MODELING TOPICS AND KNOWLEDGE BASES WITH VECTOR REPRESENTATIONS

by

Dat Quoc Nguyen

A thesis presented for the degree of

DOCTOR OF PHILOSOPHY

Department of Computing
Faculty of Science and Engineering
Macquarie University
Sydney, Australia

May 2017

ORIGINALITY STATEMENT

Except where acknowledged in the customary manner, the material presented in this thesis is, to the best of my knowledge, original and has not been submitted in whole or part for a degree in any university.

Sydney, 29 May 2017
Dat Quoc Nguyen

ABSTRACT

Motivated by the recent success of utilizing latent feature vector representations (i.e. embeddings) in various natural language processing tasks, this thesis investigates how latent feature vector representations can help build better topic models and improve link prediction models for knowledge base completion. The first objective of this thesis is to incorporate latent feature word representations that contain external information from a large corpus in order to improve topic inference in a small corpus. The second objective is to develop new embedding models for predicting missing relationships between entities in knowledge bases. In particular, the major contributions of this thesis are summarized as follows:

- We propose two latent feature topic models which integrate word representations trained on large external corpora into two Dirichlet multinomial topic models: a Latent Dirichlet Allocation model and a one-topic-per-document Dirichlet Multinomial Mixture model.
- We introduce a triple-based embedding model named STransE to improve complex relationship prediction in knowledge bases. In addition, we also describe a new embedding approach, which combines the Latent Dirichlet Allocation model and the STransE model, to improve search personalization.
- We present a neighborhood mixture model where we formalize an entity representation as a mixture of its neighborhood in the knowledge base.

Extensive experiments show that our proposed models and approach obtain better performance than well-established baselines in a wide range of evaluation tasks.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest thanks to my principal supervisor, Mark Johnson, for his patient guidance and continuous support throughout the last three years. I have learned immensely from his incredible depth of knowledge. I am very grateful for his encouragement and helpful feedback on my research, in writing my papers and on the draft of this thesis. I feel very fortunate to have had an opportunity to work with him.

I would like to greatly thank my associate supervisor, Mark Dras, for his valuable advice regarding research and beyond. I am also grateful to Mark Dras, Kairit Sirts, Hegler Tissot, Kinzang Chhogyal and Tu Vu for their suggestions and comments on this thesis draft.

I really appreciate the opportunities to work with Thanh Vu in search personalization, Didi Surian and Adam Dunn in characterizing Twitter discussions. I also specifically enjoyed working with my twin brother Dai Quoc Nguyen in word representation learning.

I would like to express my gratitude to my colleagues Kairit Sirts, Richard Billingsley, Lizhen Qu, Lan Du, Hegler Tissot, John Pate, Anish Kumar, Tony Zhao, Mac Kim, Mehdi Parviz, Hiroshi Noji, Peter Anderson, Sonit Singh, Paria Jamshid Lou, Jonas Groschwitz, Shervin Malmasi, Yasaman Motazedi, Jette Viethen, Diego Molla- Aliod and Rolf Schwitter, and to my E6A347 lab mates Scott Buckley, Damian Jurd, Mahmood Yousefiazar, Mahmud Hasan, Charles Liu, Ali Siahvashi, Pablo Gonzalez, Adeline Fan and Mitchell Buckley.

I am also grateful to the department HDR directors and administrators Yan Wang, Donna Hua, Jackie Walsh, Sylvian Chow, Fiona Yang, Melina Chan, Christophe Doche, Abhaya Nayak, as well as all of the Science IT staff.

I would like to acknowledge the funding received towards my PhD candidature from the Australian Government IPRS Scholarship and the NICTA NRPA Top-Up Scholarship.

A special thanks goes to Kinzang Chhogyal, Nghia Quach, Carolyn Hamer-Smith, Phuntsho Namgay, Samdrup Dema, Anish Kumar, Vaidehi Seth, Surendra Shrestha, Alaska Pokhrel, Lan Du, Hiroshi Noji, Moe Haque, Daniel Sutantyo, Long Duong and Anh Dang. I enjoyed all the fun times and discussions we had during lunches, dinners and Christmases.

Finally, I dedicate this thesis to my parents Bao Quoc Nguyen and En Thi Nguyen. This thesis would not have been possible without their unconditional love and support.

Contents

Table of Contents **ix**

List of Figures **xiii**

List of Tables **xv**

Abbreviations **xix**

1 Introduction **1**

1.1 Motivation 1

1.1.1 Topic models 3

1.1.2 Knowledge base completion 6

1.2 Aims and Contributions 10

1.3 Outline and Origins 12

2 Background **17**

2.1 Word vector representations 18

2.1.1 Word2Vec Skip-gram model 19

2.1.2 GloVe model 20

2.2 Optimization algorithms 20

2.2.1 Gradient descent variants 21

2.2.2 AdaGrad 22

2.2.3 RMSProp 23

2.2.4 L-BFGS . . . 24
2.3 Bayesian inference for Dirichlet-Multinomials . . . 24
2.3.1 Bayesian inference . . . 25
2.3.2 Dirichlet-Multinomials . . . 25
2.3.3 A simple Dirichlet-multinomial model . . . 28
2.3.4 Inference via Gibbs sampling . . . 29
2.4 Probabilistic topic models . . . 30
2.4.1 Latent Dirichlet Allocation . . . 30
2.4.2 Advanced topic models . . . 32
2.4.3 Dirichlet Multinomial Mixture for short texts . . . 33
2.4.4 Topic models with word representations . . . 34
2.5 Embedding models for KB completion . . . 35
2.5.1 A general approach . . . 36
2.5.2 Specific models . . . 36
2.5.3 Other KB completion models . . . 39
2.6 Summary . . . 40

3 Improving topic models with word representations 41
3.1 Introduction . . . 42
3.2 New latent feature topic models . . . 43
3.2.1 Generative process for the LF-LDA model . . . 44
3.2.2 Generative process for the LF-DMM model . . . 45
3.2.3 Inference in the LF-LDA model . . . 45
3.2.4 Inference in the LF-DMM model . . . 50
3.2.5 Learning latent feature vectors for topics . . . 54
3.3 Experiments . . . 54
3.3.1 Experimental setup . . . 55
3.3.2 Topic coherence evaluation . . . 57
3.3.3 Document clustering evaluation . . . 62

3.3.4 Document classification evaluation 67
3.4 Discussion 69
3.5 Summary 70

4 STransE: a novel embedding model of entities and relationships 75
4.1 Introduction 76
4.2 The embedding model STransE 78
4.3 Link prediction evaluation 81
4.3.1 Task and evaluation protocol 81
4.3.2 Main results 82
4.4 Application for search personalization 85
4.4.1 Motivation 85
4.4.2 A new embedding approach for search personalization 88
4.4.3 Experimental methodology 91
4.4.4 Experimental results 93
4.5 Summary 93

5 Neighborhood mixture model for knowledge base completion 95
5.1 Introduction 96
5.2 Neighborhood mixture model 98
5.2.1 Neighbor-based entity representation 98
5.2.2 TransE-NMM: applying neighborhood mixtures to TransE 99
5.2.3 Parameter optimization 101
5.3 Experiments 102
5.3.1 Datasets 103
5.3.2 Evaluation tasks 103
5.3.3 Hyper-parameter tuning 104
5.4 Results 106
5.4.1 Quantitative results 106
5.4.2 Qualitative results 108

5.4.3 Discussion 109
5.5 Summary 111

6 Conclusion **113**
6.1 Answers and Key findings 113
6.2 Future work 115
6.3 Conclusion 117

Bibliography **119**

List of Figures

1.1 Two-dimensional projection of some word vectors. 2
1.2 An illustrative example of topics and topic assignments in topic modeling. 4
1.3 Two-dimensional projection of vectors of countries and their capital cities. . 7
1.4 An illustration of (incomplete) knowledge base, with 4 person entities, 2 place entities, 2 relation types and total 6 triple facts. 8
1.5 Dependency diagram of chapters and sections. 15

2.1 Graphical representation of the simple Dirichlet-multinomial mixture model 28
2.2 Graphical representations of Latent Dirichlet Allocation (LDA) and Dirichlet Multinomial Mixture (DMM) models. 31

3.1 Graphical representations of our new latent feature topic models. 44
3.2 NPMI scores on the N20short dataset with 20 topics and 40 topics, varying the mixture weight λ from 0.0 to 1.0. 58
3.3 Purity and NMI results (mean and standard deviation) on the N20short dataset with number of topics $T = 20$, varying the mixture weight λ from 0.0 to 1.0. 62
3.4 Purity and NMI results on the N20short dataset with number of topics $T = 40$, varying the mixture weight λ from 0.0 to 1.0. 63

4.1 Top-4 search results for query "acl 2017." 86
4.2 An illustration of the re-ranking process. "R" denotes that the document is relevant to the user. 87

5.1 An example fragment of a KB. 97

5.2 Relative improvement of TransE-NMM against TransE for entity prediction task in WN11 when the filtering threshold $\tau = \{10, 100, 500\}$ (with other hyper-parameters being the same as selected in Section 5.3.3). Prefixes "R-" and "F-" denote the "Raw" and "Filtered" settings, respectively. Suffixes "-MR", "-MRR" and "-H@10" abbreviate the mean rank, the mean reciprocal rank and Hits@10, respectively. 110

List of Tables

1.1 Top 15 topical words when fitting a topic model with 20 topics on a small news title corpus. **T** denotes topic index. In each topic, the probabilities of words given the topic decrease from left to right. 5

2.1 The score functions $f(h, r, t)$ and the optimization methods (Opt.) of several prominent embedding models for KB completion. In all of these models, the entities h and t are represented by vectors $\boldsymbol{v}_h$ and $\boldsymbol{v}_t \in \mathbb{R}^k$, respectively. 37

3.1 Details of experimental datasets. #**g**: number of ground truth labels; #**docs**: number of documents; #**w**/**d**: the average number of words per document. 56

3.2 NPMI scores (mean and standard deviation) for N20 and N20small datasets. The "*Improve.*" row denotes the absolute improvement accounted for the best result produced by our latent feature model over the baselines. 59

3.3 NPMI scores for TMN and TMNtitle datasets. 59

3.4 NPMI scores for Twitter dataset. 60

3.5 Examples of the 15 most probable topical words on the TMNtitle dataset with $T = 20$. InitDMM denotes the output from the 1500^{th} sample produced by the DMM model, which we use to initialize the w2v-DMM model. Iter=1, Iter=2, Iter=3 and the like refer to the output of our w2v-DMM model after running 1, 2, 3 sampling iterations, respectively. The words found in InitDMM and not found in Iter=500 are underlined. Words found by the w2v-DMM model but not found by the DMM model are in **bold**. 61

3.6 Purity and NMI results (mean and standard deviation) on the N20 and N20small datasets with $\lambda = 0.6$. The "*Improve.*" row denotes the difference between the best result obtained by our model and the baseline model. . . 64
3.7 Purity and NMI results on the TMN and TMNtitle datasets with $\lambda = 0.6$. 65
3.8 Purity and NMI results on the Twitter dataset with $\lambda = 0.6$. 66
3.9 F_1 scores (mean and standard deviation) for N20 and N20small datasets. . 68
3.10 F_1 scores for TMN and TMNtitle datasets. 68
3.11 F_1 scores for Twitter dataset. 69
3.12 Terminology explanations. "General" denotes the common notations used for both LF-LDA and LF-DMM models. 72
3.13 Statistics (Stats.) notations. The first 17 rows describe notations used for both LF-LDA and LF-DMM while the last 2 rows present notations used only for LF-DMM. 73

4.1 Statistics of the experimental datasets used in this study (and previous works). #E is the number of entities, #R is the number of relation types, and #Train, #Valid and #Test are the numbers of correct triples in training, validation and test sets, respectively. 81
4.2 Link prediction results. MR and H@10 denote evaluation metrics of mean rank and Hits@10 (in %), respectively. "NLFeat" abbreviates Node+LinkFeat. The results for NTN (Socher et al., 2013b) listed in this table are taken from Yang et al. (2015) since NTN was originally evaluated on different datasets. [*]: Results from the implementation of Nickel et al. (2016b) because these results are higher than those previously published in Bordes et al. (2013). [our]: We also report the baseline TransE's results (in tenth row from bottom), where we set the relation matrices to identity matrices and only learn the entity and relation vectors, i.e., STransE reduces to the plain TransE. 83

4.3 Hits@10 (in %) for each relation category on the FB15k dataset. The "*Improve.*" row denotes the absolute improvement accounted for STransE over the baseline TransE. 84
4.4 Basic statistics of the dataset after pre-processing. # denotes "number of." 92
4.5 Overall performances of the methods in the test set. **Our method**$_{-W}$ denotes the simplified version of our method. The subscripts denote the relative improvement over the baseline **SE**. 93

5.1 Statistics of the experimental datasets used in this study (and *previous works*). #E is the number of entities, #R is the number of relation types, and #Train, #Valid and #Test are the numbers of correct triples in the training, validation and test sets, respectively. Each validation and test set also contains the same number of incorrect triples as the number of correct triples. 103
5.2 Experimental results of TransE (i.e., TransE-NMM with $\tau = 0$) and TransE-NMM with $\tau = 10$. Micro-averaged (labeled as **Mic.**) and Macro-averaged (labeled as **Mac.**) accuracy results are for the triple classification task. MR, MRR and H@10 abbreviate the mean rank, the mean reciprocal rank and Hits@10 (in %), respectively. "R" and "F" denote the "Raw" and "Filtered" settings used in the entity prediction and relation prediction tasks, respectively. 106
5.3 Micro-averaged accuracy results (in %) for triple classification on WN11 (labeled as **W11**) and FB13 (labeled as **F13**) test sets. Scores in **bold** and underlineare the best and second best scores, respectively. "Avg." denotes the averaged accuracy. 107
5.4 Results on on the NELL186 test set. Results for the entity prediction task are in the "Raw" setting. "-SkipG" abbreviates "-Skip-gram". 108
5.5 Qualitative examples. 109

Abbreviations

NLP	natural language processing
KB	knowledge base
LDA	latent Dirichlet allocation
DMM	Dirichlet multinomial mixture
SGD	stochastic gradient descent
AdaGrad	adaptive gradient
RMSProp	root mean square propagation
L-BFGS	limited-memory Broyden-Fletcher-Goldfarb-Shanno
MAP	maximum a posteriori
WN	Wordnet
FB	Freebase
MRR	mean reciprocal rank
w.r.t.	with respect to

Chapter 1

Introduction

1.1 Motivation

In recent years, natural language processing (NLP) has witnessed one of the strongest research trends towards *representation learning* (Manning, 2015; Goth, 2016; Goldberg, 2016). The goal of representation learning is to "learn representations of the data that make it easier to extract useful information when building classifiers or other predictors" (Bengio et al., 2013). Traditional data representation approaches rely on designing a set of *features* for the data based on human knowledge and prior knowledge, resulting in a feature vector where each dimension represents a unique feature. For example, consider a set of features consisting of a vocabulary of 50,000 word types. A document containing only one word will be represented by a 50,000-dimensional *one-hot* vector where all dimensional entries are zero except the single entry corresponding to the word, which is 1. For a document of 1,000 words, its feature vector can be considered as a linear combination of one-hot vector representations each corresponding to a word in the document, so the feature vector of the document is a high-dimensional sparse vector, i.e., a 50,000-dimensional vector where most entries are zero. In the one-hot representation, words are completely independent of each other, and thus lose their meaning.

In contrast, representation learning makes use of low-dimensional and dense feature representations, i.e., *distributed representations* for features (Hinton, 1986; Bengio et al.,

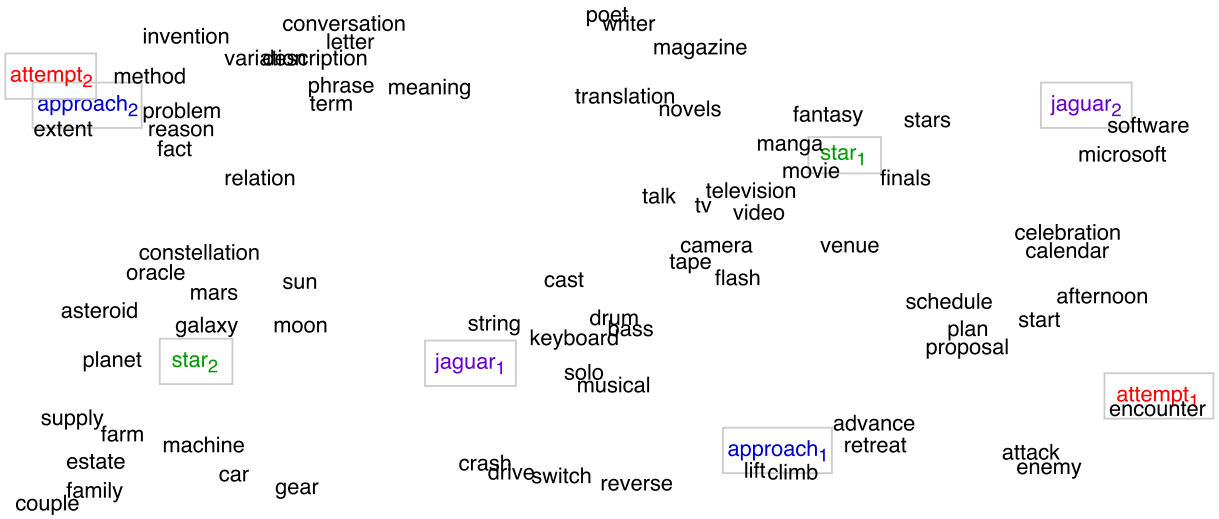


Figure 1.1: Two-dimensional projection of some word vectors. This figure is drawn based on www.socher.org/uploads/Main/MultipleVectorWordEmbedding.pdf (Huang et al., 2012).

2003; Goodfellow et al., 2016). In natural language, a distributed representation of a word—which is also called word *embedding*, word vector or latent feature word representation—captures word meaning by representing the word in a form of low-dimensional real-valued vector.[1] Using distributed word representations thus is able to model relationships between words. For example, the words "king" and "queen" are *royal* forms of "man" and "woman", respectively. We could model this "royal" relational similarity by proposing 300-dimensional word vectors such that $\boldsymbol{v}_{king} - \boldsymbol{v}_{man} \approx \boldsymbol{v}_{queen} - \boldsymbol{v}_{woman}$ where $\boldsymbol{v}_w$ denotes the vector representation of word type w. This kind of semantic relationship plays an important role in NLP (Turney, 2006). Consequently, many unsupervised models have been proposed for learning word vector representations efficiently (Mikolov et al., 2013a,b; Levy and Goldberg, 2014; Pennington et al., 2014). Trained on large unlabeled corpora, these models can produce "good" vector representations that can model the semantic relatedness or similarity of words which occur in similar contexts (Harris, 1954; Firth, 1957). Figure 1.1 illustrates word vectors learned from a billion-word corpus: words such as "moon", "sun", "mars", "planet" and "asteroid" lie close together in the projected vector space. Word vector

[1] In this thesis, the terms *embedding*, *vector representation* and *latent feature representation* will be used interchangeably.

representations learned from large corpora capture various aspects of word meanings, and thus help improve the performance of many downstream NLP tasks such as sentiment analysis, document classification, part-of-speech tagging, named entity recognition, parsing and machine translation (Collobert et al., 2011; Maas et al., 2011; Irsoy and Cardie, 2014; Kim, 2014; Chen and Manning, 2014; Lewis and Steedman, 2014; Weiss et al., 2015; Liu et al., 2015a; Lample et al., 2016; Luong et al., 2016).

This thesis will look at two problems where word embeddings can help with performance. The two problems, building better topic models and knowledge base completion, are ones that can be viewed as vector space models, and the overall task of the thesis is to examine how word embedding vector space models can be helpfully integrated into these.

1.1.1 Topic models

Topic modeling algorithms are statistical methodologies "for analyzing documents, where a document is viewed as a collection of words, and the words in the document are viewed as being generated by an underlying set of topics" (Jordan and Mitchell, 2015).[2] For example, the document in Figure 1.2, entitled "Obit: Baruch Blumberg, Nobel winner and extraterrestrial researcher", briefly summarizes Baruch Blumberg's scientific career and his passing. We highlighted words used in the document as follows: words connected to his scientific career, such as "virus", "disease", "vaccine" and "research", which appear in a topic containing **health**-related words, are colored *red*. Words about his work at NASA, such as "lunar" and "NASA" which are related to **space**, are in *green*. Words also about his passing, such as "died" and "attack" which are related to instances of violence, like **war**, are in *yellow*. If we highlight all words except function words (e.g., "a", "in", "at" and "of"), we would find that the document is an admixture of topics of health, space and war (i.e., the admixture of colors red, green and yellow) with different proportions. Mathematically, the document is characterized by a probability distribution over topics, in which each topic is modeled as a probability distribution over words in a fixed vocabulary (Blei et al., 2003).

[2]In fact, topic models are also used for other kinds of data (Blei, 2012). However, in this thesis we discuss topic modeling in the context of text analysis.

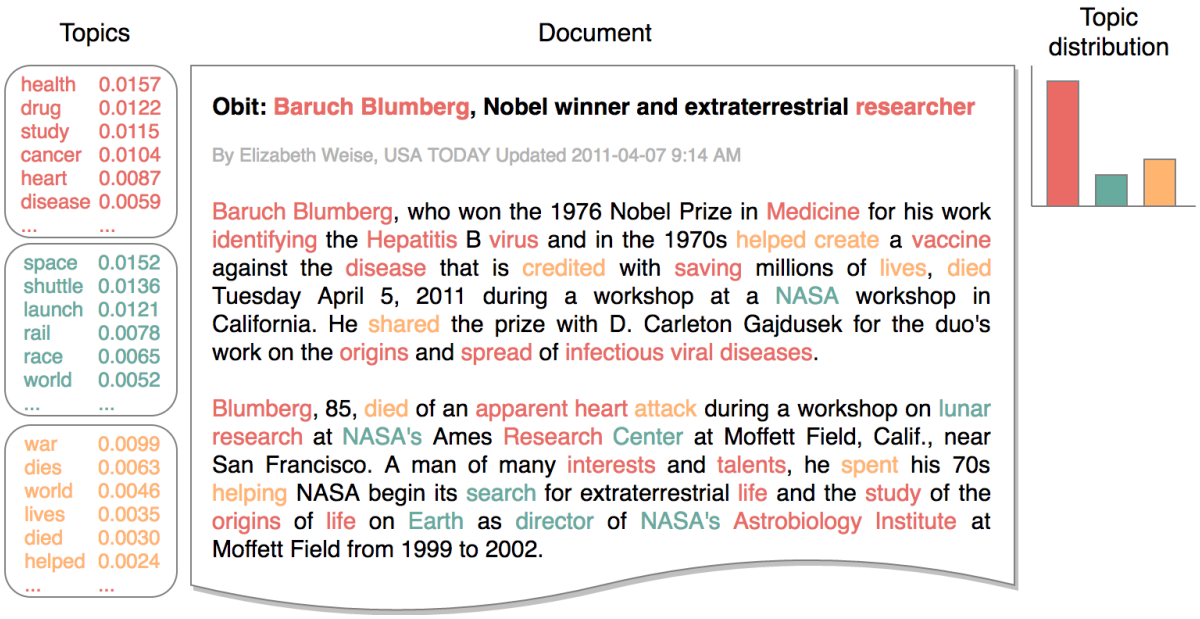


Figure 1.2: An illustrative example of topics and topic assignments in topic modeling.

As illustrated in Figure 1.2, the topic distribution clearly shows that the document mainly focuses on Baruch Blumberg's health research accomplishments because most words in the document are assigned to the health topic, while the word distribution for the health topic associates high probabilities to health-related words such as "drug" and "disease." Formally, given a collection of documents, topic models learn a set of *latent* topics, and infer topic distributions and word distributions from co-occurrence of words within documents (Blei et al., 2003; Steyvers and Griffiths, 2007; Blei, 2012). Then the topic distributions are used to categorize or cluster documents by topic, while the word distributions are used to group words which occur in similar document contexts.

Given the word distributions produced by a topic model, we can represent each word by a distributed vector where each dimension represents a topic and its corresponding value is the probability of the word given the topic. Thus a topic model can be also viewed as a distributed word vector model (Steyvers and Griffiths, 2007; Maas and Ng, 2010). As both assign a latent vector to words, in principle these vectors can be compared. However, the relationship between distributed word representations and topic models is still largely unknown because they come from two different research communities and have different

T	Top-15 topical words
4	egypt china u.s. mubarak bin libya laden france bahrain air report rights court u.n. war
5	critic corner office video game star lady gaga show weekend sheen box park takes man
19	nfl idol draft american show film season sheen n.f.l. back top star charlie players men

Table 1.1: Top 15 topical words when fitting a topic model with 20 topics on a small news title corpus. **T** denotes topic index. In each topic, the probabilities of words given the topic decrease from left to right.

goals. Distributed word vectors come from the neural net research tradition and typically have NLP applications (Manning, 2015; Goldberg, 2016), while topic models come from the Bayesian modeling research tradition and typically have information retrieval applications (Wei and Croft, 2006; Wang et al., 2007; Yi and Allan, 2009). The word representations in a distributed word vector model are typically evaluated by (i) whether words with similar vectors have similar meanings, and (ii) whether "distance" in word representation space is meaningful, e.g., $\boldsymbol{v}_{king} - \boldsymbol{v}_{queen} \approx \boldsymbol{v}_{man} - \boldsymbol{v}_{woman}$ (Mikolov et al., 2013a,b; Baroni et al., 2014; Levy and Goldberg, 2014; Pennington et al., 2014; Österlund et al., 2015). But the word representations in a topic model are typically evaluated by how well they assign words to topics, i.e., to measure how coherent the assignment of words to topics is (Chang et al., 2009; Newman et al., 2010; Stevens et al., 2012; Lau et al., 2014).

In conventional topic models, when the collection of documents is small and/or the documents are short, such as Tweets, instant messages and forum messages, the learned topics will be "noisy" due to the limited information of word co-occurrence within documents. For example, Table 1.1 presents top-15 topical words (i.e., high-probability words) in some topics when fitting a topic model on a small and short corpus. Table 1.1 shows that these topics are not easy to manually label, i.e. it is difficult to interpreter these topics based on top topical words. That is, the top topical words are not semantically coherent. Thus, in this case, the topic model is not able to produce good topics. Motivated by the success of utilizing pre-trained distributed word vectors in various NLP tasks, our first research question is:

RQ 1: *How can word vectors learned on a large external corpus be used to improve topic models estimated from a smaller corpus or from a corpus of short documents ?*

1.1.2 Knowledge base completion

Before introducing this thesis's contributions regarding the first research question, let us return to the example of the "royal" relationship between "king" and "man", and between "queen" and "woman", which was mentioned earlier in this chapter. As illustrated in this example: $\boldsymbol{v}_{king} - \boldsymbol{v}_{man} \approx \boldsymbol{v}_{queen} - \boldsymbol{v}_{woman}$, word vectors learned from a large corpus can also model relational similarities or linguistic regularities between pairs of words as translations in the projected vector space (Turney, 2006; Mikolov et al., 2013a,b; Levy and Goldberg, 2014; Pennington et al., 2014). Figure 1.3 shows another example of a relational similarity with respect to (w.r.t.) word pairs of countries and capital cities:

$$\begin{aligned} \boldsymbol{v}_{Japan} - \boldsymbol{v}_{Tokyo} &\approx \boldsymbol{v}_{Germany} - \boldsymbol{v}_{Berlin} \\ \boldsymbol{v}_{Germany} - \boldsymbol{v}_{Berlin} &\approx \boldsymbol{v}_{Italy} - \boldsymbol{v}_{Rome} \\ \boldsymbol{v}_{Italy} - \boldsymbol{v}_{Rome} &\approx \boldsymbol{v}_{Portugal} - \boldsymbol{v}_{Lisbon} \end{aligned}$$

Let us consider the country and capital pairs in Figure 1.3 to be pairs of entities rather than word types. That is, we represent country and capital entities by low-dimensional and dense vectors. The relational similarity between word pairs is presumably to capture a "capital_of" relationship between country and capital entities. Also, we represent this relationship by a translation vector $\boldsymbol{v}_{capital\_of}$ in the entity vector space. So, we expect:

$$\begin{aligned} \boldsymbol{v}_{Japan} - \boldsymbol{v}_{Tokyo} \approx \boldsymbol{v}_{capital\_of} \quad &\text{, i.e.,} \quad \boldsymbol{v}_{Tokyo} + \boldsymbol{v}_{capital\_of} \approx \boldsymbol{v}_{Japan} \\ \boldsymbol{v}_{Germany} - \boldsymbol{v}_{Berlin} \approx \boldsymbol{v}_{capital\_of} \quad &\text{, i.e.,} \quad \boldsymbol{v}_{Berlin} + \boldsymbol{v}_{capital\_of} \approx \boldsymbol{v}_{Germany} \\ \boldsymbol{v}_{Italy} - \boldsymbol{v}_{Rome} \approx \boldsymbol{v}_{capital\_of} \quad &\text{, i.e.,} \quad \boldsymbol{v}_{Rome} + \boldsymbol{v}_{capital\_of} \approx \boldsymbol{v}_{Italy} \end{aligned}$$

This intuition inspired the TransE model—a well-known embedding model for *knowledge base completion* or *link prediction* in knowledge bases (Bordes et al., 2013).

Knowledge bases (KBs) are collections of real-world triples, where each triple or fact (h, r, t) in KBs represents some relation r between a head entity h and a tail entity t. KBs can thus be formalized as directed multi-relational graphs, where nodes correspond to

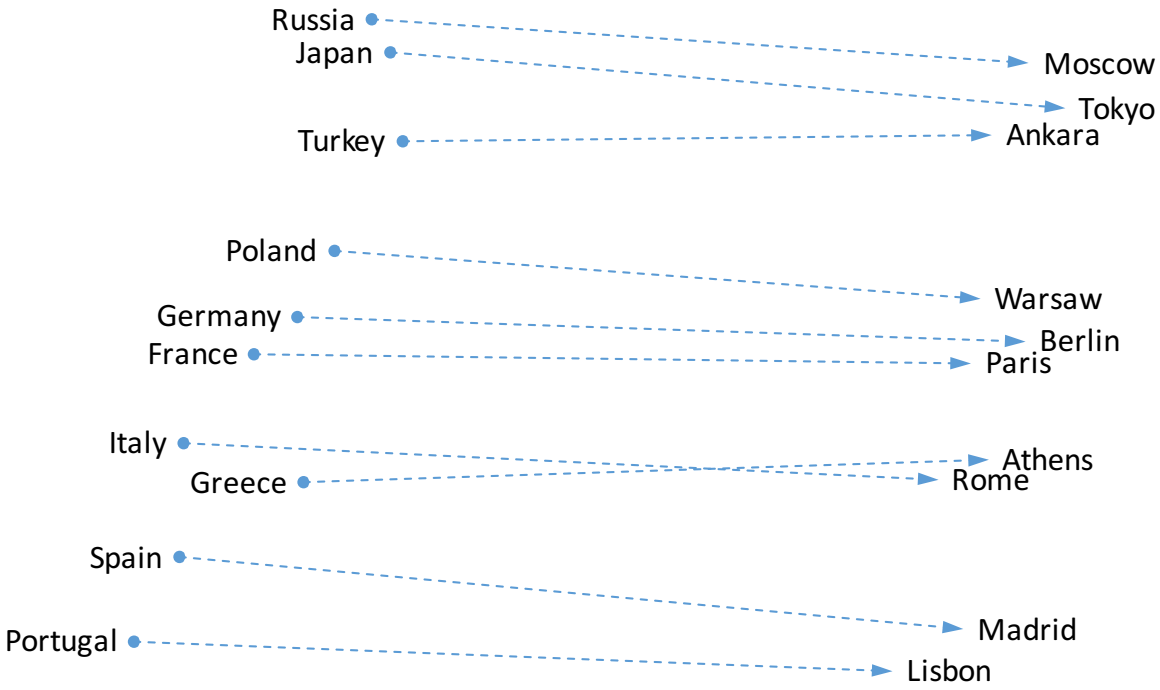


Figure 1.3: Two-dimensional projection of vectors of countries and their capital cities. This figure is drawn based on Mikolov et al. (2013b).

entities and edges linking the nodes encode various kinds of relationship (García-Durán et al., 2016; Nickel et al., 2016a). Here entities are real-world things or objects such as persons, places, organizations, music tracks or movies. Each relation type defines a certain relationship between entities. For example, as illustrated in Figure 1.4, the relation type "child_of" relates person entities with each other, while the relation type "born_in" relates person entities with place entities. Several KB examples include the domain-specific KB GeneOntology[3] and popular generic KBs of WordNet (Fellbaum, 1998), ConceptNet (Liu and Singh, 2004), YAGO (Suchanek et al., 2007), Freebase (Bollacker et al., 2008), NELL (Carlson et al., 2010) and DBpedia (Lehmann et al., 2015) as well as commercial KBs such as Google's Knowledge Graph, Microsoft's Satori and Facebook's Open Graph. Nowadays, KBs are used in a number of commercial applications including search engines such as Google, Microsoft's Bing and Facebook's Graph search. They also are useful

[3] http://www.geneontology.org

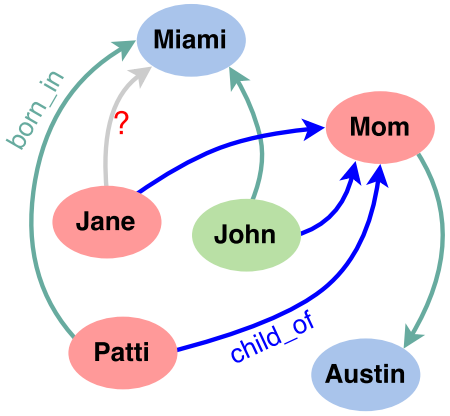


Figure 1.4: An illustration of (incomplete) knowledge base, with 4 person entities, 2 place entities, 2 relation types and total 6 triple facts. This figure is drawn based on Weston and Bordes (2014).

resources for many NLP tasks such as question answering (Ferrucci, 2012; Fader et al., 2014), word sense disambiguation (Navigli and Velardi, 2005; Agirre et al., 2013), semantic parsing (Krishnamurthy and Mitchell, 2012; Berant et al., 2013) and co-reference resolution (Ponzetto and Strube, 2006; Dutta and Weikum, 2015).

A main issue is that even very large KBs, such as Freebase and DBpedia, which contain billions of fact triples about the world, are still far from complete. Specifically, in English DBpedia 2014, 60% of person entities miss a place of birth and 58% of the scientists do not have a fact about what they are known for (Krompaß et al., 2015). In Freebase, 71% of 3 million person entities miss a place of birth, 75% do not have a nationality while 94% have no facts about their parents (West et al., 2014). So, in terms of a specific application, question answering systems based on incomplete KBs would not provide a correct answer given a correctly interpreted question. For example, given the incomplete KB in Figure 1.4, it would be impossible to answer the question "where was Jane born ?", although the question is completely matched with existing entity and relation type information (i.e., "Jane" and "born_in") in KB. Consequently, much work has been devoted towards knowledge base completion to perform link prediction in KBs, which attempts to predict whether a relationship/triple not in the KB is likely to be true, i.e., to add new triples by leveraging existing triples in the KB (Lao and Cohen, 2010; Bordes et al., 2012;

Gardner et al., 2014; García-Durán et al., 2016). For example, we would like to predict the missing tail entity in the incomplete triple (Jane, born_in, ?) or predict whether the triple (Jane, born_in, Miami) is correct or not.

Recently, embedding models for knowledge base completion have been proven to give state-of-the-art link prediction performances (Nickel et al., 2011; Jenatton et al., 2012; Bordes et al., 2013; Socher et al., 2013b; Wang et al., 2014a; Dong et al., 2014; Lin et al., 2015b; Guu et al., 2015; Toutanova and Chen, 2015; García-Durán et al., 2016; Trouillon et al., 2016; Toutanova et al., 2016; Nickel et al., 2016b). In all these models, entities are represented by latent feature vectors while relation types are represented by latent feature vectors and/or matrices and/or third-order tensors. Among these models, the well-known TransE model (Bordes et al., 2013) is a simple and powerful model. TransE learns low-dimensional and dense vectors for every entity and relation type, so that each relation type corresponds to a translation vector operating on the vectors representing the entities, i.e., $\boldsymbol{v}_h + \boldsymbol{v}_r \approx \boldsymbol{v}_t$ for each fact triple (h, r, t). TransE thus is suitable for 1-to-1 relationships, such as "capital_of", where a head entity is linked to at most one tail entity given a relation type. Because of using only one translation vector to represent each relation type, TransE is not well-suited for Many-to-1, 1-to-Many and Many-to-Many relationships,[4] such as for relation types "born_in", "place_of_birth" and "research_fields." For example in Figure 1.4, using one vector representing the relation type "born_in" cannot capture both the translating direction from "Patti" to "Miami" and its inverse direction from "Mom" to "Austin." Thus, our second research question is:

RQ 2: *How can we develop a new embedding model for KB completion to better capture Many-to-1, 1-to-Many and Many-to-Many relationships ?*

It is worth noting that most embedding models only take triples into account. So, these models ignore useful information implicitly presented by the structure of the KB. For

[4]A relation type r is classified Many-to-1 if multiple head entities can be connected by r to at most one tail entity. A relation type r is classified 1-to-Many if multiple tail entities can be linked by r from at most one head entity. A relation type r is classified Many-to-Many if multiple head entities can be connected by r to a tail entity and vice versa.

example, the relation path $h \xrightarrow{\text{born\_in\_city}} e \xrightarrow{\text{city\_in\_country}} t$ should indicate a relationship "nationality" between the h and t entities. Also, neighborhood information of entities could be useful for predicting the relationship between two entities as well. For example in the KB NELL (Carlson et al., 2010), we have information such as that if a person works for an organization and that this person also leads that organization, then it is likely that this person is the CEO of that organization. Our third research question is:

RQ 3: *How can we develop a new embedding model using such useful information as relation path or neighborhood for better link prediction in KBs ?*

It might also be noted that there already exist dozens of embedding models proposed for KB completion (see Section 2.5). So, it would be interesting to further explore those models for a new application where we could formulate its corresponding data into triples. For example, in Web search engines, we observe *user*-oriented relationships between submitted *queries* and *documents* returned by the search engines. That is, we have triple representations (query, user, document) in which for each user-oriented relationship, we would have many queries and documents, resulting in a lot of Many-to-Many relationships. Thus a new model developed for answering the second research question **RQ 2** is particularly well-suited for this application. Our fourth research question is:

RQ 4: *How can we adapt a KB completion model to some new application such as in Web search engines ?*

1.2 Aims and Contributions

By investigating the research questions mentioned above, the aims of this thesis are:

- To develop new topic models that incorporate word representations containing external information from a large corpus, with the goal of improving topic inference in a small corpus or a corpus of short documents like Tweets.

- To develop new embedding models for improving link prediction in KBs, and to investigate a new application task where these models might be useful.

In order to satisfy the aims, we present two topic models to answer **RQ 1** in Chapter 3. In addition, we propose a triple-based embedding model and a neighborhood mixture model for KB completion to address **RQ 2** and **RQ 3** in chapters 4 and 5, respectively. We also explore a new application task on search personalization in Web search engines to answer **RQ 4** in Chapter 4, where we apply our triple-based KB completion model to improve search results. The major contributions of this thesis are summarized as follows:

- **Improving topic models with word representations:** We propose two novel latent feature topic models which integrate latent feature word vectors into two different topic models. In addition, we introduce our inference procedures for the new topic models. Furthermore, we compare the use of two well-known sets of pre-trained word vectors with the new topic models. We obtain significant improvements on topic coherence evaluation, document clustering and document classification tasks. In particular, we find the highest improvements are on collections with few documents and collections of short documents.

- **STransE—a novel embedding model of entities and relationships:** We present a new triple-based embedding model, which we call *STransE*, for KB completion. Our STransE model extends the TransE model (Bordes et al., 2013), to represent each relation type by a low-dimensional translation vector and two projection matrices. STransE achieves better link prediction results on two benchmark datasets than previous embedding models. More specifically, we find the highest improvements of STransE over the baseline TransE are for Many-to-1, 1-to-Many and Many-to-Many relationships.

- **Application to search personalization:** We also present a new embedding approach for the search personalization task, by applying STransE to learn user topical interests from input queries and returned documents. Here, we represent each

user by a user vector embedding and two user matrices, while vector representations of queries and documents are pre-learned by utilizing the well-known topic model Latent Dirichlet Allocation (Blei et al., 2003). In fact, this application of search personalization serves as a bridge for connecting a topic model and a KB completion model. Experimental results show that our new approach significantly improves the search performance.

- **Neighborhood mixture model for knowledge base completion:** We propose a new neighborhood mixture model where we formalize an entity representation as a mixture of its neighborhood in the KB. We demonstrate its usefulness by applying it to the TransE model. On three benchmark datasets, experiments show that the neighborhood information significantly helps to improve the results of TransE, leading to better performance over competitive baselines for triple classification, entity prediction and relation prediction tasks.

1.3 Outline and Origins

The remainder of this thesis is outlined as follows:

- **Chapter 2** gives necessary technical background used in this thesis, and also briefly reviews related work w.r.t. topic models and KB completion models.

- **Chapter 3** presents our latent feature topic models. Chapter 3 also provides experimental results on six document collections for three evaluation tasks of topic coherence, document clustering and document classification.

- **Chapter 4** introduces our embedding model STransE, and compares its link prediction results on two benchmark datasets to previously published results. Chapter 4 also explores a new application of STransE for the search personalization task.

- **Chapter 5** presents our neighborhood mixture model for link prediction in KBs and experimental results on three benchmark datasets for three KB completion tasks.

- **Chapter 6** recaps our major findings and draws possible directions for future work.

Figure 1.5 illustrates the dependencies of sections in chapters 2, 3, 4 and 5. The following peer-reviewed publications form the basis of chapters 3, 4 and 5 in this thesis:

- Chapter 3 is based on Nguyen et al. (2015a):

  Dat Quoc Nguyen, Richard Billingsley, Lan Du and Mark Johnson. Improving Topic Models with Latent Feature Word Representations. *Transactions of the Association for Computational Linguistics*, 3:299-313, June 2015. URL `https://transacl.org/ojs/index.php/tacl/article/view/582/158`

- Chapter 4 is based on Nguyen et al. (2016b) and Vu et al. (2017a):

  Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu and Mark Johnson. STransE: a novel embedding model of entities and relationships in knowledge bases. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT 2016, pages 460-466, June 2016. URL `http://www.aclweb.org/anthology/N16-1054`

  Thanh Vu*, Dat Quoc Nguyen*, Mark Johnson, Dawei Song and Alistair Willis. Search Personalization with Embeddings. In *Proceedings of the 39th European Conference on Information Retrieval*, ECIR 2017, pages 598-604, April 2017. URL `https://doi.org/10.1007/978-3-319-56608-5_54` (*: The first two authors contributed **equally** to this work)

- Chapter 5 is based on Nguyen et al. (2016a):

  Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu and Mark Johnson. Neighborhood Mixture Model for Knowledge Base Completion. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, CoNLL 2016, pages 40-50, August 2016. URL `http://www.aclweb.org/anthology/K16-1005`

Also, this thesis draws insights and/or experiences from some related material:

- Dat Quoc Nguyen. jLDADMM: A Java package for the LDA and DMM topic models. Technical report, June 2015. URL `http://www.jldadmm.sourceforge.net/jldadmmV1.pdf`

- Dat Quoc Nguyen, Kairit Sirts and Mark Johnson. Improving Topic Coherence with Latent Feature Word Representations in MAP Estimation for Topic Modeling. In *Proceedings of the 13th Australasian Language Technology Association Workshop*, ALTA 2015, pages 116-121, December 2015. URL `http://www.aclweb.org/anthology/U15-1014`

- Didi Surian, Dat Quoc Nguyen, Georgina Kennedy, Mark Johnson, Enrico Coiera and Adam G. Dunn. Characterizing Twitter Discussions About HPV Vaccines Using Topic Modeling and Community Detection. *Journal of Medical Internet Research*, 18(8):e232, August 2016. URL `https://doi.org/10.2196/jmir.6045`

- Dat Quoc Nguyen. An overview of embedding models of entities and relationships for knowledge base completion. *arXiv preprint*, arXiv:1703.08098, March 2017. URL `https://arxiv.org/abs/1703.08098`

- Dai Quoc Nguyen, Dat Quoc Nguyen, Ashutosh Modi, Stefan Thater and Manfred Pinkal. A Mixture Model for Learning Multi-Sense Word Embeddings. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics*, *SEM 2017, to appear.

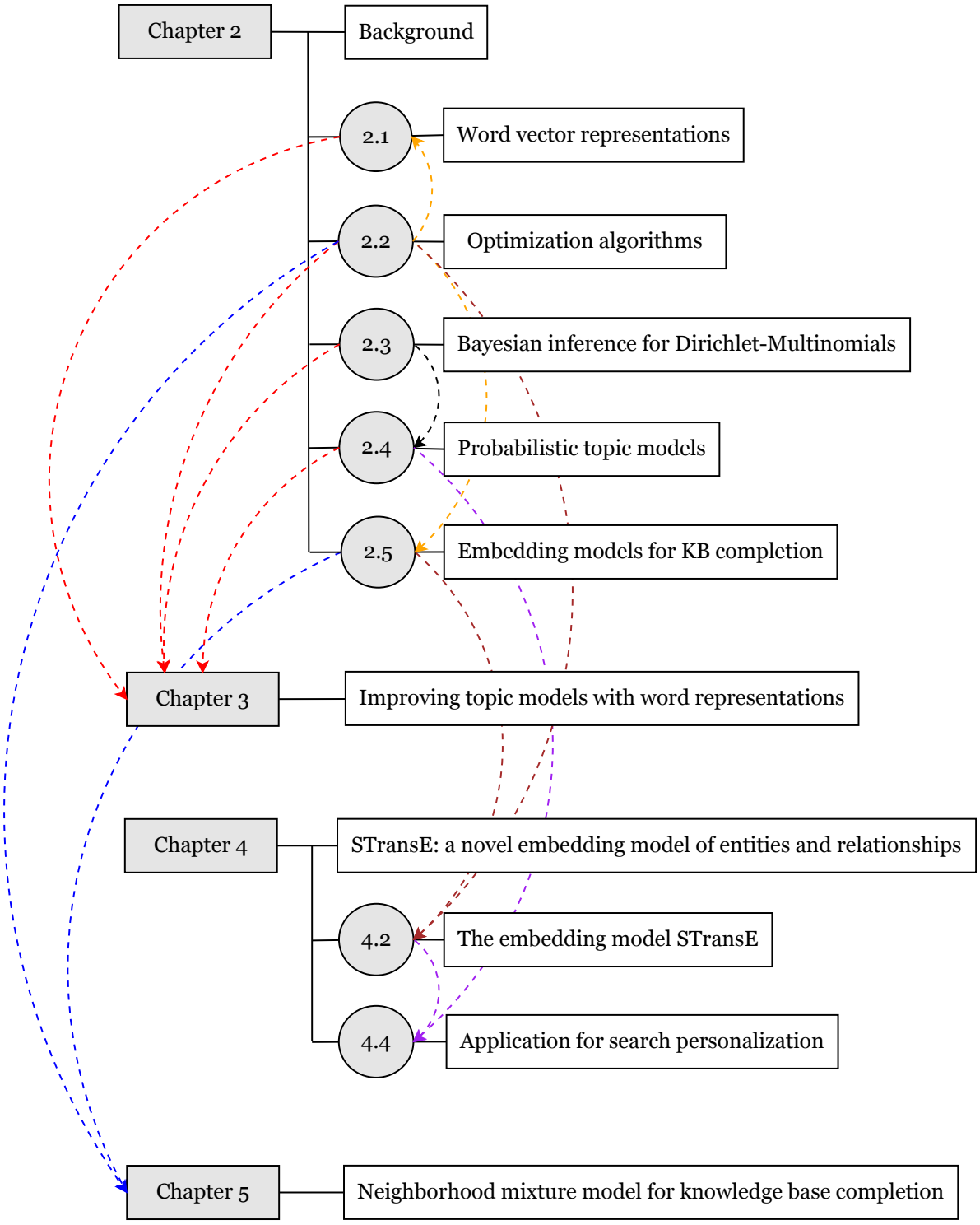


Figure 1.5: Dependency diagram of chapters and sections.

Chapter 2

Background

Contents

2.1 Word vector representations 18
2.1.1 Word2Vec Skip-gram model 19
2.1.2 GloVe model 20
2.2 Optimization algorithms 20
2.2.1 Gradient descent variants 21
2.2.2 AdaGrad 22
2.2.3 RMSProp 23
2.2.4 L-BFGS 24
2.3 Bayesian inference for Dirichlet-Multinomials 24
2.3.1 Bayesian inference 25
2.3.2 Dirichlet-Multinomials 25
2.3.3 A simple Dirichlet-multinomial model 28
2.3.4 Inference via Gibbs sampling 29
2.4 Probabilistic topic models 30
2.4.1 Latent Dirichlet Allocation 30
2.4.2 Advanced topic models 32
2.4.3 Dirichlet Multinomial Mixture for short texts 33
2.4.4 Topic models with word representations 34
2.5 Embedding models for KB completion 35
2.5.1 A general approach 36
2.5.2 Specific models 36
2.5.3 Other KB completion models 39
2.6 Summary 40

This chapter provides a brief overview of related work and the required technical background. We start by presenting two widely-used unsupervised models for learning word vectors in Section 2.1. Next we present optimization algorithms to minimize an objective

function in Section 2.2, in which these algorithms are used by the two word vector models and later used in our new topic models in Chapter 3 and knowledge base (KB) completion models in chapters 4 and 5. We then focus on the fundamentals of probabilistic topic models by introducing basics of Bayesian inference in Section 2.3 and discussing baseline and advanced topic models in Section 2.4. Finally, we overview embedding models for KB completion in Section 2.5.

2.1 Word vector representations

Latent feature representations of words have been already used successfully in many NLP tasks (Manning, 2015; Goth, 2016). Many methods have been proposed for learning real-valued latent feature word vectors (Goldberg, 2016). The general hypothesis behind those methods is that words which occur in similar contexts share semantic relatedness or similarity (Harris, 1954; Firth, 1957). Traditional count-based methods rely on word co-occurrence counts in the contexts, e.g., methods, which are based on Pointwise Mutual Information[1] or matrix factorization, use window contexts of 5 or 10 words (Lund and Burgess, 1996; Bullinaria and Levy, 2007; Turney and Pantel, 2010). Recent prediction-based models maximize the probability of predicting contexts where a target word occurs, or vice versa, predicting the target word given its contexts (Bengio et al., 2003; Collobert and Weston, 2008; Mikolov et al., 2013a,b). Baroni et al. (2014) showed that the prediction-based models outperform the count-based models. However, Levy and Goldberg (2014), Pennington et al. (2014) and Österlund et al. (2015) later showed that the count-based methods and prediction-based methods are not qualitatively different on a wider range of semantic evaluation tasks.

The following subsections present two recent, widely-used models for learning word vector representations. We utilize the pre-trained word vectors produced by these models in Chapter 3.

[1]See Church and Hanks (1990) for the definition of Pointwise Mutual Information.

2.1.1 Word2Vec Skip-gram model

Given a corpus of words $D = \{w_1, w_2, ..., w_M\}$, the prediction-based Word2Vec Skip-gram model[2] (Mikolov et al., 2013b) minimizes the following negative log-likelihood objective function:

$$\mathcal{L} = -\sum_{t=1}^{M} \sum_{\substack{-k \leq j \leq k \\ j \neq 0}} \log \mathrm{P}(w_{t+j} \mid w_t) \tag{2.1}$$

where w_{t+j} is a *context* word given the *target* word w_t with k to be the context size.

Let $\boldsymbol{\omega} \in \mathbb{R}^{V \times d}$ be a target word-vector matrix and $\mathbf{C} \in \mathbb{R}^{V \times d}$ be a context word-vector matrix, where V is the size of the vocabulary W, i.e., $V = |W|$. Here $\boldsymbol{\omega}_i$ and $\mathbf{C}_i$ represent (row) vectors associated with the word type with vocabulary index i. The model defines the probability $\mathrm{P}(c \mid w)$ of predicting the context word c given the target word w using the softmax function as follows:

$$\mathrm{P}(c \mid w) = \frac{\exp(\mathbf{C}_{i(c)} \cdot \boldsymbol{\omega}_{i(w)})}{\sum_{w' \in W} \exp(\mathbf{C}_{i(w')} \cdot \boldsymbol{\omega}_{i(w)})} \tag{2.2}$$

where the subscript $i(x)$ refers to the index of the word type x in the vocabulary.

Computing $\log \mathrm{P}(c \mid w)$ is expensive for each training target word, so the Word2Vec Skip-gram model approximates $\log \mathrm{P}(c \mid w)$ with a negative-sampling objective:

$$\mathcal{O}_{c,w} = \log \sigma\left(\mathbf{C}_{i(c)} \cdot \boldsymbol{\omega}_{i(w)}\right) + \sum_{n=1}^{n=N} \mathbb{E}_{w'_n \sim \mathrm{P}_W}\left[\log \sigma\left(-\mathbf{C}_{i(w'_n)} \cdot \boldsymbol{\omega}_{i(w)}\right)\right] \tag{2.3}$$

where σ is the sigmoid function: $\sigma(u) = \dfrac{1}{1 + e^{-u}}$ and words $w'_1, w'_2, ..., w'_N$ are randomly sampled from the vocabulary W using a noise distribution P_W. A typical noise distribution is the *unigram distribution* raised to the 3/4 power (Mikolov et al., 2013b).[3]

The model is then trained to learn word vectors (i.e., $\boldsymbol{\omega}$ and $\mathbf{C}$) using vanilla Stochastic Gradient Descent (SGD—see Section 2.2.1).

[2] `https://code.google.com/p/word2vec`

[3] The unigram distribution can be thought as a categorical distribution (see Section 2.3.2) parameterized by a vector of unigram probabilities.

2.1.2 GloVe model

The GloVe model[4] (Pennington et al., 2014) is another widely-used model for learning word vectors, by combining advantages of both count- and prediction-based methods. Let $\mathbf{X}$ be the word-context co-occurrence matrix where X_{ij} denotes the number of times the i^{th} word type occurs near the j^{th} word type in a word corpus. GloVe learns $\boldsymbol{\omega}$ and $\mathbf{C}$ from $\mathbf{X}$ by minimizing the following objective function:

$$\mathcal{L} = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(\boldsymbol{\omega}_i \cdot \mathbf{C}_j + b_i^{(\boldsymbol{\omega})} + b_j^{(\mathbf{C})} - \log X_{ij}\right)^2$$

where $b_i^{(\boldsymbol{\omega})}$ and $b_j^{(\mathbf{C})}$ are the unknown target and context bias terms associated with the i^{th} and j^{th} word types, respectively. In addition, $f\left(X_{ij}\right)$ is defined as the weighting function:

$$f\left(X_{ij}\right) = \begin{cases} \left(\frac{X_{ij}}{100}\right)^{3/4} & \text{if } X_{ij} < 100 \\ 1 & \text{otherwise.} \end{cases}$$

The GloVe model is trained to learn $\boldsymbol{\omega}$ and $\mathbf{C}$ by using SGD with AdaGrad adaptive learning (AdaGrad—see Section 2.2.2), and then using $\boldsymbol{\omega} + \mathbf{C}$ to obtain final word vectors.

2.2 Optimization algorithms

This section presents optimization algorithms for finding the minimum of an objective function $\mathcal{F}(\boldsymbol{\theta})$ parameterized by the parameter vector $\boldsymbol{\theta} \in \mathbb{R}^k$. These algorithms are used in learning the word vector representations in Section 2.1, and also in chapters 3, 4 and 5. We describe different variants of gradient descent (Robbins and Monro, 1951) in Section 2.2.1. We then present adaptive learning algorithms AdaGrad (Duchi et al., 2011) and RMSProp (Tieleman and Hinton, 2012) in sections 2.2.2 and 2.2.3, respectively. Finally, we present the Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm (Liu and Nocedal, 1989) in Section 2.2.4. The material in Section 2.2 is based on Ruder (2016) and Nocedal and Wright (2006, Section 7.2).

[4] `http://nlp.stanford.edu/projects/glove/`

We consider to minimizing the objective function $\mathcal{F}(\boldsymbol{\theta})$ which can be factorized into functions $\mathcal{F}_i(\boldsymbol{\theta})$ as follows:

$$\mathcal{F}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \mathcal{F}_i(\boldsymbol{\theta}) \tag{2.4}$$

where each function $\mathcal{F}_i(\boldsymbol{\theta})$ is associated with the i^{th} observation in the training dataset. Here, each training observation (i.e., training example) could be a pair of document and its label in a document classification task, or a pair of correct and incorrect triples $(\mathsf{head\ entity}, \mathsf{relation}, \mathsf{tail\ entity})$ in the link prediction task (see chapters 4 and 5), or a tuple of a word and its negative-sampled words as in the Word2Vec Skip-gram model (see Section 2.1.1).

2.2.1 Gradient descent variants

Gradient descent algorithms update the parameters $\boldsymbol{\theta}$ in the opposite direction of the gradient $\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta})$ of the objective function with respect to (w.r.t.) the parameters (Ruder, 2016). As shown in Algorithm 1, for each update iteration, the "batch" gradient descent algorithm first computes the gradient $\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta})$, then updates the parameters $\boldsymbol{\theta}$ in the direction of the negative gradient with the learning rate η which controls how large we perform an update. In Algorithm 1, hyper-parameter $\eta > 0$ is a scalar *learning rate*.

Algorithm 1: Batch gradient descent algorithm

Require: Choose an initial parameter vector $\boldsymbol{\theta}$ and learning rate $\eta > 0$

repeat

$$\boldsymbol{\theta} \coloneqq \boldsymbol{\theta} - \eta\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta}) = \boldsymbol{\theta} - \eta\sum_{i=1}^{n}\nabla_{\boldsymbol{\theta}}\mathcal{F}_i(\boldsymbol{\theta})$$

until *convergence*

Batch methods, such as batch gradient descent or L-BFGS which is presented in Section 2.2.4, use the whole training dataset to compute the gradient. They tend to converge very well to the global minimum for convex functions and to a local optima for non-convex functions. However, if the number of training examples n is large, computing the gradient for a single parameter update becomes very slow in such batch methods.

Algorithm 2: Vanilla stochastic gradient descent (SGD) algorithm

Require: Choose an initial parameter vector $\boldsymbol{\theta}$ and learning rate $\eta > 0$.

`// Repeat until convergence`

for *t = 1, 2, ...* **do**

  $\boldsymbol{\theta} \coloneqq \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{F}_{i(t)}(\boldsymbol{\theta})$ `//` $i(t) \in [1, n]$ `is function index at time step` t.

In Algorithm 2 presenting the stochastic gradient descent (SGD) algorithm, the index $i(t) \in [1, n]$ of the function $\mathcal{F}_{i(t)}$ to be evaluated at the time step t is selected either sequentially or at random. Unlike batch methods, as detailed in Algorithm 2, SGD updates parameters after seeing each single training example. SGD thus is usually faster and could be used in an "online" setting when incorporating new training data. Alternatively, the mini-batch gradient descent algorithm computes the gradient for each parameter update w.r.t. every mini-batch of more than one training example. Here the mini-batch sizes vary for different applications, commonly ranging from 10 to 256.

One of the challenges for SGD and mini-batch gradient descent is to choose a proper learning rate η. Setting this hyper-parameter too low could make the algorithm slow to converge, while a too large learning rate could cause the algorithm to diverge. Additionally, the same learning rate is applied for all frequent or infrequent parameters, thus not suitable for dealing with sparse data. In next sections 2.2.2 and 2.2.3, we present two improvements on the basic SGD algorithm, including AdaGrad and RMSProp. Other well-known variants including AdaDelta and Adam algorithms can be found in Zeiler (2012) and Kingma and Ba (2014), respectively. See detailed reviews of these algorithms in Ruder (2016).

2.2.2 AdaGrad

AdaGrad (Duchi et al., 2011) adapts the learning rate separately for every parameter based on historical information, resulting in small learning rates for the frequently appearing parameters and larger learning rates for the infrequent parameters.

Let $g_j^{(t)}$ denote the partial derivative w.r.t the j^{th} parameter θ_j at time step t: $g_j^{(t)} = \frac{\partial \mathcal{F}_{i(t)}}{\partial \theta_j}(\boldsymbol{\theta}^{(t)})$. The SGD update rule for every parameter θ_j at each time step t also is:

$$\theta_j^{(t+1)} = \theta_j^{(t)} - \eta \cdot g_j^{(t)} \tag{2.5}$$

AdaGrad modifies the per-parameter update rule of SGD as follows:

$$\theta_j^{(t+1)} = \theta_j^{(t)} - \frac{\eta}{\sqrt{G_{j,j}^{(t)} + \epsilon}} \cdot g_j^{(t)} \tag{2.6}$$

where ϵ is a smoothing term to avoid division by zero,[5] and $\boldsymbol{G}^{(t)} \in \mathbb{R}^{k \times k}$ is a diagonal matrix where the j, j diagonal element $G_{j,j}^{(t)}$ is defined as $G_{j,j}^{(t)} = \sum_{t'=1}^{t} (g_j^{(t')})^2 = G_{j,j}^{(t-1)} + (g_j^{(t)})^2$, i.e., the sum of the squares of all historical partial derivatives w.r.t. θ_j. So the AdaGrad update rule may be written by using element-wise multiplication operator $\odot$ as follows:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta\big(\mathsf{diag}(\boldsymbol{G}^{(t)}) + \boldsymbol{\epsilon}\big)^{-1/2} \odot \boldsymbol{g}^{(t)} \tag{2.7}$$

where $\boldsymbol{\epsilon} \in \mathbb{R}^k$ is the smoothing vector such that $\epsilon_j = \epsilon \; \forall \; j = 1, 2, ..., k$. And $\mathsf{diag}(\boldsymbol{G}^{(t)})$ is the vector of the diagonal elements of $\boldsymbol{G}^{(t)}$ while $\boldsymbol{g}^{(t)}$ is the gradient $\nabla_{\boldsymbol{\theta}} \mathcal{F}_{i(t)}(\boldsymbol{\theta}^{(t)})$ w.r.t the parameters $\boldsymbol{\theta}$ at the time step t.

2.2.3 RMSProp

Similar to AdaGrad, RMSProp (Tieleman and Hinton, 2012) is also a method for adapting the learning rate separately for each of the parameters. RMSProp update rule for every parameter θ_j at each time step t is as follows:

$$\theta_j^{(t+1)} = \theta_j^{(t)} - \frac{\eta}{\sqrt{v_j^{(t)} + \epsilon}} \cdot g_j^{(t)}, \text{ in which} \tag{2.8}$$

$$v_j^{(t)} = \rho \, v_j^{(t-1)} + (1 - \rho) \, (g_j^{(t)})^2 \tag{2.9}$$

where ρ is a scalar constant controlling the magnitude of the squared partial derivative for every parameter.[6]

[5] ϵ is commonly set to 10^{-8}.

[6] Tieleman and Hinton (2012) suggests to set ρ to be 0.9.

2.2.4 L-BFGS

L-BFGS (Liu and Nocedal, 1989) is another popular batch method for parameter estimation. It is the algorithm of choice for fitting log-linear models (Andrew and Gao, 2007). L-BFGS has the update rule for parameters $\boldsymbol{\theta}$ each time step t as follows:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta\, \mathbf{H}_{(t)} \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{\theta}^{(t)}) \tag{2.10}$$

where $\mathbf{H}_{(t)} \in \mathbb{R}^{k \times k}$, which is an approximation to the inverse Hessian, is updated at every time step t based on the past m update steps as:

$$\begin{aligned}
\mathbf{H}_{(t)} \;=\; & \big(\mathbf{V}_{(t-1)}^{\top} \cdots \mathbf{V}_{(t-m)}^{\top}\big) \mathbf{H}_{(t)}^{0} \big(\mathbf{V}_{(t-m)} \cdots \mathbf{V}_{(t-1)}\big) \\
& + \rho_{(t-m)} \big(\mathbf{V}_{(t-1)}^{\top} \cdots \mathbf{V}_{(t-m+1)}^{\top}\big) \boldsymbol{s}_{(t-m)} \boldsymbol{s}_{(t-m)}^{\top} \big(\mathbf{V}_{(t-m+1)} \cdots \mathbf{V}_{(t-1)}\big) \\
& + \rho_{(t-m+1)} \big(\mathbf{V}_{(t-1)}^{\top} \cdots \mathbf{V}_{(t-m+2)}^{\top}\big) \boldsymbol{s}_{(t-m+1)} \boldsymbol{s}_{(t-m+1)}^{\top} \big(\mathbf{V}_{(t-m+2)} \cdots \mathbf{V}_{(t-1)}\big) \\
& + \cdots + \rho_{(t-1)} \boldsymbol{s}_{(t-1)} \boldsymbol{s}_{(t-1)}^{\top}
\end{aligned} \tag{2.11}$$

where $\mathbf{H}_{(t)}^{0} \in \mathbb{R}^{k \times k}$ is a positive initial matrix, and:

$$\mathbf{V}_{(t)} = \mathbf{I} - \rho_{(t)} \boldsymbol{y}_{(t)} \boldsymbol{s}_{(t)}^{\top} \tag{2.12}$$

$$\rho_{(t)} = \frac{1}{\boldsymbol{y}_{(t)}^{\top} \boldsymbol{s}_{(t)}} \tag{2.13}$$

$$\boldsymbol{s}_{(t)} = \boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)} \tag{2.14}$$

$$\boldsymbol{y}_{(t)} = \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{\theta}^{(t+1)}) - \nabla_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{\theta}^{(t)}) \tag{2.15}$$

More details of the L-BFGS algorithm can be also found in Nocedal and Wright (2006).

2.3 Bayesian inference for Dirichlet-Multinomials

In this section, we introduce necessary background relevant to topic modeling—one of most common successful applications of Bayesian inference. The material in Section 2.3 is based on Heinrich (2005), Johnson (2011) and Lim (2016).

2.3.1 Bayesian inference

A Bayesian model treats its unknown parameters as random variables, and allows a *prior* distribution over these parameters. Inference in Bayesian models is based on the *posterior* distribution $\mathrm{P}(\boldsymbol{\theta} \mid \mathcal{W})$ over the model parameters $\boldsymbol{\theta}$ conditional on the observed data $\mathcal{D}$, according to Bayes' rule:

$$\begin{aligned} \mathrm{P}(\boldsymbol{\theta} \mid \mathcal{D}) &= \frac{\mathrm{P}(\mathcal{D} \mid \boldsymbol{\theta})\mathrm{P}(\boldsymbol{\theta})}{\mathrm{P}(\mathcal{D})} \\ \text{i.e., } \mathsf{posterior} &= \frac{\mathsf{likelihood} \cdot \mathsf{prior}}{\mathsf{evidence}} \end{aligned} \tag{2.16}$$

where $\mathrm{P}(\mathcal{D} \mid \boldsymbol{\theta})$ is the *likelihood*, $\mathrm{P}(\boldsymbol{\theta})$ is the prior distribution, and $\mathrm{P}(\mathcal{D})$ is the marginal data likelihood also called *evidence*, and:[7]

$$\mathrm{P}(\mathcal{D}) = \int \mathrm{P}(\mathcal{D} \mid \boldsymbol{\theta}')\mathrm{P}(\boldsymbol{\theta}') \, \mathrm{d}\boldsymbol{\theta}' \tag{2.17}$$

Because $\mathrm{P}(\mathcal{D})$ is often hard or intractable to calculate, and $\mathrm{P}(\mathcal{D})$ does not depend on the parameters $\boldsymbol{\theta}$, the posterior is commonly written in the form of a proportionality:

$$\begin{aligned} \mathrm{P}(\boldsymbol{\theta} \mid \mathcal{D}) &\propto \mathrm{P}(\mathcal{D} \mid \boldsymbol{\theta})\mathrm{P}(\boldsymbol{\theta}) \\ \text{i.e., } \mathsf{posterior} &\propto \mathsf{likelihood} \cdot \mathsf{prior} \end{aligned} \tag{2.18}$$

Writing the posterior in this proportionality formula allows us to avoid evaluating the evidence $\mathrm{P}(\mathcal{D})$, but we still can analyze the posterior, e.g., by using Gibbs sampling algorithm (see Section 2.3.4).

2.3.2 Dirichlet-Multinomials

We give an introduction on necessary probability distributions used in our new topic models. See Walck (2007) for more details of these distributions and other important ones.

[7] For consistency, we then use the integral as a generalization for both continuous and discrete random variable spaces. In fact, the evidence is computed for the discrete case as: $\mathrm{P}(\mathcal{D}) = \sum_{\boldsymbol{\theta}'} \mathrm{P}(\mathcal{D} \mid \boldsymbol{\theta}')\mathrm{P}(\boldsymbol{\theta}')$.

Categorical and multinomial distributions

A *categorical distribution*—which is also referred to as a *discrete distribution*—has a finite set of m outcomes.[8] For convenience, we use the set of m integer outcomes: $1, 2, ..., m$. A categorical distribution thus is parameterized by a vector $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_m)$ where $\theta_j \in [0, 1]$ represents the probability of seeing j for each $j = 1, 2, ..., m$, and so $\sum_{j=1}^{m} \theta_j = 1$. That is, for a random variable X which is generated conditional on the parameters $\boldsymbol{\theta}$ with the categorical distribution, we denote this as:

$$X \mid \boldsymbol{\theta} \sim \text{Cat}(\boldsymbol{\theta}) \tag{2.19}$$

and its probability mass function is:

$$\text{P}(X = j \mid \boldsymbol{\theta}) = \text{Cat}(X = j \mid \boldsymbol{\theta}) = \theta_j \tag{2.20}$$

Consider $\mathcal{D} = (X_1, X_2, ..., X_n)$ of n draws from a categorical distribution given its m-dimensional parameter vector $\boldsymbol{\theta}$, i.e., $X_i \mid \boldsymbol{\theta} \sim \text{Cat}(\boldsymbol{\theta})$. Then the likelihood is:

$$\text{P}(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{i=1}^{n} \text{Cat}(X_i \mid \boldsymbol{\theta}) = \prod_{j=1}^{m} \theta_j^{N_j} \tag{2.21}$$

where N_j is the number of times j occurs in $\mathcal{D}$, and so $n = \sum_{j=1}^{m} N_j$. It can be conceived of as sampling n words from a vocabulary W of size $m = |W|$, and so N_j is the number of times the word type with vocabulary index j is sampled while θ_j is the probability that it occurs as a word in a document.

Let $\mathbf{N} = (N_1, N_2, ..., N_m)$ be a vector of frequencies. If $\mathbf{N}$ is said to be generated according to a *multinomial distribution* given the parameters $\boldsymbol{\theta}$ and n, then we write as:

$$\mathbf{N} \mid n, \boldsymbol{\theta} \sim \text{Multi}(\boldsymbol{\theta}, n) \tag{2.22}$$

and the probability mass function of $\mathbf{N}$ is:

[8] The categorical distribution reduces to the *Bernoulli distribution* when $m = 2$, i.e., the categorical distribution with only two outcomes: 0 and 1.

$$\mathrm{P}(\mathbf{N} \mid n, \boldsymbol{\theta}) = \mathrm{Multi}(\mathbf{N} \mid \boldsymbol{\theta}, n) = \frac{n!}{\prod_{j=1}^{m} N_j!} \prod_{j=1}^{m} \theta_j^{N_j} \tag{2.23}$$

The categorical distribution thus is equivalent to the multinomial distribution where $n = 1$.

Dirichlet distribution

Let $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_m)$ be a m-dimensional parameter vector, where $\alpha_j > 0$ for $j = 1, 2, ..., m$. A random vector $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_m)$, where $\theta_j \in [0, 1]$ for each $j = 1, 2, ..., m$ and $\sum_{j=1}^{m} \theta_j = 1$, to be distributed according to a *Dirichlet distribution* parameterized by the vector $\boldsymbol{\alpha}$ is denoted by:

$$\boldsymbol{\theta} \mid \boldsymbol{\alpha} \sim \mathrm{Dir}(\boldsymbol{\alpha}) \tag{2.24}$$

with its probability density function is:

$$\mathrm{P}(\boldsymbol{\theta} \mid \boldsymbol{\alpha}) = \mathrm{Dir}(\boldsymbol{\theta} \mid \boldsymbol{\alpha}) = \frac{1}{\Delta(\boldsymbol{\alpha})} \prod_{j=1}^{m} \theta_j^{\alpha_j - 1} \tag{2.25}$$

where the Dirichlet delta function $\Delta(\boldsymbol{\alpha})$ used to normalize the Dirichlet is defined as:

$$\Delta(\boldsymbol{\alpha}) = \int \prod_{j=1}^{m} \theta_j^{\alpha_j - 1} \, \mathrm{d}\boldsymbol{\theta} = \frac{\prod_{j=1}^{m} \Gamma(\alpha_j)}{\Gamma(\sum_{j=1}^{m} \alpha_j)} \tag{2.26}$$

in which Γ is the Gamma function with properties $\Gamma(k) = (k-1)!$ for any positive integer k and $\Gamma(x + 1) = (x)\Gamma(x)$ for any positive real number x.

And the expectation of the Dirichlet parameterized by $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_m)$ is:

$$\mathbb{E}[\boldsymbol{\theta}] = \left(\frac{\alpha_1}{\sum_{j=1}^{m} \alpha_j}, \frac{\alpha_2}{\sum_{j=1}^{m} \alpha_j}, ..., \frac{\alpha_m}{\sum_{j=1}^{m} \alpha_j} \right), \text{ or simply} \tag{2.27}$$

$$\mathbb{E}[\theta_j] = \frac{\alpha_j}{\sum_{k=1}^{m} \alpha_k} \tag{2.28}$$

In topic modeling, a *symmetric* Dirichlet distribution is commonly used, where all elements of $\boldsymbol{\alpha}$ have the same value, i.e., $\alpha_j = \alpha$ for $j = 1, 2, ..., m$. In this case, we have:

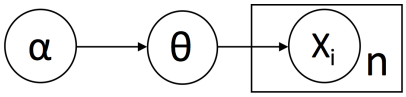


Figure 2.1: Graphical representation of the simple Dirichlet-multinomial mixture model

$$\mathrm{Dir}(\boldsymbol{\theta} \mid \boldsymbol{\alpha}) = \frac{1}{\Delta(\boldsymbol{\alpha})} \prod_{j=1}^{m} \theta_j^{\alpha-1} \tag{2.29}$$

$$\Delta(\boldsymbol{\alpha}) = \frac{\Gamma(\alpha)^m}{\Gamma(m\alpha)} \tag{2.30}$$

2.3.3 A simple Dirichlet-multinomial model

Now let consider to a simple Dirichlet-multinomial mixture model as follows:

$$\begin{aligned} \boldsymbol{\theta} &\mid \boldsymbol{\alpha} \sim \mathrm{Dir}(\boldsymbol{\alpha}) \\ X_i &\mid \boldsymbol{\theta} \sim \mathrm{Cat}(\boldsymbol{\theta}) \ , \quad \text{for } i = 1, 2, ..., n \end{aligned}$$

The generative process can be imagined as follows: we first generate a vector $\boldsymbol{\theta}$ of probabilities—that word types in a vocabulary would appear as a word in a document—from a Dirichlet distribution parameterized by $\boldsymbol{\alpha}$, then we sample n words from the vocabulary using a categorical distribution parameterized by the probability vector $\boldsymbol{\theta}$. We can represent the generative process as a Bayes net using plates, which indicate replication as in Figure 2.1. With the observed data $\mathcal{D} = (X_1, X_2, ..., X_n)$, we can *integrate out* (or *collapse*) $\boldsymbol{\theta}$ to directly compute the evidence $\mathrm{P}(\mathcal{D} \mid \boldsymbol{\alpha})$:

$$\begin{aligned} \mathrm{P}(\mathcal{D} \mid \boldsymbol{\alpha}) &= \int \mathrm{P}(\mathcal{D}, \boldsymbol{\theta} \mid \boldsymbol{\alpha}) \, \mathrm{d}\boldsymbol{\theta} = \int \mathrm{P}(\mathcal{D} \mid \boldsymbol{\theta}) \, \mathrm{P}(\boldsymbol{\theta} \mid \boldsymbol{\alpha}) \, \mathrm{d}\boldsymbol{\theta} \\ &= \int \Big(\prod_{j=1}^{m} \theta_j^{N_j} \Big) \Big(\frac{1}{\Delta(\boldsymbol{\alpha})} \prod_{j=1}^{m} \theta_j^{\alpha_j - 1} \Big) \mathrm{d}\boldsymbol{\theta} \\ &= \frac{1}{\Delta(\boldsymbol{\alpha})} \int \prod_{j=1}^{m} \theta_j^{N_j + \alpha_j - 1} \, \mathrm{d}\boldsymbol{\theta} \\ &= \frac{\Delta(\mathbf{N} + \boldsymbol{\alpha})}{\Delta(\boldsymbol{\alpha})} \end{aligned} \tag{2.31}$$

Then the posterior distribution is computed using Bayes' rule:

$$\begin{aligned}
\mathrm{P}(\boldsymbol{\theta} \mid \mathcal{D}, \boldsymbol{\alpha}) &= \frac{\mathrm{P}(\mathcal{D} \mid \boldsymbol{\theta})\, \mathrm{P}(\boldsymbol{\theta} \mid \boldsymbol{\alpha})}{\mathrm{P}(\mathcal{D} \mid \boldsymbol{\alpha})} \\
&= \frac{\Big(\prod_{j=1}^{m} \theta_j^{N_j}\Big)\Big(\frac{1}{\Delta(\boldsymbol{\alpha})} \prod_{j=1}^{m} \theta_j^{\alpha_j - 1}\Big)}{\frac{\Delta(\mathbf{N} + \boldsymbol{\alpha})}{\Delta(\boldsymbol{\alpha})}} \\
&= \frac{1}{\Delta(\mathbf{N} + \boldsymbol{\alpha})} \prod_{j=1}^{m} \theta_j^{N_j + \alpha_j - 1} \\
&= \mathrm{Dir}(\boldsymbol{\theta} \mid \mathbf{N} + \boldsymbol{\alpha}) \qquad (2.32)
\end{aligned}$$

When using the Dirichlet distribution as a conjugate prior for the parameters of the categorical (or multinomial) distribution, we find that the posterior and prior belong to the same family of Dirichlet distributions. This Dirichlet-multinomial conjugacy thus is the key to computing the posteriors in Dirichlet-multinomial mixture models like probabilistic topic models.

2.3.4 Inference via Gibbs sampling

For such a simple model as presented in Section 2.3.3, we can exactly compute the marginal data likelihood (i.e., the evidence $\mathrm{P}(\mathcal{D})$), resulting in an exact posterior inference $\mathrm{P}(\boldsymbol{\theta} \mid \mathcal{D})$ over the model parameter vector $\boldsymbol{\theta}$. However, for more complicated models such as the Latent Dirichlet Allocation model (Blei et al., 2003) as presented in Section 2.4.1, it is intractable to derive the marginal likelihood, so we cannot perform an exact inference. The solution is to use algorithms for approximate Bayesian inference, such as the expectation-maximization algorithm (Dempster et al., 1977), variational Bayes methods (Blei et al., 2003; Bishop, 2006) and sampling algorithms (Griffiths and Steyvers, 2004; Robert and Casella, 2004). We present in this section a brief introduction of the Gibbs sampling algorithm only, which is used later for inference in our new topic models. See MacKay (2003) for an overview of the other approximate algorithms.

Gibbs sampling is a particular Markov chain Monte Carlo algorithm (MCMC) (Robert

Algorithm 3: Gibbs sampling algorithm

Require: Choose an initial value for the m-dimensional parameter vector $\boldsymbol{\theta}$

for *iteration i = 1, 2, ...* **do**

  for *parameter index j = 1, 2, ..., m* **do**

    $\theta_j \coloneqq \theta_j \sim \mathrm{P}(\theta_j \mid \boldsymbol{\theta}_{\neg j})$ `// j could also be chosen randomly`

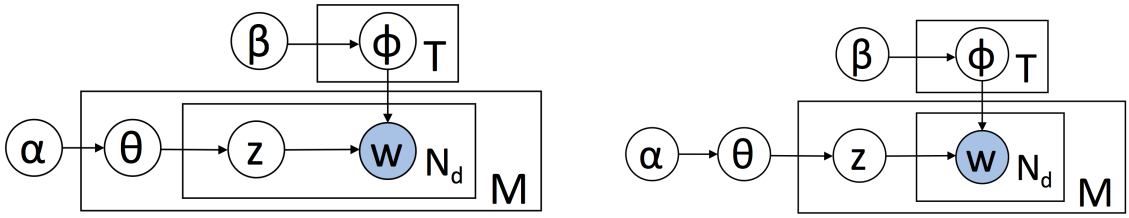
and Casella, 2004). The general idea of a MCMC method is to use a Markov chain to produce samples $\boldsymbol{\theta}$, where each state of the chain is a possible value of $\boldsymbol{\theta}$. After a stationary state has been reach (i.e., discarding first *burn-in* samples) for which the sampling distribution converges to the desired posterior distribution, the chain will start to sample $\boldsymbol{\theta}$ from the true posterior $\mathrm{P}(\boldsymbol{\theta} \mid \mathcal{D})$. The Gibbs sampling is a widely-used MCMC method where each parameter element θ_j is sampled conditional on all other elements, which we denote $\boldsymbol{\theta}_{\neg j}$. As shown in Algorithm 3, each *Gibbs sweep* or iteration produces a new configuration of $\boldsymbol{\theta}$, where the values of all parameter vector elements are resampled, i.e., replacing the current value θ_j with a sample from $\mathrm{P}(\theta_j \mid \boldsymbol{\theta}_{\neg j})$. Here the Gibbs sampling is useful when it is easy to sample from the conditional probabilities $\mathrm{P}(\theta_j \mid \boldsymbol{\theta}_{\neg j})$, i.e., we know how to compute these probabilities.

2.4 Probabilistic topic models

This section briefly overviews probabilistic topic models. Sections 2.4.1 and 2.4.3 present baseline topic models that we extend to integrate pre-trained word vectors learned from a large external corpus. Section 2.4.2 presents other advanced topic models while Section 2.4.4 discusses topic models that also incorporate the word vectors. The material in Section 2.4 is based on Blei (2012) and Nguyen et al. (2015a).

2.4.1 Latent Dirichlet Allocation

The most well-known topic model Latent Dirichlet Allocation (LDA) (Blei et al., 2003) represents each document d in the document collection D as a probability distribution $\boldsymbol{\theta}_d$



(LDA) (DMM)

Figure 2.2: Graphical representations of Latent Dirichlet Allocation (LDA) and Dirichlet Multinomial Mixture (DMM). These figures are drawn based on Nguyen et al. (2015a).

over topics, where each topic z is modeled by a probability distribution $\boldsymbol{\phi}_z$ over words in a fixed vocabulary W. As presented in Figure 2.2, where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are hyper-parameters and T is the number of topics, the generative process for LDA is described as follows:

$$\begin{array}{rcccll} \boldsymbol{\phi}_z & | & \boldsymbol{\beta} & \sim & \text{Dir}(\boldsymbol{\beta}) & z = 1, 2, ..., T \\ \boldsymbol{\theta}_d & | & \boldsymbol{\alpha} & \sim & \text{Dir}(\boldsymbol{\alpha}) & d = 1, 2, ..., M \text{ where } M = |D| \\ z_{d_i} & | & \boldsymbol{\theta}_d & \sim & \text{Cat}(\boldsymbol{\theta}_d) & d = 1, 2, ..., M \ ; \ i = 1, 2, ..., N_d \\ w_{d_i} & | & z_{d_i} \ , \ \boldsymbol{\phi} & \sim & \text{Cat}(\boldsymbol{\phi}_{z_{d_i}}) & d = 1, 2, ..., M \ ; \ i = 1, 2, ..., N_d \end{array}$$

where Dir and Cat stand for a Dirichlet distribution and a categorical distribution, N_d is the number of word tokens in document d, and z_{d_i} is the topic indicator for the i^{th} word w_{d_i} in document d. Here, the topic-to-word Dirichlet multinomial component generates the word w_{d_i} by drawing it from the categorical distribution $\text{Cat}(\boldsymbol{\phi}_{z_{d_i}})$ for topic z_{d_i}.

Griffiths and Steyvers (2004) described a collapsed Gibbs sampling algorithm for estimating the LDA topic model. By integrating out $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, the algorithm samples the topic z_{d_i} for the current i^{th} word w_{d_i} in document d using the conditional distribution $\text{P}(z_{d_i} \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d_i})$, where $\boldsymbol{w}$ denotes the word observations for the whole document collection D and $\boldsymbol{z}_{\neg d_i}$ denotes the topic assignments of all the other words in D, so:

$$\begin{aligned} \text{P}(z_{d_i} = t \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d_i}) &= \frac{\text{P}(\boldsymbol{w}, \boldsymbol{z})}{\text{P}(\boldsymbol{w}, \boldsymbol{z}_{\neg d_i})} \\ &\propto \frac{\text{P}(\boldsymbol{w}, \boldsymbol{z} \mid \boldsymbol{\alpha}, \boldsymbol{\beta})}{\text{P}(\boldsymbol{w}_{\neg d_i}, \boldsymbol{z}_{\neg d_i} \mid \boldsymbol{\alpha}, \boldsymbol{\beta})} \\ &\propto (N^t_{d_{\neg i}} + \alpha) \frac{N^{t,w_{d_i}}_{\neg d_i} + \beta}{N^t_{\neg d_i} + V\beta} \end{aligned} \tag{2.33}$$

Notation: $N_d^{t,w}$ is the rank-3 tensor that counts the number of times that word w is generated from topic t in document d. When an index is omitted, it indicates summation over that index (so N_d is the number of words in document d). We write the subscript $\neg d$ for the document collection D with document d removed, and the subscript $\neg d_i$ for D with just the i^{th} word in document d removed, while the subscript $d_{\neg i}$ represents document d without its i^{th} word. For example, $N_{\neg d_i}^t$ is the number of words labeled with topic t, ignoring the i^{th} word of document d. In addition, V is the number of word types: $V = |W|$, and M is the number of documents in the collection, i.e., $M = |D|$.

2.4.2 Advanced topic models

Following LDA, various topic models have been explored such as supervised topic models (Blei and McAuliffe, 2008; Perotte et al., 2011), interactive topic models (Hu et al., 2011), collaborative topic models (Wang and Blei, 2011), conditional topic random field models (Zhu and Xing, 2010), multilingual topic models (Mimno et al., 2009; Boyd-Graber and Blei, 2009; Zhang et al., 2010), partially labeled Dirichlet allocation models (Ramage et al., 2011) and structured topic models (Du et al., 2013).

Although the majority of topic models assume that documents belong to a single corpus, Wang et al. (2009) described Markov topic models to learn topics simultaneously from multiple corpora. Because the number of topics T is pre-fixed in LDA, Teh et al. (2006a) presented hierarchical Dirichlet process mixture models to automatically identify T during posterior inference from the training corpus. Whereas in LDA, word order is not taken into account, Wallach (2006), Johnson (2010) and Zhao et al. (2015b) developed topical collocation models in which the word generation process depends not only on the chosen topic, but also short sequences of adjacent words. Also, the LDA model does not take the temporal ordering of the documents into account, thus the dynamic topic models (Blei and Lafferty, 2006) and the continuous time dynamic topic models (Wang et al., 2008) are introduced to track how the topics change over time. In addition, LDA does not capture the correlations between topics, so the correlated topic model (Blei and

Lafferty, 2007) and the Pachinko allocation machine (Li and McCallum, 2006) have been proposed to discover the connections between topics. Furthermore, Chang and Blei (2010) constructed a relational topic model to incorporate useful information about the content of the documents with the connections between them. Blei (2012) reviews other probabilistic topic models and LDA-like models which have been adapted to other kinds of data.

2.4.3 Dirichlet Multinomial Mixture for short texts

As shown in Tang et al. (2014), LDA obtains poor performance when the data presents extreme properties (e.g., very short or very few documents). That is, applying topic models to short documents, such as Tweets or instant messages, is challenging because of data sparsity and the limited contexts in such texts. One approach is to assume that there is only one topic per document (Nigam et al., 2000; Zhao et al., 2011; Yin and Wang, 2014; Surian et al., 2016). Another approach is to combine short texts into long pseudo-documents before training LDA (Hong and Davison, 2010; Weng et al., 2010; Mehrotra et al., 2013; Bicalho et al., 2017). In addition, Chen et al. (2015), Quan et al. (2015) and Zuo et al. (2016) presented topic models which implicitly aggregate short texts into pseudo-documents based on their topic assignments instead of auxiliary information as in Jin et al. (2011). Other LDA-extended models proposed for short text topic modeling can be found in Vosecky et al. (2013), Yan et al. (2013), Lin et al. (2014) and Li et al. (2016b).

In the Dirichlet Multinomial Mixture (DMM) model (Nigam et al., 2000), each document is assumed to only have one topic. The process of generating a document d in the collection D, as shown in Figure 2.2, is to first select a topic assignment for the document, and then the topic-to-word Dirichlet multinomial component generates all the words in the document from the same selected topic:

$$
\begin{aligned}
\boldsymbol{\phi}_z \;&|\; \boldsymbol{\beta} \;&&\sim \mathrm{Dir}(\boldsymbol{\beta}) && z = 1, 2, ..., T \\
\boldsymbol{\theta} \;&|\; \boldsymbol{\alpha} \;&&\sim \mathrm{Dir}(\boldsymbol{\alpha}) && \\
z_d \;&|\; \boldsymbol{\theta} \;&&\sim \mathrm{Cat}(\boldsymbol{\theta}) && d = 1, 2, ..., M \\
w_{d_i} \;&|\; z_d\ ,\ \boldsymbol{\phi} \;&&\sim \mathrm{Cat}(\boldsymbol{\phi}_{z_d}) && d = 1, 2, ..., M\ ;\ i = 1, 2, ..., N_d
\end{aligned}
$$

Yin and Wang (2014) introduced a collapsed Gibbs sampling algorithm for the DMM model in which a topic z_d is sampled for the document d using the conditional probability $\mathrm{P}(z_d \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d})$, where $\boldsymbol{z}_{\neg d}$ denotes the topic assignments of all the other documents, so:

$$\begin{aligned}
\mathrm{P}(z_d = t \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d}) &= \frac{\mathrm{P}(\boldsymbol{w}, \boldsymbol{z})}{\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}_{\neg d})} \\
&\propto \frac{\mathrm{P}(\boldsymbol{w}, \boldsymbol{z} \mid \boldsymbol{\alpha}, \boldsymbol{\beta})}{\mathrm{P}(\boldsymbol{w}_{\neg d}, \boldsymbol{z}_{\neg d} \mid \boldsymbol{\alpha}, \boldsymbol{\beta})} \\
&\propto (M^t_{\neg d} + \alpha) \frac{\Gamma(N^t_{\neg d} + V\beta)}{\Gamma(N^t_{\neg d} + N_d + V\beta)} \prod_{w=1}^{V} \frac{\Gamma(N^{t,w}_{\neg d} + N^w_d + \beta)}{\Gamma(N^{t,w}_{\neg d} + \beta)} \quad (2.34)
\end{aligned}$$

Notation: Tensor N and negation $\neg$ are defined similar to those used in the LDA model (see Section 2.4.1). $M^t_{\neg d}$ is the number of documents assigned to topic t excluding the current document d. In addition, Γ is the Gamma function.

2.4.4 Topic models with word representations

To the best of our knowledge, our novel latent feature topic models LF-LDA and LF-DMM (Nguyen et al., 2015a)—which are presented in Chapter 3—were among the first models to integrate vector representations of words into Dirichlet multinomial topic models. Later, Nguyen et al. (2015b) presented an extension of LDA with a MAP estimation approach to improve topic coherence with word representations. Simultaneously, Das et al. (2015) proposed the Gaussian-LDA model which also integrates pre-trained word vectors into the LDA model. The difference between the LF-LDA model and the Gaussian-LDA model is that Gaussian-LDA defines a latent feature model based on a multivariate Gaussian distribution with the topic vector as the mean vector, while LF-LDA uses a log-linear model. The probability that it generates a word given a topic in Gaussian-LDA is defined by the Euclidean distance between the word vector and topic vector, while in LF-LDA it uses the dot product of the word vector and topic vector. On document clustering and classification evaluation tasks, Li et al. (2016c) and Hu and Tsujii (2016) later showed that LF-LDA produces significantly higher results than Gaussian-LDA.

Li et al. (2016c), Fu et al. (2016) and Jiang et al. (2016) proposed new topic models where a latent feature topic-word distribution defines the probability of a word given a topic based on the vector representations of the word, its context words and the topic. Hu and Tsujii (2016) introduced a latent concept topic model with word vectors, in which a new latent variable is introduced to capture the conceptual similarity of words and is modeled as a multivariate Gaussian distribution over the word vector space. Rather than modeling each topic by a categorical distribution or Gaussian distribution over the word vector space as in previous models, Li et al. (2016d) and Batmanghelich et al. (2016) used the von Mises-Fisher distributions (Banerjee et al., 2005) on the word vector space to represent topics. In addition, Li et al. (2016a) extended the DMM model by incorporating auxiliary word embeddings through the generalized Pólya urn model (Kotz et al., 2000) in the inference process.

Latent topics have also been used to learn word vectors. Recent research has extended the Word2Vec Skip-gram model (see Section 2.1.1) to integrate pre-trained LDA-based latent topics of words for learning word representations (Liu et al., 2015c; Cheng et al., 2015; Liu et al., 2015b; Zhang and Zhong, 2016; Nguyen et al., 2017). More specifically, Liu et al. (2015c) used each topic as a target to predict context words. Cheng et al. (2015) extended Liu et al. (2015c)'s model to treat topics as pseudo-words for predicting contexts of both words and topics. Zhang and Zhong (2016) used pairs consisting of a word and its corresponding topic to predict other word and topic pairs. Nguyen et al. (2017) proposed a mixture model to jointly learning vector representations of words and topics. In addition, Liu et al. (2015b) applied a bilinear tensor operator to model the interaction between vector representations of words and their latent topics.

2.5 Embedding models for KB completion

This section serves as a brief overview of embedding models for knowledge base (KB) completion, which represent entities and/or relations with dense feature vectors or matrices. The material in Section 2.5 is based on Nguyen (2017).

2.5.1 A general approach

Let $\mathcal{E}$ denote the set of entities and $\mathcal{R}$ the set of relation types. Denote by $\mathcal{G}$ the knowledge base consisting of a set of correct triples (h, r, t), such that $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$. For each triple (h, r, t), the embedding models define a *score function* $f(h, r, t)$ of its implausibility. Their goal is to choose f such that the score $f(h, r, t)$ of a plausible triple (h, r, t) is smaller than the score $f(h', r', t')$ of an implausible triple (h', r', t').

Table 2.1 summarizes different score functions $f(h, r, t)$ and the optimization algorithms used to estimate model parameters. To learn model parameters (i.e., entity vectors, relation vectors or matrices), the embedding models minimize an objective function. A common objective function is the following margin-based function (Lin, 2004):

$$\mathcal{L} = \sum_{\substack{(h,r,t)\in\mathcal{G} \\ (h',r,t')\in\mathcal{G}'_{(h,r,t)}}} \max\Big(0, \gamma + f(h, r, t) - f(h', r, t')\Big) \tag{2.35}$$

where $\mathcal{G}'_{(h,r,t)}$ is the set of incorrect triples generated by corrupting the correct triple (h, r, t).

2.5.2 Specific models

The Unstructured model (Bordes et al., 2012) assumes that the head and tail entity vectors are similar. As the Unstructured model does not take the relationship into account, it cannot distinguish different relation types. The Structured Embedding (SE) model (Bordes et al., 2011) assumes that the head and tail entities are similar only in a relation-dependent subspace, where each relation is represented by two different matrices. Furthermore, the SME model (Bordes et al., 2012) uses four different matrices to project entity and relation vectors into a subspace. The TransE model (Bordes et al., 2013) is inspired by models such as the Word2Vec Skip-gram model (Mikolov et al., 2013b) where relationships between words often correspond to translations in latent feature space.

The TransH model (Wang et al., 2014a) associates each relation with a relation-specific hyperplane and uses a projection vector to project entity vectors onto that hyperplane. TransD (Ji et al., 2015) and TransR/CTransR (Lin et al., 2015b) extend the TransH model

Model	Score function $f(h,r,t)$	**Opt.**
Unstructured	$\|\boldsymbol{v}_h - \boldsymbol{v}_t\|_{\ell_{1/2}}$	SGD
SE	$\|\mathbf{W}_{r,1}\boldsymbol{v}_h - \mathbf{W}_{r,2}\boldsymbol{v}_t\|_{\ell_{1/2}}$; $\mathbf{W}_{r,1}$, $\mathbf{W}_{r,2} \in \mathbb{R}^{k\times k}$	SGD
SME	$(\mathbf{W}_{1,1}\boldsymbol{v}_h + \mathbf{W}_{1,2}\boldsymbol{v}_r + \mathbf{b}_1)^\top(\mathbf{W}_{2,1}\boldsymbol{v}_t + \mathbf{W}_{2,2}\boldsymbol{v}_r + \mathbf{b}_2)$ $\mathbf{b}_1$, $\mathbf{b}_2 \in \mathbb{R}^n$; $\mathbf{W}_{1,1}$, $\mathbf{W}_{1,2}$, $\mathbf{W}_{2,1}$, $\mathbf{W}_{2,2} \in \mathbb{R}^{n\times k}$	SGD
TransE	$\|\boldsymbol{v}_h + \boldsymbol{v}_r - \boldsymbol{v}_t\|_{\ell_{1/2}}$; $\boldsymbol{v}_r \in \mathbb{R}^k$	SGD
TransH	$\|(\mathbf{I} - \boldsymbol{r}_p\boldsymbol{r}_p^\top)\boldsymbol{v}_h + \boldsymbol{v}_r - (\mathbf{I} - \boldsymbol{r}_p\boldsymbol{r}_p^\top)\boldsymbol{v}_t\|_{\ell_{1/2}}$ $\boldsymbol{r}_p$, $\boldsymbol{v}_r \in \mathbb{R}^k$; $\mathbf{I}$: Identity matrix size $k \times k$	SGD
TransR	$\|\mathbf{W}_r\boldsymbol{v}_h + \boldsymbol{v}_r - \mathbf{W}_r\boldsymbol{v}_t\|_{\ell_{1/2}}$; $\mathbf{W}_r \in \mathbb{R}^{n\times k}$; $\boldsymbol{v}_r \in \mathbb{R}^n$	SGD
TransD	$\|(\mathbf{I} + \boldsymbol{r}_p\boldsymbol{h}_p^\top)\boldsymbol{v}_h + \boldsymbol{v}_r - (\mathbf{I} + \boldsymbol{r}_p\boldsymbol{t}_p^\top)\boldsymbol{v}_t\|_{\ell_{1/2}}$ $\boldsymbol{r}_p$, $\boldsymbol{v}_r \in \mathbb{R}^n$; $\boldsymbol{h}_p, \boldsymbol{t}_p \in \mathbb{R}^k$; $\mathbf{I}$: Identity matrix size $n \times k$	AdaDelta
lppTransD	$\|(\mathbf{I} + \boldsymbol{r}_{p,1}\boldsymbol{h}_p^\top)\boldsymbol{v}_h + \boldsymbol{v}_r - (\mathbf{I} + \boldsymbol{r}_{p,2}\boldsymbol{t}_p^\top)\boldsymbol{v}_t\|_{\ell_{1/2}}$ $\boldsymbol{r}_{p,1}$, $\boldsymbol{r}_{p,2}$, $\boldsymbol{v}_r \in \mathbb{R}^n$; $\boldsymbol{h}_p, \boldsymbol{t}_p \in \mathbb{R}^k$; $\mathbf{I}$: Identity matrix size $n \times k$	SGD
TranSparse	$\|\mathbf{W}_r^h(\theta_r^h)\boldsymbol{v}_h + \boldsymbol{v}_r - \mathbf{W}_r^t(\theta_r^t)\boldsymbol{v}_t\|_{\ell_{1/2}}$; $\mathbf{W}_r^h$, $\mathbf{W}_r^t \in \mathbb{R}^{n\times k}$; θ_r^h, $\theta_r^t \in \mathbb{R}$; $\boldsymbol{v}_r \in \mathbb{R}^n$	SGD
[Our] STransE	$\|\mathbf{W}_{r,1}\boldsymbol{v}_h + \boldsymbol{v}_r - \mathbf{W}_{r,2}\boldsymbol{v}_t\|_{\ell_{1/2}}$; $\mathbf{W}_{r,1}$, $\mathbf{W}_{r,2} \in \mathbb{R}^{k\times k}$; $\boldsymbol{v}_r \in \mathbb{R}^k$	SGD
DISTMULT	$\boldsymbol{v}_h^\top\mathbf{W}_r\boldsymbol{v}_t$; $\mathbf{W}_r$ is a diagonal matrix $\in \mathbb{R}^{k\times k}$	AdaGrad
NTN	$\boldsymbol{v}_r^\top tanh(\boldsymbol{v}_h^\top\mathbf{M}_r\boldsymbol{v}_t + \mathbf{W}_{r,1}\boldsymbol{v}_h + \mathbf{W}_{r,2}\boldsymbol{v}_t + \mathbf{b}_r)$ $\boldsymbol{v}_r$, $\mathbf{b}_r \in \mathbb{R}^n$; $\mathbf{M}_r \in \mathbb{R}^{k\times k\times n}$; $\mathbf{W}_{r,1}$, $\mathbf{W}_{r,2} \in \mathbb{R}^{n\times k}$	L-BFGS
HolE	$sigmoid(\boldsymbol{v}_r^\top(\boldsymbol{v}_h \circ \boldsymbol{v}_t))$; $\boldsymbol{v}_r \in \mathbb{R}^k$, $\circ$ denotes *circular correlation*	AdaGrad
Bilinear-COMP	$\boldsymbol{v}_h^\top\mathbf{W}_{r_1}\mathbf{W}_{r_2}...\mathbf{W}_{r_m}\boldsymbol{v}_t$; $\mathbf{W}_{r_1}, \mathbf{W}_{r_2}, ..., \mathbf{W}_{r_m} \in \mathbb{R}^{k\times k}$	AdaGrad
TransE-COMP	$\|\boldsymbol{v}_h + \boldsymbol{v}_{r_1} + \boldsymbol{v}_{r_2} + ... + \boldsymbol{v}_{r_m} - \boldsymbol{v}_t\|_{\ell_{1/2}}$; $\boldsymbol{v}_{r_1}, \boldsymbol{v}_{r_2}, ..., \boldsymbol{v}_{r_m} \in \mathbb{R}^k$	AdaGrad

Table 2.1: The score functions $f(h,r,t)$ and the optimization methods (Opt.) of several prominent embedding models for KB completion. In all of these models, the entities h and t are represented by vectors $\boldsymbol{v}_h$ and $\boldsymbol{v}_t \in \mathbb{R}^k$, respectively.

by using two projection vectors and a matrix to project entity vectors into a relation-specific space, respectively. Similar to TransR, TransR-FT (Feng et al., 2016a) also uses a matrix to project head and tail entity vectors. TEKE_H (Wang and Li, 2016) extends TransH to incorporate rich context information in an external text corpus. lppTransD (Yoon et al., 2016) extends TransD to additionally use two projection vectors for representing each relation. Our STransE model (Nguyen et al., 2016b)—which is detailed in Chapter 4—and the TranSparse model (Ji et al., 2016) can be viewed as direct extensions of the TransR model, where head and tail entities are associated with their own projection matrices. Unlike STransE, the TranSparse model uses adaptive sparse matrices, whose sparse degrees are defined based on the number of entities linked by relations.

DISTMULT (Yang et al., 2015) is based on the Bilinear model (Nickel et al., 2011; Bordes et al., 2012; Jenatton et al., 2012) where each relation is represented by a diagonal rather than a full matrix. The neural tensor network (NTN) model (Socher et al., 2013b) uses a bilinear tensor operator to represent each relation while ER-MLP (Dong et al., 2014) and ProjE (Shi and Weninger, 2017) can be viewed as simplified versions of NTN. Such quadratic forms are also used to model entities and relations in KG2E (He et al., 2015), TransG (Xiao et al., 2016), ComplEx (Trouillon et al., 2016), TATEC (García-Durán et al., 2016) and RSTE (Tay et al., 2017). In addition, HolE (Nickel et al., 2016b) uses circular correlation—a compositional operator—which can be interpreted as a compression of the tensor product.

Recent research has shown that relation paths between entities in KBs provide richer context information and improve the performance of embedding models for KB completion (Luo et al., 2015; Lin et al., 2015a; Liang and Forbus, 2015; García-Durán et al., 2015; Guu et al., 2015; Toutanova et al., 2016; Nguyen et al., 2016a). Luo et al. (2015) constructed relation paths between entities and, viewing entities and relations in the path as pseudo-words, then applied Word2Vec algorithms (Mikolov et al., 2013b) to produce pre-trained vectors for these pseudo-words. Luo et al. (2015) showed that using these pre-trained vectors for initialization helps to improve the performance of models TransE (Bordes et al., 2013), SME (Bordes et al., 2012) and SE (Bordes et al., 2011). In addition, Liang and Forbus (2015) used the implausibility score produced by SME to compute the weights of relation paths. Furthermore, RTransE (García-Durán et al., 2015), PTransE (Lin et al., 2015a) and TransE-COMP (Guu et al., 2015) are extensions of the TransE model. These models similarly represent a relation path by a vector which is the sum of the vectors of all relations in the path, whereas in the Bilinear-COMP model (Guu et al., 2015) and the PRUNED-PATHS model (Toutanova et al., 2016), each relation is a matrix and so it represents the relation path by matrix multiplication. Our neighborhood mixture TransE-NMM model (Nguyen et al., 2016a)—which is detailed in Chapter 5—can be also viewed as a three-relation path model as it takes into account the neighborhood entity and relation information of both head and tail entities in each triple. The neighborhood information is

also exploited in graph convolutional networks (Schlichtkrull et al., 2017). If we consider to entity types as local neighborhoods, then our TransE-NMM and the graph convolutional networks could also be used for entity type prediction (Yao et al., 2013). Note that our new models STransE (Nguyen et al., 2016b) and TransE-NMM (Nguyen et al., 2016a) were published in June 2016 and August 2016, respectively. So, all other models published in 2016 were developed simultaneously with (or, after) STransE and TransE-NMM.

A note on experimental datasets: Toutanova and Chen (2015) noted that some experimental datasets for KB completion are easy because they contain many reversible relations. Dettmers et al. (2017) showed a concrete example: A test triple (feline, hyponym, cat) can be mapped to a training triple (cat, hypernym, feline), thus knowing that "hyponym" and "hypernym" are reversible allows us to easily predict the majority of test triples. So, future study should focus on realistic KB completion datasets which represent a more challenging learning setting. See Dettmers et al. (2017) for a detail discussion.

2.5.3 Other KB completion models

The Path Ranking Algorithm (PRA) (Lao and Cohen, 2010) is a random walk inference technique which was proposed to predict a new relationship between two entities in KBs. Lao et al. (2011) used PRA to estimate the probability of an unseen triple as a combination of weighted random walks that follow different paths linking the head entity and tail entity in the KB. Gardner et al. (2014) made use of an external text corpus to increase the connectivity of the KB used as the input to PRA. Gardner and Mitchell (2015) improved PRA by proposing a subgraph feature extraction technique to make the generation of random walks in KBs more efficient and expressive, while Wang et al. (2016) extended PRA to couple the path ranking of multiple relations. PRA can also be used in conjunction with first-order logic in the discriminative Gaifman model (Niepert, 2016). In addition, Neelakantan et al. (2015) used a recurrent neural network to learn vector representations of PRA-style relation paths between entities in the KB. Other random-walk based learning algorithms for KB completion can be also found in Feng et al. (2016b), Liu et al. (2016) and

Wei et al. (2016). See other methods for learning from knowledge bases and multi-relational data in Nickel et al. (2016a).

2.6 Summary

In this chapter, we briefly described technical background used in this thesis. We also presented related work w.r.t. probabilistic topic models and embedding models for KB completion. The technical background and related work serve as a foundation for new models presented in chapters 3 to 5.

Chapter 3

Improving topic models with word representations

Contents

3.1 Introduction 42
3.2 New latent feature topic models 43
3.2.1 Generative process for the LF-LDA model 44
3.2.2 Generative process for the LF-DMM model 45
3.2.3 Inference in the LF-LDA model 45
3.2.4 Inference in the LF-DMM model 50
3.2.5 Learning latent feature vectors for topics 54
3.3 Experiments 54
3.3.1 Experimental setup 55
3.3.2 Topic coherence evaluation 57
3.3.3 Document clustering evaluation 62
3.3.4 Document classification evaluation 67
3.4 Discussion 69
3.5 Summary 70

Probabilistic topic models are widely used to discover latent topics in document collections, while latent feature vector representations of words have been used to obtain high performance in many NLP tasks. In this chapter, we extend two different Dirichlet multinomial topic models by incorporating word vectors trained on very large corpora to improve the word-topic mapping learned on a smaller corpus. Experimental results show that by using information from the external corpora, our new models produce significant improvements on topic coherence, document clustering and document classification tasks,

especially on datasets with few or short documents. The work presented in this chapter has been published in Nguyen et al. (2015a).

3.1 Introduction

Topic modeling algorithms, such as the Latent Dirichlet Allocation model (Blei et al., 2003) and related methods (Blei, 2012), are often used to learn a set of latent topics for a corpus, and predict the probabilities of each word in each document belonging to each topic (Teh et al., 2006b; Newman et al., 2006; Toutanova and Johnson, 2008; Porteous et al., 2008; Johnson, 2010; Xie and Xing, 2013; Hingmire et al., 2013).

Conventional topic modeling algorithms such as these infer document-to-topic and topic-to-word distributions from the co-occurrence of words within documents. However, when the training corpus is small or when the training documents are short such as Tweets, instant messages and forum messages, the resulting distributions might be based on little evidence. Phan et al. (2011) showed that it helps to exploit external knowledge to improve the topic representations. Phan et al. (2011) assumed that a small or short corpus is a sample of topics from a larger corpus like Wikipedia, and then use the topics discovered in the larger corpus to help shape the topic representations in the small or short corpus. However, if the larger corpus has many irrelevant topics, this will "use up" the topic space of the model. Refer to Section 2.4.3 for a brief discussion of topic modeling on short texts. In addition, Petterson et al. (2010) proposed an extension of the Latent Dirichlet Allocation model, using external information about word similarity, such as thesauri and dictionaries, in order to smooth the topic-to-word distribution.

Latent feature word representations have been used for a wide range of NLP tasks such as sentiment analysis, dependency parsing, sequence labeling and machine translation (Glorot et al., 2011; Socher et al., 2013a; Sutskever et al., 2014; Dyer et al., 2015; Ma and Hovy, 2016). The combination of values permitted by latent features forms a high dimensional space which makes it well-suited to model topics of very large corpora. Rather than relying solely on a multinomial or latent feature model, we explore how to take advantage

of both latent feature and multinomial models by using a latent feature representation trained on a large external corpus to supplement a multinomial topic model estimated from a smaller corpus.

We propose two new latent feature topic models which integrate latent feature word representations into two Dirichlet multinomial topic models: a Latent Dirichlet Allocation (LDA) model (Blei et al., 2003) and a one-topic-per-document Dirichlet Multinomial Mixture (DMM) model (Nigam et al., 2000).[1] This is our main contribution. Specifically, we replace the topic-to-word Dirichlet multinomial component which generates the words from topics in each Dirichlet multinomial topic model by a two-component mixture of a Dirichlet multinomial component and a latent feature component. In addition to presenting a sampling procedure for the new models, we also compare the use of two well-known sets of pre-trained latent feature word vectors—Google Word2Vec and Stanford GloVe—with our models. We achieve significant improvements on topic coherence evaluation, document clustering and document classification tasks, especially on corpora of short documents and corpora with few documents.

3.2 New latent feature topic models

We propose two novel probabilistic topic models, which we call the LF-LDA and the LF-DMM (Nguyen et al., 2015a). LF-LDA combines a latent feature model with the LDA model, while LF-DMM combines a latent feature model with the DMM model. Additionally, we also present Gibbs sampling procedures for LF-LDA and LF-DMM.

In general, LF-LDA and LF-DMM are formed by taking the original Dirichlet multinomial topic models LDA and DMM, and replacing their topic-to-word Dirichlet multinomial component that generates words from topics with a two-component mixture of a topic-to-word Dirichlet multinomial component and a latent feature component. Informally, as shown in Figure 3.1, the new models have the structure of the original Dirichlet multinomial topic models with the addition of two matrices $\boldsymbol{\tau}$ and $\boldsymbol{\omega}$ of latent feature weights. Here,

[1] See brief descriptions of the LDA and DMM models in sections 2.4.1 and 2.4.3, respectively.

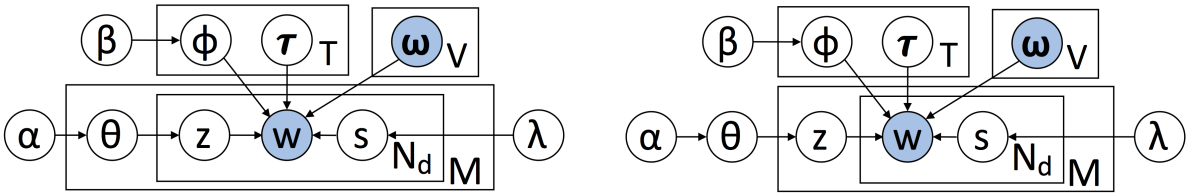


Figure 3.1: Graphical representations of our new latent feature topic models.

$\boldsymbol{\tau}_t$ and $\boldsymbol{\omega}_w$ are the latent-feature vectors associated with topic t and word w, respectively. Our latent feature model defines the probability that it generates a word given the topic as the categorical distribution CatE with:

$$\text{CatE}(w \mid \boldsymbol{\tau}_t \boldsymbol{\omega}^\top) = \frac{\exp(\boldsymbol{\omega}_w \cdot \boldsymbol{\tau}_t)}{\sum_{w' \in W} \exp(\boldsymbol{\omega}_{w'} \cdot \boldsymbol{\tau}_t)} \tag{3.1}$$

i.e., CatE is a categorical distribution with log-space parameters: $\text{CatE}(w \mid \boldsymbol{u}) \propto \exp(u_w)$. As $\boldsymbol{\tau}_t$ and $\boldsymbol{\omega}_w$ are (row) vectors of latent feature weights, so $\boldsymbol{\tau}_t \boldsymbol{\omega}^\top$ is a vector of "scores" indexed by words. In LF-LDA and LF-DMM, we utilize pre-trained word vectors—that are trained on external billion-word corpora—in order to incorporate the rich information from very large datasets. Therefore, the word-vector matrix $\boldsymbol{\omega}$ is fixed.

We explain the generative processes of LF-LDA and LF-DMM in sections 3.2.1 and 3.2.2, and then present our Gibbs sampling procedures for LF-LDA and LF-DMM in sections 3.2.3 and 3.2.4, respectively. Finally, Section 3.2.5 explains how we estimate $\boldsymbol{\tau}$.

3.2.1 Generative process for the LF-LDA model

The LF-LDA model generates a document as follows: a distribution over topics $\boldsymbol{\theta}_d$ is drawn for document d; then, for each i^{th} word w_{d_i} (in the sequential order that words appear in the document), the model chooses a topic indicator z_{d_i}, a binary indicator variable s_{d_i} is sampled from a Bernoulli distribution to determine whether the word w_{d_i} is to be generated by the Dirichlet multinomial or latent feature component, and finally the word is generated from the chosen topic by the determined topic-to-word component. The generative process is:

$$
\begin{aligned}
\boldsymbol{\phi}_z &\mid \boldsymbol{\beta} &&\sim \text{Dir}(\boldsymbol{\beta}) \\
\boldsymbol{\theta}_d &\mid \boldsymbol{\alpha} &&\sim \text{Dir}(\boldsymbol{\alpha}) \\
z_{d_i} &\mid \boldsymbol{\theta}_d &&\sim \text{Cat}(\boldsymbol{\theta}_d) \\
s_{d_i} &\mid \lambda &&\sim \text{Ber}(\lambda) \\
w_{d_i} &\mid z_{d_i}\ ,\ s_{d_i}\ ,\ \boldsymbol{\phi}\ ,\ \boldsymbol{\tau}\ ,\ \boldsymbol{\omega} &&\sim (1 - s_{d_i})\text{Cat}(\boldsymbol{\phi}_{z_{d_i}}) + s_{d_i}\text{CatE}(\boldsymbol{\tau}_{z_{d_i}}\, \boldsymbol{\omega}^\top)
\end{aligned}
$$

where the hyper-parameter λ is the probability of a word being generated by the latent feature topic-to-word component and $\text{Ber}(\lambda)$ is a Bernoulli distribution with success probability λ.

3.2.2 Generative process for the LF-DMM model

The LF-DMM model uses the DMM's assumption that all the words in a document share the same topic. So, the process of generating a document in a document collection with LF-DMM is as follows: a distribution over topics $\boldsymbol{\theta}$ is drawn for the document collection; next, the model draws a topic indicator z_d for the entire document d; then, for every i^{th} word w_{d_i} in the document d, a binary indicator variable s_{d_i} is sampled from a Bernoulli distribution to determine whether the Dirichlet multinomial or latent feature component will be used to generate the word w_{d_i}, and finally the word is generated from the same topic z_d by the determined component. The generative process is summarized as:

$$
\begin{aligned}
\boldsymbol{\phi}_z &\mid \boldsymbol{\beta} &&\sim \text{Dir}(\boldsymbol{\beta}) \\
\boldsymbol{\theta} &\mid \boldsymbol{\alpha} &&\sim \text{Dir}(\boldsymbol{\alpha}) \\
z_d &\mid \boldsymbol{\theta} &&\sim \text{Cat}(\boldsymbol{\theta}) \\
s_{d_i} &\mid \lambda &&\sim \text{Ber}(\lambda) \\
w_{d_i} &\mid z_d\ ,\ s_{d_i}\ ,\ \boldsymbol{\phi}\ ,\ \boldsymbol{\tau}\ ,\ \boldsymbol{\omega} &&\sim (1 - s_{d_i})\text{Cat}(\boldsymbol{\phi}_{z_d}) + s_{d_i}\text{CatE}(\boldsymbol{\tau}_{z_d}\, \boldsymbol{\omega}^\top)
\end{aligned}
$$

3.2.3 Inference in the LF-LDA model

From the generative model of LF-LDA in Figure 3.1, by integrating out $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, we use the Gibbs sampling algorithm (Robert and Casella, 2004) to perform inference to calculate the conditional topic assignment probabilities for each word. The outline of the Gibbs

sampling algorithm for LF-LDA is detailed in Algorithm 4. Instead of sampling $\boldsymbol{\tau}_t$ from the posterior $\mathrm{P}(\boldsymbol{\tau}_t \mid \boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s})$, we perform MAP estimation as described in Section 3.2.5.

Algorithm 4: An approximate Gibbs sampling algorithm for the LF-LDA model

```
Initialize the word-topic variables z_{d_i} using the LDA sampling algorithm
for iteration iter = 1, 2, ... do
    for topic t = 1, 2, ..., T do
        τ_t = arg max_{τ_t} P(τ_t | w, z, s)
    for document d = 1, 2, ..., M do
        for word index i = 1, 2, ..., N_d do
            sample z_{d_i} and s_{d_i} from P(z_{d_i}, s_{d_i} | w, z_{¬d_i}, s_{¬d_i}, τ, ω)
```

By applying Bayes' rule (see Section 2.3.1), the conditional probability $\mathrm{P}(z_{d_i}, s_{d_i} \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d_i}, \boldsymbol{s}_{\neg d_i}, \boldsymbol{\tau}, \boldsymbol{\omega})$ for sampling the topic variable and binary selection variable for the i^{th} word w_{d_i} in the document d is derived as:

$$\begin{aligned} & \mathrm{P}(z_{d_i} = t, s_{d_i} = s \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d_i}, \boldsymbol{s}_{\neg d_i}, \boldsymbol{\tau}, \boldsymbol{\omega}) \\ = \; & \frac{\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s} \mid \boldsymbol{\tau}, \boldsymbol{\omega})}{\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}_{\neg d_i}, \boldsymbol{s}_{\neg d_i} \mid \boldsymbol{\tau}, \boldsymbol{\omega})} \\ \propto \; & \frac{\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega})}{\mathrm{P}(\boldsymbol{w}_{\neg d_i}, \boldsymbol{z}_{\neg d_i}, \boldsymbol{s}_{\neg d_i} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega})} \end{aligned} \tag{3.2}$$

So, the inference process involves computing the joint distribution of all observed and latent variables. As represented in Figure 3.1 and Section 3.2.1, the joint distribution of all observed and latent variables in the LF-LDA model can be factored as follows:

$$\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega}) = \mathrm{P}(\boldsymbol{z} \mid \boldsymbol{\alpha})\mathrm{P}(\boldsymbol{s} \mid \lambda)\mathrm{P}(\boldsymbol{w} \mid \boldsymbol{z}, \boldsymbol{s}, \boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{\omega}) \tag{3.3}$$

Notation: Due to the new models' two-component mixture of a topic-to-word Dirichlet multinomial component and a latent feature component, we separate out the counts for each of the two components of each model. We define the rank-3 tensor $K_d^{t,w}$ as the number of times a word w in document d is generated from topic t by the latent feature component of the generative LF-LDA or LF-DMM model. We also modify the earlier definition of the tensor $N_d^{t,w}$ to be the number of times a word w in document d is generated from topic

t by the Dirichlet multinomial component of LF-LDA or LF-DMM. For both tensors K and N, omitting an index refers to summation over that index, e.g., $N_d^w + K_d^w$ is the total number of times the word type w appears in document d while $N_d + K_d$ is the number of word tokens in document d. In addition, negation $\neg$ indicates exclusion as before: $\neg d$ represents the document collection D with the document d removed, and $\neg d_i$ represents D with just the i^{th} word in d removed, while $d_{\neg i}$ denotes the document d without its i^{th} word. Notations are explained in details in tables 3.12 and 3.13 at the end of this chapter.

To obtain the first term $\mathrm{P}(\boldsymbol{z} \mid \boldsymbol{\alpha})$, we integrate over the distributions $\boldsymbol{\theta}$ as follows:

$$\begin{aligned}
\mathrm{P}(\boldsymbol{z} \mid \boldsymbol{\alpha}) &= \int \mathrm{P}(\boldsymbol{z}, \boldsymbol{\theta} \mid \boldsymbol{\alpha})\ \mathrm{d}\boldsymbol{\theta} = \int \mathrm{P}(\boldsymbol{z} \mid \boldsymbol{\theta})\mathrm{P}(\boldsymbol{\theta} \mid \boldsymbol{\alpha})\ \mathrm{d}\boldsymbol{\theta} \\
&= \int \Big(\prod_{d=1}^{M}\prod_{t=1}^{T} \theta_{d,t}^{N_d^t + K_d^t}\Big)\Big(\prod_{d=1}^{M} \frac{1}{\Delta(\boldsymbol{\alpha})}\prod_{t=1}^{T} \theta_{d,t}^{\alpha-1}\Big)\ \mathrm{d}\boldsymbol{\theta} \\
&= \prod_{d=1}^{M} \frac{\Delta(\mathbf{N}_d^* + \mathbf{K}_d^* + \boldsymbol{\alpha})}{\Delta(\boldsymbol{\alpha})} \int \frac{1}{\Delta(\mathbf{N}_d^* + \mathbf{K}_d^* + \boldsymbol{\alpha})} \prod_{t=1}^{T} \theta_{d,t}^{N_d^t + K_d^t + \alpha - 1}\ \mathrm{d}\boldsymbol{\theta} \\
&= \prod_{d=1}^{M} \frac{\Delta(\mathbf{N}_d^* + \mathbf{K}_d^* + \boldsymbol{\alpha})}{\Delta(\boldsymbol{\alpha})} \qquad (3.4)
\end{aligned}$$

where $\mathbf{N}_d^*$ and $\mathbf{K}_d^*$ are the document-to-topic count vectors: $\mathbf{N}_d^* = \{N_d^t\}_{t=1}^T$ and $\mathbf{K}_d^* = \{K_d^t\}_{t=1}^T$. Note that for convenience, we use symmetric Dirichlet distributions with the Dirichlet delta function Δ to be defined as:

$$\begin{aligned}
\Delta(\boldsymbol{\alpha}) &= \frac{\Gamma(\alpha)^T}{\Gamma(T\alpha)} \\
\Delta(\mathbf{N}_d^* + \mathbf{K}_d^* + \boldsymbol{\alpha}) &= \frac{\prod_{t=1}^{T} \Gamma(N_d^t + K_d^t + \alpha)}{\Gamma(N_d + K_d + T\alpha)}
\end{aligned}$$

The second term $\mathrm{P}(\boldsymbol{s} \mid \lambda)$ is obtained as follows:

$$\mathrm{P}(\boldsymbol{s} \mid \lambda) = \prod_{d=1}^{M} (1-\lambda)^{N_d}\, \lambda^{K_d} \qquad (3.5)$$

Finally, we integrate over $\boldsymbol{\phi}$ to obtain the term $\mathrm{P}(\boldsymbol{w} \mid \boldsymbol{z}, \boldsymbol{s}, \boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{\omega})$ as follows:

$$
\begin{aligned}
&\mathrm{P}(\boldsymbol{w} \mid \boldsymbol{z}, \boldsymbol{s}, \boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{\omega}) \\
&\quad = \int \mathrm{P}(\boldsymbol{w}, \boldsymbol{\phi} \mid \boldsymbol{z}, \boldsymbol{s}, \boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{\omega}) \, \mathrm{d}\boldsymbol{\phi} \\
&\quad = \int \mathrm{P}(\boldsymbol{w} \mid \boldsymbol{\phi}, \boldsymbol{z}, \boldsymbol{s}, \boldsymbol{\tau}, \boldsymbol{\omega}) \mathrm{P}(\boldsymbol{\phi} \mid \boldsymbol{\beta}) \, \mathrm{d}\boldsymbol{\phi} \\
&\quad = \int \Big(\prod_{t=1}^{T} \prod_{w=1}^{V} \phi_{t,w}^{N^{t,w}} \, \mathrm{CatE}\big(w \mid \boldsymbol{\tau}_t \, \boldsymbol{\omega}^\top\big)^{K^{t,w}} \Big) \Big(\prod_{t=1}^{T} \frac{1}{\Delta(\boldsymbol{\beta})} \prod_{w=1}^{V} \phi_{t,w}^{\beta-1} \Big) \, \mathrm{d}\boldsymbol{\phi} \\
&\quad = \prod_{t=1}^{T} \prod_{w=1}^{V} \mathrm{CatE}\big(w \mid \boldsymbol{\tau}_t \, \boldsymbol{\omega}^\top\big)^{K^{t,w}} \prod_{t=1}^{T} \frac{\Delta(\mathbf{N}^{t,\bullet} + \boldsymbol{\beta})}{\Delta(\boldsymbol{\beta})} \int \frac{1}{\Delta(\mathbf{N}^{t,\bullet} + \boldsymbol{\beta})} \prod_{w=1}^{V} \phi_{t,w}^{N^{t,w}+\beta-1} \, \mathrm{d}\boldsymbol{\phi} \\
&\quad = \prod_{t=1}^{T} \prod_{w=1}^{V} \mathrm{CatE}\big(w \mid \boldsymbol{\tau}_t \, \boldsymbol{\omega}^\top\big)^{K^{t,w}} \prod_{t=1}^{T} \frac{\Delta(\mathbf{N}^{t,\bullet} + \boldsymbol{\beta})}{\Delta(\boldsymbol{\beta})} \qquad (3.6)
\end{aligned}
$$

where $\mathbf{N}^{t,\bullet}$ is the topic-to-word count vector: $\mathbf{N}^{t,\bullet} = \{N^{t,w}\}_{w=1}^{V}$, and:

$$
\begin{aligned}
\Delta(\boldsymbol{\beta}) &= \frac{\Gamma(\beta)^V}{\Gamma(V\beta)} \\
\Delta(\mathbf{N}^{t,\bullet} + \boldsymbol{\beta}) &= \frac{\prod_{w=1}^{V} \Gamma(N^{t,w} + \beta)}{\Gamma\big(N^t + V\beta\big)}
\end{aligned}
$$

Given the equations 3.4, 3.5 and 3.6, the joint distribution in Equation 3.3 becomes:

$$
\begin{aligned}
&\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega}) \qquad (3.7) \\
&= \prod_{d=1}^{M} \frac{\Delta(\mathbf{N}_d^* + \mathbf{K}_d^* + \boldsymbol{\alpha})}{\Delta(\boldsymbol{\alpha})} \prod_{d=1}^{M} \big(1-\lambda\big)^{N_d} \, \lambda^{K_d} \prod_{t=1}^{T} \prod_{w=1}^{V} \mathrm{CatE}\big(w \mid \boldsymbol{\tau}_t \, \boldsymbol{\omega}^\top\big)^{K^{t,w}} \prod_{t=1}^{T} \frac{\Delta(\mathbf{N}^{t,\bullet} + \boldsymbol{\beta})}{\Delta(\boldsymbol{\beta})}
\end{aligned}
$$

From equations 3.2 and 3.7, the conditional probability $\mathrm{P}(z_{d_i}, s_{d_i} \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d_i}, \boldsymbol{s}_{\neg d_i}, \boldsymbol{\tau}, \boldsymbol{\omega})$ is derived as follows:

$$
\begin{aligned}
&\mathrm{P}(z_{d_i} = t, s_{d_i} = s \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d_i}, \boldsymbol{s}_{\neg d_i}, \boldsymbol{\tau}, \boldsymbol{\omega}) \\
&\quad \propto \frac{\Delta(\mathbf{N}_d^* + \mathbf{K}_d^* + \boldsymbol{\alpha})}{\Delta(\mathbf{N}_{d_{\neg i}}^* + \mathbf{K}_{d_{\neg i}}^* + \boldsymbol{\alpha})} (1-\lambda)^{1-s} \, \lambda^s \, \mathrm{CatE}\big(w_{d_i} \mid \boldsymbol{\tau}_t \, \boldsymbol{\omega}^\top\big)^s \, \Big(\frac{\Delta(\mathbf{N}^{t,\bullet} + \boldsymbol{\beta})}{\Delta(\mathbf{N}_{\neg d_i}^{t,\bullet} + \boldsymbol{\beta})} \Big)^{1-s} \\
&\quad = \frac{\Gamma(N_d^t + K_d^t + \alpha)}{\Gamma(N_{d_{\neg i}}^t + K_{d_{\neg i}}^t + \alpha)} \frac{\Gamma(N_{d_{\neg i}} + K_{d_{\neg i}} + T\alpha)}{\Gamma(N_d + K_d + T\alpha)} (1-\lambda)^{1-s} \, \lambda^s \, \mathrm{CatE}\big(w_{d_i} \mid \boldsymbol{\tau}_t \, \boldsymbol{\omega}^\top\big)^s \\
&\qquad \Big(\frac{\prod_{w=1}^{V} \Gamma(N^{t,w} + \beta)}{\prod_{w=1}^{V} \Gamma(N_{\neg d_i}^{t,w} + \beta)} \frac{\Gamma(N_{\neg d_i}^t + V\beta)}{\Gamma(N^t + V\beta)} \Big)^{1-s} \qquad (3.8)
\end{aligned}
$$

where $N_d^t + K_d^t + \alpha = N_{d_{\neg i}}^t + K_{d_{\neg i}}^t + \alpha + 1$, and $N_d + K_d + T\alpha = N_{d_{\neg i}} + K_{d_{\neg i}} + T\alpha + 1$ is independent of t and s, and when $z_{d_i} = t$ and $s_{d_i} = 0$ we also have: $N^{t,w_{d_i}} + \beta = N_{\neg d_i}^{t,w_{d_i}} + \beta + 1$ and $N^t + V\beta = N_{\neg d_i}^t + V\beta + 1$. Because the Gamma function Γ has the property of $\Gamma(x+1) = x\Gamma(x)$, Equation 3.8 can be rewritten as:

$$\begin{aligned}
&\mathrm{P}(z_{d_i} = t, s_{d_i} = s \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d_i}, \boldsymbol{s}_{\neg d_i}, \boldsymbol{\tau}, \boldsymbol{\omega}) \\
&\quad \propto (N_{d_{\neg i}}^t + K_{d_{\neg i}}^t + \alpha)\Big((1-\lambda)\frac{N_{\neg d_i}^{t,w_{d_i}} + \beta}{N_{\neg d_i}^t + V\beta}\Big)^{1-s} \Big(\lambda \ \mathrm{CatE}(w_{d_i} \mid \boldsymbol{\tau}_t \boldsymbol{\omega}^\top)\Big)^s
\end{aligned} \tag{3.9}$$

Given Equation 3.9, for sampling the topic z_{d_i} and the binary indicator variable s_{d_i} of the i^{th} word w_{d_i} in the document d, we integrate out s_{d_i} in order to sample z_{d_i} and then sample s_{d_i} given z_{d_i}. We sample the topic z_{d_i} using the following conditional distribution:

$$\begin{aligned}
&\mathrm{P}(z_{d_i} = t \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d_i}, \boldsymbol{\tau}, \boldsymbol{\omega}) \\
&\quad \propto (N_{d_{\neg i}}^t + K_{d_{\neg i}}^t + \alpha)\Big((1-\lambda)\frac{N_{\neg d_i}^{t,w_{d_i}} + \beta}{N_{\neg d_i}^t + V\beta} + \lambda \ \mathrm{CatE}(w_{d_i} \mid \boldsymbol{\tau}_t \boldsymbol{\omega}^\top)\Big)
\end{aligned} \tag{3.10}$$

Then we sample s_{d_i} conditional on $z_{d_i} = t$ with:

$$\mathrm{P}(s_{d_i}{=}s \mid z_{d_i}{=}t) \propto \begin{cases} (1-\lambda)\dfrac{N_{\neg d_i}^{t,w_{d_i}} + \beta}{N_{\neg d_i}^t + V\beta} \text{ for } s = 0 \\ \lambda \ \mathrm{CatE}(w_{d_i} | \boldsymbol{\tau}_t \boldsymbol{\omega}^\top) \text{ for } s = 1 \end{cases} \tag{3.11}$$

Estimating topic and word distributions

Now we need to estimate the multinomial parameter sets $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ given $\boldsymbol{z}$. Applying Bayes' rule on the component d in Equation 3.4, we have:

$$\begin{aligned}
\mathrm{P}(\boldsymbol{\theta}_d \mid \boldsymbol{z}_d, \boldsymbol{\alpha}) &= \frac{\mathrm{P}(\boldsymbol{z}_d \mid \boldsymbol{\theta}_d)\mathrm{P}(\boldsymbol{\theta}_d \mid \boldsymbol{\alpha})}{\mathrm{P}(\boldsymbol{z}_d \mid \boldsymbol{\alpha})} \\
&= \frac{\Delta(\boldsymbol{\alpha})}{\Delta(\mathbf{N}_d^* + \mathbf{K}_d^* + \boldsymbol{\alpha})}\Big(\prod_{t=1}^T \theta_{d,t}^{N_d^t + K_d^t}\Big)\frac{1}{\Delta(\boldsymbol{\alpha})}\prod_{t=1}^T \theta_{d,t}^{\alpha-1} \\
&= \mathrm{Dir}(\boldsymbol{\theta}_d \mid \mathbf{N}_d^* + \mathbf{K}_d^* + \boldsymbol{\alpha})
\end{aligned} \tag{3.12}$$

Similarly, applying Bayes' rule on the component t in Equation 3.6, we also have:

$$\mathrm{P}(\boldsymbol{\phi}_t \mid \boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s}, \boldsymbol{\beta}) = \mathrm{Dir}(\boldsymbol{\phi}_t \mid \mathbf{N}^{t,\bullet} + \boldsymbol{\beta}) \tag{3.13}$$

Using the expectation of the Dirichlet distribution (see Equation 2.28) yields:

$$\hat{\theta}_{d,t} = \frac{N_d^t + K_d^t + \alpha}{N_d + K_d + T\alpha} \tag{3.14}$$

$$\hat{\phi}_{t,w} = \frac{N^{t,w} + \beta}{N^t + V\beta} \tag{3.15}$$

So the document-to-topic and topic-to-word distributions in the LF-LDA model are:

$$\mathrm{P}(t \mid d) = \hat{\theta}_{d,t} \tag{3.16}$$

$$\mathrm{P}(w \mid t) = (1-\lambda)\hat{\phi}_{t,w} + \lambda\, \mathrm{CatE}(w \mid \boldsymbol{\tau}_t \boldsymbol{\omega}^\top) \tag{3.17}$$

3.2.4 Inference in the LF-DMM model

For the LF-DMM model, we integrate out $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, and then sample the topic z_d and the distribution selection variables $\boldsymbol{s}_d$ for document d using Gibbs sampling as outlined in Algorithm 5. Similar to the method presented in Algorithm 4, we also use MAP estimation of $\boldsymbol{\tau}$ as detailed in Section 3.2.5 rather than sampling from the posterior $\mathrm{P}(\boldsymbol{\tau}_t \mid \boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s})$.

Algorithm 5: An approximate Gibbs sampling algorithm for the LF-DMM model

```
Initialize the word-topic variables z_{d_i} using the DMM sampling algorithm
for iteration iter = 1, 2, ... do
    for topic t = 1, 2, ..., T do
        τ_t = arg max_{τ_t} P(τ_t | w, z, s)
    for document d = 1, 2, ..., M do
        sample z_d and s_d from P(z_d = t, s_d | w, z_{¬d}, s_{¬d}, τ, ω)
```

By applying Bayes' rule, the conditional distribution $\mathrm{P}(z_d = t, \boldsymbol{s}_d \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d}, \boldsymbol{s}_{\neg d}, \boldsymbol{\tau}, \boldsymbol{\omega})$ for sampling topic variable and binary selection variables for document d is:

$$\begin{aligned}
&\mathrm{P}(z_d = t, \boldsymbol{s}_d \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d}, \boldsymbol{s}_{\neg d}, \boldsymbol{\tau}, \boldsymbol{\omega}) \\
&\quad = \frac{\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s} \mid \boldsymbol{\tau}, \boldsymbol{\omega})}{\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}_{\neg d}, \boldsymbol{s}_{\neg d} \mid \boldsymbol{\tau}, \boldsymbol{\omega})} \\
&\quad \propto \frac{\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega})}{\mathrm{P}(\boldsymbol{w}_{\neg d}, \boldsymbol{z}_{\neg d}, \boldsymbol{s}_{\neg d} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega})}
\end{aligned} \tag{3.18}$$

Similar to LF-LDA, we factor the joint distribution in LF-DMM as follows:

$$\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega}) = \mathrm{P}(\boldsymbol{z} \mid \boldsymbol{\alpha})\mathrm{P}(\boldsymbol{s} \mid \lambda)\mathrm{P}(\boldsymbol{w} \mid \boldsymbol{z}, \boldsymbol{s}, \boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{\omega}) \tag{3.19}$$

in which the distributions $\mathrm{P}(\boldsymbol{s} \mid \lambda)$ and $\mathrm{P}(\boldsymbol{w} \mid \boldsymbol{z}, \boldsymbol{s}, \boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{\omega})$ are computed by using Equations 3.5 and 3.6, respectively. $\mathrm{P}(\boldsymbol{z} \mid \boldsymbol{\alpha})$ in LF-DMM is derived by integrating over $\boldsymbol{\theta}$:

$$\begin{aligned}
\mathrm{P}(\boldsymbol{z} \mid \boldsymbol{\alpha}) &= \int \mathrm{P}(\boldsymbol{z}, \boldsymbol{\theta} \mid \boldsymbol{\alpha})\ \mathrm{d}\boldsymbol{\theta} = \int \mathrm{P}(\boldsymbol{z} \mid \boldsymbol{\theta})\mathrm{P}(\boldsymbol{\theta} \mid \boldsymbol{\alpha})\ \mathrm{d}\boldsymbol{\theta} \\
&= \int \prod_{t=1}^{T} \theta_t^{M^t} \frac{1}{\Delta(\boldsymbol{\alpha})} \prod_{t=1}^{T} \theta_d^{\alpha-1}\ \mathrm{d}\boldsymbol{\theta} \\
&= \frac{\Delta(\mathbf{M}^* + \boldsymbol{\alpha})}{\Delta(\boldsymbol{\alpha})} \int \frac{1}{\Delta(\mathbf{M}^* + \boldsymbol{\alpha})} \prod_{t=1}^{T} \theta_d^{M^t+\alpha-1}\ \mathrm{d}\boldsymbol{\theta} \\
&= \frac{\Delta(\mathbf{M}^* + \boldsymbol{\alpha})}{\Delta(\boldsymbol{\alpha})}
\end{aligned} \tag{3.20}$$

where $\mathbf{M}^*$ is the vector of topic observation counts for the collection D: $\mathbf{M}^* = \{M^t\}_{t=1}^T$ in which M^t is the number of documents assigned to topic t, and:

$$\Delta(\mathbf{M}^* + \boldsymbol{\alpha}) = \frac{\prod_{t=1}^{T} \Gamma(M^t + \alpha)}{\Gamma(M + T\alpha)}$$

Given Equations 3.5, 3.6 and 3.20, the joint distribution $\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega})$ in Equation 3.19 becomes:

$$\begin{aligned}
&\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega}) \\
&= \frac{\Delta(\mathbf{M}^* + \boldsymbol{\alpha})}{\Delta(\boldsymbol{\alpha})} \prod_{d=1}^{M} (1-\lambda)^{N_d}\ \lambda^{K_d} \prod_{t=1}^{T} \prod_{w=1}^{V} \mathrm{CatE}\big(w \mid \boldsymbol{\tau}_t\, \boldsymbol{\omega}^\top\big)^{K^{t,w}} \prod_{t=1}^{T} \frac{\Delta(\mathbf{N}^{t,\bullet} + \boldsymbol{\beta})}{\Delta(\boldsymbol{\beta})}
\end{aligned} \tag{3.21}$$

By applying Equation 3.21, the conditional distribution of topic variable and binary selection variables for document d in Equation 3.18 is obtained as:

$$\begin{aligned}
&\mathrm{P}(z_d = t, \boldsymbol{s}_d \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d}, \boldsymbol{s}_{\neg d}, \boldsymbol{\tau}, \boldsymbol{\omega}) \\
&\quad \propto \frac{\mathrm{P}(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{s} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega})}{\mathrm{P}(\boldsymbol{w}_{\neg d}, \boldsymbol{z}_{\neg d}, \boldsymbol{s}_{\neg d} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \boldsymbol{\tau}, \boldsymbol{\omega})} \\
&\quad = \frac{\Delta(\mathbf{M}^* + \boldsymbol{\alpha})}{\Delta(\mathbf{M}^*_{\neg d} + \boldsymbol{\alpha})} (1-\lambda)^{N_d}\, \lambda^{K_d} \prod_{w=1}^{V} \mathrm{CatE}\big(w \mid \boldsymbol{\tau}_t\, \boldsymbol{\omega}^\top\big)^{K_d^{t,w}} \frac{\Delta(\mathbf{N}^{t,\bullet} + \boldsymbol{\beta})}{\Delta(\mathbf{N}^{t,\bullet}_{\neg d} + \boldsymbol{\beta})} \\
&\quad = \frac{\Gamma(M^t + \alpha)}{\Gamma(M^t_{\neg d} + \alpha)} \frac{\Gamma(M_{\neg d} + V\alpha)}{\Gamma(M + V\alpha)} (1-\lambda)^{N_d}\, \lambda^{K_d} \prod_{w=1}^{V} \mathrm{CatE}\big(w \mid \boldsymbol{\tau}_t\, \boldsymbol{\omega}^\top\big)^{K_d^{t,w}} \\
&\qquad \frac{\prod_{w=1}^{V} \Gamma(N^{t,w} + \beta)}{\prod_{w=1}^{V} \Gamma(N^{t,w}_{\neg d} + \beta)} \frac{\Gamma(N^t_{\neg d} + V\beta)}{\Gamma(N^t + V\beta)}
\end{aligned} \tag{3.22}$$

where $M^t + \alpha = M^t_{\neg d} + \alpha + 1$ and $M + V\alpha = M_{\neg d} + V\alpha + 1$. LF-DMM assumes that all words in a document share the same topic, so when $z_d = t$, we have: $N^{t,w} + \beta = N^{t,w}_{\neg d} + N^w_d + \beta$, $N^t + V\beta = N^t_{\neg d} + N_d + V\beta$ and $K^{t,w}_d = K^w_d$. So Equation 3.22 can be rewritten as follows:

$$\begin{aligned}
&\mathrm{P}(z_d = t, \boldsymbol{s}_d \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d}, \boldsymbol{s}_{\neg d}, \boldsymbol{\tau}, \boldsymbol{\omega}) \\
&\quad \propto (M^t_{\neg d} + \alpha)(1-\lambda)^{N_d}\, \lambda^{K_d} \prod_{w=1}^{V} \mathrm{CatE}\big(w \mid \boldsymbol{\tau}_t\, \boldsymbol{\omega}^\top\big)^{K_d^w} \\
&\qquad \frac{\Gamma(N^t_{\neg d} + V\beta)}{\Gamma(N^t_{\neg d} + N_d + V\beta)} \prod_{w=1}^{V} \frac{\Gamma(N^{t,w}_{\neg d} + N^w_d + \beta)}{\Gamma(N^{t,w}_{\neg d} + \beta)} \\
&\quad = (M^t_{\neg d} + \alpha)(1-\lambda)^{N_d}\, \lambda^{K_d} \prod_{w=1}^{V} \mathrm{CatE}\Big(w \mid \boldsymbol{\tau}_t\, \boldsymbol{\omega}^\top\Big)^{K_d^w} \\
&\qquad \frac{\prod_{w=1}^{V} \prod_{j=1}^{N^w_d} (N^{t,w}_{\neg d} + \beta + j - 1)}{\prod_{j=1}^{N_d} (N^t_{\neg d} + V\beta + j - 1)}
\end{aligned} \tag{3.23}$$

Unfortunately it is difficult to integrate out $\boldsymbol{s}_d$ in this distribution P because of the ratios of Gamma functions. As z_d and $\boldsymbol{s}_d$ are not independent, it is computationally expensive to directly sample from this distribution, as there are $2^{(N^w_d + K^w_d)}$ different values of $\boldsymbol{s}_d$. So we approximate P with a distribution Q that factorizes across words as follows:

$$
\begin{aligned}
& Q(z_d = t, \boldsymbol{s}_d \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d}, \boldsymbol{s}_{\neg d}, \boldsymbol{\tau}, \boldsymbol{\omega}) \\
& \quad \propto \ (M^t_{\neg d} + \alpha)\ (1-\lambda)^{N_d}\ \lambda^{K_d} \prod_{w=1}^{V} \mathrm{CatE}\big(w \mid \boldsymbol{\tau}_t\, \boldsymbol{\omega}^\top\big)^{K^w_d} \prod_{w=1}^{V} \left(\frac{N^{t,w}_{\neg d} + \beta}{N^t_{\neg d} + V\beta}\right)^{N^w_d} \\
& \quad = \ (M^t_{\neg d} + \alpha) \prod_{w=1}^{V} \left((1-\lambda)\frac{N^{t,w}_{\neg d} + \beta}{N^t_{\neg d} + V\beta}\right)^{N^w_d} \Big(\lambda\ \mathrm{CatE}(w \mid \boldsymbol{\tau}_t\, \boldsymbol{\omega}^\top)\Big)^{K^w_d}
\end{aligned} \tag{3.24}
$$

This simpler distribution Q can be viewed as an approximation to P in which the topic-word "counts" are "frozen" within a document. This approximation is reasonably accurate for short documents. This distribution Q simplifies the coupling between z_d and $\boldsymbol{s}_d$. This enables us to integrate out $\boldsymbol{s}_d$ in Q. We first sample the document topic z_d for document d using $Q(z_d)$, marginalizing over $\boldsymbol{s}_d$:

$$
\begin{aligned}
& Q(z_d = t \mid \boldsymbol{w}, \boldsymbol{z}_{\neg d}, \boldsymbol{\tau}, \boldsymbol{\omega}) \\
& \quad \propto \ (M^t_{\neg d} + \alpha) \prod_{w=1}^{V} \left((1-\lambda)\frac{N^{t,w}_{\neg d} + \beta}{N^t_{\neg d} + V\beta} + \ \lambda\ \mathrm{CatE}(w \mid \boldsymbol{\tau}_t\, \boldsymbol{\omega}^\top)\right)^{(N^w_d + K^w_d)}
\end{aligned} \tag{3.25}
$$

Then we sample the binary indicator variable s_{d_i} for each i^{th} word w_{d_i} in document d conditional on $z_d = t$ from the following distribution:

$$
Q(s_{d_i}{=}s \mid z_d = t) \propto \begin{cases} (1-\lambda)\dfrac{N^{t,w_{d_i}}_{\neg d} + \beta}{N^t_{\neg d} + V\beta} \text{ for } s = 0 \\ \lambda\ \mathrm{CatE}(w_{d_i} \mid \boldsymbol{\tau}_t\, \boldsymbol{\omega}^\top) \text{ for } s = 1 \end{cases} \tag{3.26}
$$

Estimating topic and word distributions

The topic-to-word distribution $\mathrm{P}(w \mid t)$ in LF-DMM can be obtained following Equation 3.17 as in the LF-LDA model. It is also straightforward to obtain the distribution $\boldsymbol{\theta}$ over topics as follows:

$$
\hat{\theta}_t = \frac{M^t + \alpha}{M + T\alpha} \tag{3.27}
$$

So the document-to-topic distribution can be computed by applying Bayes' rule:

$$\mathrm{P}(t \mid d) \quad \propto \quad \mathrm{P}(t)\mathrm{P}(d \mid t) = \hat{\theta}_t \prod_{w=1}^{V} \mathrm{P}\big(w \mid t\big)^{(N_d^w + K_d^w)} \tag{3.28}$$

3.2.5 Learning latent feature vectors for topics

Bayesian uncertainty is problematic due to the log transformation (Eisenstein et al., 2011). So, to estimate the topic vectors after each Gibbs sampling iteration through the data, we apply regularized maximum likelihood estimation. Applying MAP estimation to learn log-linear models for topic models is also used in SAGE (Eisenstein et al., 2011) and SPRITE (Paul and Dredze, 2015). However, unlike our models, those models do not use latent feature word vectors to characterize topic-word distributions. The negative log likelihood of the corpus $\mathcal{L}$ under our model factorizes topic-wise into factors $\mathcal{L}_t$ for each topic. With L_2 regularization[2] for topic t, these are:

$$\mathcal{L}_t = -\sum_{w=1}^{V} K^{t,w} \Big(\boldsymbol{\tau}_t \cdot \boldsymbol{\omega}_w - \log \big(\sum_{w'=1}^{V} \exp(\boldsymbol{\tau}_t \cdot \boldsymbol{\omega}_{w'}) \big) \Big) + \mu \parallel \boldsymbol{\tau}_t \parallel_2^2 \tag{3.29}$$

The MAP estimate of topic vectors $\boldsymbol{\tau}_t$ is obtained by minimizing the regularized negative log likelihood. We used L-BFGS (Liu and Nocedal, 1989) to find the topic vector $\boldsymbol{\tau}_t$ that minimizes $\mathcal{L}_t$. See Section 2.2.4 for a brief introduction to L-BFGS.[3]

3.3 Experiments

To investigate the performance of our new models LF-LDA and LF-DMM, we compared their performance against baseline LDA and DMM models on common evaluation tasks of topic coherence, document clustering and document classification.[4] The topic coherence evaluation measures the coherence of topic-word associations, i.e., it directly evaluates how coherent the assignment of words to topics is (Chang et al., 2009; Stevens et al., 2012). The document clustering and document classification tasks evaluate how useful the topics

[2]The L_2 regularizer constant was set to $\mu = 0.01$.
[3]We used the L-BFGS implementation from the Mallet toolkit (McCallum, 2002).
[4]For experiments with LDA and DMM, we use the jLDADMM package (Nguyen, 2015).

assigned to documents are in clustering and classification tasks (Lu et al., 2011). Because we expect LF-LDA and LF-DMM to perform comparatively well in situations where there is little data about topic-to-word distributions, our experiments focus on corpora with few or short documents. We also investigated which values of the mixture weight hyper-parameter λ perform well, and compared the performance when using two different sets of pre-trained word vectors in these new models.

3.3.1 Experimental setup

3.3.1.1 Distributed word representations

We experimented with two state-of-the-art sets of pre-trained word vectors. Word2Vec word vectors are pre-trained 300-dimensional vectors for 3 million words and phrases.[5] These vectors were trained on a 100 billion word subset of the Google News corpus by using the Google Word2Vec Skip-gram model (Mikolov et al., 2013b). GloVe word vectors are pre-trained 300-dimensional vectors for 2 million words.[6] These vectors were learned from 42-billion tokens of Common Crawl data using the Stanford GloVe model (Pennington et al., 2014). We refer to the LF-LDA and LF-DMM models using the pre-trained Word2Vec and GloVe word vectors as: **w2v-LDA**, **glove-LDA**, **w2v-DMM** and **glove-DMM**.

3.3.1.2 Experimental datasets

We conducted experiments on the 20-Newsgroups dataset, the TagMyNews news dataset and the Sanders Twitter corpus. The 20-Newsgroups dataset contains about 19,000 newsgroup documents evenly grouped into 20 different categories.[7] The TagMyNews news dataset consists of about 32,600 English RSS news items grouped into 7 categories, where each news document has a news title and a short description (Vitale et al., 2012).[8] In our

[5]Download at: `https://code.google.com/p/word2vec/`

[6]Download at: `http://www-nlp.stanford.edu/projects/glove/`

[7]We used the "all-terms" version of the 20-Newsgroups dataset available at `http://web.ist.utl.pt/acardoso/datasets/` (Cardoso-Cachopo, 2007).

[8]The TagMyNews news dataset is unbalanced, where the largest category contains 8,200 news items while the smallest category contains about 1,800 items. Download at: `http://acube.di.unipi.it/tmn-dataset/`

Dataset	**#g**	**#docs**	**#w/d**	V
N20	20	18,820	103.3	19,572
N20short	20	1,794	13.6	6,377
N20small	20	400	88.0	8,157
TMN	7	32,597	18.3	13,428
TMNtitle	7	32,503	4.9	6,347
Twitter	4	2,520	5.0	1,390

Table 3.1: Details of experimental datasets. #**g**: number of ground truth labels; #**docs**: number of documents; #**w/d**: the average number of words per document.

experiments, we also used a news title dataset which consists of just the news titles from the TagMyNews news dataset. Each dataset was down-cased, and we removed non-alphabetic characters and stop-words found in the stop-word list in the Mallet toolkit (McCallum, 2002). We also removed words shorter than 3 characters and words appearing less than 10 times in the 20-Newsgroups corpus, and under 5 times in the TagMyNews news and news titles datasets. In addition, words not found in both Google and Stanford vector representations were also removed.[9] We refer to the cleaned 20-Newsgroups, TagMyNews news and news title datasets as **N20**, **TMN** and **TMNtitle**, respectively. We also performed experiments on two subsets of the N20 dataset. The **N20short** dataset consists of all documents from N20 with less than 21 words. The **N20small** dataset contains 400 documents consisting of 20 randomly selected documents from each group of N20.

Finally, we also experimented on the publicly available Sanders Twitter corpus.[10] This corpus consists of 5,512 Tweets grouped into four different topics (Apple, Google, Microsoft, and Twitter). Due to restrictions in Twitter's Terms of Service, the actual Tweets need to be downloaded using 5,512 Tweet IDs. There are 850 Tweets not available to download. After removing the non-English Tweets, 3,115 Tweets remain. In addition to converting into lowercase and removing non-alphabetic characters, words were normalized by using a lexical normalization dictionary for microblogs (Han et al., 2012). We then removed stop-words, words shorter than 3 characters or appearing less than 3 times in the corpus.

[9] 1366, 27 and 12 words were correspondingly removed out of the 20-Newsgroups, TagMyNews news and news title datasets.

[10] Download at: `http://www.sananalytics.com/lab/index.php`

The four words *apple*, *google*, *microsoft* and *twitter* were removed as these four words occur in every Tweet in the corresponding topic. Moreover, words not found in both Google and Stanford vector lists were also removed.[11] In all experiments, after removing words from documents, any document with a zero word count was also removed from the corpus. For the Twitter corpus, this resulted in just 2,520 remaining Tweets.

3.3.1.3 General settings

The hyper-parameter β used in baseline LDA and DMM models was set to 0.01, as this is a common setting in the literature (Griffiths and Steyvers, 2004). We set the hyper-parameter $\alpha = 0.1$, as this can improve performance relative to the standard setting $\alpha = \frac{50}{T}$, as noted by Lu et al. (2011) and Yin and Wang (2014). We ran each baseline model for 2000 iterations and evaluated the topics assigned to words in the last sample. For our new models, we ran the baseline models for 1500 iterations, then used the outputs from the last sample to initialize our models, which we ran for 500 further iterations. We report the mean and standard deviation of the results of ten repetitions of each experiment (so the standard deviation is approximately 3 standard errors, or a 99% confidence interval).

3.3.2 Topic coherence evaluation

This section examines the quality of the topic-word mappings induced by our models. In topic models, topics are distributions over words. The topic coherence evaluation measures to what extent the high-probability words in each topic are semantically coherent (Chang et al., 2009; Stevens et al., 2012).

3.3.2.1 Quantitative analysis

Newman et al. (2010), Mimno et al. (2011) and Lau et al. (2014) describe methods for automatically evaluating the semantic coherence of sets of words. The method presented in Lau et al. (2014) uses the normalized pointwise mutual information (NPMI) score and

[11] There are 91 removed words.

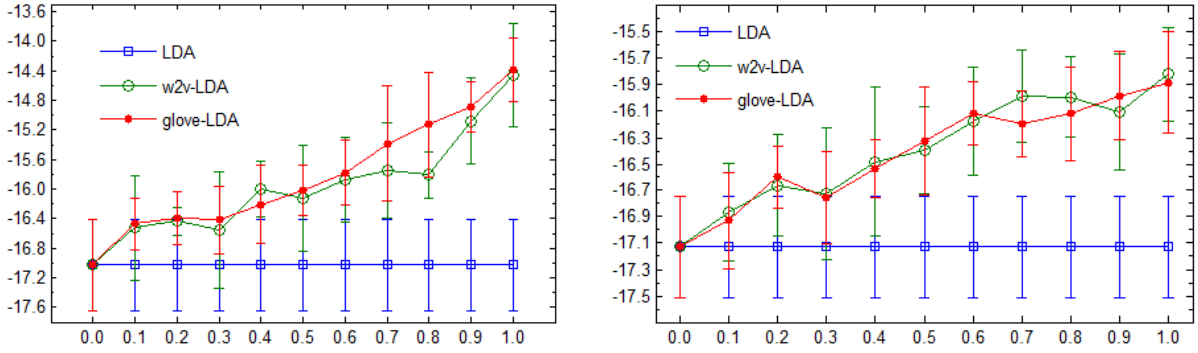


Figure 3.2: NPMI scores on the N20short dataset with 20 topics and 40 topics, varying the mixture weight λ from 0.0 to 1.0.

has a strong correlation with human-judged coherence. A higher NPMI score indicates that the topic distributions are semantically more coherent. Given a topic t represented by its top-N topic words $w_1, w_2, ..., w_N$, the NPMI score for t is computed as follows:

$$\text{NPMI-Score}(t) = \sum_{1 \leqslant i < j \leqslant N} \frac{\log \dfrac{\mathrm{P}(w_i, w_j)}{\mathrm{P}(w_i)\mathrm{P}(w_j)}}{-\log \mathrm{P}(w_i, w_j)} \tag{3.30}$$

where the probabilities in Equation 3.30 are derived from a 10-word sliding window over an external corpus. The NPMI score for a topic model is the average score for all topics. We compute the NPMI score based on top-15 most probable words of each topic and use the English Wikipedia of 4.6 million articles as our external corpus.[12]

Figure 3.2 shows NPMI scores computed for the LDA, w2v-LDA and glove-LDA models on the N20short dataset for 20 and 40 topics. We see that $\lambda = 1.0$ gives the highest NPMI score. In other words, using only the latent feature model produces the most coherent topic distributions.

Tables 3.2, 3.3 and 3.4 present the NPMI scores produced by the models on the other experimental datasets, where we vary the number of topics in steps from 4 to 80.[13] Tables 3.3 and 3.4 show that the DMM model performs better than the LDA model on the TMN,

[12]We used the Wikipedia-articles dump of July 8, 2014.

[13]We perform with $T = 6$ on the N20 and N20small datasets as the 20-Newsgroups dataset could be also grouped into 6 larger topics instead of 20 fine-grained categories.

Data	**Method**	$\lambda = 1.0$			
		T=6	T=20	T=40	T=80
N20	LDA	-16.7 ± 0.9	-11.7 ± 0.7	-11.5 ± 0.3	-11.4 ± 0.4
	w2v-LDA	-14.5 ± 1.2	-9.0 ± 0.8	-10.0 ± 0.5	-10.7 ± 0.4
	glove-LDA	**-11.6** ± 0.8	**-7.4** ± 1.0	**-8.3** ± 0.7	**-9.7** ± 0.4
	Improve.	5.1	4.3	3.2	1.7
N20small	LDA	-18.4 ± 0.6	-16.7 ± 0.6	-17.8 ± 0.4	-17.6 ± 0.3
	w2v-LDA	**-12.0** ± 1.1	**-12.7** ± 0.7	-15.5 ± 0.4	**-16.3** ± 0.3
	glove-LDA	-13.0 ± 1.1	-12.8 ± 0.7	**-15.0** ± 0.5	-16.6 ± 0.2
	Improve.	6.4	4.0	2.8	1.3

Table 3.2: NPMI scores (mean and standard deviation) for N20 and N20small datasets. The "*Improve.*" row denotes the absolute improvement accounted for the best result produced by our latent feature model over the baselines.

Data	**Method**	$\lambda = 1.0$			
		T=7	T=20	T=40	T=80
TMN	LDA	-17.3 ± 1.1	-12.7 ± 0.8	-12.3 ± 0.5	-13.0 ± 0.3
	w2v-LDA	-14.7 ± 1.5	-12.8 ± 0.8	-12.2 ± 0.5	-13.1 ± 0.2
	glove-LDA	**-13.0** ± 1.8	**-9.7** ± 0.7	**-11.5** ± 0.5	**-12.9** ± 0.4
	Improve.	4.3	3.0	0.8	0.1
TMN	DMM	-17.4 ± 1.5	-12.2 ± 1.0	-10.6 ± 0.6	-11.2 ± 0.4
	w2v-DMM	**-11.5** ± 1.6	-7.0 ± 0.7	**-5.8** ± 0.5	**-5.8** ± 0.3
	glove-DMM	-13.4 ± 1.5	**-6.2** ± 1.2	-6.6 ± 0.5	-6.3 ± 0.5
	Improve.	5.9	6.0	4.8	5.4
TMNtitle	LDA	-17.2 ± 0.8	-15.4 ± 0.7	-15.3 ± 0.3	-15.6 ± 0.3
	w2v-LDA	-14.2 ± 1.0	-14.0 ± 0.7	**-15.0** ± 0.3	**-14.9** ± 0.4
	glove-LDA	**-13.9** ± 0.9	**-13.4** ± 0.7	-15.2 ± 0.5	-15.2 ± 0.2
	Improve.	3.3	2.0	0.3	0.7
TMNtitle	DMM	-16.5 ± 0.9	-13.6 ± 1.0	-13.1 ± 0.5	-13.7 ± 0.3
	w2v-DMM	**-9.6** ± 0.6	**-7.5** ± 0.8	**-8.1** ± 0.4	-9.7 ± 0.4
	glove-DMM	-10.9 ± 1.3	-8.1 ± 0.5	**-8.1** ± 0.5	**-9.1** ± 0.3
	Improve.	5.6	6.1	5.0	4.6

Table 3.3: NPMI scores for TMN and TMNtitle datasets.

TMNtitle and Twitter datasets. These results show that our latent feature models produce significantly higher scores than the baseline models on all the experimental datasets.

Data	Method	$\lambda = 1.0$			
		T=4	T=20	T=40	T=80
Twitter	LDA	-8.5 ± 1.1	-14.5 ± 0.4	-15.1 ± 0.4	-15.9 ± 0.2
	w2v-LDA	-7.3 ± 1.0	**-13.2** ± 0.6	**-14.0** ± 0.3	**-14.1** ± 0.3
	glove-LDA	**-6.2** ± 1.6	-13.9 ± 0.6	-14.2 ± 0.4	-14.2 ± 0.2
	Improve.	2.3	1.3	1.1	1.8
Twitter	DMM	-5.9 ± 1.1	-10.4 ± 0.7	-12.0 ± 0.3	-13.3 ± 0.3
	w2v-DMM	-5.5 ± 0.7	-10.5 ± 0.5	-11.2 ± 0.5	**-12.5** ± 0.1
	glove-DMM	**-5.1** ± 1.2	**-9.9** ± 0.6	**-11.1** ± 0.3	**-12.5** ± 0.4
	Improve.	0.8	0.5	0.9	0.8

Table 3.4: NPMI scores for Twitter dataset.

Google Word2Vec vs. Stanford GloVe word vectors: In general, our latent feature models obtain competitive NPMI results in using pre-trained Google Word2Vec and Stanford GloVe word vectors for a large value of T, for example $T = 80$. With small values of T, for example with $T \leq 7$, using Google word vectors produces better scores than using Stanford word vectors on the small N20small dataset of normal texts and on the short text TMN and TMNtitle datasets. However, the opposite pattern holds on the full N20 dataset. Both sets of the pre-trained word vectors produce similar scores on the small and short Twitter dataset.

3.3.2.2 Qualitative analysis

Here we provide an example of how our new models improve topic coherence. Table 3.5 compares the top-15 words[14] produced by the baseline DMM model and our w2v-DMM model with $\lambda = 1.0$ on the TMNtitle dataset with $T = 20$ topics.

Topic 1 of the DMM model consists of words related to "nuclear crisis in Japan" together with other unrelated words. The w2v-DMM model produced a purer topic 1 focused on "Japan earthquake and nuclear crisis," presumably related to the "Fukushima Daiichi nuclear disaster." Topic 3 is about "oil prices" in both models. However, all top-15 words are qualitatively more coherent in the w2v-DMM model. While topic 4 of the DMM model

[14]In the baseline model, the top-15 topical words output from the 1500^{th} sample are similar to top-15 words from the 2000^{th} sample if we do not take the order of the most probable words into account.

Topic 1									Topic 3		
InitDMM	Iter=1	Iter=2	Iter=5	Iter=10	Iter=20	Iter=50	Iter=100	Iter=500	InitDMM	Iter=50	Iter=500
japan	japan	japan	japan	japan	japan	japan	japan	japan	u.s.	prices	prices
nuclear	nuclear	nuclear	nuclear	nuclear	nuclear	nuclear	nuclear	nuclear	oil	sales	sales
u.s.	u.s.	u.s.	u.s.	u.s.	u.s.	plant	u.s.	u.s.	japan	oil	oil
crisis	**russia**	crisis	plant	plant	plant	u.s.	plant	plant	prices	u.s.	u.s.
plant	radiation	china	crisis	radiation	**quake**	**quake**	**quake**	**quake**	stocks	stocks	profit
china	**nuke**	**russia**	radiation	crisis	radiation	radiation	radiation	radiation	sales	profit	stocks
libya	**iran**	plant	china	china	crisis	**earthquake**	**earthquake**	**earthquake**	profit	japan	japan
radiation	crisis	radiation	**russia**	**nuke**	**nuke**	**tsunami**	**tsunami**	**tsunami**	fed	rise	rise
u.n.	china	**nuke**	**nuke**	**russia**	china	**nuke**	**nuke**	**nuke**	rise	**gas**	**gas**
vote	libya	libya	**power**	**power**	**tsunami**	crisis	crisis	crisis	growth	growth	growth
korea	plant	**iran**	u.n.	u.n.	**earthquake**	**disaster**	**disaster**	**disaster**	wall	**profits**	shares
europe	u.n.	u.n.	**iran**	**iran**	**disaster**	**plants**	**oil**	**power**	street	shares	**price**
government	**mideast**	**power**	**reactor**	**earthquake**	**power**	**power**	**plants**	**oil**	china	**price**	**profits**
election	**pakistan**	**pakistan**	**earthquake**	**reactor**	**reactor**	**oil**	**power**	**japanese**	fall	**rises**	**rises**
deal	**talks**	**talks**	libya	**quake**	**japanese**	**japanese**	**tepco**	**plants**	shares	**earnings**	**earnings**

Topic 4			Topic 5			Topic 19			Topic 14		
InitDMM	Iter=50	Iter=500	InitDMM	Iter=50	Iter=500	InitDMM	Iter=50	Iter=500	InitDMM	Iter=50	Iter=500
egypt	libya	libya	critic	**dies**	star	nfl	nfl	nfl	nfl	law	law
china	egypt	egypt	corner	star	sheen	idol	draft	**sports**	court	bill	texas
u.s.	**mideast**	**iran**	office	**broadway**	**idol**	draft	**lockout**	draft	law	governor	bill
mubarak	**iran**	**mideast**	video	**american**	**broadway**	american	players	players	bill	texas	governor
bin	**opposition**	**opposition**	game	**idol**	show	show	**coach**	**lockout**	wisconsin	senate	senate
libya	**leader**	**protests**	star	lady	**american**	film	**nba**	**football**	players	union	union
laden	u.n.	**leader**	lady	gaga	gaga	season	**player**	**league**	judge	**obama**	**obama**
france	**protests**	**syria**	gaga	show	**tour**	sheen	sheen	n.f.l.	governor	wisconsin	budget
bahrain	**syria**	u.n.	show	**news**	**cbs**	n.f.l.	**league**	**player**	union	budget	wisconsin
air	**tunisia**	**tunisia**	weekend	critic	**hollywood**	back	n.f.l.	**baseball**	house	**state**	**immigration**
report	**protesters**	**chief**	sheen	**film**	**mtv**	top	**coaches**	**court**	texas	**immigration**	**state**
rights	**chief**	**protesters**	box	**hollywood**	lady	star	**football**	**coaches**	lockout	**arizona**	**vote**
court	**asia**	mubarak	park	**fame**	**wins**	charlie	**judge**	**nflpa**	budget	**california**	**washington**
u.n.	**russia**	**crackdown**	takes	**actor**	**charlie**	players	**nflpa**	**basketball**	peru	**vote**	**arizona**
war	**arab**	bahrain	man	**movie**	**stars**	men	**court**	**game**	senate	**federal**	**california**

Table 3.5: Examples of the 15 most probable topical words on the TMNtitle dataset with $T = 20$. InitDMM denotes the output from the 1500^{th} sample produced by the DMM model, which we use to initialize the w2v-DMM model. Iter=1, Iter=2, Iter=3 and the like refer to the output of our w2v-DMM model after running 1, 2, 3 sampling iterations, respectively. The words found in InitDMM and not found in Iter=500 are underlined. Words found by the w2v-DMM model but not found by the DMM model are in **bold**.

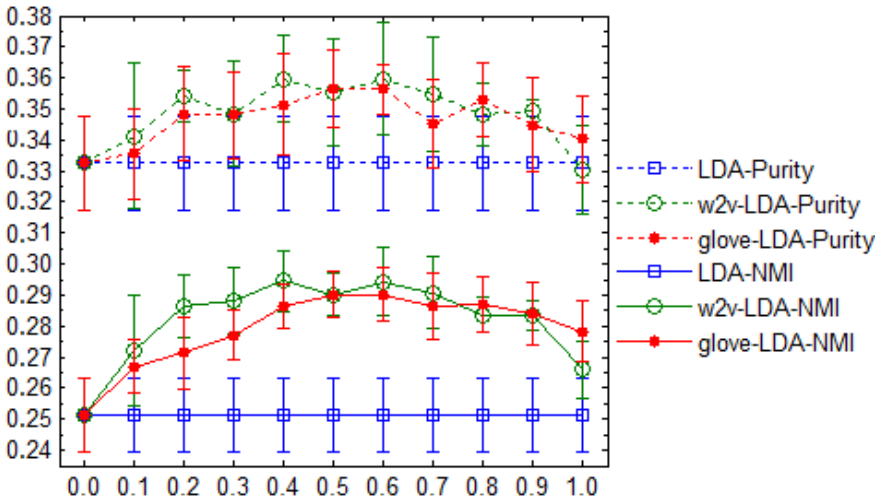


Figure 3.3: Purity and NMI results (mean and standard deviation) on the N20short dataset with number of topics $T = 20$, varying the mixture weight λ from 0.0 to 1.0.

is difficult to manually label, topic 4 of the w2v-DMM model is about the "Arab Spring" event. Topics 5, 19 and 14 of the DMM model are not easy to label. Topic 5 relates to "entertainment", topic 19 is generally a mixture of "entertainment" and "sport", and topic 14 is about "sport" and "politics." However, the w2v-DMM model more clearly distinguishes these topics: topic 5 is about "entertainment", topic 19 is only about "sport" and topic 14 is only about "politics."

3.3.3 Document clustering evaluation

We compared our models to the baseline models in a document clustering task. After using a topic model to calculate the topic probabilities of a document, we assign every document the topic with the highest probability given the document (Cai et al., 2008; Lu et al., 2011; Xie and Xing, 2013; Yan et al., 2013). We use two common metrics to evaluate clustering performance: *Purity* and *normalized mutual information* (NMI).[15] Purity and NMI scores always range from 0.0 to 1.0, and higher scores reflect better clustering performance.

Figures 3.3 and 3.4 present Purity and NMI results obtained by the LDA, w2v-LDA and glove-LDA models on the N20short dataset with the numbers of topics T set to either

[15]See Manning et al. (2008, Section 16.3) for details of the Purity and NMI metrics.

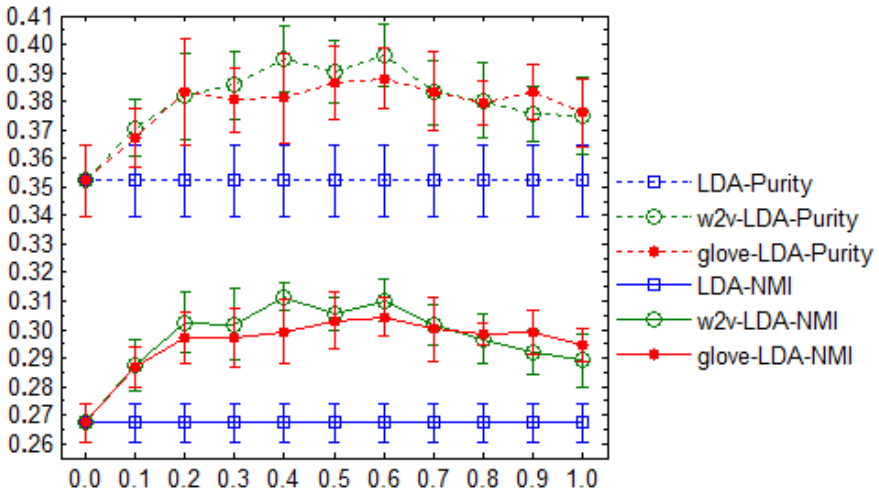


Figure 3.4: Purity and NMI results on the N20short dataset with number of topics $T = 40$, varying the mixture weight λ from 0.0 to 1.0.

20 or 40, and the value of the mixture weight λ varied from 0.0 to 1.0. We found that setting λ to 1.0 (i.e., using only the latent features to model words), the glove-LDA produced 1%+ higher scores on both Purity and NMI results than the w2v-LDA when using 20 topics. However, the two models glove-LDA and w2v-LDA returned equivalent results with 40 topics where they gain 2%+ absolute improvement[16] on both Purity and NMI against the baseline LDA model. By varying λ, as shown in figures 3.3 and 3.4, the w2v-LDA and glove-LDA models obtain their best results at $\lambda = 0.6$ where the w2v-LDA model does slightly better than the glove-LDA. Both models significantly outperform their baseline LDA models; for example with 40 topics, the w2v-LDA model attains 4.4% and 4.3% over the LDA model on Purity and NMI metrics, respectively.

We fix the mixture weight λ at 0.6, and report experimental results based on this value for the rest of this section. Tables 3.6, 3.7 and 3.8 show clustering results produced by our models and the baseline models on the remaining datasets with different numbers of topics. As expected, the DMM model is better than the LDA model on the short datasets of TMN, TMNtitle and Twitter. For example with 80 topics on the TMNtitle dataset, the DMM achieves about 7+% higher Purity and NMI scores than LDA.

[16]Using the Student's t-Test, the improvement is significant ($p < 0.01$).

Data	Method	Purity				NMI			
		T=6	T=20	T=40	T=80	T=6	T=20	T=40	T=80
N20	LDA	0.293 ± 0.002	0.573 ± 0.019	**0.639** ± 0.017	**0.646** ± 0.005	0.516 ± 0.009	0.582 ± 0.009	**0.557** ± 0.007	**0.515** ± 0.003
	w2v-LDA	0.291 ± 0.002	0.569 ± 0.021	0.616 ± 0.017	0.638 ± 0.006	0.500 ± 0.008	0.563 ± 0.009	0.535 ± 0.008	0.505 ± 0.004
	glove-LDA	**0.295** ± 0.001	**0.604** ± 0.031	0.632 ± 0.017	0.638 ± 0.007	**0.522** ± 0.003	**0.596** ± 0.012	0.550 ± 0.010	0.507 ± 0.003
	Improve.	0.002	0.031	-0.007	-0.008	0.006	0.014	-0.007	-0.008
N20small	LDA	0.232 ± 0.011	0.408 ± 0.017	0.477 ± 0.015	0.559 ± 0.018	0.376 ± 0.016	0.474 ± 0.013	0.513 ± 0.009	0.563 ± 0.008
	w2v-LDA	0.229 ± 0.005	**0.439** ± 0.015	**0.516** ± 0.024	**0.595** ± 0.016	0.406 ± 0.023	**0.519** ± 0.014	**0.548** ± 0.017	**0.585** ± 0.009
	glove-LDA	**0.235** ± 0.008	0.427 ± 0.022	0.492 ± 0.022	0.579 ± 0.011	**0.436** ± 0.019	0.504 ± 0.020	0.527 ± 0.013	0.576 ± 0.006
	Improve.	0.003	0.031	0.039	0.036	0.06	0.045	0.035	0.022

Table 3.6: Purity and NMI results (mean and standard deviation) on the N20 and N20small datasets with $\lambda = 0.6$. The "*Improve.*" row denotes the difference between the best result obtained by our model and the baseline model.

Data	Method	Purity				NMI			
		T=7	T=20	T=40	T=80	T=7	T=20	T=40	T=80
TMN	LDA	0.648 ± 0.029	0.717 ± 0.009	**0.721** ± 0.003	0.719 ± 0.007	0.436 ± 0.019	0.393 ± 0.008	0.354 ± 0.003	0.320 ± 0.003
	w2v-LDA	**0.658** ± 0.020	0.716 ± 0.012	0.720 ± 0.008	**0.725** ± 0.004	0.446 ± 0.014	0.399 ± 0.006	0.355 ± 0.005	**0.325** ± 0.003
	glove-LDA	**0.658** ± 0.034	**0.722** ± 0.007	0.719 ± 0.008	**0.725** ± 0.006	**0.448** ± 0.017	**0.403** ± 0.004	**0.356** ± 0.004	0.324 ± 0.004
	Improve.	0.01	0.005	-0.001	0.006	0.012	0.01	0.002	0.005
TMN	DMM	0.637 ± 0.029	0.699 ± 0.015	0.707 ± 0.014	0.715 ± 0.009	0.445 ± 0.024	0.422 ± 0.007	0.393 ± 0.009	0.364 ± 0.006
	w2v-DMM	0.623 ± 0.020	0.737 ± 0.018	**0.760** ± 0.010	0.772 ± 0.005	0.426 ± 0.015	0.428 ± 0.009	0.405 ± 0.006	0.378 ± 0.003
	glove-DMM	**0.641** ± 0.042	**0.749** ± 0.011	0.758 ± 0.008	**0.776** ± 0.006	**0.449** ± 0.028	**0.441** ± 0.008	**0.408** ± 0.005	**0.381** ± 0.003
	Improve.	0.004	0.05	0.053	0.061	0.004	0.019	0.015	0.017
TMNtitle	LDA	0.572 ± 0.014	0.599 ± 0.015	0.593 ± 0.011	0.580 ± 0.006	0.314 ± 0.008	0.262 ± 0.006	0.228 ± 0.006	0.196 ± 0.003
	w2v-LDA	0.579 ± 0.020	0.619 ± 0.015	**0.611** ± 0.007	0.598 ± 0.004	0.321 ± 0.012	0.279 ± 0.006	**0.239** ± 0.005	**0.210** ± 0.002
	glove-LDA	**0.584** ± 0.026	**0.623** ± 0.012	0.600 ± 0.008	**0.601** ± 0.004	**0.322** ± 0.015	**0.280** ± 0.004	0.235 ± 0.006	0.209 ± 0.003
	Improve.	0.012	0.024	0.018	0.021	0.008	0.018	0.011	0.014
TMNtitle	DMM	0.558 ± 0.015	0.600 ± 0.010	0.634 ± 0.011	0.658 ± 0.006	0.338 ± 0.012	0.327 ± 0.006	0.304 ± 0.004	0.271 ± 0.002
	w2v-DMM	0.552 ± 0.022	0.653 ± 0.012	0.678 ± 0.007	0.682 ± 0.005	0.314 ± 0.016	0.325 ± 0.006	0.305 ± 0.004	**0.282** ± 0.003
	glove-DMM	**0.586** ± 0.019	**0.672** ± 0.013	**0.679** ± 0.009	**0.683** ± 0.004	**0.343** ± 0.015	**0.339** ± 0.007	**0.307** ± 0.004	**0.282** ± 0.002
	Improve.	0.028	0.072	0.045	0.025	0.005	0.012	0.003	0.011

Table 3.7: Purity and NMI results on the TMN and TMNtitle datasets with $\lambda = 0.6$.

Data	Method	Purity				NMI			
		T=4	T=20	T=40	T=80	T=4	T=20	T=40	T=80
Twitter	LDA	0.559 ± 0.020	0.614 ± 0.016	0.626 ± 0.011	0.631 ± 0.008	0.196 ± 0.018	0.174 ± 0.008	0.170 ± 0.007	0.160 ± 0.004
	w2v-LDA	**0.598** ± 0.023	**0.635** ± 0.016	**0.638** ± 0.009	**0.637** ± 0.012	**0.249** ± 0.021	**0.191** ± 0.011	0.176 ± 0.003	**0.167** ± 0.006
	glove-LDA	0.597 ± 0.016	**0.635** ± 0.014	0.637 ± 0.010	**0.637** ± 0.007	0.242 ± 0.013	**0.191** ± 0.007	**0.177** ± 0.007	0.165 ± 0.005
	Improve.	0.039	0.021	0.012	0.006	0.053	0.017	0.007	0.007
Twitter	DMM	0.523 ± 0.011	0.619 ± 0.015	0.660 ± 0.008	0.684 ± 0.010	0.222 ± 0.013	0.213 ± 0.011	0.198 ± 0.008	0.196 ± 0.004
	w2v-DMM	**0.589** ± 0.017	0.655 ± 0.015	**0.668** ± 0.008	0.694 ± 0.009	0.243 ± 0.014	0.215 ± 0.009	**0.203** ± 0.005	0.204 ± 0.006
	glove-DMM	0.583 ± 0.023	**0.661** ± 0.019	0.667 ± 0.009	**0.697** ± 0.009	**0.250** ± 0.020	**0.223** ± 0.014	0.201 ± 0.006	**0.206** ± 0.005
	Improve.	0.066	0.042	0.008	0.013	0.028	0.01	0.005	0.01

Table 3.8: Purity and NMI results on the Twitter dataset with $\lambda = 0.6$.

New models vs. baseline models: On most tests, our models score higher than the baseline models, particularly on the small N20small dataset where we get 6.0% improvement on NMI at $T = 6$, and on the short text TMN and TMNtitle datasets we obtain 6.1% and 2.5% higher Purity at $T = 80$. In addition, on the short and small Twitter dataset with $T = 4$, we achieve 3.9% and 5.3% improvements in Purity and NMI scores, respectively. Those results show that an improved model of topic-word mappings also improves the document-topic assignments. For the small value of $T \leq 7$, on the large datasets of N20, TMN and TMNtitle, our models and baseline models obtain similar clustering results. However, with higher values of T, our models perform better than the baselines on TMN and TMNtitle, while on N20, the baseline LDA model attains a slightly higher clustering results than ours. In contrast, on the short and small Twitter dataset, our models obtain considerably better clustering results than the baseline models with a small value of T.

Google Word2Vec vs. Stanford GloVe word vectors: On the small N20short and N20small datasets, using the Google pre-trained word vectors produces higher clustering scores than using Stanford pre-trained word vectors. However, on the large datasets N20, TMN and TMNtitle, using Stanford word vectors produces higher scores than using Google word vectors when using a smaller number of topics, for example $T \leq 20$. With more topics, for instance $T = 80$, the pre-trained Google and Stanford word vectors produce similar clustering results. In addition, on the Twitter dataset, both sets of pre-trained word vectors produce similar results.

3.3.4 Document classification evaluation

Unlike the document clustering task, the document classification task evaluates the distribution over topics for each document. Following Lacoste-Julien et al. (2009), Lu et al. (2011), Huh and Fienberg (2012) and Zhai and Boyd-graber (2013), we used Support Vector Machines (SVM) to predict the ground truth labels from the topic-proportion vector of each document. We used the WEKA's implementation (Hall et al., 2009) of the fast Sequential Minimal Optimization algorithm (Platt, 1999) for learning a classifier with

Data	Method	$\lambda = 0.6$			
		T=6	T=20	T=40	T=80
N20	LDA	0.312 ± 0.013	0.635 ± 0.016	**0.742** ± 0.014	0.763 ± 0.005
	w2v-LDA	**0.316** ± 0.013	0.641 ± 0.019	0.730 ± 0.017	**0.768** ± 0.004
	glove-LDA	0.288 ± 0.013	**0.650** ± 0.024	0.733 ± 0.011	0.762 ± 0.006
	Improve.	0.004	0.015	-0.009	0.005
N20small	LDA	0.204 ± 0.020	0.392 ± 0.029	0.459 ± 0.030	0.477 ± 0.025
	w2v-LDA	**0.213** ± 0.018	**0.442** ± 0.025	**0.502** ± 0.031	**0.509** ± 0.022
	glove-LDA	0.181 ± 0.011	0.420 ± 0.025	0.474 ± 0.029	0.498 ± 0.012
	Improve.	0.009	0.05	0.043	0.032

Table 3.9: F_1 scores (mean and standard deviation) for N20 and N20small datasets.

Data	Method	$\lambda = 0.6$			
		T=7	T=20	T=40	T=80
TMN	LDA	0.658 ± 0.026	0.754 ± 0.009	0.768 ± 0.004	0.778 ± 0.004
	w2v-LDA	0.663 ± 0.021	0.758 ± 0.009	**0.769** ± 0.005	**0.780** ± 0.004
	glove-LDA	**0.664** ± 0.025	**0.760** ± 0.006	0.767 ± 0.003	0.779 ± 0.004
	Improve.	0.006	0.006	0.001	0.002
TMN	DMM	0.607 ± 0.040	0.694 ± 0.026	0.712 ± 0.014	0.721 ± 0.008
	w2v-DMM	0.607 ± 0.019	0.736 ± 0.025	**0.760** ± 0.011	0.771 ± 0.005
	glove-DMM	**0.621** ± 0.042	**0.750** ± 0.011	0.759 ± 0.006	**0.775** ± 0.006
	Improve.	0.014	0.056	0.048	0.054
TMNtitle	LDA	0.564 ± 0.015	0.625 ± 0.011	0.626 ± 0.010	0.624 ± 0.006
	w2v-LDA	0.563 ± 0.029	**0.644** ± 0.010	**0.643** ± 0.007	0.640 ± 0.004
	glove-LDA	**0.568** ± 0.028	**0.644** ± 0.010	0.632 ± 0.008	**0.642** ± 0.005
	Improve.	0.004	0.019	0.017	0.018
TMNtitle	DMM	0.500 ± 0.021	0.600 ± 0.015	0.630 ± 0.016	0.652 ± 0.005
	w2v-DMM	0.528 ± 0.028	0.663 ± 0.008	0.682 ± 0.008	**0.681** ± 0.006
	glove-DMM	**0.565** ± 0.022	**0.680** ± 0.011	**0.684** ± 0.009	**0.681** ± 0.004
	Improve.	0.065	0.08	0.054	0.029

Table 3.10: F_1 scores for TMN and TMNtitle datasets.

ten-fold cross-validation and WEKA's default parameters. We present the macro-averaged F_1 score (Manning et al., 2008, Section 13.6) as the evaluation metric for this task.

Just as in the document clustering task, the mixture weight $\lambda = 0.6$ obtains the highest classification performances on the N20short dataset. For example, with $T = 40$, our w2v-LDA and glove-LDA obtain F_1 scores at 40.0% and 38.9% which are 4.5% and 3.4% higher

Data	Method	$\lambda = 0.6$			
		T=4	T=20	T=40	T=80
Twitter	LDA	0.526 ± 0.021	0.636 ± 0.011	0.650 ± 0.014	0.653 ± 0.008
	w2v-LDA	**0.578** ± 0.047	0.651 ± 0.015	0.661 ± 0.011	**0.664** ± 0.010
	glove-LDA	0.569 ± 0.037	**0.656** ± 0.011	**0.662** ± 0.008	0.662 ± 0.006
	Improve.	0.052	0.02	0.012	0.011
Twitter	DMM	0.469 ± 0.014	0.600 ± 0.021	0.645 ± 0.009	0.665 ± 0.014
	w2v-DMM	**0.539** ± 0.016	0.649 ± 0.016	0.656 ± 0.007	0.676 ± 0.012
	glove-DMM	0.536 ± 0.027	**0.654** ± 0.019	**0.657** ± 0.008	**0.680** ± 0.009
	Improve.	0.07	0.054	0.012	0.015

Table 3.11: F_1 scores for Twitter dataset.

than F_1 score at 35.5% obtained by the LDA model, respectively.

We report classification results on the remaining experimental datasets with mixture weight $\lambda = 0.6$ in tables 3.9, 3.10 and 3.11. Unlike the clustering results, the LDA model does better than the DMM model for classification on the TMN dataset.

New models vs. baseline models: On most evaluations, our models perform better than the baseline models. In particular, on the small N20small and Twitter datasets, when the number of topics T is equal to number of ground truth labels (i.e., 20 and 4 correspondingly), our w2v-LDA obtains $5^{+}\%$ higher F_1 score than the LDA model. In addition, our w2v-DMM model achieves 5.4% and 2.9% higher F_1 score than the DMM model on short TMN and TMNtitle datasets with $T = 80$, respectively.

Google Word2Vec vs. Stanford GloVe word vectors: The comparison of the Google and Stanford word vectors for classification is similar to the one for clustering.

3.4 Discussion

We found that the topic coherence evaluation produced the best results with a mixture weight $\lambda = 1$, which corresponds to using topic-word distributions defined in terms of the latent-feature word vectors. This is not surprising, since the topic coherence evaluation

we used (Lau et al., 2014) is based on word co-occurrences in an external corpus (here, Wikipedia), and it is reasonable that the billion-word corpora used to train the latent feature word vectors are more useful for this task than the much smaller topic-modeling corpora, from which the topic-word multinomial distributions are trained.

On the other hand, the document clustering and document classification tasks depend more strongly on possibly idiosyncratic properties of the smaller topic-modeling corpora, since these evaluations reflect how well the document-topic assignments can group or distinguish documents within the topic-modeling corpus. Smaller values of λ enable the models to learn topic-word distributions that include an arbitrary multinomial topic-word distribution, enabling the models to capture idiosyncratic properties of the topic-modeling corpus. Even in these evaluations we found that an intermediate value of $\lambda = 0.6$ produced the best results, indicating that better word-topic distributions were produced when information from the large external corpus is combined with corpus-specific topic-word multinomials. We found that using the latent feature word vectors produced significant performance improvements even when the domain of the topic-modeling corpus was quite different to that of the external corpus from which the word vectors were derived, as was the case in our experiments on Twitter data.

We found that using either the pre-trained Google or the Stanford word vectors produced very similar results. As far as we could tell, there is no reason to prefer either one of these in our topic modeling applications. In order to compare the pre-trained Google and Stanford word vectors, we excluded words that did not appear in both sets of vectors. It would be interesting for future work to learn vectors for these unseen words. In addition, it is worth fine-tuning the seen-word vectors on the dataset of interest.

3.5 Summary

In this chapter, we showed that latent feature word representations containing external information from billion-word corpora can be used to improve topic modeling on smaller datasets. We proposed two novel latent feature topic models, namely LF-LDA and LF-

DMM, that integrate a latent feature model within two topic models LDA and DMM. We compared the performance of LF-LDA and LF-DMM to the baselines LDA and DMM on topic coherence, document clustering and document classification evaluations. In the topic coherence evaluation, LF-LDA and LF-DMM outperformed the baseline models on all six experimental datasets, showing that our method for exploiting external information from very large corpora helps improve the topic-to-word mapping. Meanwhile, document clustering and document classification results show that LF-LDA and LF-DMM improve the document-topic assignments compared to the baseline models, especially on datasets with few or short documents.

Notation		Description
General	M	Number of documents in the document collection D, i.e., $M = \|D\|$
	V	Size of the vocabulary W, i.e., number of word types $V = \|W\|$
	T	Number of topics
	$\neg d$	Document collection D with document d removed
	$\neg d_i$	Document collection D with just the i^{th} word in document d removed
	$d_{\neg i}$	Document d without its i^{th} word
	α	A scalar hyper-parameter
	$\boldsymbol{\alpha}$	T-dimensional parameter vector of the symmetric Dirichlet prior: $\boldsymbol{\alpha}_j = \alpha$ for $j = 1, 2, ..., T$
	β	A scalar hyper-parameter
	$\boldsymbol{\beta}$	V-dimensional parameter vector of the symmetric Dirichlet prior: $\boldsymbol{\beta}_k = \beta$ for $k = 1, 2, ..., V$
	λ	Mixture weight hyper-parameter
	$\boldsymbol{w}$	Word observations for the whole document collection D
	w_{d_i}	The i^{th} word in document d
	$\boldsymbol{s}$	Distribution indicator variables for the document collection D
	s_{d_i}	The binary indicator variable that determines whether the Dirichlet multinomial or latent feature component will be used to generate w_{d_i}
	$\boldsymbol{\phi}$	Topic-word distributions as a $T \times V$ matrix
	$\boldsymbol{\phi}_t$	Topic-word distribution for topic t, entries in $\boldsymbol{\phi}$
	$\phi_{t,w}$	Probability of a word type with vocabulary index w given topic t
LF-LDA	$\boldsymbol{z}$	Topic assignments of all words in the document collection D
	$\boldsymbol{z}_d$	Topic assignments of all words in document d
	z_{d_i}	Topic for the i^{th} word in document d
	$\boldsymbol{\theta}$	Document-topic distributions as a $M \times T$ matrix
	$\boldsymbol{\theta}_d$	Document-topic distribution for document d, entries in $\boldsymbol{\theta}$
	$\theta_{d,t}$	Probability of a topic t given document d
LF-DMM	$\boldsymbol{z}$	Topic assignments of all documents in the document collection D
	z_d	Topic for entire document d
	$\boldsymbol{\theta}$	Topic distribution for the document collection D as a T-dimensional vector
	θ_t	Probability of a topic t, entries in $\boldsymbol{\theta}$

Table 3.12: Terminology explanations. "General" denotes the common notations used for both LF-LDA and LF-DMM models.

Stats.	Description
Δ	Dirichlet delta function
Γ	Gamma function
$N_d^{t,w}$	Number of times a word type with vocabulary index w in document d is generated from topic t by the Dirichlet multinomial component (Dir-Multi)
N_d^w	Number of times a word type with vocabulary index w in document d is generated by Dir-Multi: $N_d^w = \sum_{t=1}^{T} N_d^{t,w}$
N_d^t	Number of word tokens in document d that are generated from topic t by Dir-Multi: $N_d^t = \sum_{w=1}^{V} N_d^{t,w}$
$\mathbf{N}_d^*$	T-dimensional Dir-Multi document-topic count vector: $\mathbf{N}_d^* = \{N_d^t\}_{t=1}^T$
N_d	Number of word tokens in document d that are generated by Dir-Multi: $N_d = \sum_{w=1}^{V} N_d^w = \sum_{t=1}^{T} N_d^t$
$N^{t,w}$	Number of times a word type with vocabulary index w in the document collection D is generated from topic t by Dir-Multi: $N^{t,w} = \sum_{d=1}^{M} N_d^{t,w}$
$\mathbf{N}^{t,\bullet}$	V-dimensional Dir-Multi topic-word count vector: $\mathbf{N}^{t,\bullet} = \{N^{t,w}\}_{w=1}^V$
N^t	Number of word tokens in the document collection D that are generated from topic t by Dir-Multi: $N^t = \sum_{w=1}^{V} N^{t,w} = \sum_{d=1}^{M} N_d^t$
$K_d^{t,w}$	Number of times a word type with vocabulary index w in document d is generated from topic t by the latent feature (LF) component
K_d^w	Number of times a word type with vocabulary index w in document d is generated by the LF component: $K_d^w = \sum_{t=1}^{T} K_d^{t,w}$
K_d^t	Number of word tokens in document d that are generated from topic t by the LF component: $K_d^t = \sum_{w=1}^{V} K_d^{t,w}$
$\mathbf{K}_d^*$	T-dimensional LF document-topic count vector: $\mathbf{K}_d^* = \{K_d^t\}_{t=1}^T$
K_d	Number of word tokens in document d that are generated by the LF component: $K_d = \sum_{w=1}^{V} K_d^w = \sum_{t=1}^{T} K_d^t$
$K^{t,w}$	Number of times a word type with index w in the document collection D is generated from topic t by the LF component: $K^{t,w} = \sum_{d=1}^{M} K_d^{t,w}$
K^t	Number of word tokens in the document collection D that are generated from topic t by the LF component: $K^t = \sum_{w=1}^{V} K^{t,w} = \sum_{d=1}^{M} K_d^t$
M^t	Number of documents assigned to topic t, so the total number of documents: $M = \sum_{t=1}^{T} M^t$
$\mathbf{M}^*$	T-dimensional vector of topic observation counts for the document collection D: $\mathbf{M}^* = \{M^t\}_{t=1}^T$

Table 3.13: Statistics (Stats.) notations. The first 17 rows describe notations used for both LF-LDA and LF-DMM while the last 2 rows present notations used only for LF-DMM.

Chapter 4

STransE: a novel embedding model of entities and relationships

Contents

4.1 Introduction 76
4.2 The embedding model STransE 78
4.3 Link prediction evaluation 81
4.3.1 Task and evaluation protocol 81
4.3.2 Main results 82
4.4 Application for search personalization 85
4.4.1 Motivation 85
4.4.2 A new embedding approach for search personalization 88
4.4.3 Experimental methodology 91
4.4.4 Experimental results 93
4.5 Summary 93

The work we presented in Chapter 3 can be also viewed as a combination of different embedding vector space models together. Unlike Chapter 3 which focuses on topic models, this chapter combines insights from several previous embedding models for *link prediction* or *knowledge base completion* into a new embedding model called *STransE*. Our STransE represents each entity as a low-dimensional vector, and each relation by two matrices and a translation vector. STransE thus is a simple combination of the SE and TransE models, but it obtains better link prediction performance—especially for Many-to-1, 1-to-Many and Many-to-Many relationships—on two benchmark datasets than previous

embedding models. Thus, STransE can serve as a new baseline for the more complex models in knowledge base completion. Furthermore, we also present an experiment of applying STransE for a new task of search personalization. Portions of the work presented in this chapter has been published in Nguyen et al. (2016b) and Vu et al. (2017a).[1]

4.1 Introduction

Knowledge bases (KBs) of real-world facts, such as WordNet (Fellbaum, 1998), YAGO (Suchanek et al., 2007), Freebase (Bollacker et al., 2008) and DBpedia (Lehmann et al., 2015), represent relationships between entities as triples (head entity, relation, tail entity). Thus knowledge bases are useful resources for a variety of NLP tasks. However, because knowledge bases are typically incomplete (Socher et al., 2013b; West et al., 2014), it is useful to be able to perform *link prediction* or *knowledge base completion*, i.e., predict whether a relationship not in the knowledge base is likely to be true (Taskar et al., 2004; Bordes et al., 2011; Nickel et al., 2016a). A variety of different kinds of information is potentially useful here, including information extracted from external corpora (Riedel et al., 2013; Wang et al., 2014b) or from a Web-search-based question answering system (West et al., 2014) and other relationships that hold between the entities (Angeli and Manning, 2013; Zhao et al., 2015a). For example, Toutanova et al. (2015) used information from the external ClueWeb-12 corpus to significantly enhance performance.

While integrating a wide variety of information sources can produce excellent results (Toutanova and Chen, 2015; Wang and Li, 2016; Das et al., 2017; Xiao et al., 2017), there are several reasons for studying simpler models that directly optimize a score function for the triples in a knowledge base, such as the one presented here. First, additional information sources might not be available, e.g., for knowledge bases for specialized domains. Second, models that do not exploit external resources are simpler and thus typically much faster to train than the more complex models using additional information. Third, complex models

[1]The first two authors Thanh Vu and Dat Quoc Nguyen contributed **equally** to the work published in Vu et al. (2017a).

that exploit external information are typically extensions of simpler models, and are often initialized with parameters estimated by such simpler models, so improvements to simpler models should yield corresponding improvements to complex models as well.

Embedding models for KB completion associate entities and/or relations with dense feature vectors or matrices. Such models obtain state-of-the-art performance (Nickel et al., 2011; Bordes et al., 2011, 2012, 2013; Socher et al., 2013b; Wang et al., 2014a; Lin et al., 2015b) and generalize to large KBs (Krompaß et al., 2015). See Section 2.5 for a summary of prominent embedding models for KB completion.

Let (h, r, t) represent a triple. In all of the models discussed here, the head entity h and the tail entity t are represented by vectors $\boldsymbol{v}_h$ and $\boldsymbol{v}_t \in \mathbb{R}^k$ respectively. The *Unstructured* model (Bordes et al., 2012) assumes that $\boldsymbol{v}_h \approx \boldsymbol{v}_t$, i.e., $\boldsymbol{v}_h$ is approximately equal to $\boldsymbol{v}_t$. As the Unstructured model does not take the relationship r into account, it cannot distinguish different relation types. The *Structured Embedding* (SE) model (Bordes et al., 2011) extends the unstructured model by assuming that h and t are similar only in a relation-dependent subspace. It represents each relation r with two matrices $\mathbf{W}_{r,1}$ and $\mathbf{W}_{r,2} \in \mathbb{R}^{k \times k}$, which are chosen so that $\mathbf{W}_{r,1}\boldsymbol{v}_h \approx \mathbf{W}_{r,2}\boldsymbol{v}_t$. The *TransE* model (Bordes et al., 2013) is inspired by models such as Word2Vec Skip-gram (Mikolov et al., 2013a,b) where relationships between words often correspond to translations in latent feature space. The TransE model represents each relation r by a translation vector $\boldsymbol{v}_r \in \mathbb{R}^k$, which is chosen so that $\boldsymbol{v}_h + \boldsymbol{v}_r \approx \boldsymbol{v}_t$.

The primary contribution of this work is to demonstrate that two simple link prediction models, SE and TransE, can be combined into a single model, which we call *STransE* (Nguyen et al., 2016b). Specifically, we use relation-specific matrices $\mathbf{W}_{r,1}$ and $\mathbf{W}_{r,2}$ as in the SE model to identify the relation-dependent aspects of both h and t, and use a vector $\boldsymbol{v}_r$ as in the TransE model to describe the relationship between h and t in this subspace. Specifically, our new KB completion model STransE chooses $\mathbf{W}_{r,1}$, $\mathbf{W}_{r,2}$ and $\boldsymbol{v}_r$ so that $\mathbf{W}_{r,1}\boldsymbol{v}_h + \boldsymbol{v}_r \approx \mathbf{W}_{r,2}\boldsymbol{v}_t$. That is, a TransE-style relationship holds in some relation-dependent subspace, and crucially, this subspace may involve very different projections of the head h and tail t. So $\mathbf{W}_{r,1}$ and $\mathbf{W}_{r,2}$ can highlight, suppress, or even change the sign

of, relation-specific attributes of h and t. For example, for the "purchases" relationship, certain attributes of individuals h (e.g., age, gender, marital status) are presumably strongly correlated with very different attributes of objects t (e.g., sports car, washing machine and the like). As we show below, STransE performs better than the SE and TransE models and other state-of-the-art KB completion models on two standard link prediction datasets WN18 and FB15k. More specifically, the highest improvements of STransE over the baselines are for complex Many-to-1, 1-to-Many and Many-to-Many relationships. So, STransE can serve as a new baseline for KB completion. We expect that the STransE will also be able to serve as the basis for extended models that exploit a wider variety of information sources, just as TransE does.

In addition, we also present an experiment which applies STransE to improve a search personalization task (Vu et al., 2017a). In this application task, we would observe a lot of Many-to-Many user-oriented relationships between submitted queries and clicked documents. So, STransE is particularly well-suited for this task. Previous work has shown that the performance of search personalization depends on the richness of user profiles which normally represent the user's topical interests (Teevan et al., 2005). We thus propose a new embedding approach to apply STransE for learning user profiles, where users are embedded on a topical interest space. We then directly utilize the user profiles for search personalization. Experiments on query logs from a major Web search engine demonstrate that our embedding approach improves the performance of the search engine and also achieves better search performance than other strong baselines.

4.2 The embedding model STransE

Let $\mathcal{E}$ denote the set of entities and $\mathcal{R}$ the set of relation types. For each triple (h, r, t), where $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$, the STransE model defines a *score function* $f(h, r, t)$ of its implausibility. Our goal is to choose f such that the score $f(h, r, t)$ of a plausible triple (h, r, t) is smaller than the score $f(h', r', t')$ of an implausible triple (h', r', t'). We define the STransE score function f as follows:

$$f(h,r,t) \;=\; \|\mathbf{W}_{r,1}\boldsymbol{v}_h + \boldsymbol{v}_r - \mathbf{W}_{r,2}\boldsymbol{v}_t\|_{\ell_{1/2}} \tag{4.1}$$

using either the ℓ_1 or the ℓ_2-norm (the choice is made using validation data; in our experiments we found that the ℓ_1 norm gave slightly better results). To learn the vectors and matrices we minimize the following margin-based objective function:

$$\mathcal{L} \;=\; \sum_{\substack{(h,r,t)\in\mathcal{G} \\ (h',r,t')\in\mathcal{G}'_{(h,r,t)}}} \max\Big(0, \gamma + f(h,r,t) - f(h',r,t')\Big) \tag{4.2}$$

where γ is the margin hyper-parameter, $\mathcal{G}$ is the training set consisting of correct triples, and:

$$\mathcal{G}'_{(h,r,t)} = \{(h',r,t) \mid h' \in \mathcal{E}, (h',r,t) \notin \mathcal{G}\} \cup \{(h,r,t') \mid t' \in \mathcal{E}, (h,r,t') \notin \mathcal{G}\} \tag{4.3}$$

is the set of incorrect triples generated by corrupting the head entity or the tail entity in a correct triple $(h,r,t) \in \mathcal{G}$.

We use Stochastic Gradient Descent (SGD) to minimize $\mathcal{L}$,[2] and impose the following constraints during training: $\|\boldsymbol{v}_h\|_2 \leqslant 1$, $\|\boldsymbol{v}_r\|_2 \leqslant 1$, $\|\boldsymbol{v}_t\|_2 \leqslant 1$, $\|\mathbf{W}_{r,1}\boldsymbol{v}_h\|_2 \leqslant 1$ and $\|\mathbf{W}_{r,2}\boldsymbol{v}_t\|_2 \leqslant 1$. Following previous research (Lin et al., 2015b; Ji et al., 2015; García-Durán et al., 2015; Lin et al., 2015a; García-Durán et al., 2016; Yoon et al., 2016; Ji et al., 2016), we use the entity and relation vectors produced by TransE (Bordes et al., 2013) to initialize the entity and relation vectors in STransE. That is, as shown in Algorithm 6, we first fix the relation matrices as identity matrices and only optimize the entity and relation vectors (i.e., STransE now reduces to the plain TransE). Then, in the second step, we learn all model parameters of vectors and matrices jointly. In all experiments presented in Section 4.3, we train for 2000 epochs during each of the two optimization steps.

We apply the "*Bernoulli* trick" used also in previous work to set different probabilities for generating head or tail entities when sampling incorrect triples (Wang et al., 2014a; Lin et al., 2015b; Ji et al., 2015; Yoon et al., 2016; Ji et al., 2016; Xiao et al., 2017). More

[2]See Section 2.2.1 for a brief introduction to SGD.

Algorithm 6: Parameter optimization for STransE

```
// Randomly initialize entity and relation vectors as in TransE (Bordes
   et al., 2013), and set relation matrices to be the identity matrices.
```

for $e \in \mathcal{E}$ **do**
  $\boldsymbol{v}_e \leftarrow$ **uniform**$(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$

for $r \in \mathcal{R}$ **do**
  $\boldsymbol{v}_r \leftarrow$ **uniform**$(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$
  $\mathbf{W}_{r,1} = \mathbf{I}$
  $\mathbf{W}_{r,2} = \mathbf{I}$

// First optimization step: learning TransE for STransE initialization

for *epoch i = 1, 2, ..., 2000* **do**
  for $(h, r, t) \in \mathcal{G}$ **do**
    $(h', r, t') \leftarrow$ **sample**$(\mathcal{G}'_{(h,r,t)})$ // Sample a corrupted triple
    Update entity and relation vectors w.r.t: $\nabla \max\Big(0, \gamma + f(h, r, t) - f(h', r, t')\Big)$
    Normalize the updated vectors to length 1 if their length is larger than 1.

// Second optimization step

for *epoch i = 1, 2, ..., 2000* **do**
  for $(h, r, t) \in \mathcal{G}$ **do**
    $(h', r, t') \leftarrow$ **sample**$(\mathcal{G}'_{(h,r,t)})$
    Update entity and relation vectors, and relation matrices w.r.t:
      $\nabla \max\Big(0, \gamma + f(h, r, t) - f(h', r, t')\Big)$
    Normalize the updated vectors to length 1 if their length is larger than 1.
    Update entity vectors and relation matrices w.r.t:
      $\nabla\Big(\max(0, \|\mathbf{W}_{r,1}\boldsymbol{v}_h\|_2^2 - 1) + \max(0, \|\mathbf{W}_{r,1}\boldsymbol{v}_{h'}\|_2^2 - 1) + \max(0, \|\mathbf{W}_{r,2}\boldsymbol{v}_t\|_2^2 - 1)$
        $+ \max(0, \|\mathbf{W}_{r,2}\boldsymbol{v}_{t'}\|_2^2 - 1)\Big)$

specifically, for each relation type r, we calculate the averaged number $a_{r,1}$ of heads h for a pair (r, t) and the averaged number $a_{r,2}$ of tails t for a pair (h, r). We then define a Bernoulli distribution with success probability $\lambda_r = \dfrac{a_{r,1}}{a_{r,1} + a_{r,2}}$ for sampling: given a correct triple (h, r, t), we corrupt this triple by replacing head entity with probability λ_r while replacing the tail entity with probability $(1 - \lambda_r)$.

Dataset	#E	#R	#Train	#Valid	#Test
WN18	40,943	18	141,442	5,000	5,000
FB15k	14,951	1,345	483,142	50,000	59,071

Table 4.1: Statistics of the experimental datasets used in this study (and previous works). #E is the number of entities, #R is the number of relation types, and #Train, #Valid and #Test are the numbers of correct triples in training, validation and test sets, respectively.

4.3 Link prediction evaluation

We conduct experiments and compare the performance of our STransE model with published results on the benchmark datasets WN18 and FB15k (Bordes et al., 2013). WN18 is derived from the large lexical KB WordNet (Miller, 1995) involving 18 relation types. FB15k is derived from the large real-world fact KB FreeBase (Bollacker et al., 2008) covering about 15k entities. Information about these datasets is given in Table 4.1.

4.3.1 Task and evaluation protocol

The link prediction task, which is also referred to as the *entity prediction* task, predicts the head or tail entity given the relation type and the other entity, i.e., predicting h given $(?, r, t)$ or predicting t given $(h, r, ?)$ where ? denotes the missing element (Bordes et al., 2011, 2012, 2013). The results are evaluated using the ranking induced by the score function $f(h, r, t)$ on test triples.

Each correct test triple (h, r, t) is corrupted by replacing either its head or tail entity by each of the possible entities in turn, and then we rank these candidates in ascending order of their implausibility score. This is called as the "Raw" setting protocol. For the "Filtered" setting protocol described in Bordes et al. (2013), we also *filtered* out before ranking any corrupted triples that appear in the KB. Ranking a corrupted triple appearing in the KB (i.e., a correct triple) higher than the original test triple is also correct, but is penalized by the "Raw" score, thus the "Filtered" setting provides a clearer view on the ranking performance.

In addition to the mean rank and the Hits@10 (i.e., the proportion of test triples for

which the target entity was ranked in the top 10 predictions), which were originally used in this task (Bordes et al., 2013), we also report the mean reciprocal rank (**MRR**), which is commonly used in information retrieval.[3] In both "Raw" and "Filtered" settings, mean rank is always greater than or equal to 1 and lower mean rank indicates better entity prediction performance. The MRR and Hits@10 scores always range from 0.0 to 1.0, and higher score reflects better prediction result.

Following Bordes et al. (2013), we used a grid search on the validation set to choose either the l_1 or l_2 norm in the score function f, as well as to set the SGD learning rate $\eta \in \{0.0001, 0.0005, 0.001, 0.005, 0.01\}$, the margin hyper-parameter $\gamma \in \{1, 3, 5\}$ and the vector size $k \in \{50, 100\}$. The *lowest filtered mean rank* on the validation set was obtained when using the l_1 norm in f on both WN18 and FB15k, and when $\eta = 0.0005, \gamma = 5,$ and $k = 50$ for WN18, and $\eta = 0.0001, \gamma = 1,$ and $k = 100$ for FB15k.

4.3.2 Main results

Table 4.2 compares the link prediction results of our STransE model with results reported in prior work, using the same experimental setup. The first 19 rows report the performance of the models that do not exploit information about alternative paths between head and tail entities. The next 5 rows report results of the models that exploit information about relation paths. The last 3 rows present results for the models which make use of textual mentions derived from a large external corpus.

It is clear that the models with the additional external corpus information obtained best results. In future work we plan to extend the STransE model to incorporate such additional information. Table 4.2 also shows that the models employing path information generally achieve better results than models that do not use such information. In terms of models not exploiting path information or external information, the STransE model produces the highest filtered mean rank on WN18 and the second highest filtered Hits@10 and mean reciprocal rank on FB15k. Compared to the closely related models SE, TransE,

[3]See Baeza-Yates and Ribeiro-Neto (2011) for definitions of the mean rank, Hits@10, and MRR.

Method	Raw						Filtered					
	WN18			FB15k			WN18			FB15k		
	MR	H@10	MRR	MR	H@10	MRR	MR	H@10	MRR	MR	H@10	MRR
SE (Bordes et al., 2011)	1011	68.5	-	273	28.8	-	985	80.5	-	162	39.8	-
Unstructured (Bordes et al., 2012)	315	35.3	-	1074	4.5	-	304	38.2	-	979	6.3	-
SME (Bordes et al., 2012)	545	65.1	-	274	30.7	-	533	74.1	-	154	40.8	-
TransH (Wang et al., 2014a)	401	73.0	-	212	45.7	-	303	86.7	-	87	64.4	-
TransR (Lin et al., 2015b)	238	79.8	-	198	48.2	-	225	92.0	-	77	68.7	-
CTransR (Lin et al., 2015b)	231	79.4	-	199	48.4	-	218	92.3	-	75	70.2	-
KG2E (He et al., 2015)	342	80.2	-	**174**	48.9	-	331	92.8	-	59	74.0	-
TransD (Ji et al., 2015)	224	79.6	-	194	53.4	-	212	92.2	-	91	77.3	-
lppTransD (Yoon et al., 2016)	283	80.5	-	195	53.0	-	270	94.3	-	78	78.7	-
TranSparse (Ji et al., 2016)	223	80.1	-	187	**53.5**	-	211	93.2	-	82	79.5	-
TATEC (García-Durán et al., 2016)	-	-	-	-	-	-	-	-	-	**58**	76.7	-
NTN (Socher et al., 2013b)	-	-	-	-	-	-	-	66.1	0.53	-	41.4	0.25
DISTMULT (Yang et al., 2015)	-	-	-	-	-	-	-	94.2	0.83	-	57.7	0.35
ComplEx (Trouillon et al., 2016)	-	-	0.587	-	-	0.242	-	94.7	**0.941**	-	**84.0**	**0.692**
HolE (Nickel et al., 2016b)	-	-	**0.616**	-	-	0.232	-	**94.9**	0.938	-	73.9	0.524
RESCAL (Nickel et al., 2011) [*]	-	-	0.603	-	-	0.189	-	92.8	0.890	-	58.7	0.354
TransE (Bordes et al., 2013) [*]	-	-	0.351	-	-	0.222	-	94.3	0.495	-	74.9	0.463
TransE [our]	264	79.3	0.375	220	49.0	0.226	250	92.5	0.525	70	74.6	0.465
Our STransE model	**217**	**80.9**	0.469	219	51.6	**0.252**	**206**	93.4	0.657	69	79.7	0.543
RTransE (García-Durán et al., 2015)	-	-	-	-	-	-	-	-	-	**50**	76.2	-
PTransE (Lin et al., 2015a)	-	-	-	207	51.4	-	-	-	-	58	84.6	-
GAKE (Feng et al., 2016b)	-	-	-	228	44.5	-	-	-	-	119	64.8	-
Gaifman (Niepert, 2016)	-	-	-	-	-	-	352	93.9	-	75	84.2	-
Hiri (Liu et al., 2016)	-	-	-	-	-	-	-	90.8	0.691	-	70.3	0.603
NLFeat (Toutanova and Chen, 2015)	-	-	-	-	-	-	-	**94.3**	**0.940**	-	**87.0**	**0.822**
TEKE_H (Wang and Li, 2016)	**127**	80.3	-	212	51.2	-	**114**	92.9	-	108	73.0	-
SSP (Xiao et al., 2017)	168	**81.2**	-	**163**	**57.2**	-	156	93.2	-	82	79.0	-

Table 4.2: Link prediction results. MR and H@10 denote evaluation metrics of mean rank and Hits@10 (in %), respectively. "NLFeat" abbreviates Node+LinkFeat. The results for NTN (Socher et al., 2013b) listed in this table are taken from Yang et al. (2015) since NTN was originally evaluated on different datasets. [*]: Results from the implementation of Nickel et al. (2016b) because these results are higher than those previously published in Bordes et al. (2013). [our]: We also report the baseline TransE's results (in tenth row from bottom), where we set the relation matrices to identity matrices and only learn the entity and relation vectors, i.e., STransE reduces to the plain TransE.

Method	Predicting head h				Predicting tail t			
	1-1	1-M	M-1	M-M	1-1	1-M	M-1	M-M
SE	35.6	62.6	17.2	37.5	34.9	14.6	68.3	41.3
Unstructured	34.5	2.5	6.1	6.6	34.3	4.2	1.9	6.6
SME	35.1	53.7	19.0	40.3	32.7	14.9	61.6	43.3
TransH	66.8	87.6	28.7	64.5	65.5	39.8	83.3	67.2
TransR	78.8	89.2	34.1	69.2	79.2	37.4	90.4	72.1
CTransR	81.5	89.0	34.7	71.2	80.8	38.6	90.1	73.8
KG2E	**92.3**	94.6	**66.0**	69.6	**92.6**	**67.9**	94.4	73.4
TATEC	79.3	93.2	42.3	77.2	78.5	51.5	92.7	80.7
TransD	86.1	**95.5**	39.8	78.5	85.4	50.6	94.4	81.2
lppTransD	86.0	94.2	54.4	**82.2**	79.7	43.2	**95.3**	79.7
TranSparse	86.8	**95.5**	44.3	80.9	86.6	56.6	94.4	**83.3**
TransE [our]	80.2	85.0	47.7	75.9	80.2	53.5	82.4	78.2
STransE	82.8	94.2	50.4	80.1	82.4	56.9	93.4	83.1
Improve.	2.6	9.2	2.7	4.2	2.2	3.4	11.0	4.9

Table 4.3: Hits@10 (in %) for each relation category on the FB15k dataset. The "*Improve.*" row denotes the absolute improvement accounted for STransE over the baseline TransE.

TransR, CTransR, TransD and TranSparse, our STransE model does better than these models on both WN18 and FB15k.[4]

Following Bordes et al. (2013), Table 4.3 analyzes Hits@10 results on FB15k with respect to the relation categories defined as follows: for each relation type r, we computed the averaged number $a_{r,1}$ of heads h for a pair (r, t) and the averaged number $a_{r,2}$ of tails t for a pair (h, r). If $a_{r,1} < 1.5$ and $a_{r,2} < 1.5$, then r is labeled **1-1** (i.e. a 1-to-1 relationship). If $a_{r,1} \geq 1.5$ and $a_{r,2} < 1.5$, then r is labeled **M-1** (i.e. a Many-to-1 relationship). If $a_{r,1} < 1.5$ and $a_{r,2} \geq 1.5$, then r is labeled as **1-M** (i.e. a 1-to-Many relationship). If $a_{r,1} \geq 1.5$ and $a_{r,2} \geq 1.5$, then r is labeled as **M-M** (i.e. a Many-to-Many relationship). 1.4%, 8.9%, 14.6% and 75.1% of the test triples belong to a relation type classified as **1-1**, **1-M**, **M-1** and **M-M**, respectively.

Table 4.3 shows that in comparison to closely related models not using path information, STransE obtains the second highest Hits@10 result for **M-M** relation category at (80.1% +

[4]In fact, as shown in Table 4.2, TransE obtains a very competitive link prediction performance. A similar observation was also made by García-Durán et al. (2015), Lin et al. (2015a), García-Durán et al. (2016) and Nickel et al. (2016b). The reason is probably due to a careful grid search.

83.1%)/2 = 81.6% which is 0.5% smaller than the Hits@10 result of TranSparse for **M-M**. However, STransE obtains 2.5% higher Hits@10 result than TranSparse for **M-1**. In addition, STransE also performs better than TransD for **1-M** and **M-1** relation categories. We believe the improved performance of the STransE model is due to its use of full matrices, rather than just projection vectors as in TransD. This permits STransE to model diverse and complex relation categories (such as **1-M**, **M-1** and especially **M-M**) better than TransD and other similar models. However, STransE is not as good as TransD for the **1-1** relations. Perhaps the extra parameters in STransE hurt performance in this case (note that 1-1 relations are relatively rare, so STransE does better overall).

4.4 Application for search personalization

4.4.1 Motivation

Using a Web search engine such as Google[5] or Bing[6] is straightforward (Teevan et al., 2010): a user types an input query which consists of a few words into the search engine's search box, and then the search engine returns to the user a list of documents. However, given the same input query, different users with different search interests might desire different information from the search engine. For example, Figure 4.1 shows the top-4 documents returned by a search engine for the query "acl 2017." Given this query, a music fan would select either the first or third document which is about the **ACL** music festival 2017, while a research student working on natural language processing would click to the second document which is about the 55th annual meeting of the **A**ssociation for **C**omputational **L**inguistics, whereas a football fan would click to the fourth document which is about the 2017 **A**FC **C**hampions **L**eague. Here, the football fan might expect to get results about the 2017 AFC Champions League in the first places rather than in the fourth. So identifying the user's search interest plays a crucial role in Web search engines.

Personalized search engines thus utilize users' personal data, such as their historical

[5] `https://www.google.com`
[6] `https://www.bing.com/`

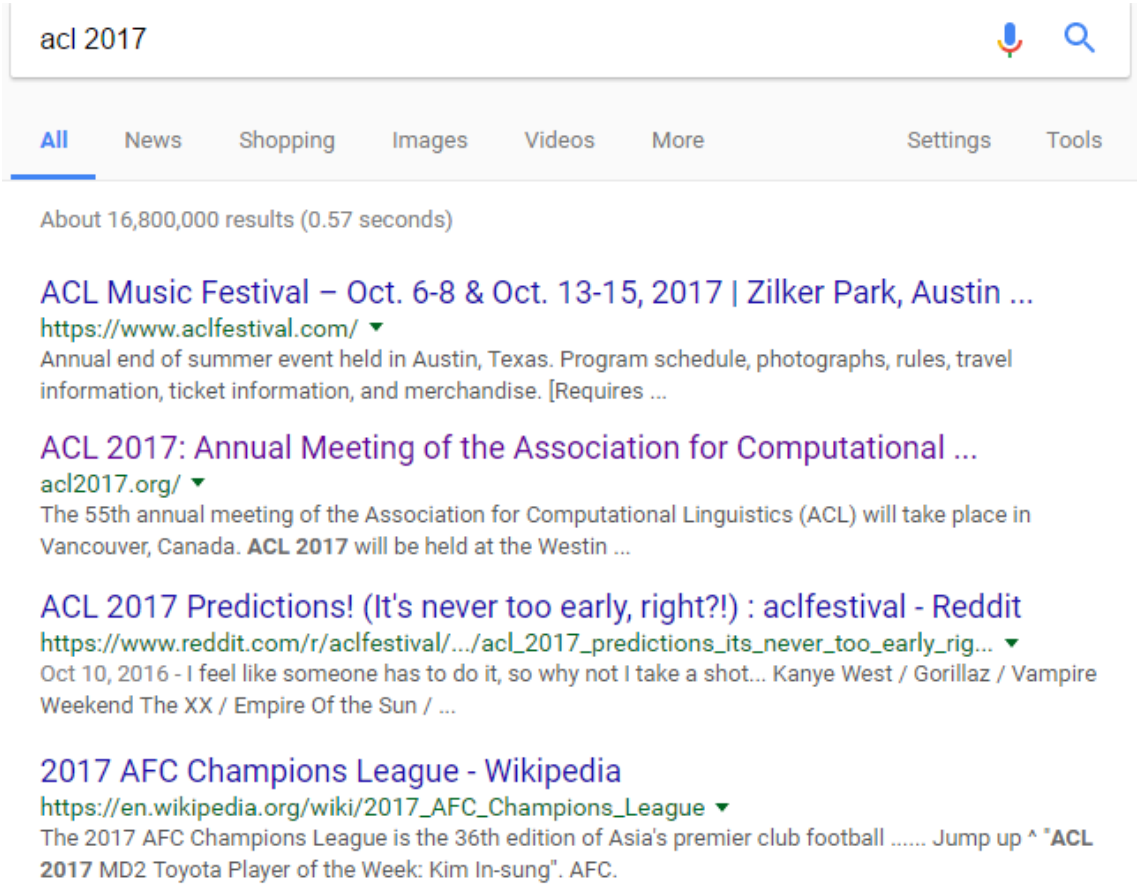


Figure 4.1: Top-4 search results for query "acl 2017."

interaction with the search engine (e.g., submitted queries, clicked documents), to satisfy the users' needs with better search results (Teevan et al., 2005; Dou et al., 2007; Teevan et al., 2009; Hassan and White, 2013; Shokouhi et al., 2013; Vu et al., 2014; Ustinovskiy et al., 2015; White and Awadallah, 2015; Salehi et al., 2015; Yang et al., 2016; Lofgren et al., 2016). Crucial to effective search personalization is the construction of user profiles to represent individual users' interests (Sieg et al., 2007; Bennett et al., 2012; Harvey et al., 2013; Liu, 2015; Cheng et al., 2016). A common approach is to use topical interests expressed in the user's clicked documents, which can be obtained by using a human generated ontology (Bennett et al., 2012; White et al., 2013; Yan et al., 2014) or using a topic modeling technique (Harvey et al., 2011, 2013; Vu et al., 2015b; Cheng et al., 2016).

However, using the user profile to directly personalize a search has been not very

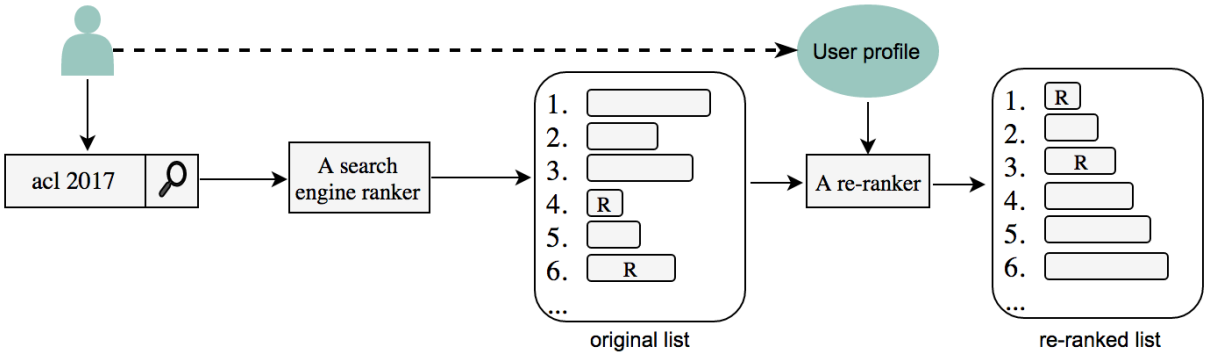


Figure 4.2: An illustration of the re-ranking process. "R" denotes that the document is relevant to the user.

successful. For example, Harvey et al. (2013) and Vu et al. (2014) obtained minor overall improvements when using user profiles. The reason is that each user profile is often constructed using only the user's relevant documents (e.g., clicked documents), ignoring user interest-dependent information related to input queries. Alternatively, the user profile is utilized as a feature of a multi-feature learning-to-rank framework (Bennett et al., 2012; White et al., 2013; Vu et al., 2015b, 2017b). Apart from the user profile, dozens of other features have been proposed as the input of a learning-to-rank algorithm (Bennett et al., 2012), so the contribution of the user profile to final ranking performance is not very clear.

To handle these problems, we propose a new *embedding* approach which applies the STransE model to constructing user profiles. We represent each user profile using two projection matrices and a user embedding. The two projection matrices are to identify the user interest-dependent aspects of input queries and relevant documents (e.g., clicked documents), while the user embedding is to capture the relationship between the queries and documents in this user interest-dependent subspace. We then *directly* utilize the user profile to re-rank the search results returned by a search engine, as illustrated in Figure 4.2. Experiments on the query logs of a Web search engine demonstrate that modeling user profile with embeddings helps to significantly improve the performance of the search engine and also achieve better results than other competitive baselines (Teevan et al., 2011; Bennett et al., 2012; Vu et al., 2015b) do.

4.4.2 A new embedding approach for search personalization

We start with our new embedding approach to building user profiles in Section 4.4.2.1, using pre-fixed document embeddings and query embeddings. We then detail the processes of using the LDA topic model (Blei et al., 2003) to learn document embeddings and query embeddings in sections 4.4.2.2 and 4.4.2.3, respectively. We finally use the user profiles to personalize original search results returned by a search engine in Section 4.4.2.4.

4.4.2.1 Building user profiles with embeddings

Let $\mathcal{Q}$ denote the set of queries, $\mathcal{U}$ be the set of users, and $\mathcal{D}$ be the set of documents. Let (q, u, d) represent a triple $(\mathsf{query}, \mathsf{user}, \mathsf{document})$. The query $q \in \mathcal{Q}$, user $u \in \mathcal{U}$ and document $d \in \mathcal{D}$ are represented by vector embeddings $\boldsymbol{v}_q$, $\boldsymbol{v}_u$ and $\boldsymbol{v}_d \in \mathbb{R}^k$, respectively.

Our goal is to select a *score function* f such that the implausibility value $f(q, u, d)$ of a correct triple (q, u, d), where d is a relevant document of u given q, is *smaller* than the implausibility value $f(q', u', d')$ of an incorrect triple (q', u', d') where d' is not a relevant document of u' given q'. Inspired by the embedding models of entities and relationships in knowledge bases, especially the STransE model, the score function f thus is similarly defined following STransE's as follows:

$$f(q, u, d) = \|\mathbf{W}_{u,1}\boldsymbol{v}_q + \boldsymbol{v}_u - \mathbf{W}_{u,2}\boldsymbol{v}_d\|_{\ell_{1/2}} \tag{4.4}$$

here we represent the profile for the user u by two matrices $\mathbf{W}_{u,1}$ and $\mathbf{W}_{u,2} \in \mathbb{R}^{k\times k}$ and a vector embedding $\boldsymbol{v}_u$, which represents the user's topical interests. Specifically, we use the interest-specific matrices $\mathbf{W}_{u,1}$ and $\mathbf{W}_{u,2}$ to identify the interest-dependent aspects of both query q and document d, and use vector $\boldsymbol{v}_u$ to describe the relationship between q and d in this interest-dependent subspace.

Here we pre-determine $\boldsymbol{v}_d$ and $\boldsymbol{v}_q$ by employing the LDA topic model. Our model parameters are only the user embeddings $\boldsymbol{v}_u$ and matrices $\mathbf{W}_{u,1}$ and $\mathbf{W}_{u,2}$. To learn these user embeddings and matrices, we minimize the margin-based objective function:

$$\mathcal{L} = \sum_{\substack{(q,u,d)\in\mathcal{G} \\ (q',u,d')\in\mathcal{G}'_{(q,u,d)}}} \max\big(0, \gamma + f(q,u,d) - f(q',u,d')\big) \tag{4.5}$$

where γ is the margin hyper-parameter, $\mathcal{G}$ is the training set that contains only correct triples, and $\mathcal{G}'_{(q,u,d)}$ is the set of incorrect triples generated by corrupting the correct triple (q,u,d), i.e., we replace the relevant document/query d/q in (q,u,d) by irrelevant documents/queries d'/q', respectively. We use SGD to minimize $\mathcal{L}$ with the following constraints during training: $\|\boldsymbol{v}_u\|_2 \leqslant 1$, $\|\mathbf{W}_{u,1}\boldsymbol{v}_q\|_2 \leqslant 1$ and $\|\mathbf{W}_{u,2}\boldsymbol{v}_d\|_2 \leqslant 1$. First, we initialize user matrices as identity matrices and then fix them to only learn the randomly initialized user embeddings. Then in the next step, we fine-tune the user embeddings and user matrices together. In all experiments shown in Section 4.4.3, we train for 200 epochs during each of the two optimization steps.

4.4.2.2 Using LDA to learn document embeddings

We model document embeddings by using topics extracted from relevant documents. We use the LDA topic model (Blei et al., 2003) to *automatically* learn k topics from the *relevant* document collection. After training an LDA model to estimate the probability distribution over topics for each document, we use the topic proportion vector of each document as its document embedding. Specifically, the z^{th} element $(z = 1, 2, ..., k)$ of the vector embedding for document d is:

$$\boldsymbol{v}_{d,z} = \mathrm{P}(z \mid d) \tag{4.6}$$

where $\mathrm{P}(z \mid d)$ is the probability of the topic z given d.

4.4.2.3 Modeling search queries with embeddings

We also represent each query as a probability distribution $\boldsymbol{v}_q$ over topics, i.e., the z^{th} element of the vector embedding for query q is defined as:

$$\boldsymbol{v}_{q,z} = \mathrm{P}(z \mid q) \tag{4.7}$$

where $\mathrm{P}(z \mid q)$ is the probability of the topic z given the query q. One approach is to directly apply a pre-trained LDA model to infer topic distributions for new documents (Phan et al., 2011). That is, we consider queries as new documents, thus applying the LDA model trained on the collection of relevant documents to infer $\mathrm{P}(z \mid q)$. However, the queries are too short compared to the documents, so in this approach the topic distribution for each query might not be informative.

Following Bennett et al. (2012) and Vu et al. (2015b), we define $\mathrm{P}(z \mid q)$ as a mixture of LDA topic probabilities of z given documents related to q. Let $\mathcal{D}_q = \{d_1, d_2, ..., d_n\}$ be the set of top-n ranked documents returned for a query q (in the experiments, we select $n = 10$). We define $\mathrm{P}(z \mid q)$ as follows:

$$\mathrm{P}(z \mid q) = \sum\nolimits_{i=1}^{n} \lambda_i \mathrm{P}(z \mid d_i) \tag{4.8}$$

where $\lambda_i = \dfrac{\delta^{i-1}}{\sum_{j=1}^{n} \delta^{j-1}}$ is the exponential decay function of i which is the rank of d_i in $\mathcal{D}_q$, and δ is the decay hyper-parameter ($0 < \delta < 1$). The decay function is to specify the fact that a higher ranked document is more relevant to user in term of the lexical matching, i.e., we here set the larger mixture weights to higher ranked documents.

4.4.2.4 Personalizing search results

As illustrated in Figure 4.2, we utilize the user profiles (i.e., the learned user embeddings and matrices) to re-rank the original list of documents produced by a search engine for user u with query q as follows:

1. We download top-n ranked documents given the input query q. For each downloaded document d, we apply the trained LDA model to infer topic distribution $\boldsymbol{v}_d$.

2. We then model the query q as a topic distribution $\boldsymbol{v}_q$ as in Section 4.4.2.3.

3. For each triple (q, u, d), we calculate the implausibility value $f(q, u, d)$ as defined in Equation 4.4. We then sort the values of n triples (q, u, d) in the ascending order to obtain a new ranked list.

4.4.3 Experimental methodology

Dataset: We evaluate our new approach using the search results returned by the Bing search engine. We use a dataset of query logs of 106 anonymous users in 15 days from 01 July 2012 to 15 July 2012 (Vu et al., 2014, 2015a). A log entity contains a user identifier, a query, top-10 URLs ranked by the search engine, and clicked URLs along with the user's dwell time. We also download the content documents of these URLs for training LDA (Blei et al., 2003) to learn document and query embeddings.

Bennett et al. (2012) indicated that short-term (i.e., session) profiles achieved better search performance than the longer-term profiles. Short-term profiles are usually constructed using the user's search interactions within a search session and used to personalize the search within the session (Bennett et al., 2012). To identify a search session, we use 30 minutes of user inactivity to demarcate the session boundary. In our experiments, we build short-term profiles and utilize the profiles to personalize the returned results. Specifically, we uniformly separate the last log entries within search sessions into a *test set* and a *validation set*. The remainder of log entities within search sessions are used for *training* (e.g., to learn user embeddings and matrices in our approach).

Evaluation methodology: We use the SAT criteria detailed in Fox et al. (2005) to identify whether a clicked URL is relevant from the query logs (i.e., a SAT click). That is either a click with a dwell time of at least 30 seconds or the last result click in a search session. We assign a positive (relevant) label to a returned URL if it is a SAT click. The remainder of the top-10 URLs is assigned negative (irrelevant) labels. We use the rank positions of the positive labeled URLs as the ground truth to evaluate the search performance before and after re-ranking. We also apply a simple pre-processing on these datasets as follows. At first, we remove the queries whose positive label set is empty from

#days	#users	#distinct queries	#SAT clicks	#sessions	#distinct documents
15	106	6,632	8,052	2,394	33,591

Table 4.4: Basic statistics of the dataset after pre-processing. # denotes "number of."

the dataset. After that, we discard the domain-related queries (e.g., Facebook, YouTube). To this end, we have total 8,052 correct triples in which the training set consists of 5,658 correct triples, while the test and validation sets contain 1,210 and 1,184 correct triples, respectively. Table 4.4 presents the dataset statistics after pre-processing.

Evaluation metrics: We use two standard evaluation metrics in document ranking (Manning et al., 2008; Bennett et al., 2012): *mean reciprocal rank* (**MRR**) and *precision* (**Hits@1**).[7] For each metric, the higher value indicates the better ranking performance.

Hyper-parameter tuning: We perform a grid search to select optimal hyper-parameters on the validation set. We train the LDA model using only the relevant documents (i.e., SAT clicks) extracted from the query logs, with the number of topics (i.e., the number of vector dimensions) $k \in \{50, 100, 200\}$. We apply the trained LDA model to infer document embeddings (i.e. topic distributions) for all documents, and then calculate query embeddings for all queries. We then choose either the ℓ_1 or ℓ_2 norm in the score function f, and select SGD learning rate $\eta \in \{0.001, 0.005, 0.01\}$, the margin hyper-parameter $\gamma \in \{1, 3, 5\}$ and the decay hyper-parameter $\delta \in \{0.7, 0.8, 0.9\}$. The highest MRR on the validation set is obtained when using $k = 200$, ℓ_1 in f, $\eta = 0.005$, $\gamma = 5$, and $\delta = 0.8$.

Baselines: We employ three comparative baselines with the same experimental setup: (1) **SE**: The main baseline is the original rank from the search engine (2) **CI**: We promote returned documents previously clicked by the user. This baseline is similar to the personalized navigation method in Teevan et al. (2011). (3) **L2R-SP**: The learning-to-rank method for search personalization uses short-term user profile as an extra feature (Bennett et al., 2012; Vu et al., 2015b). These are strong baselines given that they start with the

[7]We re-rank the list of top-10 documents returned by the search engine, so Hits@10 scores are the same for all baselines and our approach.

ranking provided by the search engine and add other signals (e.g., clicked documents) to get a better ranking result (Teevan et al., 2011; Bennett et al., 2012; Vu et al., 2015b).

4.4.4 Experimental results

Table 4.5 shows the performances of the baselines and our proposed method. Using the previously clicked documents **CI** helps to improve the search performance with the relative improvements of about $7^{+}\%$ in both MRR and Hits@1 metrics. With the use of short-term user profile as a feature in a multi-feature learning-to-rank framework, **L2R-SP** (Bennett et al., 2012; Vu et al., 2015b) achieves better scores than **CI**'s.

Metric	**SE**	**CI**	**L2R-SP**	**Our method**	**Our method**$_{-W}$
MRR	0.559	$0.597_{+6.9\%}$	$0.631_{+12.9\%}$	$\mathbf{0.656}_{+17.3\%}$	$0.645_{+15.4\%}$
Hits@1	0.385	$0.416_{+8.1\%}$	$0.452_{+17.4\%}$	$\mathbf{0.501}_{+30.3\%}$	$0.481_{+24.9\%}$

Table 4.5: Overall performances of the methods in the test set. **Our method**$_{-W}$ denotes the simplified version of our method. The subscripts denote the relative improvement over the baseline **SE**.

By directly learning user profiles and applying them to re-rank the search results, our embedding approach achieves the highest performance of search personalization. Specifically, our MRR and Hits@1 scores are higher than those of **L2R-SP**, with the relative improvements over **L2R-SP** at 4% for MRR and 11% for Hits@1. In Table 4.5, we also present the performances of a simplified version of our embedding approach where we fix the user matrices as identity matrices and then only learn the user vectors. Table 4.5 shows that the simplified version achieves the second highest scores compared to all others.

4.5 Summary

In this chapter, we presented a new triple-based embedding model for KB completion. Our STransE model combines insights from several simpler embedding models, in particular the Structured Embedding model (Bordes et al., 2011) and the TransE model (Bordes et al., 2013), by using a low-dimensional vector embedding and two projection matrices

to represent each relation. STransE, while being conceptually simple, produces highly competitive results on standard link prediction evaluations, and scores better than the embedding-based models it builds on. In particular, we find the highest improvements of STransE are for the complex relationships Many-to-1, 1-to-Many and Many-to-Many. Thus STransE is a suitable candidate for serving as future baseline for more complex models in the link prediction task.

We also presented a new embedding approach for a search personalization task by applying STransE to build user profiles which represent user topical interests. We model each user profile using a user embedding together with two user matrices. The user embedding and matrices are then learned using LDA-based vector embeddings of the user's submitted queries and returned documents. Applying it to Web search, we use the profile to re-rank search results returned by a Web search engine. Our experimental results show that the proposed method can significantly improve the ranking quality.

Chapter 5

Neighborhood mixture model for knowledge base completion

Contents

5.1 Introduction 96
5.2 Neighborhood mixture model 98
5.2.1 Neighbor-based entity representation 98
5.2.2 TransE-NMM: applying neighborhood mixtures to TransE . . . 99
5.2.3 Parameter optimization 101
5.3 Experiments 102
5.3.1 Datasets 103
5.3.2 Evaluation tasks 103
5.3.3 Hyper-parameter tuning 104
5.4 Results 106
5.4.1 Quantitative results 106
5.4.2 Qualitative results 108
5.4.3 Discussion 109
5.5 Summary 111

We define a novel entity representation as a mixture of its neighborhood in the knowledge base (KB) and apply this technique to TransE—a well-known embedding model for KB completion described in Section 2.5 and the basis for the work of Chapter 4. Experiments show that the neighborhood information significantly helps improve the results of TransE, leading to better performance than obtained by other state-of-the-art embedding models on three benchmark datasets for triple classification, entity prediction and relation prediction tasks. The work presented in this chapter has been published in Nguyen et al. (2016a).

5.1 Introduction

As noted in Chapter 4, embedding models for KB completion represent entities and/or relations with dense feature vectors or matrices, and obtain state-of-the-art performance (Bordes et al., 2013; Socher et al., 2013b; Wang et al., 2014a; Guu et al., 2015; Nguyen et al., 2016b). See Section 2.5 for a brief overview of embedding models for KB completion. Most embedding models for KB completion learn only from triples and by doing so, ignore lots of information implicitly provided by the structure of the knowledge graph. Recently, several authors have addressed this issue by incorporating relation path information into model learning (García-Durán et al., 2015; Lin et al., 2015a; Guu et al., 2015; Toutanova et al., 2016) and have shown that the relation paths between entities in KBs provide useful information and improve knowledge base completion. For instance, a three-relation path:

$$(\mathsf{Harrison\_Ford}, \mathbf{born\_in\_hospital}/\mathsf{r}_1, \mathsf{Swedish\_Covenant\_Hospital})$$
$$(\mathsf{Swedish\_Covenant\_Hospital}, \mathbf{hospital\_located\_in\_city}/\mathsf{r}_2, \mathsf{Chicago})$$
$$(\mathsf{Chicago}, \mathbf{city\_in\_country}/\mathsf{r}_3, \mathsf{United\_States})$$

is likely to indicate that the fact $(\mathsf{Harrison\_Ford}, \mathsf{nationality}, \mathsf{United\_States})$ could be true, so the relation path here $\mathsf{p} = \{\mathsf{r}_1, \mathsf{r}_2, \mathsf{r}_3\}$ is useful for predicting the relationship "$\mathsf{nationality}$" between the head entity "$\mathsf{Harrison\_Ford}$" and the tail entity "$\mathsf{United\_States}$."

Besides the relation paths, there could be other useful information implicitly presented in the knowledge base that could be exploited for better KB completion. For instance, the whole neighborhood of entities could provide lots of useful information for predicting the relationship between two entities. Consider for example a KB fragment given in Figure 5.1. If we know that Ben Affleck has won an Oscar award and Ben Affleck lives in Los Angeles, then this can help us to predict that Ben Affleck is an actor or a film maker, rather than a lecturer or a doctor. If we additionally know that Ben Affleck's gender is male then there is a higher chance for him to be a film maker, given that film makers are more likely to be male. This intuition can be formalized by representing an entity vector as a

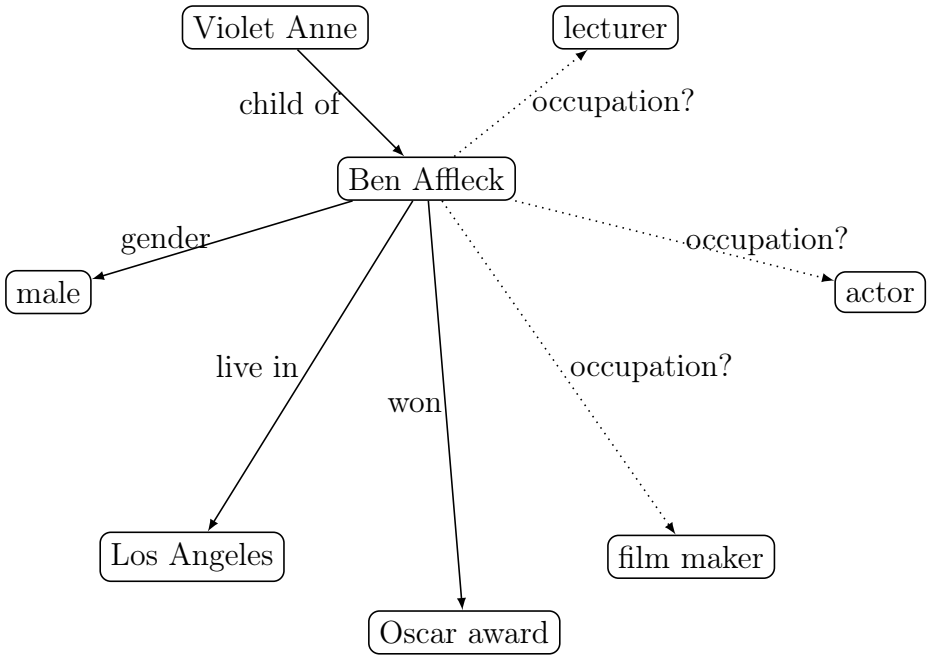


Figure 5.1: An example fragment of a KB.

relation-specific mixture of its neighborhood as follows:

$$\begin{aligned} \mathsf{Ben\_Affleck} &= \omega_{r,1}(\mathsf{Violet\_Anne}, \mathsf{child\_of}) \\ &+ \omega_{r,2}(\mathsf{male}, \mathsf{gender}^{-1}) \\ &+ \omega_{r,3}(\mathsf{Los\_Angeles}, \mathsf{lives\_in}^{-1}) \\ &+ \omega_{r,4}(\mathsf{Oscar\_award}, \mathsf{won}^{-1}), \end{aligned}$$

where $\omega_{r,i}$ are the mixing weights that indicate how important each neighboring relation is for predicting the relation r, and the subscript $^{-1}$ means the incoming edge. For example, for predicting the "occupation" relationship, the knowledge about the "child_of" relationship might not be that informative and thus the corresponding mixing coefficient can be close to zero, whereas it could be relevant for predicting some other relationships, such as "parent" or "spouse", in which case the relation-specific mixing coefficient for the "child_of" relationship could be high.

Our primary contribution is introducing and formalizing the neighborhood mixture

model. We demonstrate its usefulness by applying it to the well-known TransE model (Bordes et al., 2013). However, it could be applied to other embedding models as well, such as Bilinear models (Bordes et al., 2012; Yang et al., 2015) and STransE (see Chapter 4). While relation path models exploit extra information using longer paths existing in the KB, the neighborhood mixture model effectively incorporates information about many paths simultaneously. Our extensive experiments on three benchmark datasets show that it achieves superior performance over competitive baselines in three KB completion tasks: triple classification, entity prediction and relation prediction.

5.2 Neighborhood mixture model

In this section, we start by explaining how to formally construct the neighbor-based entity representations in section 5.2.1, and then describe the **N**eighborhood **M**ixture **M**odel applied to the TransE model (Bordes et al., 2013) in section 5.2.2. Section 5.2.3 explains how we train our model.

5.2.1 Neighbor-based entity representation

Let $\mathcal{E}$ denote the set of entities and $\mathcal{R}$ the set of relation types. Denote by $\mathcal{R}^{-1}$ the set of inverse relations r^{-1}. Denote by $\mathcal{G}$ the knowledge graph consisting of a set of correct triples (h, r, t), such that $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$. Let $\mathcal{K}$ denote the symmetric closure of $\mathcal{G}$, i.e., if a triple $(h, r, t) \in \mathcal{G}$, then both (h, r, t) and $(t, r^{-1}, h) \in \mathcal{K}$.

We define:

$$\mathcal{N}_{e,r} = \{e' | (e', r, e) \in \mathcal{K}\} \tag{5.1}$$

as a set of neighboring entities connected to entity e with relation r. Then

$$\mathcal{N}_e = \{(e', r) | r \in \mathcal{R} \cup \mathcal{R}^{-1}, e' \in \mathcal{N}_{e,r}\} \tag{5.2}$$

is the set of all entity and relation pairs that are neighbors for entity e.

Each entity e is associated with a k-dimensional vector $\boldsymbol{v}_e \in \mathbb{R}^k$ and relation-dependent vectors $\boldsymbol{u}_{e,r} \in \mathbb{R}^k, r \in \mathcal{R} \cup \mathcal{R}^{-1}$. Now we can define the neighborhood-based entity representation $\boldsymbol{\vartheta}_{e,r}$ for an entity $e \in \mathcal{E}$ for predicting the relation $r \in \mathcal{R}$ as follows:

$$\boldsymbol{\vartheta}_{e,r} = a_e \boldsymbol{v}_e + \sum_{(e',r') \in \mathcal{N}_e} b_{r,r'} \boldsymbol{u}_{e',r'}, \tag{5.3}$$

a_e and $b_{r,r'}$ are the mixture weights that are constrained to sum to 1 for each neighborhood:

$$a_e \propto \delta + \exp \alpha_e \tag{5.4}$$

$$b_{r,r'} \propto \exp \beta_{r,r'} \tag{5.5}$$

where $\delta \geqslant 0$ is a hyper-parameter that controls the contribution of the entity vector $\boldsymbol{v}_e$ to the neighbor-based mixture, α_e and $\beta_{r,r'}$ are the learnable exponential mixture parameters.

In real-world factual KBs, e.g., Freebase (Bollacker et al., 2008), some entities, such as "male", can have thousands or millions neighboring entities sharing the same "gender" relation. For such entities, computing the neighbor-based vectors can be computationally very expensive. To overcome this problem, we introduce in our implementation a filtering threshold τ and consider in the neighbor-based entity representation construction only those relation-specific neighboring entity sets for which $|\mathcal{N}_{e,r}| \leq \tau$.

5.2.2 TransE-NMM: applying neighborhood mixtures to TransE

Embedding models define for each triple $(h, r, t) \in \mathcal{G}$, a *score function* $f(h, r, t)$ that measures its implausibility. The goal is to choose f such that the score $f(h, r, t)$ of a plausible triple (h, r, t) is smaller than the score $f(h', r', t')$ of an implausible triple (h', r', t').

TransE (Bordes et al., 2013) is a simple embedding model for KB completion, which, despite of its simplicity, obtains very competitive results (García-Durán et al., 2016; Nickel et al., 2016b). In TransE, both entities e and relations r are represented with k-dimensional vectors $\boldsymbol{v}_e \in \mathbb{R}^k$ and $\boldsymbol{v}_r \in \mathbb{R}^k$, respectively. These vectors are chosen such that for each triple $(h, r, t) \in \mathcal{G}$, we have: $\boldsymbol{v}_h + \boldsymbol{v}_r \approx \boldsymbol{v}_t$. The score function of the TransE model is the

norm of this translation:

$$f(h,r,t)_{\mathsf{TransE}} = \|\boldsymbol{v}_h + \boldsymbol{v}_r - \boldsymbol{v}_t\|_{\ell_{1/2}} \tag{5.6}$$

We define the score function of our new model TransE-NMM in terms of the neighbor-based entity vectors as follows:

$$f(h,r,t) = \|\boldsymbol{\vartheta}_{h,r} + \boldsymbol{v}_r - \boldsymbol{\vartheta}_{t,r^{-1}}\|_{\ell_{1/2}} \tag{5.7}$$

using either the ℓ_1 or the ℓ_2-norm, and $\boldsymbol{\vartheta}_{h,r}$ and $\boldsymbol{\vartheta}_{t,r^{-1}}$ are defined following the Equation 5.3. The relation-specific entity vectors $\boldsymbol{u}_{e,r}$ used to construct the neighbor-based entity vectors $\boldsymbol{\vartheta}_{e,r}$ are defined based on the TransE translation operator:

$$\boldsymbol{u}_{e,r} = \boldsymbol{v}_e + \boldsymbol{v}_r \tag{5.8}$$

$$\boldsymbol{v}_{r^{-1}} = -\boldsymbol{v}_r \tag{5.9}$$

$$\boldsymbol{u}_{e,r^{-1}} = \boldsymbol{v}_e - \boldsymbol{v}_r \tag{5.10}$$

For each correct triple (h,r,t), the sets of neighboring entities $\mathcal{N}_{h,r}$ and $\mathcal{N}_{t,r^{-1}}$ exclude the entities t and h, respectively. If we set the filtering threshold $\tau = 0$ then $\boldsymbol{\vartheta}_{h,r} = \boldsymbol{v}_h$ and $\boldsymbol{\vartheta}_{t,r^{-1}} = \boldsymbol{v}_t$ for all triples. In this case, TransE-NMM reduces to the plain TransE model. In all our experiments presented in Section 5.3, the baseline TransE results are obtained with the TransE-NMM with $\tau = 0$.

Our TransE-NMM model can be also viewed as a three-relation path model because it uses the neighborhood entity and relation information of both head and tail entities in each triple. Furthermore, our neighborhood mixture model can be adapted to other state-of-the-art embedding models for KB completion which are discussed in Section 2.5. For example, it can be adapted to relation path models Bilinear-COMP and TransE-COMP (Guu et al., 2015), by replacing head and tail entity vectors by the neighbor-based vector representations, thus combining advantages of both path and neighborhood information.

5.2.3 Parameter optimization

The TransE-NMM model parameters include the vectors $\boldsymbol{v}_e, \boldsymbol{v}_r$ for entities and relation types, the entity-specific weights $\boldsymbol{\alpha} = \{\alpha_e | e \in \mathcal{E}\}$ and relation-specific weights $\boldsymbol{\beta} = \{\beta_{r,r'} | r, r' \in \mathcal{R} \cup \mathcal{R}^{-1}\}$. To learn these parameters, we minimize the L_2-regularized margin-based objective function:

$$\mathcal{L} = \sum_{\substack{(h,r,t)\in\mathcal{G} \\ (h',r,t')\in\mathcal{G}'_{(h,r,t)}}} \max\Big(0, \gamma + f(h,r,t) - f(h',r,t')\Big) + \frac{\lambda}{2}\Big(\|\boldsymbol{\alpha}\|_2^2 + \|\boldsymbol{\beta}\|_2^2\Big), \tag{5.11}$$

where γ is the margin hyper-parameter, λ is the L_2 regularization parameter, and

$$\mathcal{G}'_{(h,r,t)} = \{(h', r, t) \mid h' \in \mathcal{E}, (h', r, t) \notin \mathcal{G}\} \cup \{(h, r, t') \mid t' \in \mathcal{E}, (h, r, t') \notin \mathcal{G}\}$$

is the set of incorrect triples generated by corrupting the correct triple $(h, r, t) \in \mathcal{G}$. We applied the "*Bernoulli* trick" to choose whether to generate the head or tail entity when sampling an incorrect triple (Wang et al., 2014a; Lin et al., 2015b; He et al., 2015; Ji et al., 2015, 2016). A description of the Bernoulli trick is previously presented in Section 4.2.

We use Stochastic Gradient Descent with RMSProp adaptive learning (Tieleman and Hinton, 2012) to minimize $\mathcal{L}$, and impose the following hard constraints during training: $\|\boldsymbol{v}_e\|_2 \leqslant 1$ and $\|\boldsymbol{v}_r\|_2 \leqslant 1$.[1] We employ alternating optimization to minimize $\mathcal{L}$. As shown in Algorithm 7, we first initialize the entity and relation-specific mixing parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ to zero and only learn the randomly initialized entity and relation vectors $\boldsymbol{v}_e$ and $\boldsymbol{v}_r$. Then we fix the learned vectors and only optimize the mixing parameters. In the final step, we fix again the mixing parameters and fine-tune the vectors. In all experiments presented in Section 5.3, we train for 200 epochs during each of the three optimization steps.

[1] See Section 2.2.3 for a brief introduction to RMSProp.

Algorithm 7: Parameter optimization for TransE-NMM

```
// Randomly initialize entity and relation vectors as in TransE (Bordes
   et al., 2013), and set entity and relation-specific weights to be zero.
```

for $e \in \mathcal{E}$ **do**
  $\boldsymbol{v}_e \leftarrow$ **uniform**$(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$

for $r \in \mathcal{R}$ **do**
  $\boldsymbol{v}_r \leftarrow$ **uniform**$(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$

// First optimization step

for *epoch i = 1, 2, ..., 200* **do**
  for $(h, r, t) \in \mathcal{G}$ **do**
    $(h', r, t') \leftarrow$ **sample**$(\mathcal{G}'_{(h,r,t)})$ // Sample a corrupted triple
    Update entity and relation vectors w.r.t: $\nabla \max\Big(0, \gamma + f(h, r, t) - f(h', r, t')\Big)$
    Normalize the updated vectors to length 1 if their length is larger than 1.

// Second optimization step

for *epoch i = 1, 2, ..., 200* **do**
  for $(h, r, t) \in \mathcal{G}$ **do**
    $(h', r, t') \leftarrow$ **sample**$(\mathcal{G}'_{(h,r,t)})$
    Update entity and relation-specific weights w.r.t:
      $\nabla \max\Big(0, \gamma + f(h, r, t) - f(h', r, t')\Big) + \frac{\lambda}{2}\Big(\|\boldsymbol{\alpha}\|_2^2 + \|\boldsymbol{\beta}\|_2^2\Big)$

// Last optimization step

for *epoch i = 1, 2, ..., 200* **do**
  for $(h, r, t) \in \mathcal{G}$ **do**
    $(h', r, t') \leftarrow$ **sample**$(\mathcal{G}'_{(h,r,t)})$
    Update entity and relation vectors w.r.t: $\nabla \max\Big(0, \gamma + f(h, r, t) - f(h', r, t')\Big)$
    Normalize the updated vectors to length 1 if their length is larger than 1.

5.3 Experiments

To investigate the usefulness of the neighborhood mixtures, we compare the performance of our TransE-NMM model against the results of the baseline model TransE (Bordes et al., 2013) and other state-of-the-art embedding models on the triple classification, entity prediction and relation prediction tasks.

Dataset:	WN11	FB13	NELL186
#R	11	13	186
#E	38,696	75,043	14,463
#Train	112,581	316,232	31,134
#Valid	2,609	5,908	5,000
#Test	10,544	23,733	5,000

Table 5.1: Statistics of the experimental datasets used in this study (and *previous works*). #E is the number of entities, #R is the number of relation types, and #Train, #Valid and #Test are the numbers of correct triples in the training, validation and test sets, respectively. Each validation and test set also contains the same number of incorrect triples as the number of correct triples.

5.3.1 Datasets

We conduct experiments using three publicly available datasets WN11, FB13 and NELL186. For all of them, the validation and test sets containing both correct and incorrect triples have already been constructed. Table 5.1 gives the statistical information about these datasets. WN11 and FB13 were produced by Socher et al. (2013b) for triple classification.[2] WN11 is derived from the large lexical KB WordNet (Miller, 1995) involving 11 relation types. FB13 is derived from the large real-world fact KB FreeBase (Bollacker et al., 2008) covering 13 relation types. NELL186 was introduced by Guo et al. (2015) for both triple classification and entity prediction tasks,[3] containing the 186 most frequent relations in the KB NELL (Carlson et al., 2010).

5.3.2 Evaluation tasks

We evaluate TransE-NMM on three commonly used benchmark tasks: triple classification, entity prediction and relation prediction. This subsection describes those tasks in detail.

Triple classification: The triple classification task was first introduced by Socher et al. (2013b), and since then it has been used to evaluate various embedding models. The aim of the task is to predict whether a triple (h, r, t) is correct or not.

[2] `http://cs.stanford.edu/people/danqi/data/nips13-dataset.tar.bz2`
[3] `http://aclweb.org/anthology/attachments/P/P15/P15-1009.Datasets.zip`

For classification, we set a relation-specific threshold θ_r for each relation type r. If the implausibility score of an unseen test triple (h, r, t) is smaller than θ_r then the triple will be classified as correct, otherwise incorrect. Following Socher et al. (2013b), the relation-specific thresholds are determined by maximizing the micro-averaged accuracy, which is a per-triple average, on the validation set. We also report the macro-averaged accuracy, which is a per-relation average.

Entity prediction: The standard entity prediction task, which is also called the link prediction task (Bordes et al., 2013), predicts the head or the tail entity given the relation type and the other entity. The results are evaluated using a ranking induced by the function $f(h, r, t)$ on test triples. Note that the incorrect triples in the validation and test sets are not used for evaluating the entity prediction task nor the relation prediction task. As detailed in Section 4.3.1, for this entity prediction task, we use two settings "Raw" and "Filtered" with three ranking-based evaluation metrics: mean rank, Hits@10 and mean reciprocal rank (**MRR**). In both "Raw" and "Filtered" settings, lower mean rank, higher MRR or higher Hits@10 indicates better entity prediction performance.

Relation prediction: The relation prediction task (Lin et al., 2015a) predicts the relation type given the head and tail entities, i.e., predicting r given $(h, ?, t)$ where ? denotes the missing element. We corrupt each correct test triple (h, r, t) by replacing its relation r by each possible relation type in turn, and then rank these candidates in ascending order of their implausibility score. Just as in the entity prediction task, we use two setting protocols "Raw" and "Filtered", and evaluate on the mean rank, MRR and Hits@10 metrics.

5.3.3 Hyper-parameter tuning

For all evaluation tasks, experimental results for TransE are obtained with TransE-NMM with the filtering threshold $\tau = 0$ as mentioned in Section 5.2.2, while we set $\tau = 10$ for TransE-NMM.

For triple classification, we first performed a grid search to choose the optimal hyper-parameters for TransE by monitoring the micro-averaged triple classification accuracy after each training epoch on the validation set. For all datasets, we chose either the ℓ_1 or ℓ_2 norm in the score function f and the initial RMSProp learning rate $\eta \in \{0.001, 0.01\}$. Following the previous work (Wang et al., 2014a; Lin et al., 2015b; Ji et al., 2015; He et al., 2015; Ji et al., 2016), we selected the margin hyper-parameter $\gamma \in \{1, 2, 4\}$ and the number of vector dimensions $k \in \{20, 50, 100\}$ on WN11 and FB13. On NELL186, we set $\gamma = 1$ and $k = 50$ (Guo et al., 2015; Luo et al., 2015). The highest accuracy on the validation set was obtained when using $\eta = 0.01$ for all three datasets, and when using ℓ_2 norm for NELL186, $\gamma = 4$, $k = 20$ and ℓ_1 norm for WN11, and $\gamma = 1$, $k = 100$ and ℓ_2 norm for FB13.

We set the hyper-parameters η, γ, k, and the ℓ_1 or the ℓ_2-norm in our TransE-NMM model to the same optimal hyper-parameters searched for TransE. We then used a grid search to select the hyper-parameter $\delta \in \{0, 1, 5, 10\}$ and L_2 regularizer $\lambda \in \{0.005, 0.01, 0.05\}$ for TransE-NMM. By monitoring the micro-averaged accuracy after each training epoch, we obtained the highest accuracy on validation set when using $\delta = 1$ and $\lambda = 0.05$ for both WN11 and FB13, and $\delta = 0$ and $\lambda = 0.01$ for NELL186.

For both entity prediction and relation prediction tasks, we set the hyper-parameters η, γ, k, and the ℓ_1 or the ℓ_2-norm for both TransE and TransE-NMM to be the same as the optimal parameters found for the triple classification task. We then monitored on TransE the filtered MRR on validation set after each training epoch. We chose the model with highest validation MRR, which was then used to evaluate the test set. For TransE-NMM, we searched the hyper-parameter $\delta \in \{0, 1, 5, 10\}$ and L_2 regularizer $\lambda \in \{0.005, 0.01, 0.05\}$. By monitoring the filtered MRR after each training epoch, we selected the best model with the highest filtered MRR on the validation set. For the entity prediction task, we selected $\delta = 10$ and $\lambda = 0.005$ for WN11, $\delta = 5$ and $\lambda = 0.01$ for FB13, and $\delta = 5$ and $\lambda = 0.005$ for NELL186. For the relation prediction task, we selected $\delta = 10$ and $\lambda = 0.005$ for WN11, $\delta = 10$ and $\lambda = 0.05$ for FB13, and $\delta = 1$ and $\lambda = 0.05$ for NELL186.

Data	Method		Triple class.		Entity prediction			Relation prediction		
			Mic.	Mac.	MR	MRR	H@10	MR	MRR	H@10
WN11	R	TransE	85.21	82.53	4324	**0.102**	**19.21**	2.37	0.679	**99.93**
		TransE-NMM	**86.82**	**84.37**	**3687**	0.094	17.98	**2.14**	**0.687**	99.92
	F	TransE			4304	**0.122**	**21.86**	2.37	0.679	**99.93**
		TransE-NMM			**3668**	0.109	20.12	**2.14**	**0.687**	99.92
FB13	R	TransE	87.57	86.66	9037	0.204	35.39	1.01	0.996	99.99
		TransE-NMM	**88.58**	**87.99**	**8289**	**0.258**	**35.53**	1.01	0.996	**100.0**
	F	TransE			5600	0.213	36.28	1.01	0.996	99.99
		TransE-NMM			**5018**	**0.267**	**36.36**	1.01	0.996	**100.0**
NELL186	R	TransE	92.13	88.96	309	0.192	36.55	8.43	0.580	77.18
		TransE-NMM	**94.57**	**90.95**	**238**	**0.221**	**37.55**	**6.15**	**0.677**	**82.16**
	F	TransE			279	0.268	47.13	8.32	0.602	77.26
		TransE-NMM			**214**	**0.292**	**47.82**	**6.08**	**0.690**	**82.20**

Table 5.2: Experimental results of TransE (i.e., TransE-NMM with $\tau = 0$) and TransE-NMM with $\tau = 10$. Micro-averaged (labeled as **Mic.**) and Macro-averaged (labeled as **Mac.**) accuracy results are for the triple classification task. MR, MRR and H@10 abbreviate the mean rank, the mean reciprocal rank and Hits@10 (in %), respectively. "R" and "F" denote the "Raw" and "Filtered" settings used in the entity prediction and relation prediction tasks, respectively.

5.4 Results

5.4.1 Quantitative results

Table 5.2 presents the results of TransE and TransE-NMM on triple classification, entity prediction and relation prediction tasks on all experimental datasets. The results show that TransE-NMM generally performs better than TransE in all three evaluation tasks.

Specifically, TransE-NMM obtains higher triple classification results than TransE in all three experimental datasets, for example, with 2.44% absolute improvement in the micro-averaged accuracy on the NELL186 dataset (i.e., 31% reduction in error). In terms of entity prediction, TransE-NMM obtains better mean rank, MRR and Hits@10 scores than TransE on both FB13 and NELL186 datasets. Specifically, on NELL186 TransE-NMM gains a significant improvement of $279 - 214 = 65$ in the filtered mean rank (which is about 23% relative improvement), while on the FB13 dataset, TransE-NMM improves with $0.267 - 0.213 = 0.054$ in the filtered MRR (which is about 25% relative improvement).

Method	**W11**	**F13**	Avg.
TransR (Lin et al., 2015b)	85.9	82.5	84.2
CTransR (Lin et al., 2015b)	85.7	-	-
TransD (Ji et al., 2015)	86.4	**89.1**	**87.8**
TEKE_H (Wang and Li, 2016)	84.8	84.2	84.5
TranSparse-S (Ji et al., 2016)	86.4	88.2	87.3
TranSparse-US (Ji et al., 2016)	<u>86.8</u>	87.5	87.2
NTN (Socher et al., 2013b)	70.6	87.2	78.9
TransH (Wang et al., 2014a)	78.8	83.3	81.1
SLogAn (Liang and Forbus, 2015)	75.3	85.3	80.3
KG2E (He et al., 2015)	85.4	85.3	85.4
Bilinear-COMP (Guu et al., 2015)	77.6	86.1	81.9
TransE-COMP (Guu et al., 2015)	80.3	87.6	84.0
TransR-FT (Feng et al., 2016a)	86.6	82.9	84.8
TransG (Xiao et al., 2016)	**87.4**	87.3	87.4
lppTransD (Yoon et al., 2016)	86.2	<u>88.6</u>	87.4
TransE	85.2	87.6	86.4
TransE-NMM	<u>86.8</u>	<u>88.6</u>	<u>87.7</u>

Table 5.3: Micro-averaged accuracy results (in %) for triple classification on WN11 (labeled as **W11**) and FB13 (labeled as **F13**) test sets. Scores in **bold** and <u>underline</u> are the best and second best scores, respectively. "Avg." denotes the averaged accuracy.

On the WN11 dataset, TransE-NMM only achieves better mean rank for entity prediction. The relation prediction results of TransE-NMM and TransE are relatively similar on both WN11 and FB13 because the number of relation types is small in these two datasets. On NELL186, however, TransE-NMM does significantly better than TransE.

In Table 5.3, we compare the micro-averaged triple classification accuracy of our TransE-NMM model with the previously reported results on the WN11 and FB13 datasets. The first 6 rows report the performance of models that use TransE to initialize the entity and relation vectors. The last 11 rows present the accuracy of models with randomly initialized parameters. Table 5.3 shows that TransE-NMM obtains the second highest result on both WN11 and FB13. Note that there are higher results reported for NTN (Socher et al., 2013b), Bilinear-COMP and TransE-COMP (Guu et al., 2015) when entity vectors are initialized by averaging the pre-trained word vectors (Mikolov et al., 2013b; Pennington et al., 2014). It is not surprising as many entity names in WordNet and

Method	Triple class.		Entity pred.	
	Mic.	Mac.	MR	H@10
TransE-LLE	90.08	84.50	535	20.02
SME-LLE	93.64	89.39	253	37.14
SE-LLE	93.95	88.54	447	31.55
TransE-SkipG	85.33	80.06	385	30.52
SME-SkipG	92.86	89.65	293	**39.70**
SE-SkipG	93.07	87.98	412	31.12
TransE	92.13	88.96	309	36.55
TransE-NMM	**94.57**	**90.95**	**238**	37.55

Table 5.4: Results on on the NELL186 test set. Results for the entity prediction task are in the "Raw" setting. "-SkipG" abbreviates "-Skip-gram".

FreeBase are lexically meaningful. It is possible for all other embedding models to utilize the pre-trained word vectors as well. However, as pointed out by Wang et al. (2014a) and Guu et al. (2015), averaging the pre-trained word vectors for initializing entity vectors is an open problem and it is not always useful since entity names in many domain-specific KBs are not lexically meaningful.

Table 5.4 compares the accuracy for triple classification, the raw mean rank and raw Hits@10 scores for entity prediction on the NELL186 dataset. The first three rows present the best results reported in Guo et al. (2015), while the next three rows present the best results reported in Luo et al. (2015). TransE-NMM obtains the highest triple classification accuracy, the best raw mean rank and the second highest raw Hits@10 on the entity prediction task in this comparison.

5.4.2 Qualitative results

Table 5.5 presents some examples to illustrate the useful information modeled by the neighbors. We took the relation-specific mixture weights from the learned TransE-NMM model optimized on the entity prediction task, and then extracted three neighbor relations with the largest mixture weights given a relation. Table 5.5 shows that those relations are semantically coherent. For example, if we know the place of birth and/or the place of death of a person and/or the location where the person is living, it is likely that we can

Relation	Top 3-neighbor relations
has_instance (WN11)	type_of subordinate_instance_of domain_topic
synset_domain_topic (WN11)	domain_region member_holonym member_meronym
nationality (FB13)	place_of_birth place_of_death location
spouse (FB13)	children, spouse, parents
CEOof (NELL186)	WorksFor TopMemberOfOrganization PersonLeadsOrganization
AnimalDevelopDisease (NELL186)	AnimalSuchAsInsect AnimalThatFeedOnInsect AnimalDevelopDisease

Table 5.5: Qualitative examples.

predict the person's nationality. On the other hand, if we know that a person works for an organization and that this person is also the top member of that organization, then it is possible that this person is the CEO of that organization.

5.4.3 Discussion

Despite of the lower triple classification scores of TransE reported in Wang et al. (2014a), Table 5.3 shows that TransE in fact obtains a very competitive accuracy. Particularly, compared to the relation path model TransE-COMP (Guu et al., 2015), when model parameters were randomly initialized, TransE obtains $85.2-80.3 = 4.9\%$ absolute accuracy improvement on the WN11 dataset while achieving similar score on the FB13 dataset. Our high results of the TransE model are probably due to a careful grid search and using the *Bernoulli* trick. Note that Lin et al. (2015b), Ji et al. (2015) and Ji et al. (2016) did not report the TransE results used for initializing TransR, TransD and TranSparse, respectively. They directly copied the TransE results previously reported in Wang et al.

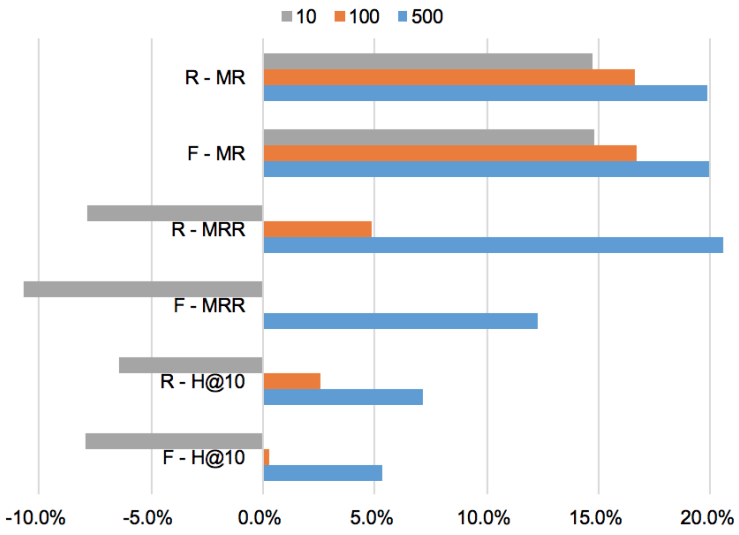


Figure 5.2: Relative improvement of TransE-NMM against TransE for entity prediction task in WN11 when the filtering threshold $\tau = \{10, 100, 500\}$ (with other hyper-parameters being the same as selected in Section 5.3.3). Prefixes "R-" and "F-" denote the "Raw" and "Filtered" settings, respectively. Suffixes "-MR", "-MRR" and "-H@10" abbreviate the mean rank, the mean reciprocal rank and Hits@10, respectively.

(2014a). So it is difficult to determine exactly how much TransR, TransD and TranSparse gain over TransE. These models might obtain better results than previously reported when the TransE used for initialization performs as well as reported in this chapter. Also, García-Durán et al. (2015), Lin et al. (2015a), García-Durán et al. (2016) and Nickel et al. (2016b) showed that for entity prediction TransE obtains very competitive results which are much higher than the TransE results originally published in Bordes et al. (2013).[4]

As presented in Table 5.2, for entity prediction using WN11, TransE-NMM with the filtering threshold $\tau = 10$ only obtains better mean rank than TransE (about 15% relative improvement) but lower Hits@10 and mean reciprocal rank. The reason might be that in semantic lexical KBs such as WordNet where relationships between words or word groups are manually constructed, whole neighborhood information might be useful. So when using a small filtering threshold, the model ignores a lot of potential information that could help predicting relationships. Figure 5.2 presents relative improvements in entity prediction of TransE-NMM over TransE on WN11 when varying the filtering threshold τ. Figure

[4]They did not report the results on WN11 and FB13 datasets, which are used in this chapter, though.

5.2 shows that TransE-NMM gains better scores with higher τ value. Specifically, when $\tau = 500$ TransE-NMM does better than TransE in all entity prediction metrics.

5.5 Summary

In this chapter, we introduced a neighborhood mixture model for knowledge base completion by constructing neighbor-based vector representations for entities. We demonstrated its effect by extending TransE (Bordes et al., 2013) with our neighborhood mixture model. On three different datasets, experimental results show that our model significantly improves TransE and generally obtains better results than the other state-of-the-art embedding models on triple classification, entity prediction and relation prediction tasks.

Chapter 6

Conclusion

To answer the research questions set out in Chapter 1, we presented two new probabilistic topic models, two new embedding models for KB completion, and a new embedding approach for personalized search in chapters 3, 4 and 5. In this chapter, we highlight our major findings and discuss possible future research directions.

6.1 Answers and Key findings

In Chapter 3, we showed that latent feature word vectors can be used to improve topic models. More specifically, we addressed the research question on developing new topic models that incorporate pre-trained word representations learned on a large corpus to improve topic inference on a smaller corpus:

RQ 1: *How can word vectors learned on a large external corpus be used to improve topic models estimated from a smaller corpus or from a corpus of short documents ?*

We proposed two new topic models LF-LDA and LF-DMM, that integrate latent feature representations of words within two baseline topic models: the LDA model for normal texts (Blei et al., 2003) and the DMM model for short texts (Nigam et al., 2000; Yin and Wang, 2014). We investigated the use of two well-known sets of pre-trained word vectors—Google Word2Vec (Mikolov et al., 2013b) and Stanford GloVe (Pennington et al., 2014)—with our

LF-LDA and LF-DMM models for three evaluation tasks on six experimental datasets. In topic coherence evaluation, our models performed significantly better the baseline topic models LDA and DMM on all experimental datasets. This confirmed that our approach for utilizing external information from very large corpora via pre-trained word vectors helps improve the topic-to-word mapping on smaller corpora. Also, document clustering and classification evaluations showed that LF-LDA and LF-DMM produced better document-topic assignments than the baseline models LDA and DMM did. In particular, we found the highest improvements on document clustering and classification results are for datasets with few or short documents. We also found that utilizing the pre-trained Google Word2Vec and Stanford Glove word vectors produced similar results in all three evaluation tasks.

In Chapter 4, one research question we addressed focused on developing a new embedding model for KB completion to deal with complex relationships:

RQ 2: *How can we develop a new embedding model for KB completion to better capture Many-to-1, 1-to-Many and Many-to-Many relationships ?*

We proposed a new triple-based embedding model, namely STransE, for link prediction in KBs. Our STransE model combines insights from the Structured Embedding (SE) model (Bordes et al., 2011) and the TransE model (Bordes et al., 2013) to represent each relation type by a low-dimensional vector embedding and two projection matrices. We found that STransE obtains better link prediction performances than SE, TransE and other closely related embedding models, especially for complex Many-to-1, 1-to-Many and Many-to-Many relationships. Thus, STransE can serve as a future baseline for more complex models.

In Chapter 4, we also addressed another research question of whether we could explore a new application task for a KB completion model:

RQ 4: *How can we adapt a KB completion model to some new application such as in Web search engines ?*

We explored a new application in the search personalization task, where we could make a connection between a topic model (e.g., LDA) and a KB completion model (e.g.,

STransE). Specifically, we proposed a new embedding approach for search personalization, by applying STransE to learn user profiles which represent user topical interests, i.e., to represent each user profile by a user vector embedding and two user matrices. Then we learned the user vector and matrices by utilizing LDA-based pre-determined vector representations of the user's submitted queries and returned documents. Next, we applied learned user profiles to re-rank search results produced by the Bing search engine. The experiments showed that our approach is able to significantly improve the ranking scores.

In Chapter 5, we answered the research question of whether, apart from the triples, we could make use of extra information from the structure of KBs:

RQ 3: *How can we develop a new embedding model using such useful information as relation path or neighborhood for better link prediction in KBs ?*

We introduced a neighborhood mixture model for KB completion, by formalizing neighbor-based vector representations for entities. While relation path models use extra information from longer paths existing in the KB, our neighborhood mixture model simultaneously incorporates information about many paths effectively. We demonstrated its effectiveness by applying it to the TransE model (Bordes et al., 2013). Our extensive experiments on three benchmark datasets showed that the neighborhood mixture model significantly improves TransE, resulting in better performances than other competitive embedding models in triple classification, entity prediction and relation prediction tasks.

6.2 Future work

Given the multi-faceted nature of the work described in this thesis, there are many directions for future research expansions.

Regarding topic modeling, it would be interesting to identify exactly how the latent feature word vectors improve topic modeling performance. We believe that they provide useful information about word meaning extracted from the large corpora that they are trained on, but it is possible that the performance improvements arise because the word

vectors are trained on context windows of size 5 or 10, while the LDA and DMM models view documents as bags of words, and effectively use a context window that encompasses the entire document. In preliminary experiments, where we train latent feature word vectors from the topic-modeling corpus alone using context windows of size 10, we found that performance degraded relative to the results presented here, suggesting that the use of a context window alone is not responsible for the performance improvements we reported here. Clearly, it would be valuable to investigate this further. In order to use a Gibbs sampler in Section 3.2.4, the conditional distributions needed to be distributions we can sample from cheaply, which is not the case for the ratios of Gamma functions. While we used a simple approximation, we can also explore other sampling techniques that can avoid approximations, such as Metropolis-Hastings sampling (Bishop, 2006, Section 11.2.2). Although we have not evaluated our approach on very large corpora, the corpora we have evaluated on do vary in size, and we showed that the gains from our approach are greatest when the corpora are small. A drawback of our approach is that it is slow on very large corpora. Variational Bayesian inference may provide an efficient solution to this problem (Jordan et al., 1999; Blei et al., 2003). Furthermore, we plan to tune the word vectors to fit the topic modeling corpora. We also aim to develop better latent feature models of topic-word distributions.

In term of KB completion, we plan to extend STransE to use relation path or neighborhood information in knowledge bases. One potential direction is to adapt our neighborhood mixture model to STransE by replacing the head and tail entity vectors by the neighbor-based vector representations. Especially, we plan to adapt our neighborhood mixture model to relation path models, such as TransE-COMP (Guu et al., 2015), to combine the useful information from both relation paths and entity neighborhoods. Furthermore, for the application in search personalization, as presented in Section 4.4, we employed LDA for inferring document and query embeddings, before modeling users. Instead of that two-step manner, we will represent document and query embeddings as bag-of-word vectors, and then directly learn them together with user embeddings and matrices.

6.3 Conclusion

There are many interesting research questions that remain to be addressed in understanding how representation learning can help improve topic models and knowledge base completion. We hope that this thesis serves its purpose by providing concrete answers to several specific questions and discussing possible research directions for future work.

Bibliography

Eneko Agirre, Oier López de Lacalle, and Aitor Soroa. Random Walks for Knowledge-Based Word Sense Disambiguation. *Computational Linguistics*, 40(1):57–84, 2013.

Galen Andrew and Jianfeng Gao. Scalable Training of L1-regularized Log-linear Models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 33–40, 2007.

Gabor Angeli and Christopher Manning. Philosophers are Mortal: Inferring the Truth of Unseen Facts. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 133–142, 2013.

Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. *Modern Information Retrieval - the concepts and technology behind search, Second edition.* Pearson Education Ltd., Harlow, England, 2011.

Arindam Banerjee, Inderjit S. Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the Unit Hypersphere Using Von Mises-Fisher Distributions. *Journal of Machine Learning Research*, 6:1345–1382, 2005.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, 2014.

Kayhan Batmanghelich, Ardavan Saeedi, Karthik Narasimhan, and Sam Gershman. Nonparametric Spherical Topic Modeling with Word Embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 537–542, 2016.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A Neural Probabilistic Language Model. *Journal of Artificial Intelligence Research*, 3:1137–1155, 2003.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

Paul N. Bennett, Ryen W. White, Wei Chu, Susan T. Dumais, Peter Bailey, Fedor Borisyuk, and Xiaoyuan Cui. Modeling the Impact of Short- and Long-term Behavior on Search Personalization. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 185–194, 2012.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, 2013.

Paulo Bicalho, Marcelo Pita, Gabriel Pedrosa, Anisio Lacerda, and Gisele L. Pappa. A General Framework to Expand Short Text for Topic Modeling. *Information Sciences*, 393(C):66–81, 2017.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

David M. Blei. Probabilistic Topic Models. *Communications of the ACM*, 55(4):77–84, 2012.

David M. Blei and John D. Lafferty. Dynamic Topic Models. In *Proceedings of the 23rd international conference on Machine learning*, pages 113–120. ACM, 2006.

David M. Blei and John D. Lafferty. A Correlated Topic Model of Science. *The Annals of Applied Statistics*, 1(1):17–35, 2007.

David M. Blei and Jon D. McAuliffe. Supervised Topic Models. In *Advances in Neural Information Processing Systems 20*, pages 121–128. MIT Press, 2008.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250, 2008.

Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. Learning Structured Embeddings of Knowledge Bases. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pages 301–306, 2011.

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A Semantic Matching Energy Function for Learning with Multi-relational Data. *Machine Learning*, 94(2): 233–259, 2012.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26*, pages 2787–2795. 2013.

Jordan Boyd-Graber and David M. Blei. Multilingual topic models for unaligned text. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 75–82, 2009.

John A. Bullinaria and Joseph P. Levy. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39(3): 510–526, 2007.

Deng Cai, Qiaozhu Mei, Jiawei Han, and Chengxiang Zhai. Modeling Hidden Topics on Document Manifold. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 911–920, 2008.

Ana Cardoso-Cachopo. Improving Methods for Single-label Text Categorization. PhD Thesis, Instituto Superior Tecnico, Universidade Tecnica de Lisboa, 2007.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, Jr., and Tom M. Mitchell. Toward an Architecture for Never-ending Language Learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 1306–1313, 2010.

J. Chang and David M. Blei. Hierarchical relational models for document networks. *Annals of Applied Statistics*, 4(1):124–150, 2010.

Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L. Boyd-graber, and David M. Blei. Reading Tea Leaves: How Humans Interpret Topic Models. In *Advances in Neural Information Processing Systems 22*, pages 288–296. 2009.

Danqi Chen and Christopher Manning. A Fast and Accurate Dependency Parser using Neural Networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 740–750, 2014.

Weizheng Chen, Jinpeng Wang, Yan Zhang, Hongfei Yan, and Xiaoming Li. User Based Aggregation for Biterm Topic Model. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 489–494, 2015.

Jianpeng Cheng, Zhongyuan Wang, Ji-Rong Wen, Jun Yan, and Zheng Chen. Contextual Text Understanding in Distributional Semantic Space. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 133–142, 2015.

Zhiyong Cheng, Shen Jialie, and Steven C.H. Hoi. On Effective Personalized Music Retrieval by Exploring Online User Behaviors. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 125–134, 2016.

Kenneth Ward Church and Patrick Hanks. Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistics*, 16(1):22–29, 1990.

Ronan Collobert and Jason Weston. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167, 2008.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural Language Processing (Almost) from Scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian LDA for Topic Models with Word Embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 795–804, 2015.

Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. Chains of reasoning over entities, relations, and text using recurrent neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141, 2017.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. *arXiv preprint*, abs/1707.01476, 2017.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge Vault: A Web-scale Approach to Probabilistic Knowledge Fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 601–610, 2014.

Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. A Large-scale Evaluation and Analysis of Personalized Search Strategies. In *Proceedings of the 16th International Conference on World Wide Web*, pages 581–590, 2007.

Lan Du, Wray Buntine, and Mark Johnson. Topic Segmentation with a Structured Topic Model. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 190–200, 2013.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *The Journal of Machine Learning Research*, 12: 2121–2159, 2011.

Sourav Dutta and Gerhard Weikum. Cross-Document Co-Reference Resolution using Sample-Based Clustering with Knowledge Enrichment. *Transactions of the Association for Computational Linguistics*, 3:15–28, 2015.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. Transition-Based Dependency Parsing with Stack Long Short-Term Memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 334–343, 2015.

Jacob Eisenstein, Amr Ahmed, and Eric Xing. Sparse Additive Generative Models of Text. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1041–1048, 2011.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open Question Answering over Curated and Extracted Knowledge Bases. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1156–1165, 2014.

Christiane D. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

Jun Feng, Minlie Huang, Mingdong Wang, Mantong Zhou, Yu Hao, and Xiaoyan Zhu. Knowledge Graph Embedding by Flexible Translation. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference*, pages 557–560, 2016a.

Jun Feng, Minlie Huang, Yang Yang, and Xiaoyan Zhu. GAKE: Graph Aware Knowledge Embedding. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 641–651, 2016b.

David Angelo Ferrucci. Introduction to "This is Watson". *IBM Journal of Research and Development*, 56(3):235–249, 2012.

John R. Firth. A Synopsis of Linguistic Theory, 1930-1955. *Studies in Linguistic Analysis*, pages 1–32, 1957.

Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan Dumais, and Thomas White. Evaluating Implicit Measures to Improve Web Search. *ACM Transactions on Information Systems*, 23(2):147–168, 2005.

Xianghua Fu, Ting Wang, Jing Li, Chong Yu, and Wangwang Liu. Improving Distributed Word Representation and Topic Model by Word-Topic Mixture Model. In *Proceedings of The 8th Asian Conference on Machine Learning*, pages 190–205, 2016.

Alberto García-Durán, Antoine Bordes, and Nicolas Usunier. Composing Relationships with Translations. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 286–290, 2015.

Alberto García-Durán, Antoine Bordes, Nicolas Usunier, and Yves Grandvalet. Combining Two and Three-Way Embedding Models for Link Prediction in Knowledge Bases. *Journal of Artificial Intelligence Research*, 55:715–742, 2016.

Matt Gardner and Tom Mitchell. Efficient and Expressive Knowledge Base Completion Using Subgraph Feature Extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1488–1498, 2015.

Matt Gardner, Partha P. Talukdar, Jayant Krishnamurthy, and Tom M. Mitchell. Incorporating Vector Space Similarity in Random Walk Inference over Knowledge Bases. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 397–406, 2014.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach. In *Proceedings of the 28th International Conference on Machine Learning*, pages 513–520, 2011.

Yoav Goldberg. A Primer on Neural Network Models for Natural Language Processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Gregory Goth. Deep or Shallow, NLP is Breaking out. *Communications of the ACM*, 59 (3):13–16, 2016.

Thomas L. Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1):5228–5235, 2004.

Shu Guo, Quan Wang, Bin Wang, Lihong Wang, and Li Guo. Semantically Smooth Knowledge Graph Embedding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 84–94, 2015.

Kelvin Guu, John Miller, and Percy Liang. Traversing Knowledge Graphs in Vector Space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 318–327, 2015.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA Data Mining Software: An Update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

Bo Han, Paul Cook, and Timothy Baldwin. Automatically Constructing a Normalisation Dictionary for Microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 421–432, 2012.

Zellig S. Harris. Distributional structure. *Word*, 10:146–162, 1954.

Morgan Harvey, Ian Ruthven, and Mark J. Carman. Improving Social Bookmark Search Using Personalised Latent Variable Language Models. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, pages 485–494, 2011.

Morgan Harvey, Fabio Crestani, and Mark J. Carman. Building User Profiles from Topic Models for Personalised Search. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pages 2309–2314, 2013.

Ahmed Hassan and Ryen W. White. Personalized Models of Search Satisfaction. In *Proceedings of the 22nd ACM International Conference on Conference on Information and Knowledge Management*, pages 2009–2018, 2013.

Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. Learning to Represent Knowledge Graphs with Gaussian Embedding. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 623–632, 2015.

Gregor Heinrich. Parameter Estimation for Text Analysis. Technical report, 2005. URL `http://www.arbylon.net/publications/text-est.pdf`.

Swapnil Hingmire, Sandeep Chougule, Girish K. Palshikar, and Sutanu Chakraborti. Document Classification by Topic Labeling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 877–880, 2013.

Geoffrey E. Hinton. Learning Distributed Representations of Concepts. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pages 1–12, 1986.

Liangjie Hong and Brian D. Davison. Empirical Study of Topic Modeling in Twitter. In *Proceedings of the First Workshop on Social Media Analytics*, pages 80–88, 2010.

Weihua Hu and Jun'ichi Tsujii. A Latent Concept Topic Model for Robust Topic Inference Using Word Embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 380–386, 2016.

Yuening Hu, Jordan Boyd-Graber, and Brianna Satinoff. Interactive topic modeling. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 248–257, 2011.

Eric Huang, Richard Socher, Christopher Manning, and Andrew Ng. Improving Word Representations via Global Context and Multiple Word Prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 873–882, 2012.

Seungil Huh and Stephen E. Fienberg. Discriminative Topic Modeling Based on Manifold Learning. *ACM Transactions on Knowledge Discovery from Data*, 5(4):20:1–20:25, 2012.

Ozan Irsoy and Claire Cardie. Opinion Mining with Deep Recurrent Neural Networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 720–728, 2014.

Rodolphe Jenatton, Nicolas L. Roux, Antoine Bordes, and Guillaume R Obozinski. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems 25*, pages 3167–3175. 2012.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge Graph Embedding via Dynamic Mapping Matrix. In *Proceedings of the 53rd Annual Meeting of the*

Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 687–696, 2015.

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. Knowledge Graph Completion with Adaptive Sparse Transfer Matrix. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 985–991, 2016.

Di Jiang, Lei Shi, Rongzhong Lian, and Hua Wu. Latent Topic Embedding. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2689–2698, 2016.

Ou Jin, Nathan N. Liu, Kai Zhao, Yong Yu, and Qiang Yang. Transferring Topical Knowledge from Auxiliary Long Texts for Short Text Clustering. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 775–784, 2011.

Mark Johnson. PCFGs, Topic Models, Adaptor Grammars and Learning Topical Collocations and the Structure of Proper Names. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1148–1157, 2010.

Mark Johnson. Bayesian Inference for Dirichlet-Multinomials. URL `http://web.science.mq.edu.au/~mjohnson/papers/Johnson-IPAM11-extras.pdf`, 2011.

M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37(2): 183–233, 1999.

Yoon Kim. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751, 2014.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980, 2014.

Samuel Kotz, Hosam Mahmoud, and Philippe Robert. On generalized Pólya urn models. *Statistics & Probability Letters*, 49(2):163 – 173, 2000.

Jayant Krishnamurthy and Tom Mitchell. Weakly Supervised Training of Semantic Parsers. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 754–765, 2012.

Denis Krompaß, Stephan Baier, and Volker Tresp. Type-Constrained Representation Learning in Knowledge Graphs. In *Proceedings of the 14th International Semantic Web Conference*, pages 640–655. 2015.

Simon Lacoste-Julien, Fei Sha, and Michael I. Jordan. DiscLDA: Discriminative Learning for Dimensionality Reduction and Classification. In *Advances in Neural Information Processing Systems 21*, pages 897–904. 2009.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural Architectures for Named Entity Recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, 2016.

Ni Lao and William W. Cohen. Relational retrieval using a combination of path-constrained random walks. *Machine Learning*, 81(1):53–67, 2010.

Ni Lao, Tom Mitchell, and William W. Cohen. Random Walk Inference and Learning in a Large Scale Knowledge Base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 529–539, 2011.

Han Jey Lau, David Newman, and Timothy Baldwin. Machine Reading Tea Leaves: Automatically Evaluating Topic Coherence and Topic Model Quality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 530–539, 2014.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web*, 6(2):167–195, 2015.

Omer Levy and Yoav Goldberg. Neural Word Embedding as Implicit Matrix Factorization. In *Advances in Neural Information Processing Systems 27*, pages 2177–2185. 2014.

Mike Lewis and Mark Steedman. Improved CCG Parsing with Semi-supervised Supertagging. *Transactions of the Association for Computational Linguistics*, 2:327–338, 2014.

Chenliang Li, Haoran Wang, Zhiqian Zhang, Aixin Sun, and Zongyang Ma. Topic Modeling for Short Texts with Auxiliary Word Embeddings. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 165–174, 2016a.

Chenliang Li, Jian Xing, Aixin Sun, and Zongyang Ma. Effective Document Labeling with Very Few Seed Words: A Topic Model Approach. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 85–94, 2016b.

Shaohua Li, Tat-Seng Chua, Jun Zhu, and Chunyan Miao. Generative Topic Embedding: a Continuous Representation of Documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 666–675, 2016c.

Wei Li and Andrew McCallum. Pachinko allocation: DAG-structured mixture models of topic correlations. In *Proceedings of the 23rd international conference on Machine learning*, pages 577–584, 2006.

Ximing Li, Jinjin Chi, Changchun Li, Jihong Ouyang, and Bo Fu. Integrating Topic Modeling with Word Embeddings by Mixtures of vMFs. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 151–160, 2016d.

Chen Liang and Kenneth D. Forbus. Learning Plausible Inferences from Semantic Web Knowledge by Combining Analogical Generalization with Structured Logistic Regression. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 551–557, 2015.

Kar Wai Lim. *Nonparametric Bayesian Topic Modelling with Auxiliary Data*. PhD thesis, The Australian National University, 2016.

Tianyi Lin, Wentao Tian, Qiaozhu Mei, and Hong Cheng. The Dual-sparse Topic Model: Mining Focused Topics and Focused Terms in Short Text. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 539–550, 2014.

Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling Relation Paths for Representation Learning of Knowledge Bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 705–714, 2015a.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence Learning*, pages 2181–2187. 2015b.

Yi Lin. A note on margin-based loss functions in classification. *Statistics & Probability Letters*, 68(1):73 – 82, 2004.

D. C. Liu and J. Nocedal. On the Limited Memory BFGS Method for Large Scale Optimization. *Mathematical Programming*, 45(3):503–528, 1989.

H. Liu and P. Singh. ConceptNet — A Practical Commonsense Reasoning Tool-Kit. *BT Technology Journal*, 22(4):211–226, 2004.

Pengfei Liu, Shafiq Joty, and Helen Meng. Fine-grained Opinion Mining with Recurrent Neural Networks and Word Embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1433–1443, 2015a.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Learning Context-sensitive Word Embeddings with Neural Tensor Skip-gram Model. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 1284–1290, 2015b.

Qiao Liu, Liuyi Jiang, Minghao Han, Yao Liu, and Zhiguang Qin. Hierarchical Random Walk Inference in Knowledge Graphs. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 445–454, 2016.

Xin Liu. Modeling Users' Dynamic Preference for Personalized Recommendation. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 1785–1791, 2015.

Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Topical Word Embeddings. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2418–2424, 2015c.

Peter Lofgren, Siddhartha Banerjee, and Ashish Goel. Personalized PageRank Estimation and Search: A Bidirectional Approach. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 163–172, 2016.

Yue Lu, Qiaozhu Mei, and ChengXiang Zhai. Investigating task performance of probabilistic topic models: an empirical study of PLSA and LDA. *Information Retrieval*, 14: 178–203, 2011.

Kevin Lund and Curt Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, 28(2):203–208, 1996.

Yuanfei Luo, Quan Wang, Bin Wang, and Li Guo. Context-Dependent Knowledge Graph Embedding. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1656–1661, 2015.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task Sequence to Sequence Learning. In *Proceedings of the 4th International Conference on Learning Representations*, 2016.

Xuezhe Ma and Eduard Hovy. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Long Papers)*, pages 1064–1074, 2016.

Andrew L. Maas and Andrew Y. Ng. A Probabilistic Model for Semantic Word Vectors. In *Proceedings of the NIPS 2010 Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 142–150, 2011.

David J. C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003. ISBN 978-0-521-64298-9.

Christopher D. Manning. Computational Linguistics and Deep Learning. *Computational Linguistics*, 41(4):701–707, 2015.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

Andrew McCallum. MALLET: A Machine Learning for Language Toolkit, 2002. URL `http://mallet.cs.umass.edu`.

Rishabh Mehrotra, Scott Sanner, Wray Buntine, and Lexing Xie. Improving LDA Topic Models for Microblogs via Tweet Pooling and Automatic Labeling. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 889–892, 2013.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *Proceedings of the International Conference on Learning Representations 2013: Workshop Track*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. 2013b.

George A. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41, 1995.

David Mimno, Hanna M Wallach, Jason Naradowsky, David A Smith, and Andrew McCallum. Polylingual topic models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing - Volume 2*, pages 880–889, 2009.

David Mimno, Hanna M. Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing Semantic Coherence in Topic Models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 262–272, 2011.

Roberto Navigli and Paola Velardi. Structural Semantic Interconnections: A Knowledge-Based Approach to Word Sense Disambiguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1075–1086, 2005.

Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. Compositional Vector Space Models for Knowledge Base Completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 156–166, 2015.

David Newman, Chaitanya Chemudugunta, and Padhraic Smyth. Statistical Entity-Topic Models. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 680–686, 2006.

David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. Automatic Evaluation of Topic Coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 100–108, 2010.

Dai Quoc Nguyen, Dat Quoc Nguyen, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. A Mixture Model for Learning Multi-Sense Word Embeddings. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics*, page to appear, 2017.

Dat Quoc Nguyen. jLDADMM: A Java package for the LDA and DMM topic models. URL `http://jldadmm.sourceforge.net/`, 2015.

Dat Quoc Nguyen. An overview of embedding models of entities and relationships for knowledge base completion. *arXiv preprint*, arXiv:1703.08098, 2017.

Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. Improving Topic Models with Latent Feature Word Representations. *Transactions of the Association for Computational Linguistics*, 3:299–313, 2015a.

Dat Quoc Nguyen, Kairit Sirts, and Mark Johnson. Improving Topic Coherence with Latent Feature Word Representations in MAP Estimation for Topic Modeling. In *Proceedings of the Australasian Language Technology Association Workshop 2015*, pages 116–121, 2015b.

Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. Neighborhood Mixture Model for Knowledge Base Completion. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 40–50, 2016a.

Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. STransE: a novel embedding model of entities and relationships in knowledge bases. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 460–466, 2016b.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A Three-Way Model for Collective Learning on Multi-Relational Data. In *Proceedings of the 28th International Conference on Machine Learning*, pages 809–816, 2011.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A Review of Relational Machine Learning for Knowledge Graphs. *Proceedings of the IEEE*, 104(1): 11–33, 2016a.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic Embeddings of Knowledge Graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 1955–1961, 2016b.

Mathias Niepert. Discriminative Gaifman Models. In *Advances in Neural Information Processing Systems 29*, pages 3405–3413. 2016.

Kamal Nigam, AK McCallum, S Thrun, and T Mitchell. Text Classification from Labeled and Unlabeled Documents Using EM. *Machine learning*, 39:103–134, 2000.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, second edition, 2006.

Arvid Österlund, David Ödling, and Magnus Sahlgren. Factorization of Latent Variables in Distributional Semantic Models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 227–231, 2015.

Michael Paul and Mark Dredze. SPRITE: Generalizing Topic Models with Structured Priors. *Transactions of the Association for Computational Linguistics*, 3:43–57, 2015.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

Adler J Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. Hierarchically Supervised Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems 24*, pages 2609–2617. 2011.

James Petterson, Wray Buntine, Shravan M. Narayanamurthy, Tibério S. Caetano, and Alex J. Smola. Word Features for Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems 23*, pages 1921–1929. 2010.

Xuan-Hieu Phan, Cam-Tu Nguyen, Dieu-Thu Le, Le-Minh Nguyen, Susumu Horiguchi, and Quang-Thuy Ha. A Hidden Topic-Based Framework Toward Building Applications with Short Web Documents. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):961–976, 2011.

John C. Platt. Fast Training of Support Vector Machines using Sequential Minimal Optimization. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in kernel methods*, pages 185–208. 1999. ISBN 0-262-19416-3.

Simone Paolo Ponzetto and Michael Strube. Exploiting Semantic Role Labeling, WordNet and Wikipedia for Coreference Resolution. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 192–199, 2006.

Ian Porteous, David Newman, Alexander Ihler, Arthur Asuncion, Padhraic Smyth, and Max Welling. Fast Collapsed Gibbs Sampling for Latent Dirichlet Allocation. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 569–577, 2008.

Xiaojun Quan, Chunyu Kit, Yong Ge, and Sinno Jialin Pan. Short and Sparse Text Topic Modeling via Self-aggregation. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 2270–2276, 2015.

Daniel Ramage, Christopher D Manning, and Susan Dumais. Partially labeled topic models for interpretable text mining. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–465, 2011.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. Relation Extraction with Matrix Factorization and Universal Schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84, 2013.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

Christian P. Robert and George Casella. *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Springer-Verlag New York, Inc., 2004.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.

Sara Salehi, Jia Tina Du, and Helen Ashman. Examining Personalization in Academic Web Search. In *Proceedings of the 26th ACM Conference on Hypertext & Social Media*, pages 103–111, 2015.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling Relational Data with Graph Convolutional Networks. *ArXiv preprint*, abs/1703.06103, 2017.

Baoxu Shi and Tim Weninger. ProjE: Embedding Projection for Knowledge Graph Completion. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 1236–1242, 2017.

Milad Shokouhi, Ryen W. White, Paul Bennett, and Filip Radlinski. Fighting Search Engine Amnesia: Reranking Repeated Results. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 273–282, 2013.

Ahu Sieg, Ahu Sieg, Bamshad Mobasher, Bamshad Mobasher, Robin Burke, and Robin Burke. Web search personalization with ontological user profiles. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 525–534, 2007.

Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. Parsing with Compositional Vector Grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, 2013a.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *Advances in Neural Information Processing Systems 26*, pages 926–934. 2013b.

Keith Stevens, Philip Kegelmeyer, David Andrzejewski, and David Buttler. Exploring Topic Coherence over Many Models and Many Topics. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 952–961, 2012.

Mark Steyvers and Tom Griffiths. Probabilistic topic models. *Handbook of Latent Semantic Analysis*, 427(7):424–440, 2007.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, pages 697–706, 2007.

Didi Surian, Dat Quoc Nguyen, Georgina Kennedy, Mark Johnson, Enrico Coiera, and G. Adam Dunn. Characterizing Twitter Discussions About HPV Vaccines Using Topic Modeling and Community Detection. *Journal of Medical Internet Research*, 18(8):e232, 2016.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, pages 3104–3112, 2014.

Jian Tang, Zhaoshi Meng, XuanLong Nguyen, Qiaozhu Mei, and Ming Zhang. Understanding the Limiting Factors of Topic Modeling via Posterior Contraction Analysis. In *Proceedings of the 31st International Conference on International Conference on Machine Learning*, pages 190–198, 2014.

Ben Taskar, Ming fai Wong, Pieter Abbeel, and Daphne Koller. Link Prediction in Relational Data. In *Advances in Neural Information Processing Systems 16*, pages 659–666. 2004.

Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Falk Brauer. Random Semantic Tensor Ensemble for Scalable Knowledge Graph Link Prediction. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 751–760, 2017.

Jaime Teevan, Susan T. Dumais, and Eric Horvitz. Personalizing search via automated analysis of interests and activities. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 449–456, 2005.

Jaime Teevan, Meredith Ringel Morris, and Steve Bush. Discovering and Using Groups to Improve Personalized Search. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 15–24, 2009.

Jaime Teevan, Susan T. Dumais, and Eric Horvitz. Potential for Personalization. *ACM Transactions on Computer-Human Interaction*, 17(1):4:1–4:31, 2010.

Jaime Teevan, Daniel J. Liebling, and Gayathri Ravichandran Geetha. Understanding and Predicting Personal Navigation. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, pages 85–94, 2011.

Yee W Teh, Michael I. Jordan, M.J. Beal, and David M. Blei. Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, 101:1566–1581, 2006a.

Yee W Teh, David Newman, and Max Welling. A Collapsed Variational Bayesian Inference Algorithm for Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems 19*, pages 1353–1360. 2006b.

T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

Kristina Toutanova and Danqi Chen. Observed Versus Latent Features for Knowledge Base and Text Inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, 2015.

Kristina Toutanova and Mark Johnson. A Bayesian LDA-based Model for Semi-Supervised Part-of-speech Tagging. In *Advances in Neural Information Processing Systems 20*, pages 1521–1528. 2008.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing Text for Joint Embedding of Text and Knowledge Bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, 2015.

Kristina Toutanova, Victoria Lin, Wen-tau Yih, Hoifung Poon, and Chris Quirk. Compositional Learning of Embeddings for Relation Paths in Knowledge Base and Text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1434–1444, 2016.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex Embeddings for Simple Link Prediction. In *Proceedings of the 33nd International Conference on Machine Learning*, pages 2071–2080, 2016.

Peter D. Turney. Similarity of Semantic Relations. *Computational Linguistics*, 32(3): 379–416, 2006.

Peter D. Turney and Patrick Pantel. From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188, 2010.

Yury Ustinovskiy, Gleb Gusev, and Pavel Serdyukov. An Optimization Framework for Weighting Implicit Relevance Labels for Personalized Web Search. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1144–1154, 2015.

Daniele Vitale, Paolo Ferragina, and Ugo Scaiella. Classification of Short Texts by Deploying Topical Annotations. In *Proceedings of the 34th European Conference on Advances in Information Retrieval*, pages 376–387, 2012.

Jan Vosecky, Di Jiang, Kenneth Wai-Ting Leung, and Wilfred Ng. Dynamic Multi-faceted Topic Discovery in Twitter. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pages 879–884, 2013.

Thanh Vu, Dawei Song, Alistair Willis, Son Ngoc Tran, and Jingfei Li. Improving Search Personalisation with Dynamic Group Formation. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 951–954, 2014.

Thanh Vu, Alistair Willis, and Dawei Song. Modelling Time-aware Search Tasks for Search Personalisation. In *Proceedings of the 24th International Conference on World Wide Web Companion*, pages 131–132, 2015a.

Thanh Vu, Alistair Willis, Son Ngoc Tran, and Dawei Song. Temporal Latent Topic User Profiles for Search Personalisation. In *Proceedings of the 37th European Conference on Information Retrieval*, pages 605–616, 2015b.

Thanh Vu, Dat Quoc Nguyen, Mark Johnson, Dawei Song, and Alistair Willis. Search Personalization with Embeddings. In *Proceedings of the 39th European Conference on*

Information Retrieval, pages 598–604, 2017a. The first two authors contributed equally to this work.

Thanh Vu, Alistair Willis, Udo Kruschwitz, and Dawei Song. Personalised Query Suggestion for Intranet Search with Temporal User Profiling. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, pages 265–268, 2017b.

Christian Walck. Handbook on statistical distributions for experimentalists. Technical report, Internal Report SUF-PFY/96-01, University of Stockholm, 2007.

Hanna M Wallach. Topic Modeling: Beyond Bag-of-Words. In *Proceedings of the 23rd international conference on Machine learning*, pages 977–984, 2006.

Chong Wang and David M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 448–456, 2011.

Chong Wang, David M. Blei, and David Heckerman. Continuous Time Dynamic Topic Models. In *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence*, pages 579–586, 2008.

Chong Wang, Bo Thiesson, Christopher Meek, and David M. Blei. Markov Topic Models. In *Proceedings of the 12th International Confe-rence on Artificial Intelligence and Statistics*, volume 5, pages 583–590, 2009.

Quan Wang, Jing Liu, Yuanfei Luo, Bin Wang, and Chin-Yew Lin. Knowledge Base Completion via Coupled Path Ranking. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1308–1318, 2016.

X. Wang, A. McCallum, and X. Wei. Topical N-Grams: Phrase and Topic Discovery, with an Application to Information Retrieval. In *Proceedings of the 7th IEEE International Conference on Data Mining*, pages 697–702, 2007.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge Graph Embedding by Translating on Hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1112–1119. 2014a.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge Graph and Text Jointly Embedding. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1591–1601, 2014b.

Zhigang Wang and Juan-Zi Li. Text-Enhanced Representation Learning for Knowledge Graph. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 1293–1299, 2016.

Xing Wei and W. Bruce Croft. LDA-based Document Models for Ad-hoc Retrieval. In *Proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 178–185, 2006.

Zhuoyu Wei, Jun Zhao, and Kang Liu. Mining Inference Formulas by Goal-Directed Random Walks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1379–1388, 2016.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. Structured Training for Neural Network Transition-Based Parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 323–333, 2015.

Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. TwitterRank: Finding Topic-sensitive Influential Twitterers. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, pages 261–270, 2010.

Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. Knowledge Base Completion via Search-based Question Answering. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 515–526, 2014.

Jason Weston and Antoine Bordes. Embedding Methods for NLP. In *EMNLP 2014 tutorial*, 2014.

Ryen W. White and Ahmed Hassan Awadallah. Personalizing Search on Shared Devices. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 523–532, 2015.

Ryen W. White, Wei Chu, Ahmed Hassan, Xiaodong He, Yang Song, and Hongning Wang. Enhancing Personalized Search by Mining and Modeling Task Behavior. In *Proceedings of the 22Nd International Conference on World Wide Web*, pages 1411–1420, 2013.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. TransG : A Generative Model for Knowledge Graph Embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2316–2325, 2016.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. SSP: Semantic Space Projection for Knowledge Graph Embedding with Text Descriptions. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 3104–3110, 2017.

Pengtao Xie and Eric P. Xing. Integrating Document Clustering and Topic Modeling. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pages 694–703, 2013.

Jinyun Yan, Wei Chu, and Ryen W. White. Cohort Modeling for Enhanced Personalized Search. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 505–514, 2014.

Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. A Biterm Topic Model for Short Texts. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 1445–1456, 2013.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *Proceedings of the International Conference on Learning Representations*, 2015.

Liu Yang, Qi Guo, Yang Song, Sha Meng, Milad Shokouhi, Kieran McDonald, and W. Bruce Croft. Modeling user interests for zero-query ranking. In *Proceedings of the 38th European Conference on Information Retrieval*, pages 171–184, 2016.

Limin Yao, Sebastian Riedel, and Andrew McCallum. Universal Schema for Entity Type Prediction. In *Proceedings of the 2013 Workshop on Automated Knowledge Base Construction*, pages 79–84, 2013.

Xing Yi and James Allan. A Comparative Study of Utilizing Topic Models for Information Retrieval. In Mohand Boughanem, Catherine Berrut, Josiane Mothe, and Chantal Soule-Dupuy, editors, *Proceedings of the 31st European Conference on Information Retrieval*, pages 29–41, 2009.

Jianhua Yin and Jianyong Wang. A Dirichlet Multinomial Mixture Model-based Approach for Short Text Clustering. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 233–242, 2014.

Hee-Geun Yoon, Hyun-Je Song, Seong-Bae Park, and Se-Young Park. A Translation-Based Knowledge Graph Embedding Preserving Logical Property of Relations. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 907–916, 2016.

Matthew D. Zeiler. ADADELTA: An Adaptive Learning Rate Method. *CoRR*, abs/1212.5701, 2012.

Ke Zhai and Jordan L. Boyd-graber. Online Latent Dirichlet Allocation with Infinite Vocabulary. In *Proceedings of the 30th International Conference on Machine Learning*, pages 561–569, 2013.

Duo Zhang, Qiaozhu Mei, and ChengXiang Zhai. Cross-Lingual Latent Topic Extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1128–1137, 2010.

Heng Zhang and Guoqiang Zhong. Improving short text classification by learning vector representations of both words and hidden topics. *Knowledge-Based Systems*, 102:76–86, 2016.

Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. Comparing Twitter and Traditional Media Using Topic Models. In *Proceedings of the 33rd European Conference on Advances in Information Retrieval*, pages 338–349, 2011.

Yu Zhao, Sheng Gao, Patrick Gallinari, and Jun Guo. Knowledge Base Completion by Learning Pairwise-Interaction Differentiated Embeddings. *Data Mining and Knowledge Discovery*, 29(5):1486–1504, 2015a.

Zhendong Zhao, Lan Du, Benjamin Börschinger, John K Pate, Massimiliano Ciaramita, Mark Steedman, and Mark Johnson. A Computationally Efficient Algorithm for Learning Topical Collocation Models. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1460–1469, 2015b.

Jun Zhu and Eric P. Xing. Conditional Topic Random Fields. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.

Yuan Zuo, Junjie Wu, Hui Zhang, Hao Lin, Fei Wang, Ke Xu, and Hui Xiong. Topic Modeling of Short Texts: A Pseudo-Document View. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2105–2114, 2016.