Exponential Family Embeddings

Maja Rudolph
Columbia University

Francisco J. R. Ruiz
Univ. of Cambridge
Columbia University

Stephan Mandt
Columbia University

David M. Blei
Columbia University

Abstract

Word embeddings are a powerful approach for capturing semantic similarity among terms in a vocabulary. In this paper, we develop *exponential family embeddings*, a class of methods that extends the idea of word embeddings to other types of high-dimensional data. As examples, we studied neural data with real-valued observations, count data from a market basket analysis, and ratings data from a movie recommendation system. The main idea is to model each observation conditioned on a set of other observations. This set is called the context, and the way the context is defined is a modeling choice that depends on the problem. In language the context is the surrounding words; in neuroscience the context is close-by neurons; in market basket data the context is other items in the shopping cart. Each type of embedding model defines the context, the exponential family of conditional distributions, and how the latent embedding vectors are shared across data. We infer the embeddings with a scalable algorithm based on stochastic gradient descent. On all three applications—neural activity of zebrafish, users' shopping behavior, and movie ratings—we found exponential family embedding models to be more effective than other types of dimension reduction. They better reconstruct held-out data and find interesting qualitative structure.

1 Introduction

Word embeddings are a powerful approach for analyzing language (Bengio et al., 2006; Mikolov et al., 2013a,b; Pennington et al., 2014). A word embedding method discovers distributed representations of words; these representations capture the semantic similarity between the words and reflect a variety of other linguistic regularities (Rumelhart et al., 1986; Bengio et al., 2006; Mikolov et al., 2013c). Fitted word embeddings can help us understand the structure of language and are useful for downstream tasks based on text.

There are many variants, adaptations, and extensions of word embeddings (Mikolov et al., 2013a,b; Mnih and Kavukcuoglu, 2013; Levy and Goldberg, 2014; Pennington et al., 2014; Vilnis and McCallum, 2015), but each reflects the same main ideas. Each term in a vocabulary is associated with two latent vectors, an *embedding* and a *context vector*. These two types of vectors govern conditional probabilities that relate each word to its surrounding context. Specifically, the conditional probability of a word combines its embedding and the context vectors of its surrounding words. (Different methods combine them differently.) Given a corpus, we fit the embeddings by maximizing the conditional probabilities of the observed text.

In this paper we develop the *exponential family embedding (*EF-EMB*)*, a class of models that generalizes the spirit of word embeddings to other types of high-dimensional data. Our motivation is that other types of data can benefit from the same assumptions that underlie word embeddings, namely that a data point is governed by the other data in its context. In language, this is the foundational idea that words with similar meanings will appear in similar contexts (Harris, 1954). We use the tools of exponential families (Brown, 1986) and generalized linear models (GLMS) (McCullagh and Nelder, 1989) to adapt this idea beyond language.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

As one example beyond language, we will study computational neuroscience. Neuroscientists measure sequential neural activity across many neurons in the brain. Their goal is to discover patterns in these data with the hope of better understanding the dynamics and connections among neurons. In this example, a context can be defined as the neural activities of other nearby neurons, or as neural activity in the past. Thus, it is plausible that the activity of each neuron depends on its context. We will use this idea to fit latent embeddings of neurons, representations of neurons that uncover hidden features which help suggest their roles in the brain.

Another example we study involves shoppers at the grocery store. Economists collect shopping data (called "market basket data") and are interested in building models of purchase behavior for downstream econometric analysis, e.g., to predict demand and market changes. To build such models, they seek features of items that are predictive of when they are purchased and in what quantity. Similar to language, purchasing an item depends on its context, i.e., the other items in the shopping cart. In market basket data, Poisson embeddings can capture important econometric concepts, such as items that tend not to occur together but occur in the same contexts (substitutes) and items that co-occur, but never one without the other (complements).

We define an EF-EMB, such as one for neuroscience or shopping data, with three ingredients. (1) We define the *context*, which specifies which other data points each observation depends on. (2) We define the *conditional exponential family*. This involves setting the appropriate distribution, such as a Gaussian for real-valued data or a Poisson for count data, and the way to combine embeddings and context vectors to form its natural parameter. (3) We define the *embedding structure*, how embeddings and context vectors are shared across the conditional distributions of each observation. These three ingredients enable a variety of embedding models.

We describe EF-EMB models and develop efficient algorithms for fitting them. We show how existing methods, such as continuous bag of words (CBOW) (Mikolov et al., 2013a) and negative sampling (Mikolov et al., 2013b), can each be viewed as an EF-EMB. We study our methods on three different types of data—neuroscience data, shopping data, and movie ratings data. Mirroring the success of word embeddings, EF-EMB models outperform traditional dimension reduction, such as exponential family principal component analysis (PCA) (Collins et al., 2001) and Poisson factorization (Gopalan et al., 2015), and find interpretable features of the data.

Related work. EF-EMB models generalize CBOW (Mikolov et al., 2013a) in the same way that exponential family PCA (Collins et al., 2001) generalizes PCA, GLMS (McCullagh and Nelder, 1989) generalize regression, and deep exponential families (Ranganath et al., 2015) generalize sigmoid belief networks (Neal, 1990). A linear EF-EMB (which we define precisely below) relates to context-window-based embedding methods such as CBOW or the vector log-bilinear language model (VLBL) (Mikolov et al., 2013a; Mnih and Kavukcuoglu, 2013), which model a word given its context. The more general EF-EMB relates to embeddings with a nonlinear component, such as the skip-gram (Mikolov et al., 2013a) or the inverse vector log-bilinear language model (IVLBL) (Mnih and Kavukcuoglu, 2013). (These methods might appear linear but, when viewed as a conditional probabilistic model, the normalizing constant of each word induces a nonlinearity.)

Researchers have developed different approximations of the word embedding objective to scale the procedure. These include noise contrastive estimation (Gutmann and Hyvärinen, 2010; Mnih and Teh, 2012), hierarchical softmax (Mikolov et al., 2013b), and negative sampling (Mikolov et al., 2013a). We explain in Section 2.2 and Supplement A how negative sampling corresponds to biased stochastic gradients of an EF-EMB objective.

2 Exponential Family Embeddings

We consider a matrix $\boldsymbol{x} = x_{1:I}$ of I observations, where each x_i is a D-vector. As one example, in language x_i is an indicator vector for the word at position i and D is the size of the vocabulary. As another example, in neural data x_i is the neural activity measured at index pair $i = (n, t)$, where n indexes a neuron and t indexes a time point; each measurement is a scalar ($D = 1$).

The goal of an exponential family embedding (EF-EMB) is to derive useful features of the data. There are three ingredients: a context function, a conditional exponential family, and an embedding structure. These ingredients work together to form the objective. First, the EF-EMB models each data point conditional on its context; the context function determines which other data points are at play. Second, the conditional distribution is an appropriate exponential family, e.g., a Gaussian for real-valued data. Its parameter is a function of the embeddings of both the data point and its context. Finally, the embedding structure determines which embeddings are used when the ith point appears, either as data or in the context of another point. The objective is the sum of the log probabilities of each data point given its context. We describe each ingredient, followed by the EF-EMB objective. Examples are in Section 2.1.

Context. Each data point i has a *context* c_i, which is a set of indices of other data points. The EF-EMB models the conditional distribution of x_i given the data points in its context.

The context is a modeling choice; different applications will require different types of context. In language, the data point is a word and the context is the set of words in a window around it. In neural data, the data point is the activity of a neuron at a time point and the context is the activity of its surrounding neurons at the same time point. (It can also include neurons at future time or in the past.) In shopping data, the data point is a purchase and the context is the other items in the cart.

Conditional exponential family. An EF-EMB models each data point x_i conditional on its context $\boldsymbol{x}_{c_i}$. The distribution is an appropriate exponential family,

$$x_i \mid \boldsymbol{x}_{c_i} \sim \text{ExpFam}(\eta_i(\boldsymbol{x}_{c_i}), t(x_i)), \tag{1}$$

where $\eta_i(\boldsymbol{x}_{c_i})$ is the natural parameter and $t(x_i)$ is the sufficient statistic. In language modeling, this family is usually a categorical distribution. Below, we will study Gaussian and Poisson.

We parameterize the conditional with two types of vectors, embeddings and context vectors. The *embedding* of the ith data point helps govern its distribution; we denote it $\rho[i] \in \mathbb{R}^{K \times D}$. The *context vector* of the ith data point helps govern the distribution of data for which i appears in their context; we denote it $\alpha[i] \in \mathbb{R}^{K \times D}$.

How to define the natural parameter as a function of these vectors is a modeling choice. It captures how the context interacts with an embedding to determine the conditional distribution of a data point. Here we focus on the *linear embedding*, where the natural parameter is a function of a linear combination of the latent vectors,

$$\eta_i(\boldsymbol{x}_{c_i}) = f_i\left(\rho[i]^\top \sum_{j \in c_i} \alpha[j] x_j\right). \tag{2}$$

Following the nomenclature of generalized linear models (GLMS), we call $f_i(\cdot)$ the *link function*. We will see several examples of link functions in Section 2.1.

This is the setting of many existing word embedding models, though not all. Other models, such as the skip-gram, determine the probability through a "reverse" distribution of context words given the data point. These non-linear embeddings are still instances of an EF-EMB.

Embedding structure. The goal of an EF-EMB is to find embeddings and context vectors that describe features of the data. The *embedding structure* determines how an EF-EMB shares these vectors across the data. It is through sharing the vectors that we learn an embedding for the object of primary interest, such as a vocabulary term, a neuron, or a supermarket product. In language the same parameters $\rho[i] = \rho$ and $\alpha[i] = \alpha$ are shared across all positions i. In neural data, observations share parameters when they describe the same neuron. Recall that the index connects to both a neuron and time point $i = (n, t)$. We share parameters with $\rho[i] = \rho_n$ and $\alpha[i] = \alpha_n$ to find embeddings and context vectors that describe the neurons. Other variants might tie the embedding and context vectors to find a single set of latent variables, $\rho[i] = \alpha[i]$.

The objective function. The EF-EMB objective sums the log conditional probabilities of each data point, adding regularizers for the embeddings and context vectors.[1] We use log probability functions as regularizers, e.g., a Gaussian probability leads to ℓ_2 regularization. We also use regularizers to constrain the embeddings,e.g., to be non-negative. Thus, the objective is

$$\mathcal{L}(\boldsymbol{\rho}, \boldsymbol{\alpha}) = \sum_{i=1}^{I} \left(\eta_i^\top t(x_i) - a(\eta_i)\right) + \log p(\boldsymbol{\rho}) + \log p(\boldsymbol{\alpha}). \tag{3}$$

[1] One might be tempted to see this as a probabilistic model that is conditionally specified. However, in general it does not have a consistent joint distribution (Arnold et al., 2001).

We maximize this objective with respect to the embeddings and context vectors. In Section 2.2 we explain how to fit it with stochastic gradients.

Equation (3) can be seen as a likelihood function for a bank of GLMS (McCullagh and Nelder, 1989). Each data point is modeled as a response conditional on its "covariates," which combine the context vectors and context, e.g., as in Equation (2); the coefficient for each response is the embedding itself. We use properties of exponential families and results around GLMS to derive efficient algorithms for EF-EMB models.

2.1 Examples

We highlight the versatility of EF-EMB models with three example models and their variations. We develop the Gaussian embedding (G-EMB) for analyzing real observations from a neuroscience application; we also introduce a nonnegative version, the nonnegative Gaussian embedding (NG-EMB). We develop two Poisson embedding models, Poisson embedding (P-EMB) and additive Poisson embedding (AP-EMB), for analyzing count data; these have different link functions. We present a categorical embedding model that corresponds to the continuous bag of words (CBOW) word embedding (Mikolov et al., 2013a). Finally, we present a Bernoulli embedding (B-EMB) for binary data. In Section 2.2 we explain how negative sampling (Mikolov et al., 2013b) corresponds to biased stochastic gradients of the B-EMB objective. For convenience, these acronyms are in Table 1.

EF-EMB	*exponential family embedding*
G-EMB	*Gaussian embedding*
NG-EMB	*nonnegative Gaussian embedding*
P-EMB	*Poisson embedding*
AP-EMB	*additive Poisson embedding*
B-EMB	*Bernoulli embedding*

Table 1: Acronyms used for exponential family embeddings.

Example 1: Neural data and Gaussian observations. Consider the (calcium) expression of a large population of zebrafish neurons (Ahrens et al., 2013). The data are processed to extract the locations of the N neurons and the neural activity $x_i = x_{(n,t)}$ across location n and time t. The goal is to model the similarity between neurons in terms of their behavior, to embed each neuron in a latent space such that neurons with similar behavior are close to each other.

We consider two neurons similar if they behave similarly in the context of the activity pattern of their surrounding neurons. Thus we define the context for data index $i = (n, t)$ to be the indices of the activity of nearby neurons at the same time. We find the K-nearest neighbors (KNN) of each neuron (using a Ball-tree algorithm) according to their spatial distance in the brain. We use this set to construct the context $c_i = c_{(n,t)} = \{(m, t) | m \in \text{KNN}(n)\}$. This context varies with each neuron, but is constant over time.

With the context defined, each data point x_i is modeled with a conditional Gaussian. The conditional mean is the inner product from Equation (2), where the context is the simultaneous activity of the nearest neurons and the link function is the identity. The conditionals of two observations share parameters if they correspond to the same neuron. The embedding structure is thus $\rho[i] = \rho_n$ and $\alpha[i] = \alpha_n$ for all $i = (n, t)$. Similar to word embeddings, each neuron has two distinct latent vectors: the neuron embedding $\rho_n \in \mathbb{R}^K$ and the context vector $\alpha_n \in \mathbb{R}^K$.

These ingredients, along with a regularizer, combine to form a neural embedding objective. G-EMB uses ℓ_2 regularization (i.e., a Gaussian prior); NG-EMB constrains the vectors to be nonnegative (ℓ_2 regularization on the logarithm. i.e., a log-normal prior).

Example 2: Shopping data and Poisson observations. We also study data about people shopping. The data contains the individual purchases of anonymous users in chain grocery and drug stores. There are N different items and T trips to the stores among all households. The data is a sparse $N \times T$ matrix of purchase counts. The entry $x_i = x_{(n,t)}$ indicates the number of units of item n that was purchased on trip t. Our goal is to learn a latent representation for each product that captures the similarity between them.

We consider items to be similar if they tend to be purchased in with similar groups of other items. The *context* for observation x_i is thus the other items in the shopping basket on the same trip. For the purchase count at index $i = (n, t)$, the context is $c_i = \{j = (m, t) | m \neq n\}$.

We use conditional Poisson distributions to modelthe count data. The sufficient statistic of the Poisson is $t(x_i) = x_i$, and its natural parameter is the logarithm of the rate (i.e., the mean). We set the natural parameter as in Equation (2), with the link function defined below. The embedding structure is the same as in G-EMB, producing embeddings for the items.

We explore two choices for the link function. P-EMB uses an identity link function. Since the conditional mean is the exponentiated natural parameter, this implies that the context items contribute multiplicatively to the mean. (We use ℓ_2-regularization on the embeddings.) Alternatively, we can constrain the parameters to be nonnegative and set the link function $f(\cdot) = \log(\cdot)$. This is AP-EMB, a model with an additive mean parameterization. (We use ℓ_2-regularization in log-space.) AP-EMB only captures positive correlations between items.

Example 3: Text modeling and categorical observations. EF-EMBS are inspired by word embeddings, such as CBOW (Mikolov et al., 2013a). CBOW is a special case of an EF-EMB; it is equivalent to a multivariate EF-EMB with categorical conditionals. In the notation here, each x_i is an indicator vector of the ith word. Its dimension is the vocabulary size. The context of the ith word are the other words in a window around it (of size w), $c_i = \{j \neq i | i - w \leq j \leq i + w\}$.

The distribution of x_i is categorical, conditioned on the surrounding words $\boldsymbol{x}_{c_i}$; this is a softmax regression. It has natural parameter as in Equation (2) with an identity link function. The embedding structure imposes that parameters are shared across all observed words. The embeddings are shared globally ($\rho[i] = \rho, \alpha[i] = \alpha \in \mathbb{R}^{N \times K}$). The word and context embedding of the n^{th} word is the n^{th} row of ρ and α respectively. CBOW does not use any regularizer.

Example 4: Text modeling and binary observations. One way to simplify the CBOW objective is with a model of each entry of the indicator vectors. The data are binary and indexed by $i = (n, v)$, where n is the position in the text and v indexes the vocabulary; the variable $x_{n,v}$ is the indicator that word n is equal to term v. (This model relaxes the constraint that for any n only one $x_{n,v}$ will be on.) With this notation, the context is $c_i = \{(j, v') | \forall v', j \neq n, n - w \leq j \leq n + w\}$; the embedding structure is $\rho[i] = \rho[(n, v)] = \rho_v$ and $\alpha[i] = \alpha[(n, v)] = \alpha_v$.

We can consider different conditional distributions in this setting. As one example, set the conditional distribution to be a Bernoulli with an identity link; we call this the B-EMB model for text. In Section 2.2 we show that biased stochastic gradients of the B-EMB objective recovers negative sampling (Mikolov et al., 2013b). As another example, set the conditional distribution to Poisson with link $f(\cdot) = \log(\cdot)$. The corresponding embedding model relates closely to Poisson approximations of distributed multinomial regression (Taddy et al., 2015).

2.2 Inference and Connection to Negative Sampling

We fit the embeddings $\rho[i]$ and context vectors $\alpha[i]$ by maximizing the objective function in Equation (3). We use stochastic gradient descent (SGD) with Adagrad (Duchi et al., 2011). We can derive the analytic gradient of the objective function using properties of the exponential family (see the Supplement for details). The gradients linearly combine the data in summations we can approximate using subsampled minibatches of data. This reduces the computational cost.

When the data is sparse, we can split the gradient into the summation of two terms: one term corresponding to all data entries i for which $x_i \neq 0$, and one term corresponding to those data entries $x_i = 0$. We compute the first term of the gradient exactly—when the data is sparse there are not many summations to make—and we estimate the second term by subsampling the zero entries. Compared to computing the full gradient, this reduces the complexity when most of the entries x_i are zero. But it retains the strong information about the gradient that comes from the non-zero entries.

This relates to negative sampling, which is used to approximate the skip-gram objective (Mikolov et al., 2013b). Negative sampling re-defines the skip-gram objective to distinguish target (observed) words from randomly drawn words, using logistic regression. The gradient of the stochastic objective is identical to a noisy but biased estimate of the gradient for a B-EMB model. To obtain the equivalence, preserve the terms for the non-zero data and subsample terms for the zero data. While an unbiased

	single neuron held out		25% of neurons held out	
Model	$K = 10$	$K = 100$	$K = 10$	$K = 100$
FA	0.290 ± 0.003	0.275 ± 0.003	0.290 ± 0.003	0.276 ± 0.003
G-EMB (c=10)	0.239 ± 0.006	0.239 ± 0.005	0.246 ± 0.004	0.245 ± 0.003
G-EMB (c=50)	0.227 ± 0.002	$\mathbf{0.222 \pm 0.002}$	0.235 ± 0.003	$\mathbf{0.232 \pm 0.003}$
NG-EMB (c=10)	0.263 ± 0.004	0.261 ± 0.004	0.250 ± 0.004	0.261 ± 0.004

Table 2: Analysis of neural data: mean squared error and standard errors of neural activity (on the test set) for different models. Both EF-EMB models significantly outperform FA; G-EMB is more accurate than NG-EMB.

stochastic gradient would rescale the subsampled terms, negative sampling does not. Thus, negative sampling corresponds to a biased estimate, which down-weights the contribution of the zeros. See the Supplement for the mathematical details.

3 Empirical Study

We study exponential family embedding (EF-EMB) models on real-valued and count-valued data, and in different application domains—computational neuroscience, shopping behavior, and movie ratings. We present quantitative comparisons to other dimension reduction methods and illustrate how we can glean qualitative insights from the fitted embeddings.

3.1 Real Valued Data: Neural Data Analysis

Data. We analyze the neural activity of a larval zebrafish, recorded at single cell resolution for 3000 time frames (Ahrens et al., 2013). Through genetic modification, individual neurons express a calcium indicator when they fire. The resulting calcium imaging data is preprocessed by a nonnegative matrix factorization to identify neurons, their locations, and the fluorescence activity $x_t^* \in \mathbb{R}^N$ of the individual neurons over time (Friedrich et al., 2015). Using this method, our data contains 10,000 neurons (out of a total of 200,000).

We fit all models on the lagged data $x_t = x_t^* - x_{t-1}^*$ to filter out correlations based on calcium decay and preprocessing.[2] The calcium levels can be measured with great spatial resolution but the temporal resolution is poor; the neuronal firing rate is much higher than the sampling rate. Hence we ignore all "temporal structure" in the data and model the simultaneous activity of the neurons. We use the Gaussian embedding (G-EMB) and nonnegative Gaussian embedding (NG-EMB) from Section 2.1 to model the lagged activity of the neurons conditional on the lags of surrounding neurons. We study context sizes $c \in \{10, 50\}$ and latent dimension $K \in \{10, 100\}$.

Models. We compare EF-EMB to probabilistic factor analysis (FA), fitting K-dimensional factors for each neuron and K-dimensional factor loadings for each time frame. In FA, each entry of the data matrix is Gaussian distributed, with mean equal to the inner product of the corresponding factor and factor loading.

Evaluation. We train each model on a random sample of 90% of the lagged time frames and hold out 5% each for validation and testing. With the test set, we use two types of evaluation. (1) *Leave one out*: For each neuron x_i in the test set, we use the measurements of the other neurons to form predictions. For FA this means the other neurons are used to recover the factor loadings; for EF-EMB this means the other neurons are used to construct the context. (2) *Leave* 25% *out*: We randomly split the neurons into 4 folds. Each neuron is predicted using the three sets of neurons that are out of its fold. (This is a more difficult task.) Note in EF-EMB, the missing data might change the size of the context of some neurons. See Table 5 in Supplement C for the choice of hyperparameters.

Results. Table 2 reports both types of evaluation. The EF-EMB models significantly outperform FA in terms of mean squared error on the test set. G-EMB obtains the best results with 100 components and a context size of 50. Figure 1 illustrates how to use the learned embeddings to hypothesize connections between nearby neurons.

[2] We also analyzed unlagged data but all methods resulted in better reconstruction on the lagged data.

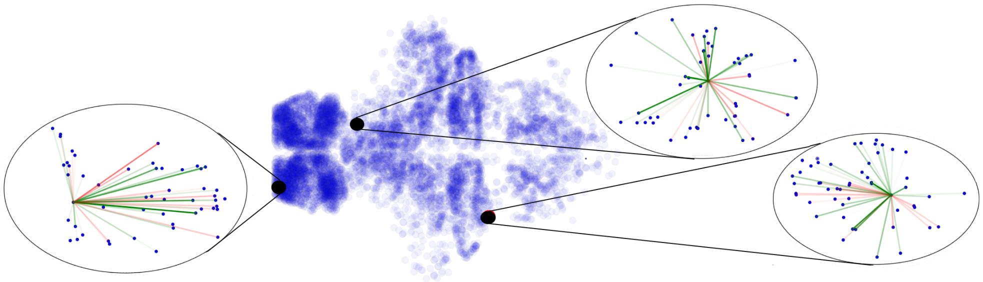

Figure 1: Top view of the zebrafish brain, with blue circles at the location of the individual neurons. We zoom on 3 neurons and their 50 nearest neighbors (small blue dots), visualizing the "synaptic weights" learned by a G-EMB model ($K = 100$). The edge color encodes the inner product of the neural embedding vector and the context vectors $\rho_n^\top \alpha_m$ for each neighbor m. Positive values are green, negative values are red, and the transparency is proportional to the magnitude. With these weights we can hypothesize how nearby neurons interact.

Model	$K = 20$	$K = 100$
P-EMB	-7.497 ± 0.007	-7.199 ± 0.008
P-EMB (dw)	-7.110 ± 0.007	$\mathbf{-6.950 \pm 0.007}$
AP-EMB	-7.868 ± 0.005	-8.414 ± 0.003
HPF	-7.740 ± 0.008	-7.626 ± 0.007
Poisson PCA	-8.314 ± 0.009	-11.01 ± 0.01

(a) Market basket analysis.

$K = 20$	$K = 100$
$\mathbf{-5.691 \pm 0.006}$	-5.726 ± 0.005
-5.790 ± 0.003	-5.798 ± 0.003
-5.964 ± 0.003	-6.118 ± 0.002
-5.787 ± 0.006	-5.859 ± 0.006
-5.908 ± 0.006	-7.50 ± 0.01

(b) Movie ratings.

Table 3: Comparison of predictive log-likelihood between P-EMB, AP-EMB, hierarchical Poisson factorization (HPF) (Gopalan et al., 2015), and Poisson principal component analysis (PCA) (Collins et al., 2001) on held out data. The P-EMB model outperforms the matrix factorization models in both applications. For the shopping data, downweighting the zeros improves the performance of P-EMB.

3.2 Count Data: Market Basket Analysis and Movie Ratings

We study the Poisson models Poisson embedding (P-EMB) and additive Poisson embedding (AP-EMB) on two applications: shopping and movies.

Market basket data. We analyze the IRI dataset[3] (Bronnenberg et al., 2008), which contains the purchases of anonymous households in chain grocery and drug stores. It contains 137, 632 trips in 2012. We remove items that appear fewer than 10 times, leaving a dataset with 7, 903 items. The context for each purchase is the other purchases from the same trip.

MovieLens data. We also analyze the MovieLens-100K dataset (Harper and Konstan, 2015), which contains movie ratings on a scale from 1 to 5. We keep only positive ratings, defined to be ratings of 3 or more (we subtract 2 from all ratings and set the negative ones to 0). The context of each rating is the other movies rated by the same user. After removing users who rated fewer than 20 movies and movies that were rated fewer than 50 times, the dataset contains 777 users and 516 movies; the sparsity is about 5%.

Models. We fit the P-EMB and the AP-EMB models using number of components $K \in \{20, 100\}$. For each K we select the Adagrad constant based on best predictive performance on the validation set. (The parameters we used are in Table 5.) In these datasets, the distribution of the context size is heavy tailed. To handle larger context sizes we pick a link function for the EF-EMB model which rescales the sum over the context in Equation (2) by the context size (the number of terms in the sum). We also fit a P-EMB model that artificially downweights the contribution of the zeros in the objective function by a factor of 0.1, as done by Hu et al. (2008) for matrix factorization. We denote it as "P-EMB (dw)."

[3] We thank IRI for making the data available. All estimates and analysis in this paper, based on data provided by IRI, are by the authors and not by IRI.

Maruchan chicken ramen	**Yoplait strawberry yogurt**	**Mountain Dew soda**	**Dean Foods 1 % milk**
M. creamy chicken ramen	Yoplait apricot mango yogurt	Mtn. Dew orange soda	Dean Foods 2 % milk
M. oriental flavor ramen	Yoplait strawberry orange smoothie	Mtn. Dew lemon lime soda	Dean Foods whole milk
M. roast chicken ramen	Yoplait strawberry banana yogurt	Pepsi classic soda	Dean Foods chocolate milk

Table 4: Top 3 similar items to a given example query words (bold face). The p-emb model successfuly captures similarities.

We compare the predictive performance with hpf (Gopalan et al., 2015) and Poisson pca (Collins et al., 2001). Both hpf and Poisson pca factorize the data into K-dimensional positive vectors of user preferences, and K-dimensional positive vectors of item attributes. ap-emb and hpf parameterize the mean additively; p-emb and Poisson pca parameterize it multiplicatively. For the ef-emb models and Poisson pca, we use stochastic optimization with ℓ_2 regularization. For hpf, we use variational inference. See Table 5 in Supplement C for details.

Evaluation. For the market basket data we hold out 5% of the trips to form the test set, also removing trips with fewer than two purchased different items. In the MovieLens data we hold out 20% of the ratings and set aside an additional 5% of the non-zero entries from the test for validation. We report prediction performance based on the normalized log-likelihood on the test set. For p-emb and ap-emb, we compute the likelihood as the Poisson mean of each nonnegative count (be it a purchase quantity or a movie rating) divided by the sum of the Poisson means for all items, given the context. To evaluate hpf and Poisson pca at a given test observation we recover the factor loadings using the other test entries we condition on, and we use the factor loading to form the prediction.

Predictive performance. Table 3 summarizes the test log-likelihood of the four models, together with the standard errors across entries in the test set. In both applications the p-emb model outperforms hpf and Poisson pca. On shopping data p-emb with $K = 100$ provides the best predictions; on MovieLens p-emb with $K = 20$ is best. For p-emb on shopping data, downweighting the contribution of the zeros gives more accurate estimates.

Item similarity in the shopping data. Embedding models can capture qualitative aspects of the data as well. Table 4 shows four example products and their three most similar items, where similarity is calculated as the cosine distance between embedding vectors. (These vectors are from p-emb with downweighted zeros and $K = 100$.) For example, the most similar items to a soda are other sodas; the most similar items to a yogurt are (mostly) other yogurts.

The p-emb model can also identify complementary and substitutable products. To see this, we compute the inner products of the embedding and the context vectors for all item pairs. A high value of the inner product indicates that the probability of purchasing one item is increased if the second item is in the shopping basket (i.e., they are complements). A low value indicates the opposite effect and the items might be substitutes for each other.

We find that items that tend to be purchased together have high value of the inner product (e.g., potato chips and beer, potato chips and frozen pizza, or two different types of soda), while items that are substitutes have negative value (e.g., two different brands of pasta sauce, similar snacks, or soups from different brands). Other items with negative value of the inner product are not substitutes, but they are rarely purchased together (e.g., toast crunch and laundry detergent, milk and a toothbrush). Supplement D gives examples of substitutes and complements.

Topics in the movie embeddings. The embeddings from MovieLens data identify thematically similar movies. For each latent dimension k, we sort the context vectors by the magnitude of the kth component. This yields a ranking of movies for each component. In Supplement E we show two example rankings. (These are from a p-emb model with $K = 50$.) The first one contains children's movies; the second contains science-fiction/action movies.

Acknowledgments

This work is supported by the EU H2020 programme (Marie Skłodowska-Curie grant agreement 706760), NFS IIS-1247664, ONR N00014-11-1-0651, DARPA FA8750-14-2-0009, DARPA N66001-15-C-4032, Adobe, the John Templeton Foundation, and the Sloan Foundation.

References

Ahrens, M. B., Orger, M. B., Robson, D. N., Li, J. M., and Keller, P. J. (2013). Whole-brain functional imaging at cellular resolution using light-sheet microscopy. *Nature Methods*, 10(5):413–420.

Arnold, B. C., Castillo, E., Sarabia, J. M., et al. (2001). Conditionally specified distributions: an introduction (with comments and a rejoinder by the authors). *Statistical Science*, 16(3):249–274.

Bengio, Y., Schwenk, H., Senécal, J.-S., Morin, F., and Gauvain, J.-L. (2006). Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer.

Bronnenberg, B. J., Kruger, M. W., and Mela, C. F. (2008). Database paper: The IRI marketing data set. *Marketing Science*, 27(4):745–748.

Brown, L. D. (1986). Fundamentals of statistical exponential families with applications in statistical decision theory. *Lecture Notes-Monograph Series*, 9:i–279.

Collins, M., Dasgupta, S., and Schapire, R. E. (2001). A generalization of principal components analysis to the exponential family. In *Neural Information Processing Systems*, pages 617–624.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.

Friedrich, J., Soudry, D., Paninski, L., Mu, Y., Freeman, J., and Ahrens, M. (2015). Fast constrained non-negative matrix factorization for whole-brain calcium imaging data. In *NIPS workshop on Neural Systems*.

Gopalan, P., Hofman, J., and Blei, D. M. (2015). Scalable recommendation with hierarchical Poisson factorization. In *Uncertainty in Artificial Intelligence*.

Gutmann, M. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Journal of Machine Learning Research*.

Harper, F. M. and Konstan, J. A. (2015). The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4):19.

Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162.

Hu, Y., Koren, Y., and Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. *Data Mining*.

Levy, O. and Goldberg, Y. (2014). Neural word embedding as implicit matrix factorization. In *Neural Information Processing Systems*, pages 2177–2185.

McCullagh, P. and Nelder, J. A. (1989). *Generalized linear models*, volume 37. CRC press.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *ICLR Workshop Proceedings. arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Neural Information Processing Systems*, pages 3111–3119.

Mikolov, T., Yih, W.-T. a., and Zweig, G. (2013c). Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751.

Mnih, A. and Kavukcuoglu, K. (2013). Learning word embeddings efficiently with noise-contrastive estimation. In *Neural Information Processing Systems*, pages 2265–2273.

Mnih, A. and Teh, Y. W. (2012). A fast and simple algorithm for training neural probabilistic language models. In *International Conference on Machine Learning*, pages 1751–1758.

Neal, R. M. (1990). Learning stochastic feedforward networks. *Department of Computer Science, University of Toronto*.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Conference on Empirical Methods on Natural Language Processing*, volume 14, pages 1532–1543.

Ranganath, R., Tang, L., Charlin, L., and Blei, D. M. (2015). Deep exponential families. *Artificial Intelligence and Statistics*.

Rumelhart, D. E., Hintont, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:9.

Taddy, M. et al. (2015). Distributed multinomial regression. *The Annals of Applied Statistics*, 9(3):1394–1414.

Vilnis, L. and McCallum, A. (2015). Word representations via Gaussian embedding. In *International Conference on Learning Representations*.

Supplement to Exponential Family Embeddings

A Inference

We fit the embeddings $\rho[i]$ and context vectors $\alpha[i]$ by maximizing the objective function in Equation (3). We use stochastic gradient descent (SGD).

We first calculate the gradient, using the identity for exponential family distributions that the derivative of the log-normalizer is equal to the expectation of the sufficient statistics, i.e., $\mathbb{E}[t(X)] = \nabla_\eta a(\eta)$. With this result, the gradient with respect to the embedding $\rho[j]$ is

$$\nabla_{\rho[j]}\mathcal{L} = \sum_{i=1}^{I} \big(t(x_i) - \mathbb{E}[t(x_i)]\big)\nabla_{\rho[j]}\eta_i + \nabla_{\rho[j]} \log p(\rho[j]). \tag{4}$$

The gradient with respect to $\alpha[j]$ has the same form. In Supplement B, we detail this expression for the particular models that we study empirically (Section 3).

The gradient in Equation (4) can involve a sum of many terms and be computationally expensive to compute. To alleviate this, we follow noisy gradients using SGD. We form a subsample $\mathcal{S}$ of the I terms in the summation, i.e.,

$$\widehat{\nabla}_{\rho[j]}\mathcal{L} = \frac{I}{|\mathcal{S}|} \sum_{i\in\mathcal{S}} \big(t(x_i) - \mathbb{E}[t(x_i)]\big)\nabla_{\rho[j]}\eta_i + \nabla_{\rho[j]} \log p(\rho[j]), \tag{5}$$

where $|\mathcal{S}|$ denotes the size of the subsample and where we scaled the summation to ensure an unbiased estimator of the gradient. Equation (5) reduces computational complexity when $|\mathcal{S}|$ is much smaller than the total number of terms. At each iteration of SGD we compute noisy gradients with respect to $\rho[j]$ and $\alpha[j]$ (for each j) and take gradient steps according to a step-size schedule. We use Adagrad (Duchi et al., 2011) to set the step-size.

Relation to negative sampling. In language, particularly when seen as a collection of binary variables, the data are sparse: each word is one of a large vocabulary. When modeling sparse data, we split the sum in Equation (4) into two contributions: those where $x_i > 0$ and those where $x_i = 0$. The gradient is

$$\begin{aligned}\nabla_{\rho[j]}\mathcal{L} = \sum_{i:x_i>0} \big(t(x_i) - \mathbb{E}[t(x_i)]\big)\nabla_{\rho[j]}\eta_i + \sum_{i:x_i=0} \big(t(0) - \mathbb{E}[t(x_i)]\big)\nabla_{\rho[j]}\eta_i \\ + \nabla_{\rho[j]} \log p(\rho[j]).\end{aligned} \tag{6}$$

We compute the first term of the gradient exactly—when the data is sparse there are not many summations to make—and we estimate the second term with subsampling. Compared to computing the full gradient, this reduces the complexity when most of the entries x_i are zero. But, it retains the strong information about the gradient that comes from the non-zero entries.

This relates to negative sampling, which is used to approximate the skip-gram objective (Mikolov et al., 2013b). Negative sampling re-defines the skip-gram objective to distinguish target (observed) words from randomly drawn words, using logistic regression. The gradient of the stochastic objective is identical to a noisy but biased estimate of the gradient in Equation (6) for a Bernoulli embedding (B-EMB) model. To obtain the equivalence, preserve the terms for the non-zero data and subsample terms for the zero data. While an unbiased stochastic gradient would rescale the subsampled terms, negative sampling does not. It is thus a biased estimate, which down-weights the contribution of the zeros.

B Stochastic Gradient Descent

To specify the gradients in Equation 4 for the SGD procedure we need the sufficient statistic $t(x)$, the expected sufficient statistic $\mathbb{E}[t(x)]$, the gradient of the natural parameter with respect to the embedding vectors and the gradient of the regularizer on the embedding vectors. In this appendix we specify these quantities for the models we study empirically in Section 3.

B.1 Gradients for Gaussian embedding (G-EMB)

Using the notation $i = (n, t)$ and reflecting the embedding structure $\rho[i] = \rho_n$, $\alpha[i] = \alpha_n$, the gradients with respect to each embedding and each context vector becomes

$$\nabla_{\rho_n} \mathcal{L} = -\lambda \rho_n + \frac{1}{\sigma^2} \sum_{t=1}^{T} \big(x_{(n,t)} - \rho_n^\top \sum_{m \in c_n} x_{(m,t)} \alpha_m \big) \big(\sum_{m \in c_n} x_{(m,t)} \alpha_m \big) \tag{7}$$

$$\nabla_{\alpha_n} \mathcal{L} = -\lambda \alpha_n + \frac{1}{\sigma^2} \sum_{t=1}^{T} \sum_{m | n \in c_m} \big(x_{(m,t)} - \rho_m^\top \sum_{r \in c_m} x_{(r,t)} \alpha_r \big) \big(x_{(n,t)} \rho_m \big) \tag{8}$$

B.2 Gradients for nonnegative Gaussian embedding (NG-EMB)

By restricting the parameters to be nonnegative we can learn nonnegative synaptic weights between neurons. For notational simplicity we write the parameters as $\exp(\rho)$ and $\exp(\alpha)$ and update them in log-space. The operator $\circ$ stands for element wise multiplication. With this notation, the gradient for the NG-EMB can be easily obtained from Equations 7 and 8 by applying the chain rule.

$$\begin{aligned} \nabla_{\rho_n} \mathcal{L} = & - \lambda \exp(\rho_n) \circ \exp(\rho_n) \\ & + \frac{1}{\sigma^2} \sum_{t=1}^{T} \big(x_{(n,t)} - \exp(\rho_n)^\top \sum_{m \in c_n} x_{(m,t)} \exp(\alpha_m) \big) \big(\sum_{m \in c_n} x_{im} \exp(\rho_n) \circ \exp(\alpha_m) \big) \end{aligned} \tag{9}$$

$$\begin{aligned} \nabla_{\alpha_n} \mathcal{L} = & - \lambda \exp(\alpha_n) \circ \exp(\alpha_n) \\ & + \frac{1}{\sigma^2} \sum_{t=1}^{T} \sum_{m | n \in c_m} \big(x_{(m,t)} - \exp(\rho_m)^\top \sum_{r \in c_m} x_{(r,t)} \exp(\alpha_r) \big) \big(x_{(n,t)} \exp(\rho_m) \circ \exp(\alpha_n) \big) \end{aligned} \tag{10}$$

B.3 Gradients for Poisson embedding (P-EMB)

We proceed similarly as for the G-EMB model.

$$\nabla_{\rho_n} \mathcal{L} = -\lambda \rho_n + \sum_{t=1}^{T} \left(x_{(n,t)} - \exp\left(\rho_n^\top \sum_{m \in c_n} x_{(m,t)} \alpha_m \right) \right) \left(\sum_{m \in c_n} x_{(m,t)} \alpha_m \right) \tag{11}$$

$$\nabla_{\alpha_n} \mathcal{L} = -\lambda \alpha_n + \sum_{t=1}^{T} \sum_{m | n \in c_m} \left(x_{(m,t)} - \exp\left(\rho_m^\top \sum_{r \in c_m} x_{(r,t)} \alpha_r \right) \right) \big(x_{(n,t)} \rho_m \big) \tag{12}$$

B.4 Gradients for additive Poisson embedding (AP-EMB)

Here, we proceed in a similar manner as for the NG-EMB model.

$$\nabla_{\rho_n} \mathcal{L} = -\lambda \exp(\rho_n) \circ \exp(\rho_n) + \sum_{t=1}^{T} \left(\frac{x_{(n,t)}}{\rho_n^\top \sum_{m \in c_n} x_{(m,t)} \alpha_m} - 1 \right) \left(\sum_{m \in c_n} x_{(m,t)} \alpha_m \right) \tag{13}$$

$$\nabla_{\alpha_n} \mathcal{L} = -\lambda \exp(\alpha_n) \circ \exp(\alpha_n) + \sum_{t=1}^{T} \sum_{m | n \in c_m} \left(\frac{x_{(m,t)}}{\rho_m^\top \sum_{r \in c_m} x_{(r,t)} \alpha_r} - 1 \right) \big(x_{(n,t)} \rho_m \big) \tag{14}$$

C Algorithm Details

	Model	minibatch size	regularization parameter	number iterations	negative samples
neuro	G-EMB	100	10	500	n/a
neuro	NG-EMB	100	0.1	500	n/a
shopping	all models	n/a	1	3000	10
movies	all models	n/a	1	3000	10

Table 5: Algorithm details for the models studied in Section 3.

D Complements and Substitutes in the Shopping Data

Table 6 shows some pairs of items with high inner product of embedding vectors and context vector. The items in the first column have higher probability of being purchased if the item in the second column is in the shopping basket. We can observe that they correspond to items that are frequently purchased together (potato chips and beer, potato chips and frozen pizza, two different sodas).

Similarly, Table 7 shows some pairs of items with low inner product. The items in the first column have lower probability of being purchased if the item in the second column is in the shopping basket. We can observe that they correspond to items that are rarely purchased together (detergent and toast crunch, milk and toothbrush), or that are substitutes of each other (two different brands of snacks, soup, or pasta sauce).

Inner product	Item 1	Item 2
2.12	Diet 7 Up lemon lime soda	Diet Squirt citrus soda
2.11	Old Dutch original potato chips	Budweiser Select 55 Lager beer
2.00	Lays potato chips	DiGiorno frozen pizza
2.00	Coca Cola zero soda	Coca Cola soda
1.99	Soyfield vanilla organic yogurt	La Yogurt low fat mango

Table 6: Market basket: List of several of the items with high inner product values. Items from the first column have higher probability of being purchased when the item in the second column is in the shopping basket.

Inner product	Item 1	Item 2
-5.06	General Mills cinnamon toast crunch	Tide Plus liquid laundry detergent
-5.00	Doritos chilli pepper	Utz cheese balls
-5.00	Land O Lakes 2% milk	Toothbrush soft adult (private brand)
-5.00	Beef Swanson Broth soup 48oz	Campbell Soup cans 10.75oz
-4.99	Ragu Robusto sautéed onion & garlic pasta sauce	Prego tomato Italian pasta sauce

Table 7: Market basket: List of several of the items with low inner product values. Items from the first column have lower probability of being purchased when the item in the second column is in the shopping basket.

E Movie Rating Results

Tables 8 and 9 show clusters of ranked movies that are learned by our P-EMB model. These rankings were generated as follows. For each latent dimension $k \in \{1, \cdots, K\}$ we sorted the context vectors according their value in this dimension. This gives us a ranking of context vectors for every k. Tables 8 and 9 show the 10 top items of the ranking for two different values of k. Similar as in topic modeling, the latent dimensions have the interpretation of topics. We see that sorting the context vectors this way reveals thematic structure in the collection of movies. While Table 8 gives a table of movies for children, Table 9 shows a cluster of science-fiction and action movies (with a few outliers).

#	Movie Name	Year	Rank
1	Winnie the Pooh and the Blustery Day	1968	0.62
2	Cinderella	1950	0.50
3	Toy Story	1995	0.46
4	Fantasia	1940	0.44
5	Dumbo	1941	0.43
6	The Nightmare Before Christmas	1993	0.37
7	Snow White and the Seven Dwarfs	1937	0.37
8	Alice in Wonderland	1951	0.35
9	James and the Giant Peach	1996	0.35

Table 8: Movielens: Cluster for “kids movies”.

#	Movie Name	Year	Rank
1	Die Hard: With a Vengeance	1995	1.25
2	Stargate	1994	1.19
3	Star Trek IV: The Voyage Home	1986	1.14
4	Manon of the Spring (Manon des sources)	1986	1.14
5	Fifth Element, The	1997	1.14
6	Star Trek VI: The Undiscovered Country	1991	1.13
7	Under Siege	1992	1.11
8	GoldenEye	1995	1.07
9	Supercop	1992	1.07

Table 9: Movielens: Cluster for “science-fiction/action movies”.