<i>Input</i>	well uh marcus lerkult is actually implementing that stuff and marcus and michael together are um leading the discussion there , yeah .
Markov	marcus lerkult is implementing marcus and michael together are leading the discussion .
Human	marcus lerkult is implementing that and marcus and michael are leading the discussion .
<i>Input</i>	so we have to get in on that .
Markov	we have to get in on that .
Human	we have to get in on that .
<i>Input</i>	because um partly those are like x-schemas .
Markov	because those are like x-schemas .
Human	those are like x-schemas .
<i>Input</i>	and it may be that um we should early on make sure that they have the flexibility that we need .

ID	Transcription
Markov	it may be we should make sure they have the flexibility that we need .
Human	we should make sure that they have the flexibility that we need .
<i>Input</i>	the word action , ok , is what 's ambiguous here .
Markov	the word action is what 's ambiguous .
Human	the word action , is what 's ambiguous .
<i>Input</i>	so that 's that form of planning , and action , and a route planner and gis , all sort of stuff .
Markov	that 's of action a route planner gis of stuff .
Human	that 's that form of planning , action , a route planner and gis
<i>Input</i>	no , in smartkom terminology that 's um called a function that 's modeled by a function modeler .
Markov	that 's called a function that 's modeled by a function modeler .
Human	in smartkom terminology that 's a function modeled by a function modeler .
<i>Input</i>	that 's simply a functionality that you give data as in a query and then you get back from that mmm , a functioning model um which might be a planner or a vcr or whatever .
Markov	that 's a functionality that you give data as in a query .
Human	that 's a functionality that you give data and then you get back a functioning model .
<i>Input</i>	so action here means dialogue act .
Markov	action means dialogue act .
Human	action here means dialogue act .
<i>Input</i>	um i think that 's not going to be good enough .
Markov	that 's not going to be good enough .
Human	i think that 's not going to be good enough .
<i>Input</i>	so i think the idea of having a , you know , transition diagram for the grammar of conversations is a good idea .
Markov	the idea of having a transition diagram for the grammar of conversations is a good idea .
Human	having a transition diagram for the grammar of conversations is good .
<i>Input</i>	and i think that we definitely have to get in on it
Markov	we have to get in on it
Human	we have to get in on it
<i>Input</i>	but i think that when you get to the tourist domain it 's not just an information retrieval system .

ID	Transcription
Markov	when you get to the tourist domain it 's an information retrieval system .
Human	but when you get to the tourist domain it 's not just an information retrieval system .
<i>Input</i>	so this is where i think people are going to have to think this through a bit more carefully .
Markov	this is where people are going to have to this .
Human	this is where people are going to have to think this through more carefully .
<i>Input</i>	but what happens when you actually get them moving and so forth and so on
Markov	what happens when you get them moving
Human	but when you get them moving
<i>Input</i>	uh i think the notion of this as a self contained functional module that interacts with where the tourism stuff is going probably is too restrictive .
Markov	as a self contained functional module that interacts with where is too restrictive .
Human	the notion of this as a self contained functional module that interacts with the tourism stuff probably is too restrictive .
<i>Input</i>	now i dunno how much people have thought ahead to the tourist domain in this
Markov	how much people have thought ahead to the tourist domain in this
Human	i dunno how much people have thought ahead to the tourist domain
<i>Input</i>	probably not enough , i mean another uh more basic point there is that the current um tasks and therefore the concepts in what 's called the action plan and what 's really the dialogue manager .
Markov	another more basic point there is the current tasks and the concepts in what 's called the action plan what 's the dialogue manager .
Human	probably not enough , the current tasks and therefore the concepts in the dialogue manager .
<i>Input</i>	um is based on slots that have to be filled and the um kind of values in these slots would be fixed things like the a time or a movie title or something like this whereas in the a um tourist domain it might be an entire route .
Markov	is based on slots that have to be filled the values in these slots would be fixed things like the a time a movie title .
Human	is based on slots that have to be filled and the values in these slots would be fixed things like the time or a movie title whereas in the tourist domain it might be an entire route .
<i>Input</i>	set-based , or even very complex structured information in these slots and i 'm not sure if complex slots of that type are really um being taken into consideration .

ID	Transcription
Markov	i 'm not sure if complex slots of that type are taken into consideration .
Human	set-based , or even very complex structured information in these slots and i 'm not sure if complex slots of that type are being taken into consideration .
<i>Input</i>	could you put a message into the right place to see if we can at least ask that question ?
Markov	could you put a message into the right place ?
Human	could you put a message to see if we can ask that question ?
<i>Input</i>	i mean nothing 's being completely settled there so this is really an ongoing discussion
Markov	this is an ongoing discussion
Human	nothing 's being completely settled
<i>Input</i>	ok i 'll talk to michael it 's what i do anyway .
Markov	it 's what i do anyway .
Human	i 'll talk to michael .
<i>Input</i>	how far is the m-three-l specification for the natural language input gone i have n't seen anything for the uh tourist path domain .
Markov	how far gone i have n't seen anything for the tourist path domain .
Human	how far is the m-three-l specification for the natural language input gone i have n't seen anything for the tourist path domain .
<i>Input</i>	yeah , there 's a meeting next week i think
Markov	there 's a meeting next week
Human	there 's a meeting next week
<i>Input</i>	those are the true key issues is how does whatever comes out of the language input pipeline look like and then what the action planner does with it and how that is uh specified .
Markov	those are the true key issues is how does whatever comes out of the language input pipeline .
Human	the key issues is how does whatever comes out of the language input pipeline look like and what the action planner does with it .
<i>Input</i>	there 're dialogue action planners that work with belief-nets that are action planners that work with you know state automata .
Markov	there 're dialogue action planners that work with belief-nets that are action planners that work with state automata .
Human	there 're dialogue action planners that work with belief-nets that are action planners that work with state automata .

ID	Transcription
<i>Input</i>	i mean it does matter because it does have to keep track of we are on part six of a route that consists of eight steps and so forth
Markov	it does matter because it does have to keep track of we are on of a route that consists of eight steps and
Human	it does have to keep track of we are on a route that consists of eight steps
<i>Input</i>	yeah , i think there are a lot of reasons why it matters .
Markov	there are a lot of reasons why it matters .
Human	there are a lot of reasons why it matters .
<i>Input</i>	ok , so that uh , for example , the action planner is going to take some spec and make some suggestions about what the user should do .
Markov	that the action planner is going to take some spec and make suggestions about what the user should do .
Human	the action planner is going to take some spec and make suggestions about what the user should do .
<i>Input</i>	what the user says after that is going to be very much caught up with what the action planner told it .
Markov	is going to be caught up with what the action planner told it .
Human	what the user says after that is going to be caught up with what the action planner told it .
<i>Input</i>	if the parser and the language end does n't know what the person 's been told
Markov	if does n't know what the person 's been told
Human	if the language end does n't know what the person 's been told
<i>Input</i>	you 're making your life much more difficult than it has to be .
Markov	you 're making your life difficult than it has to be .
Human	you 're making your life more difficult .
<i>Input</i>	i think that uh point has been realized and it 's not really um been defined yet but there 's going to be some kind of feedback and input from uh the action planner into all the analysis modules , telling them what to expect and what the current state of the discourse is .
Markov	has been realized it 's not been defined yet but there 's going to be of feedback .
Human	there 's going to be feedback from the action planner into all the analysis modules , telling them the current state of the discourse .
<i>Input</i>	beyond what 's currently being implemented which is just word lists .
Markov	beyond what 's implemented which is word lists .

ID	Transcription
Human	beyond word lists .
<i>Input</i>	we talked about this several times that the input end is going to need a fair amount of feedback from the planning end .
Markov	we talked about this several times is going to need a fair amount of feedback from the planning end .
Human	the input end is going to need feedback from the planning end .
<i>Input</i>	and even on a more basic level the action planner actually needs to be able to have um an expressive power that can deal with these structures .
Markov	the action planner needs to be able to have an expressive power that can deal with these structures .
Human	the action planner needs an expressive power that can deal with these structures .
<i>Input</i>	would there be any chance of getting the terminology changed so that the dialogue planner was called a dialogue planner ?
Markov	would there be any chance of getting the terminology changed so the dialogue planner was called a dialogue planner ?
Human	would there be any chance of getting the terminology changed so that the dialogue planner was called a dialogue planner ?
<i>Input</i>	it oughta be called a dialogue manager .
Markov	it oughta be called a dialogue manager .
Human	it oughta be called a dialogue manager .
<i>Input</i>	so what would happen if we sent a note saying gee we 've talked about this and could n't we change the whole word ?
Markov	what would happen if we change the whole word ?
Human	could n't we change the word ?
<i>Input</i>	depends on who you talk to how .
Markov	depends on who you talk to how .
Human	depends .
<i>Input</i>	i 'll go check , cause i completely agree .
Markov	i 'll go check .
Human	i 'll check .
<i>Input</i>	yeah , and i think this is just for historical reasons within uh , the preparation phase of the project and not because somebody actually believes it ought to be action planner .

ID	Transcription
Markov	this is for historical reasons within the preparation phase of the project .
Human	i think this is just for historical reasons
<i>Input</i>	so if there is resistance against changing it , that 's just because oh , we do n't want to change things .
Markov	that 's just because we do n't want to change things .
Human	if there is resistance , that 's because we do n't want to change things .
<i>Input</i>	if that persists then we 're going to need another term . for the thing that actually does the planning of the uh routes and whatever we are doing for the tourist .
Markov	for the thing that does the planning of the routes and whatever we are doing for the tourist .
Human	if that persists then we 're going to need another term for the thing that actually does the planning of the routes .
<i>Input</i>	that 's external services .
Markov	that 's external services .
Human	that 's external services .
<i>Input</i>	that has all the wrong connotations .
Markov	that has all the wrong connotations .
Human	that has the wrong connotations .
<i>Input</i>	yeah i think just the spatial planner and the route planner i showed you once the interaction among them in the deep map system so a printout of the communication between those two fills up i do n't know how many pages and that 's just part of how do i get to one place .
Markov	the spatial planner and the route planner showed you once the interaction among them in the deep map system .
Human	the spatial planner and the route planner i showed you once the interaction among them in the deep map system so a printout of the communication between those two fills up many pages .
<i>Input</i>	and he 's going to be responsible for the implementation of this action planner .
Markov	he 's going to be responsible for the implementation of this action planner .
Human	and he 's going to be responsible for the implementation of this action planner .
<i>Input</i>	no he 's completely going to rewrite everything . in java .
Markov	he 's going to rewrite everything . in java .
Human	no he 's going to rewrite everything . in java .

ID	Transcription
<i>Input</i>	yes that 's my next question whether we 're we 're going to stick to prolog or not .
Markov	that 's my next question whether we 're we 're going to stick to prolog .
Human	we 're going to stick to prolog .
<i>Input</i>	no , that 's going to be phased out .
Markov	that 's going to be phased out .
Human	no , that 's going to be phased out .
<i>Input</i>	but i do think the function modeling concept makes sense in a certain light because the dialogue manager in that case should not um have to worry about whether it 's interfacing with um something that does route planning in this way or that way
Markov	i do the function modeling concept makes sense in a certain light because the dialogue manager in that case should not have to worry
Human	but the function modeling concept makes sense because the dialogue manager should not have to worry about interfacing with something that does route planning
<i>Input</i>	there 's a logic to dialogue which is separable .
Markov	there 's a logic to dialogue which is separable .
Human	there 's a logic to dialogue which is separable .
<i>Input</i>	and it cant sort of formulate what it wants in a rather abstract uh way , you know find me a good route for this .
Markov	it cant formulate what it find me a good route for this .
Human	it cant formulate what it wants in a rather abstract way .
<i>Input</i>	it 's tricky .
Markov	it 's tricky .
Human	it 's tricky .
<i>Input</i>	it 's tricky because i think it will turn out to be the case that uh , this thing we 're talking about , the extended uh knowledge modeler will fill in some parameters about what the person wants .
Markov	it 's tricky it will turn out to be the case that will fill in parameters about what the person wants .
Human	it 's tricky because the extended knowledge modeler will fill in some parameters about what the person wants .
<i>Input</i>	but in that case the dialogue manager is sort of event driven .
Markov	the dialogue manager is event driven .

ID	Transcription
Human	the dialogue manager is event driven .
<i>Input</i>	so the dialogue manager may think it 's in a dialogue state of one sort , and one of these planning modules comes along and says hey right , now we need to ask a question .
Markov	the dialogue manager may it 's in a dialogue state of one sort .
Human	the dialogue manager may think it 's in a dialogue state of one sort , and one of these planning modules need to ask a question .
<i>Input</i>	the underlying idea of course is that there is something like kernel modules with kernel functionality that you can plug uh certain applications like tourist information or um the home scenario with uh controlling a vcr and so on .
Markov	is there is something like kernel modules with kernel functionality that you can plug certain applications .
Human	there is kernel modules with kernel functionality that you can plug certain applications like tourist information .
<i>Input</i>	that 's an additional reason to have this well-defined interface and keep these things like uh tourist information external .
Markov	that 's to have this well-defined interface and keep these things like tourist information external .
Human	that 's an additional reason to have this well-defined interface and keep things like tourist information external .
<i>Input</i>	but we are facing of course much more um realistic problems .
Markov	we are facing of course more realistic problems .
Human	we are facing more realistic problems .
<i>Input</i>	and language input for example , is of course uh crucial you know also when you do the sort of deep understanding analysis that we envision .
Markov	language input for example is of course crucial .
Human	language input , is crucial when you do the deep understanding analysis that we envision .
<i>Input</i>	and um so we 're thinking , for example how much syntactic analysis actually happens already in the parser .
Markov	we 're for example how much syntactic analysis happens in the parser .
Human	we 're thinking how much syntactic analysis happens already in the parser .
<i>Input</i>	yeah , there currently is uh no syntactic analysis but in the next release there will be some .
Markov	there is no syntactic analysis but there will be some .
Human	there is no syntactic analysis but in the next release there will be .

ID	Transcription
<i>Input</i>	so what are the plans roughly ?
Markov	what are the plans ?
Human	what are the plans ?
<i>Input</i>	um to integrate and syntactic analysis .
Markov	to integrate and syntactic analysis .
Human	to integrate and syntactic analysis .
<i>Input</i>	and um add some more features like segmentation .
Markov	add some more features like segmentation .
Human	and add features like segmentation .
<i>Input</i>	so then more than one utterance there 's often uh pause between it and a segmentation occurs .
Markov	there 's pause between it and a segmentation occurs .
Human	more than one utterance there 's often pause between it and a segmentation occurs .
<i>Input</i>	do you have a particular parser in mind ?
Markov	do you have a particular parser in mind ?
Human	do you have a parser in mind ?
<i>Input</i>	i mean have you thought through ?
Markov	have you thought through ?
Human	have you thought through ?
<i>Input</i>	no no i think it 's totally complicated for just one person
Markov	it 's totally complicated for just one person
Human	it 's complicated for one person
<i>Input</i>	yeah , ah and so things must be simpler but uh , miel syntactic analysis with um finite state transducers .
Markov	things must be simpler miel syntactic analysis with finite state transducers .
Human	things must be simpler miel syntactic analysis with finite state transducers .
<i>Input</i>	people at dfki have written a fair number of parsers .
Markov	people have written a fair number of parsers .
Human	dfki have written a number of parsers .
<i>Input</i>	other you know , people over the years . uh have written various parsers at dfki .
Markov	have written various parsers at dfki .

ID	Transcription
Human	people have written various parsers at dfki .
<i>Input</i>	none of them are suitable ?
Markov	none of them are suitable ?
Human	none of them are suitable ?
<i>Input</i>	yeah , uh the problem is that it has to be very fast because um for more than one path anywhere what 's in the latches from the speech recognizer so it 's speed is crucial .
Markov	the problem is it has to be fast because for more than one path anywhere what 's .
Human	it has to be very fast for more than one path .
<i>Input</i>	and they also have to be very robust . cuz of um speech recognition errors
Markov	they have to be robust . cuz of speech recognition errors
Human	they also have to be very robust cuz of speech recognition errors
<i>Input</i>	so there was a chunk parser in verbmobil , that was one of the uh branchers .
Markov	there was a chunk parser in verbmobil that was one of the branchers .
Human	a chunk parser in verbmobil was one of the branchers .
<i>Input</i>	yeah i guess it 's similar .
Markov	it 's similar .
Human	it 's similar .
<i>Input</i>	from michael strube , i 've heard very good stuff about the chunk parser that is done by forwiss , uh , which is in embassy doing the parsing .
Markov	from michael strube i 've heard very good stuff about the chunk parser .
Human	from michael strube , i 've heard very good stuff about the chunk parser that is done by forwiss .
<i>Input</i>	mm-hmm , yeah , it would be very interesting ,
Markov	it would be very interesting ,
Human	it would be very interesting ,
<i>Input</i>	but uh given the constraints , that you want it to be small and fast and so forth , my guess is you 're probably into some kind of chunk parsing .
Markov	given the constraints , my guess is you 're into of chunk parsing .
Human	but given the constraints , you 're probably into chunk parsing .
<i>Input</i>	yeah , and purely finite-state transducers are not so good for german since the word order is uh not fixed

ID	Transcription
Markov	finite-state transducers are not good for german since the word order is not fixed
Human	purely finite-state transducers are not good for german since the word order is not fixed
<i>Input</i>	yeah , i guess that 's the point is all the morphology and stuff .
Markov	is all the morphology and stuff .
Human	the point is the morphology .
<i>Input</i>	you 've got stemmers ?
Markov	you 've got stemmers ?
Human	you 've got stemmers ?
<i>Input</i>	um , yeah but it 's all in the lexicon .
Markov	it 's all in the lexicon .
Human	yeah but it 's all in the lexicon .
<i>Input</i>	yeah the information is available .
Markov	the information is available .
Human	the information is available .
<i>Input</i>	so you just connect to the lexicon and uh at least for german you have all of the stemming information .
Markov	you connect to the lexicon you have of the stemming information .
Human	you connect to the lexicon and at least for german you have all of the stemming information .
<i>Input</i>	we have knowledge bases from verbmobil system we can use and so .
Markov	we have knowledge bases from verbmobil system we can use so .
Human	we have knowledge bases from verbmobil system .
<i>Input</i>	so the lexicon might be derived by morphix but what 's happening on-line is just um um a retrieval from the lexicon which would give all the stemming information so it would be a full foreign lexicon .
Markov	the lexicon might be derived by morphix but what 's on-line is a retrieval from the lexicon .
Human	the lexicon might be derived by morphix but what 's happening on-line is just a retrieval from the lexicon which would give all the stemming information .
<i>Input</i>	so in german then you actually do case matching in the pattern matcher or not ?
Markov	you do case matching in the pattern matcher or not ?
Human	in german you do case matching in the pattern matcher ?

ID	Transcription
<i>Input</i>	um not yet but it 's planned to do that .
Markov	not it 's planned to do that .
Human	not yet but it 's planned .
<i>Input</i>	yeah and here 's the case where the english and the german might really be significantly different .
Markov	's the case where the english and the german be different .
Human	here 's the case where the english and the german might be significantly different .
<i>Input</i>	you really might want to do it in a significantly different way .
Markov	you might want to do it in a different way .
Human	you might want to do it in a significantly different way .
<i>Input</i>	in terms of you know , if you 're doing this for english as well as german
Markov	in of if you 're doing this for english
Human	if you 're doing this for english as well
<i>Input</i>	yes it 's um possible to do list processing .
Markov	's possible to do list processing .
Human	it 's possible to do list processing .
<i>Input</i>	some extensions uh have to be made for a english version
Markov	extensions have to be made for a english version
Human	some extensions have to be made for a english version
<i>Input</i>	not easy .
Markov	not easy .
Human	not easy .
<i>Input</i>	so we talked about the fact that there 're going to be a certain number of decisions that you want the knowledge modeler to make , that will be then fed to the function module , that does uh , route planning .
Markov	we talked about the fact that there 're going to be a certain number of decisions .
Human	there 're going to be a number of decisions that you want the knowledge modeler to make , that will be then fed to the function module , that does route planning .
<i>Input</i>	and then one half of this we talked about at little bit is how if you had the right information , if you knew something about what was said and something about was the agent a tourist or a native or a business person or uh young or old , whatever .

ID	Transcription
Markov	one half of this we talked about at bit is how if you had the right information if .
Human	one half of this is the right information , about what was said and about was the agent a tourist or a native or a business person .
<i>Input</i>	that information , and also about the
Markov	that information and about the
Human	that information
<i>Input</i>	so all that kind of information could be combined into decision networks and give you decisions .
Markov	could be combined into decision networks and give you decisions .
Human	all that information could be combined into decision networks and give you decisions .
<i>Input</i>	but the other half of the problem is how would you get that kind of information from the parsed input ?
Markov	how would you get of information from the parsed input ?
Human	the other half is how you get that information from the parsed input ?
<i>Input</i>	so what you might try to do is just you know build a template that uh somehow would capture the fact that he wants to take a picture .
Markov	is build a template would capture the fact he wants to take a picture .
Human	you might build a template that would capture the fact that he wants to take a picture .
<i>Input</i>	but uh from our point of view this is also a research project and there are a couple of people not here for various reasons who are doing doctoral dissertations on this , and the idea that we 're really after is a very deep semantics based on cognitive linguistics and the notion that there are a relatively small number of primitive conceptual schemas that characterize a lot of activity .
Markov	there are of people not here for various reasons who are doing doctoral dissertations on the idea .
Human	but there are a couple of people who are doing doctoral dissertations on this , and we 're after a very deep semantics based on cognitive linguistics and the notion that there are a relatively small number of primitive conceptual schemas that characterize a lot of activity .
<i>Input</i>	so , what we 're really trying to do is to map from the discourse to the conceptual semantics level .
Markov	is to map from the discourse to the conceptual semantics level .
Human	we 're trying to map from the discourse to the conceptual semantics level .

ID	Transcription
<i>Input</i>	so another one of these primitive , what are called image schemas , is uh goal seeking .
Markov	another one of these primitive is goal seeking .
Human	another of these primitive image schemas is goal seeking .
<i>Input</i>	and that all sorts of things , particularly in the tourist domain , can be represented in terms of uh source , path and goal .
Markov	all sorts of things can be represented in of source path and goal .
Human	all sorts of things , particularly in the tourist domain , can be represented in terms of source , path and goal .
<i>Input</i>	so if you can do this , then the notion would be that across a very large range of domains , you could use this deep conceptual basis as the interface .
Markov	if you can do this the notion would be that you could use this deep conceptual basis as the interface .
Human	across a very large range of domains , you could use this deep conceptual basis as the interface .
<i>Input</i>	and the idea of the belief-net is it combines the information from the dialogue which comes across in this general way , you know this is a goal seeking behavior , along with specific information from the ontology about the kinds of objects involved and about the situation about is it raining ?
Markov	is it combines the information from the dialogue which comes across in this general way ?
Human	the belief-net combines the information from the dialogue , along with specific information from the ontology about the kinds of objects involved and about the situation
<i>Input</i>	and so the coupling to the situation comes in this model from at the belief-net , combining evidence from the dialogue with the ontology with the situation .
Markov	comes in this model from combining evidence from the dialogue with the ontology with the situation .
Human	the situation comes from the belief-net , combining evidence from the dialogue with the ontology with the situation .

References

- Anne Abeillé, Yves Schabes, and Aravind K. Joshi. 1990. Using lexicalized tags for machine translation. In *Proceedings of the 13th conference on Computational linguistics*, pages 1–6. Association for Computational Linguistics.
- Anne Abeillé. 1994. Syntax or Semantics? Handling Nonlocal Dependencies with MCTAGS or Synchronous TAGS. *Computational Intelligence*, 10(4):471–485.
- Steven Abney. 2004. Understanding the yarowsky algorithm. *Computational Linguistics*, 30(3):365–395.
- Alfred V. Aho and Jeffrey D. Ullman. 1969. Syntax directed translations and the pushdown assembler. *Journal of Computer and System Sciences*, 3:37–56.
- James Allen and Mark Core. 1997. Draft of DAMSL: Dialog act markup in several layers. Unpublished manuscript.
- Srinivas Bangalore, Owen Rambow, and Steve Whittaker. 2000. Evaluation metrics for generation. In *INLG '00: Proceedings of the first international conference on Natural language generation*, pages 1–8. Association for Computational Linguistics.
- Michele Banko, Vibhu Mittal, and Michael Witbrock. 2000. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000)*, pages 318–325.
- Regina Barzilay, Michael Collins, Julia Hirschberg, and Steve Whittaker. 2000. The rules behind roles: Identifying speaker role in radio broadcasts. In *Proc. of the American Association for Artificial Intelligence conference (AAAI)*.
- John Bear, John Dowding, and Elizabeth Shriberg. 1992. Integrating multiple knowledge sources for detection and correction of repairs in human-computer dialog. In *Meeting of the Association for Computational Linguistics*, pages 56–63.
- Timothy C. Bell, John G. Cleary, and Ian H. Witten. 1990. *Text compression*. Prentice Hall.
- Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–72.
- A. Bies. 1995. Bracketing guidelines for treebank ii style penn treebank project. Technical report, University of Pennsylvania.
- Daniel M. Bikel. 2004. *On the parameter space of generative lexicalized statistical parsing models*. Ph.D. thesis, University of Pennsylvania. Supervisor-Mitchell P. Marcus.
- Jeff A. Bilmes and Katrin Kirchhoff. 2003. Factored language models and generalized parallel backoff. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL)*, pages 4–6. Association for Computational Linguistics.

- Ezra Black, Steven P. Abney, D. Flickenger, Claudia Gdaniec, Ralph Grishman, P. Harrison, Donald Hindle, Robert Ingria, Frederick Jelinek, Judith L. Klavans, Mark Liberman, Mitchell P. Marcus, Salim Roukos, Beatrice Santorini, and Tomek Strzalkowski. 1991. A procedure for quantitatively comparing the syntactic coverage of english grammars. In Morgan Kaufman, editor, *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 306–311.
- Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, pages 92–100.
- P. Boersma and D. Weenink. 2006. Praat: doing phonetics by computer. <http://www.praat.org/>.
- T.L. Booth and R.A. Thompson. 1973. Applying probabilistic measures to abstract languages. *IEEE Transactions on Computers*, 22(5):442–450.
- Peter F. Brown, John Cocke, Stephen Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.
- Ivan Bulyko, Mari Ostendorf, and Andreas Stolcke. 2003. Getting more mileage from web text sources for conversational speech language modeling using class-dependent mixtures. In *Proceedings of the joint conference of Human Language Technology and North American Chapter of Association for Computational Linguistics (HLT-NAACL)*.
- Susanne Burger, Victoria MacLaren, and Hua Yu. 2002. The ISL meeting corpus: The impact of meeting type on speech style. In *Proc. of the International Conference on Spoken Language Processing (ICSLP-2002)*.
- Jean Carletta, Stefan Evert, Ulrich Heid, Jonathan Kilgour, Judy Robertson, and Holger Voormann. 2003. The nite xml toolkit: flexible annotation for multi-modal language data. *Behavior Research Methods, Instruments, and Computers*, 35(3):353–363.
- Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, Guillaume Lathoud, Mike Lincoln, Agnes Lisowska, Iain McCowan, Wilfried Post, Dennis Reidsma, and Pierre Wellner. 2005. The AMI meeting corpus: A pre-announcement. In *Workshop on Multimodal Interaction and Related Machine Learning Algorithms (MLMI)*.
- Jean Carletta. 2006. Announcing the ami meeting corpus. *The ELRA Newsletter*, 11(1):3–5.
- John Carroll, Guidon Minnen, Yvonne Canning, Siobhan Devlin, and John Tait. 1998. Practical simplification of english newspaper text to assist aphasic readers. In *Proceedings of the AAAI-98 Workshop on Integrating Artificial Intelligence and Assistive Technology*.
- Raman Chandrasekar, Christine Doran, and Srinivas Bangalore. 1996. Motivations and methods for text simplification. In *Proc. of the 16th International Conference on Computational Linguistics (COLING-96)*, pages 1041–1044.

- Eugene Charniak and Mark Johnson. 2001. Edit detection and parsing for transcribed speech. In *Proc. of the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-01)*, pages 118–126.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n -best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, pages 173–180.
- Eugene Charniak, Sharon Goldwater, and Mark Johnson. 1998. Edge-based best-first chart parsing. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 127–133.
- Eugene Charniak. 1996. Tree-bank grammars. Technical report, Brown University.
- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-2000)*, pages 132–139.
- Eugene Charniak. 2001. Immediate-head parsing for language models. In *Meeting of the Association for Computational Linguistics*, pages 116–123.
- Pierre Chatain, Edward Whittaker, Joanna Mrozinski, and Sadaoki Furui. 2006a. Class model adaptation for speech summarisation. In *Proc. the Human Language Technology Conference of the NAACL*, pages 21–24.
- Pierre Chatain, Edward Whittaker, Joanna Mrozinski, and Sadaoki Furui. 2006b. Topic and stylistic adaptation for speech summarisation. In *Proc. of ICASS*.
- Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. Smote: Synthetic minority over-sampling technique.
- Stanley F. Chen and Ronald Rosenfeld. 1999. A gaussian prior for smoothing maximum entropy models. Technical Report CMUCS-99-108, Carnegie Mellon University.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*.
- David Chiang. 2006. An introduction to synchronous grammars. Unpublished manuscript.
- James Clarke and Mirella Lapata. 2006a. Constraint-based sentence compression: An integer programming approach. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 144–151.
- James Clarke and Mirella Lapata. 2006b. Models for sentence compression: a comparison across domains, training requirements and evaluation measures. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, pages 377–384. Association for Computational Linguistics.
- Norman Cliff. 1996. *Ordinal Methods for Behavioral Data Analysis*. Lawrence Erlbaum Associates.

- J. Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological measurements*, 20:37–46.
- Shuki Cohen. 2002. A computerized scale for monitoring levels of agreement during a conversation. In *Proc. of the 26th Penn Linguistics Colloquium*.
- Michael Collins, Robert E. Schapire, and Yoram Singer. 2002. Logistic regression, adaboost and bregman distances. *Machine Learning*, 48(1-3):253–285.
- Michael John Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 184–191. Association for Computational Linguistics.
- Michael Collins. 1999. *Head-driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.
- Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proceedings of the 17th International Conference on Machine Learning (ICML-2000)*, pages 175–182.
- John Conroy, Judith Schlesinger, Jade Goldstein, and Dianne O’Leary. 2004. Left-brain/right-brain multi-document summarization. In *DUC 04 Conference Proceedings*.
- Mark G. Core and Lenhart K. Schubert. 1999. A syntactic framework for speech repairs and other disruptions. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 413–420. Association for Computational Linguistics.
- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2001. *Introduction to Algorithms, Second Edition*. The MIT Press.
- Simon H. Corston-Oliver and William B. Dolan. 1999. Less is more: Eliminating index terms from subordinate clauses. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL-99)*, pages 349–356.
- Simon H. Corston-Oliver. 2001. Text compaction for display on very small screens. In *Proc. of the Workshop on Automatic Summarization at the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 89–98.
- Deborah Coughlin. 2001. Correlating automated and human assessments of machine translation quality. In *In Proceedings of MT Summit IX*, pages 63–70.
- Brooke Cowan, Ivona Kučerová, and Michael Collins. 2006. A discriminative model for tree-to-tree translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 232–241. Association for Computational Linguistics.
- Robert G. Cowell, A. Philip Dawid, Steffen L. Lauritzen, and David J. Spiegelhalter. 1999. *Probabilistic Networks and Expert Systems*. Statistics for Engineering and Information Science. Springer-Verlag.
- Koby Crammer and Yoram Singer. 2001. Pranking with ranking. In *Proc. of the conference on Neural Information Processing Systems (NIPS)*.

- J. N. Darroch and D. Ratcliff. 1972. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43(5):1470–1480.
- Hal Daumé III and Daniel Marcu. 2002. A noisy-channel model for document compression. In *Proc. of the Conference of the Association of Computational Linguistics (ACL-02)*.
- Hal Daumé III and Daniel Marcu. 2004. A tree-position kernel for document compression. In *Proceedings of the Fourth Document Understanding Conference (DUC-2004)*.
- A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(Ser B):1–38.
- R. Dhillon, S. Bhagat, H. Carvey, and E. Shriberg. 2004. Meeting recorder project: Dialog act labeling guide. Technical Report TR-04-002, ICSI.
- Pedro Domingos. 1999. MetaCost: A general method for making classifiers cost-sensitive. In *Knowledge Discovery and Data Mining*, pages 155–164.
- Bonnie J. Dorr, David Zajic, and Richard Schwartz. 2003. Hedge: A parse-and-trim approach to headline generation. In *Proc. of the HLT-NAACL Text Summarization Workshop and Document Understanding Conference (DUC-2003)*, pages 1–8.
- EARS. 2003. Effective, affordable, reusable speech-to-text (EARS). Official web site for DARPA-EARS Program: <http://projects ldc.upenn.edu/MDE/>.
- H.P. Edmundson. 1968. New methods in automatic extracting. *Journal of the Association for Computing Machinery*, 16(2):264–285.
- Jason Eisner. 1996. An empirical comparison of probability models for dependency grammar. Technical Report IRCS-96-11, Institute for Research in Cognitive Science, Univ. of Pennsylvania.
- Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proc. of the 41st Meeting of the Association for Computational Linguistics*.
- Charles Elkan. 2001. The foundations of cost-sensitive learning. In *International Joint Conferences on Artificial Intelligence*, pages 973–978.
- David Kirk Evans, Judith L. Klavans, and Kathleen R. McKeown. 2004. Columbia newsblaster: Multilingual news summarization on the web. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Demonstration Papers*, pages 1–4. Association for Computational Linguistics.
- J. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proc. of ACL*, pages 363–370.
- Michel Galley and Kathleen McKeown. 2007. Lexicalized Markov grammars for sentence compression. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 180–187. Association for Computational Linguistics.

- Michel Galley, Kathleen R. McKeown, Eric Fosler-Lussier, and Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In Erhard Hinrichs and Dan Roth, editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, pages 562–569.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004a. What’s in a translation rule? In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 273–280. Association for Computational Linguistics.
- Michel Galley, Kathleen McKeown, Julia Hirschberg, and Elizabeth Shriberg. 2004b. Identifying agreement and disagreement in conversational speech: Use of bayesian networks to model pragmatic dependencies. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL-04)*, pages 669–676.
- Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968. Association for Computational Linguistics.
- Michel Galley. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 364–372. Association for Computational Linguistics.
- Edward Gibson, Florian Wolf, Evelina Fedorenko, Doug Jones, Charlene Chuang, and Rina Patel. 2004. Two new experimental protocols for measuring speech transcript readability for timed question-answering tasks. In *DARPA EARS RT-04F Workshop*.
- Daniel Gildea. 2001. Corpus variation and parser performance. In Lillian Lee and Donna Harman, editors, *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 167–202.
- John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. SWITCHBOARD: Telephone Speech Corpus for Research and Development. In *Proc. of ICASSP*, volume 1, pages 517–520.
- David Graff. 1997. 1996 english broadcast news speech (hub4). In *Linguistic Data Consortium*.
- Gregory Grefenstette. 1998. Producing intelligent telegraphic text reduction to provide an audio scanning service for the blind. In *Proceedings of the AAAI Spring Workshop on Intelligent Text Summarization*, pages 111–115.
- Samuel S. Gross, Olga Russakovsky, Chuong B. Do, and Serafim Batzoglou. 2007. Training conditional random fields for maximum labelwise accuracy. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 529–536. MIT Press.
- Dan Gusfield. 1997. *Algorithms on strings, trees, and sequences: computer science and computational biology*. Cambridge University Press.

- Thomas Hain, Lukas Burget, John Dines, Iain McCowan, Giulia Garau, Martin Karafiát, Mike Lincoln, Darren Moore, Vincent Wan, Roeland Ordelman, and Steve Renals. 2005. The development of the ami system for the transcription of speech in meetings. In Steve Renals and Samy Bengio, editors, *Machine Learning for Multimodal Interaction, Second International Workshop, MLMI 2005, Edinburgh, UK, July 11-13, 2005, Revised Selected Papers*, volume 3869 of *Lecture Notes in Computer Science*, pages 344–356. Springer.
- John Hale, Izhak Shafran, Lisa Yung, Bonnie J. Dorr, Mary Harper, Anna Krasnyanskaya, Matthew Lease, Yang Liu, Brian Roark, Matthew Snover, and Robin Stewart. 2006. Pcfgs with syntactic and prosodic indicators of speech repairs. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 161–168. Association for Computational Linguistics.
- Donna Harman and Mark Liberman. 1993. TIPSTER complete. In *Linguistic Data Consortium. CAT: LDC93T3A*, ISBN 1-58563-020-9.
- Mary Harper, Bonnie J. Dorr, John Hale, Brian Roark, Izhak Shafran, Matthew Lease, Yang Liu, Matthew Snover, Lisa Yung, Anna Krasnyanskaya, and Robin Stewart. 2005. Parsing and spoken structural event detection. In *Final Report of 2005 Johns Hopkins Summer Workshop*.
- Randy Allen Harris. 2005. *Voice Interaction Design: Crafting the New Conversational Speech Systems*. Morgan Kaufmann.
- V. Hatzivassiloglou and K. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proc. of ACL*.
- Peter A. Heeman and James F. Allen. 1999. Speech repairs, intonational phrases, and discourse markers: modeling speakers’ utterances in spoken dialogue. *Comput. Linguist.*, 25(4):527–571.
- Ralf Herbrich, Thore Graepel, Peter Bollmann-Sdorra, and Klaus Obermayer. 1998. Learning a preference relation for information retrieval. In *Proceedings of the AAAI Workshop Text Categorization and Machine Learning*.
- D. Hillard, M. Ostendorf, and E Shriberg. 2003. Detection of agreement vs. disagreement in meetings: training with unlabeled data. In *Proc. of HLT/NAACL*.
- Julia Hirschberg and Diane J. Litman. 1994. Empirical studies on the disambiguation of cue phrases. *Computational Linguistics*, 19(3):501–530.
- Chiori Hori, Sadaoki Furui, Rob Malkin, Hua Yu, and Alex Waibel. 2002. Automatic summarization of english broadcast news speech. In *Proc. of the Human Technology Conference (HLT-02), San Diego*.
- Chiori Hori. 2002. *A Study on Statistical Methods for Automatic Speech Summarization*. Ph.D. thesis, Tokyo Institute of Technology.
- Liang Huang and David Chiang. 2005. Better k-best parsing. In *In Proceedings of 9th International Workshop of Parsing Technologies (IWPT)*.

- Rebecca Hughes. 1996. *English in Speech and Writing: investigating language and literature*. Routledge publishing.
- Akira Inoue, Takayoshi Mikami, and Yoichi Yamashita. 2004. Improvement of speech summarization using prosodic information. In *Proc. of Speech Prosody*.
- Edgar T. Irons. 1961. A syntax directed compiler for algol 60. *Commun. ACM*, 4(1):51–55.
- Adam Janin, Dan Baron, Jane Edwards, Dan Ellis, David Gelbart, Nelson Morgan, Barbara Peskin, Thilo Pfau, Elizabeth Shriberg, Andreas Stolcke, and Chuck Wooters. 2003. The ICSI meeting corpus. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP-03)*.
- Nathalie Japkowicz. 2000. The class imbalance problem: Significance and strategies. In *Proceedings of the 2000 International Conference on Artificial Intelligence (IC-AI'2000)*, volume 1, pages 111–117.
- F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. 1990. Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, 4:269–282.
- Gang Ji and Jeffrey Bilmes. 2005. Dialog act tagging using graphical models. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP-05)*.
- Hongyan Jing and Kathleen R. McKeown. 2000. Cut and paste based text summarization. In *Proceedings of the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics NAACL-2000*, pages 178–185.
- Hongyan Jing. 2000. Sentence reduction for automatic text summarization. In *Proceedings of the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-2000)*, pages 310–315.
- George H. John, Ron Kohavi, and Karl Pflieger. 1994. Irrelevant features and the subset selection problem. In *International Conference on Machine Learning (ICML)*, pages 121–129.
- M. Johnson and E. Charniak. 2004. A tag-based noisy channel model of speech repairs. In *Proc. of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*.
- Mark Johnson. 1998. PCFG models of linguistic tree representations. *Computational Linguistics*, 24(4):613–632.
- Mark Johnson. 2001. Joint and conditional estimation of tagging and parsing models. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 322–329. Association for Computational Linguistics.
- Aravind K. Joshi, Leon S. Levy, and Masako Takahashi. 1975. Tree adjunct grammar. *Journal of Computer and System Science*, 21(2):136–163.
- Aravind K. Joshi. 1987. The relevance of tree adjoining grammar to generation. *Natural Language Generation: New Directions in Artificial Intelligence, Psychology, and Linguistics*.

- Yasuhiro Kawata and Julia Bartels. 2000. Stylebook for the Japanese treebank in VERBMOBIL. Technical Report Verbmobil-Report 240, Eberhard-Karls-Universität Tübingen, Seminar für Sprachwissenschaft.
- T. Kikuchi, S. Furui, and C. Hori. 2003. Automatic speech summarization based on sentence extraction and compaction. In *ICASSP*, pages 384–387.
- Joungbum Kim, Sarah E. Schwarm, and Mari Ostendorf. 2004. Detecting structural metadata with decision trees and transformation-based learning. In *HLT-NAACL*, pages 137–144.
- Goran Kjellmer. 1991. A mint of phrases. In Karin Aijmer and Bengt Altenberg, editors, *English Corpus Linguistics*, pages 111–127. Longman.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*.
- Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization — step one: Sentence compression. In *The 17th National Conference on Artificial Intelligence (AAAI-2000)*, pages 703–710.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54. Association for Computational Linguistics.
- Jachym Kolar, Yang Liu, and Elizabeth Shriberg. 2007. Speaker adaptation of language models for automatic dialog act segmentation of meetings. In *Proceedings of Interspeech*.
- K. Koumpis and S. Renals. 2000. Transcription and summarization of voicemail speech. In *Proc. of the International Conference on Spoken Language Processing (ICSLP), Beijing*.
- K. Koumpis and S. Renals. 2003a. Evaluation of extractive voicemail summarization. In *Proc. ISCA Workshop on Multilingual Spoken Document Retrieval*, pages 19–24.
- K. Koumpis and S. Renals. 2003b. Multi-class extractive voicemail summarization. In *Proc. of Eurospeech*, pages 2785–2788.
- Miroslav Kubat and Stan Matwin. 1997. Addressing the curse of imbalanced training sets: one-sided selection. In *Proc. of the 14th International Conference on Machine Learning*, pages 179–186. Morgan Kaufmann.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA.
- John Lafferty, Xiaojin Zhu, and Yan Liu. 2004. Kernel conditional random fields: representation and clique selection. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 64. ACM Press.
- G. G. Landis and J. R. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33:159–174.

- K. Lari and S.J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4(1):35–56.
- S. L. Lauritzen and D. J. Spiegelhalter. 1988. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society*, 50(2):157–224.
- Vladimir I. Levenshtein. 1966. Binary codes capable of correcting insertions, deletions and reversals. *Soviet Physics-Doklady*, 10:707–710.
- Stephen C. Levinson. 1983. *Pragmatics*. Cambridge University Press.
- P. M. Lewis and R. E. Stearns. 1968. Syntax-directed transduction. In *Journal of the Association for Computing Machinery*, volume 15, pages 465–488.
- Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 761–768. Association for Computational Linguistics.
- Chin-Yew Lin. 2004. ROUGE: a package for automatic evaluation of summaries. In *Proc. of workshop on text summarization, ACL-04*.
- Charles X. Ling and Chenghui Li. 1998. Data mining for direct marketing: Problems and solutions. In *Knowledge Discovery and Data Mining*, pages 73–79.
- Nanci Linke-Ellis. 1999. Closed captioning in America: Looking beyond compliance. In *Proceedings of the TAO Workshop on TV Closed Captions for the hearing impaired people*, pages 43–59.
- Yang Liu, Elizabeth Shriberg, Andreas Stolcke, Dustin Hillard, Mari Ostendorf, Barbara Peskin, and Mary Harper. 2004. The ICSI-SRI-UW metadata extraction system. In *Proc. of INTER-SPEECH*, pages 577–580.
- Y. Liu, E. Shriberg, A. Stolcke, B. Peskin, J. Ang, D. Hillard, M. Ostendorf, M. Tomalin, P. Woodland, and M. Harper. 2005. Structural metadata research in the EARS program. In *Proc. of ICASSP*.
- Nitin Madhani, Necip Fazil Ayan, Philip Resnik, and Bonnie J. Dorr. 2007. Using paraphrases for parameter tuning in statistical machine translation. In *Proceedings of the Workshop on Statistical Machine Translation*. Association for Computational Linguistics.
- David M. Magerman. 1995. Statistical decision-tree models for parsing. In *Meeting of the Association for Computational Linguistics*, pages 276–283.
- Inderjeet Mani and Mark T. Maybury. 1999. *Advances in Automatic Text Summarization*. MIT Press.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.

- Sameer Maskey and Julia Hirschberg. 2005. Comparing lexical, acoustic/prosodic, discourse and structural features for speech summarization. In *Proc. of Eurospeech*.
- Andrew McCallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum entropy markov models for information extraction and segmentation. In *Proc. of ICML*.
- Andrew McCallum. 2002. MALLET: A machine learning for language toolkit. <http://mallet.cs.umass.edu>.
- David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 152–159. Association for Computational Linguistics.
- I. McCowan, S. Bengio, D. Gatica-Perez, G. Lathoud, F. Monay, D. Moore, P. Wellner, and H. Bourlard. 2003. Modeling human interaction in meetings. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP-03)*.
- Edward M. McCreight. 1976. A space-economical suffix tree construction algorithm. *J. ACM*, 23(2):262–272.
- Peter McCullagh. 1980. Regression models for ordinal data. *Journal of the Royal Statistical Society. Series B (Methodological)*, 42(2):109–142.
- Ryan McDonald. 2006. Discriminative sentence compression with soft syntactic constraints. In *Proceedings of EACL*, pages 297–304.
- Kathleen McKeown, Rebecca J. Passonneau, David K. Elson, Ani Nenkova, and Julia Hirschberg. 2005. Do summaries help? In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 210–217. ACM Press.
- Kathleen McKeown, Lokesh Shrestha, and Owen Rambow. 2007. Using question-answer pairs in extractive summarization of email conversations. In *Proceedings of CICLing*, volume 4394, pages 542–550.
- I. Dan Melamed, Giorgio Satta, and Benjamin Wellington. 2004. Generalized multitext grammars. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 661. Association for Computational Linguistics.
- Nikki Mirghafori, Andreas Stolcke, Chuck Wooters, Tuomo Pirinen, Ivan Bulyko, Dave Gelbart, Martin Graciarena, Scott Otterson, Barbara Peskin, and Mari Ostendorf. 2004. From switchboard to meetings: Development of the 2004 ICSI-SRI-UW meeting recognition system. In *Proc. of the International Conference of Spoken Language Processing (ICSLP-04)*.
- Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. 1999. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of Uncertainty in AI*, pages 467–475.
- Gabriel Murray and Steve Renals. 2006. Dialogue act compression via pitch contour preservation. In *Proceedings of Interspeech 2006*.

- Gabriel Murray, Steve Renals, and Jean Carletta. 2005a. Extractive summarization of meeting recordings. In *Proc. of Eurospeech*.
- Gabriel Murray, Steve Renals, Jean Carletta, and Johanna Moore. 2005b. Evaluating automatic summaries of meeting recordings. In *ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization*.
- Christine H. Nakatani and Julia Hirschberg. 1993. A speech-first model for repair detection and correction. In *Proceedings of the Meeting of the Association for Computational Linguistics*, pages 46–53.
- James Nattinger and Jeanette DeCarrico. 1992. *Lexical Phrases and Language Teaching*. Oxford University Press.
- J. A. Nelder and R. Mead. 1965. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313.
- Ani Nenkova and Rebecca Passonneau. 2004a. Evaluating content selection in human- or machine-generated summaries: The pyramid scoring method. Technical Report CUCS-025-03, Columbia University, CS Department.
- Ani Nenkova and Rebecca Passonneau. 2004b. Evaluating content selection in summarization: The pyramid method. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 145–152. Association for Computational Linguistics.
- Ani Nenkova. 2006. *Understanding the process of multi-document summarization: content selection, rewrite and evaluation*. Ph.D. thesis, Columbia University.
- Andrew Y. Ng and Michael I. Jordan. 2002. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Proc. of NIPS*.
- Minh Le Nguyen, Masaru Fukushi, and Susumu Horiguchi. 2005. A probabilistic sentence reduction using maximum entropy model. *IEICE - Trans. Inf. Syst.*, E88-D(2):278–288.
- NIST. 2004a. NIST pilot meeting corpus. http://www.nist.gov/speech/test_beds/mr_proj/meeting_corpus_1.
- NIST. 2004b. NIST rich transcription 2004 evaluation (RT-04). <http://nist.gov/speech/tests/rt>.
- Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 295–302.
- F. Och and H Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

- Franz Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proc. of the Joint Conf. of Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28.
- Franz Joseph Och. 2003. *Yet Another Small Maximum Entropy Toolkit (YASMET)*. <http://www.fjoch.com/YASMET.html>.
- M. Ostendorf, A. Kannan, S. Austin, O. Kimball, R. Schwartz, and J. R. Rohlicek. 1991. Integration of diverse recognition methodologies through reevaluation of n-best sentence hypotheses. In *HLT '91: Proceedings of the workshop on Speech and Natural Language*, pages 83–87. Association for Computational Linguistics.
- Maryann Overstreet. 1999. *Whales, Candlelight, and Stuff Like That: General Extenders in English Discourse*. Oxford University Press.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics.
- Judea Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman Publishers.
- A. Pomerantz. 1984. Agreeing and disagreeing with assessments: some features of preferred/dispreferred turn shapes. In J.M. Atkinson and J.C. Heritage, editors, *Structures of Social Action*, pages 57–101.
- Arjen Poutsma. 1998. Data-oriented translation. In *Proceedings of the Ninth Conference of Computational Linguistics in the Netherlands*.
- Lawrence R. Rabiner. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.
- Owen Rambow and Giorgio Satta. 1996. Synchronous models of language. In Arivind Joshi and Martha Palmer, editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 116–123. Morgan Kaufmann Publishers.
- Owen Rambow, Lokesh Shrestha, John Chen, and Christy Lauridsen. 2004. Summarizing email threads. In *Proceedings of HLT-NAACL*.
- A. Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. In *Proc. of EMNLP*.
- Adwait Ratnaparkhi. 1998. *Maximum entropy models for natural language ambiguity resolution*. Ph.D. thesis, University of Pennsylvania.
- Deepak Ravichandran, Eduard Hovy, and Franz Josef Och. 2003. Statistical QA - classifier vs reranker: What's the difference? In *Proc. of the ACL Workshop on Multilingual Summarization and Question Answering*.

- Stefan Reizler, Tracy H. King, Richard Crouch, and Annie Zaenen. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Proceedings of HLT/NAACL*, pages 197–204.
- J. Robert-Ribes, S. Pfeiffer, R. Ellison, and D. Burnham. 1999. Semi-automatic captioning of tv programs, an australian perspective. In *Proceedings of the TAO Workshop on TV Closed Captions for the Hearing Impaired People*, pages 87–100.
- Sacks, Harvey, Schegloff, Emanuel A., and Jefferson, Gail. 1974. A simplest systematics for the organization of turn-taking for conversation. *Language*, 50(4):696–735.
- Sunita Sarawagi and William W. Cohen. 2004. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems 17 (NIPS 2004)*.
- Emanuel A. Schegloff and Harvey Sacks. 1973. Opening up closings. *Semiotica*, 7-4:289–327.
- Emanuel A. Schegloff. 1968. Sequencing in conversational openings. *American Anthropologist*, 70:1075–1095.
- Jeanette P. Schmidt. 1998. All highest scoring paths in weighted grid graphs and their application to finding all approximate repeats in strings. *SIAM J. Comput.*, 27(4):972–992.
- Prithviraj Sen and Lise Getoor. 2006. Cost-sensitive learning with conditional markov networks. In *International Conference on Machine Learning*.
- Stuart M. Shieber and Yves Schabes. 1990. Generation and synchronous tree-adjointing grammars. In *Proceedings of the Fifth International Workshop on Natural Language Generation*, pages 9–14.
- Stuart M. Shieber. 1988. Evidence against the context-freeness of natural language. In J. Kulas, J. H. Fetzer, and T. L. Rankin, editors, *Philosophy, Language, and Artificial Intelligence: Resources for Processing Natural Language*, pages 79–92. Kluwer.
- Lokesh Shrestha and Kathleen McKeown. 2004. Detection of question-answer pairs in email conversations. In *Proceedings of COLING*, pages 889–895.
- Elizabeth Shriberg, Rebecca Bates, and Andreas Stolcke. 1997. A prosody-only decision-tree model for disfluency detection. In *Proceedings Eurospeech '97*, pages 2383–2386.
- E. Shriberg, A. Stolcke, and D. Baron. 2001. Observations on overlap: Findings and implications for automatic processing of multi-party conversation. In *Proc. of EUROSPEECH*.
- Elizabeth Shriberg, Raj Dhillon, Sonali Bhagat, Jeremy Ang, and Hannah Carvey. 2004. The ICSI meeting recorder dialog act (MRDA) corpus. In *SIGdial Workshop on Discourse and Dialogue*, pages 97–100.
- Elizabeth Shriberg. 1994. *Preliminaries to a Theory of Speech Disfluencies*. Ph.D. thesis, University of California.

- A. Stolcke, E. Shriberg, R. Bates, M. Ostendorf, D. Hakkani, M. Plauche, G. Tur, and Y. Lu. 1998. Automatic detection of sentence boundaries and disfluencies based on recognized words. In *Proceedings of ICSLP*, volume 5, pages 2247–2250.
- A. Stolcke, X. Anguera, K. Boakye, O. Cetin, A. Janin, M. Magimai-Doss, C. Wooters, and J. Zheng. 2007. The SRI-ICSI spring 2007 meeting and lecture recognition system. In *Proc. NIST Rich Transcription Workshop*.
- Andreas Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*.
- Charles Sutton and Andrew McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. Technical Report TR # 04-49, University of Massachusetts.
- Charles Sutton, Andrew McCallum, and Jeff Bilmes (organizers). 2006. Joint inference for natural language processing (workshop). In *HLT/NAACL*.
- Charles Sutton. 2006. GRMM: A graphical models toolkit. <http://mallet.cs.umass.edu>.
- Kuo-Chung Tai. 1979. The tree-to-tree correction problem. *Journal of the ACM*, 26(3):422–433.
- Ben Taskar, Carlos Guestrin, and Daphne Koller. 2003. Max-margin markov networks. In *Advances in Neural Information Processing Systems (NIPS 2003)*. Winner of the Best Student Paper Award.
- Christoph Tillmann and Tong Zhang. 2006. A discriminative global training algorithm for statistical mt. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, pages 721–728. Association for Computational Linguistics.
- Jenine Turner and Eugene Charniak. 2005. Supervised and unsupervised learning for sentence compression. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 290–297. Association for Computational Linguistics.
- Vladimir Vapnik. 1998. *Statistical Learning Theory*. Wiley.
- Andrew J. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13:260–269.
- Alex Waibel, Michael Bett, Florian Metze, Klaus Ries, Thomas Schaaf, Tanja Schultz, Hagen Soltau, Hua Yu, and Klaus Zechner. 2001. Advances in automatic meeting record creation and access. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP-01)*.
- Fuliang Weng, Andreas Stolcke, and Ananth Sankar. 1998. Efficient lattice representation and generation. In *Proceedings of ICSLP*.
- Michael J. Witbrock and Vibhu O. Mittal. 1999. Ultra-summarization: A statistical approach to generating highly condensed non-extractive summaries. In *Proceedings of the 22nd International Conference on Research and Development in Information Retrieval (SIGIR-99), Poster Session*, pages 315–316.

- Yuk Wah Wong and Raymond J. Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 439–446. Association for Computational Linguistics.
- Dekai Wu. 1996. A polynomial-time algorithm for statistical machine translation. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 152–158. Association for Computational Linguistics.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404.
- Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proc. of ACL*, pages 523–530.
- David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Meeting of the Association for Computational Linguistics*, pages 189–196.
- David Zajic, Bonnie J. Dorr, and Richard Schwartz. 2002. Automatic headline generation for newspaper stories. In *In Proceedings of the ACL Workshop on Text Summarization (DUC-2002)*, pages 78–85.
- Klaus Zechner and Alex Waibel. 2000a. Diasumm: Flexible summarization of spontaneous dialogues in unrestricted domains. In *Proceedings of COLING-2000, Saarbruecken, Germany*, pages 968–974.
- Klaus Zechner and Alex Waibel. 2000b. Minimizing word error rate in textual summaries of spoken language. In *ANLP*, pages 186–193.
- Klaus Zechner. 2002. Automatic summarization of open domain multi-party dialogues in diverse genres. *Computational Linguistics, Special Issue on Summarization*, 28(4):447–485.
- Kaizhong Zhang and Dennis Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262.
- Liang Zhou and Eduard Hovy. 2005. Digesting virtual “geek” culture: The summarization of technical internet relay chats. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL’05)*, pages 298–305.