Contextual Representation Learning for Text Data

A

Dissertation

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

---

in partial fulfillment

of the requirements for the degree

Doctor of Philosophy

by

Guangxu Xun

May 2021

APPROVAL SHEET

This dissertation

is submitted in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Author: Guangxu Xun

The dissertation has been read and approved by the examining committee:

Advisor: Dr. Aidong Zhang

Committee Chair: Dr. Hongning Wang

Committee Member: Dr. Yangfeng Ji

Committee Member: Dr. Jundong Li

Committee Member: Dr. Stefan Bekiranov

Committee Member: Dr. Heng Huang

Accepted for the School of Engineering and Applied Science:

Craig H. Benson, Dean, School of Engineering and Applied Science

May 2021

© Guangxu Xun 2021
ALL RIGHTS RESERVED

Acknowledgements

I would like to thank many people who have helped me along my path of pursuing a Ph.D. degree in computer science.

First and foremost, I would especially like to thank my advisor Professor Aidong Zhang, for taking me under her wing, and for running such a great lab where everyone works on cutting-edge research problems. She has been a role model researcher who is always passionate about new challenges. She is very knowledgeable and extremely professional. What makes her an even greater advisor is that she truly cares about her students. I consider myself very fortunate to have her as my advisor. Her knowledge, advice, support and encouragement have greatly helped me during my Ph.D. study.

I would like to thank my committee members Professor Hongning Wang, Professor Yangfeng Ji, Professor Jundong Li, Professor Stefan Bekiranov and Professor Heng Huang for their valuable suggestions for my research and for shaping my dissertation. I would also like to thank Professor Jing Gao for providing insightful advice.

I also appreciate the support and help from my collaborators, lab mates and friends. I would like to thank Kishlay Jha, Xiaowei Jia, Fenglong Ma, Yaqing Wang, Ye Yuan, Jinduo Liu, Nan Du, Xiaoyi Li, Vishrawas Gopalakrishnan, Yaliang Li, Houping Xiao, Chuishi Meng, Liuyi Yao, Hongfei Xue, Qiuling Suo, Mengdi Huai, Jianhui Sun, Lionel S. Lewis and many more whose names are not included here.

I would like to thank my parents, Siyuan Xun and Yuxia Zou, for raising me to value education. I would like to thank my brother, Guangyu Xun, for his support and encouragement. I would also like to thank my girlfriend, Millie Lin, for always being supportive of my decisions.

Abstract

Nowadays, text data is being generated at an increasing rate of speed. Text data is prevalent in various domains, such as social media, newspapers, clinical notes and online reviews. Text data contains rich information and understanding text data is important for Artificial Intelligence (AI) tasks, especially for Natural Language Processing (NLP) tasks. The key to understanding text data lies in the representation of the data, as the success of NLP algorithms heavily depends on the quality of the text representations. For that reason, many conventional NLP systems attempt to design preprocessing pipelines and data transformations that can provide good representations of text data. Such feature engineering is useful but requires careful design and prior knowledge. Therefore, it is desirable to learn representations of text data automatically and lessen the degree of feature engineering in NLP systems. In this way, the downstream NLP applications can be constructed faster and achieve better performances.

Context information of text data, including spatial context, temporal context and domain context, is naturally a good source to learn representations of text data. Because the context information not only contains syntactic and semantic information which are a core requirement for representations of text data, but also is easy and convenient to collect. I will first introduce how to extract semantic and syntactic features from text data based on its spatial context information, such as word-word co-occurrences and document-word co-occurrences, and also how to coordinate both kinds of spatial context information. Next, I will demonstrate how to learn time-aware representations based on the temporal context information of text data, for example, temporal representations that can capture the semantic evolutions of words. Then I will show how to learn domain-specific representations of text data based on its domain context information, for example, extracting domain-related features from documents given the task domain. Extensive evaluations are also conducted and presented to demonstrate the effectiveness of the proposed contextual representation learning algorithms.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

1.1.1 Representation Learning for Text Data

Nowadays, the amount of data generated every day is increasing at a fast speed and text data takes up a large part of it. Text data can be found everywhere, for example, social networks where users share their stories, academic conferences where research papers get published, news media, clinical notes, and online reviews. Text data contains rich information and Natural Language Processing (NLP) is a subfield of Artificial Intelligence (AI) which aims to build models for computers to understand text data. As with other machine learning models, building NLP models typically consists of three major steps: learning representations of data, formulating objective functions and optimizing model parameters. Data representation determines what information we can extract from data, for example, encoding a data sample into a feature vector. Therefore the quality of data representation has a huge impact on the performance of NLP models. Conventional NLP models adopt careful data preprocessing and transformation to obtain a representation of the text data. However, such feature engineering requires prior knowledge and careful design, and it is also time-consuming and labor-intensive. This heavily limits the ease and scope of applicability of NLP systems. Hence, it is highly desirable to automate representation learning and lessen the degree of feature engineering.

Representation learning for text data aims to automatically extract information from text data which will be subsequently used for further NLP and text mining tasks. Among

various ways to learn representations of text data, this dissertation concentrates on deep learning methods. Compared with conventional representation learning methods, deep representation learning can extract more abstract representations and has achieved huge success in many domains, such as NLP and computer vision. Deep representation learning methods can greatly reduce the effort of feature engineering and improve final model performance at the same time. To this end, many NLP researchers are devoted to deep representation learning for text data and many deep representation learning methods are proposed, such as Word2Vec [1], GloVe [2], fastText [3], ELMo [4], BERT [5] and attention [6]. Compared with traditional representations of text data, such as the one-hot representations, deep representations encode an object, e.g., a word, a sentence or a document, into a low-dimensional real-valued feature vector, which is also known as the distributed representation. Moreover, by representing text data as dense vectors, the distributed representation is able to handle the curse of dimensionality issue in large-scale data.

1.1.2 Contextual Representation Learning for Text Data

A good representation learning method for text data should have the following properties:

- Easy to collect data. The success of deep learning models is heavily dependent on the amount of training data we can collect. The more data, the better performance. Hence, to learn good representations, the data collection procedure should be easy and convenient.

- Syntax and semantics. The ultimate goal of NLP is for computers to understand human languages. Therefore, it is crucial that the learned representation of text data captures the syntactic and semantic regularities in languages.

- Efficiency. Text data normally has very high dimensionalities and faces the curse of dimensionality issue. Thus, efficient representations are desirable for text data.

- Smoothness. This is a basic requirement for most representation learning methods. We expect words, sentences, documents with similar properties to have similar representations.

- Coherence. Text data samples close to each other temporally or spatially tend to

have strong associations. When learning representations of text data, the temporal and spatial neighbors of a text sample should be carefully considered for coherence.

Learning representations based on the context information of text data can fulfill the aforementioned requirements. First of all, it is quite easy and convenient to collect the context information of text data. For example, spatial context can be efficiently constructed from the word co-occurrence patterns. Two common types of spatial context information are the global document-word co-occurrence patterns and the local word-word co-occurrence patterns, and they can be used to learn representations of documents and words. A distinct advantage of such context information is that it requires no human annotation and can be easily constructed from unstructured text data. Second, according to the distributional hypothesis [7], "words that are used and occur in the same contexts tend to purport similar meanings", representation learning based on context information is able to capture the word-level syntax and semantics. In addition, words with similar meanings tend to occur in similar contexts, and this context similarity can help us achieve the smoothness of representations. Third, since contextual representation learning considers various types of context information, e.g., the temporal context information, the coherence of representations can also be achieved. Lastly, context information works well with the distributed representations in deep models, and can further avoid the high dimension issue in text data.

Therefore, this dissertation is focused on learning deep representations of text data based on the context information. Contextual representation learning models are investigated based on three types of context information: spatial context, temporal context and domain context.

1.2 Spatial Context in Text Data

Spatially nearby words form the most commonly used context information in text data. Based on the scopes of spatial context in text data, there are global context and local context. Spatial context information typically provides a way to establish language models, where we can learn representations of the input text data directly or as a byproduct.

1.2.1 Global Context

The global context information of a text corpus refers to the document level co-occurrence information, which contains topical information in language. Contextual representation learning models based on global spatial context try to model languages and learn representations based on the document-word co-occurrences. Most of these models follow the bag-of-words assumption and can learn representations carrying topic structures.

Topic models typically take the global context as input and can learn topical representations of text data. Latent Semantic Analysis (LSA) [8] uses a document-word matrix to denote the global context information of a text corpus and uses singular value decomposition (SVD) to find a low-rank approximation of the matrix while obtaining the latent semantics of documents. The latent semantics represent the topic dimension of documents. Documents and words can then be represented as dense vectors in the latent topic space. Probabilistic Latent Semantic Indexing (PLSI), also known as Probabilistic Latent Semantic Analysis (PLSA), is a classical topic model. It is a generative model for word and document co-occurrences [9]. The basic idea of this model is to model the co-occurrences which associate a latent topic variable with the occurrence of a word in a document. Every document in the corpus corresponds to a unique distribution of topics and every topic also has a unique distribution of words in the vocabulary. Though being effective, PLSI suffers from the issue of over-fitting due to the linear growth in the number of parameters with the number of training documents. To address this issue, Latent Dirichlet Allocation (LDA) [10] is proposed. Compared with PLSI, LDA also assumes that each topic is a specific distribution over words, but LDA tackles the over-fitting issue by generating the topic distribution for each document from a Dirichlet distribution.

1.2.2 Local Context

The local context information of a text corpus refers to the word level co-occurrence information, i.e., the neighborhood words of a focus word in a context window. It contains semantic and syntactic regularities in language. Most contextual representation learning models based on local context follow the distributional hypothesis and can learn representations carrying word-level syntax and semantics.

Neural language models typically take the local context as input and can learn distributed representations of words. Distributed representations of words [11], also known

as word embeddings and word vectors, are a core technique to introduce neural networks into contextual representation learning. It is able to address the curse of dimensionality issue and the lack of semantics issue in conventional one-hot representations. By incorporating the distributed representations of words, the neural probabilistic language model (NPLM) [12] uses a multi-layer neural network to predict words given their local contexts and learn word embeddings as a byproduct. Inspired by NPLM, many more deep representation learning models are proposed based on local context. For example, Word2Vec [1] alleviates the computation burden of the last softmax layer using negative sampling and hierarchical softmax, and GloVe [2] speeds up the learning process by calculating the contextual statistics in advance. In recent years, the popularity of neural language models as a pre-training method continues to grow as they achieve remarkable success in both academia and industry. Famous examples are ELMo [4] and BERT [5]. These models adopt deeper architectures, larger corpora, more parameters, and more importantly they consider more complicated and wider context information. As a consequence, they are able to dynamically calculate representations of input data, which is especially useful for complex phrases and polysemes. Moreover, they started the practice and research of transfer learning in NLP

1.3 Temporal Context in Text Data

Representations of text data can be time-sensitive in many scenarios. For example, a series of consecutive daily headline news gradually reveal the whole picture of an event. A more common scenario is the evolution of word meanings, for instance, “apple” was initially only associated with fruits, but now it also has another meaning as a technology company. The temporal context can help us learn time-aware representations of text data and ensure the temporal coherence of representations.

The temporal context can be integrated into representation learning in many ways, the most commonly used being to minimize the difference between representations at adjacent time frames. In particular, the temporal context is constructed by splitting text corpus into multiple time slices. One way to utilize the temporal context is to first learn static representations of each time slice separately, and then try to find a linear alignment across different time slices [13, 14]. Another way to utilize the temporal context is to jointly learn representations of different time slices and circumvent the additional alignment step [15, 16]. Temporally coherent representations are useful for

many time-sensitive NLP tasks, such as language evolution analysis and trending topic tracking.

1.4 Domain Context in Text Data

Text data is generated in a countless number of domains, including but not limited to social media texts, bioinformatics research papers, machine learning research papers, log files, news articles, novels and online reviews. Moreover, text data in different languages can also be viewed as in different domains. Therefore, the quality of representations largely depends on the task domains. For example, given the same movie review text, if we are doing sentiment analysis, we would like the representations sensitive to the sentiment content of the text, such as keywords like "excellent", "recommended" and "boring". However, if we are doing genre detection, we would prefer the representations that can extract information like "comedy" and "anime".

Good representations of text data vary largely based on different domains. The domain context of text data, e.g., prior knowledge, constraints and objectives, usually comes from the source information, the publisher information and the task descriptions. The domain context information can be integrated into representation learning as auxiliary modules, such as attention modules [6], to store the domain information. Contextual representation learning based on domain context enables us to automatically extract domain-sensitive features from text data.

1.5 Contributions and Dissertation Organization

This dissertation investigates contextual representation learning for text data based on three types of context information: spatial context, temporal context and domain context. The organization and contributions of the rest of the dissertation are summarized as follows:

Chapter 2 investigates how supplementary local context information can mitigate the lack of context information problem and discover topics in short texts. Discovering topics in short texts, such as news titles and tweets, has become an important task for many content analysis applications. However, due to the lack of context information in short texts, the performance of conventional topic models on short texts is usually unsatisfying. In this chapter, we propose a novel topic model for short text corpus

using word embeddings. Continuous space word embeddings, which are proven effective at capturing regularities in language, are incorporated into our model to provide additional semantics. Thus we model each short document as a Gaussian topic over word embeddings in the vector space. In addition, considering that background words in a short text are usually not semantically related, we introduce a discrete background mode over word types to complement the continuous Gaussian topics. We evaluate our model on real-world news titles, showing that our model is able to extract more coherent topics from short texts compared with the baseline methods and learn better topic representation for each short document.

Chapter 3 investigates how supplementary local context information can assist in discovering topic correlations. Conventional correlated topic models are able to capture the correlation structure among latent topics by replacing the Dirichlet prior with the logistic normal distribution. Word embeddings have been proven to be able to capture semantic regularities in language. Therefore, the semantic relatedness and correlations between words can be directly calculated in the word embedding space. In this chapter, we propose a novel correlated topic model using word embeddings. The proposed model enables us to exploit the additional word-level correlation information in word embeddings and directly model topic correlation in the continuous word embedding space. In the model, words in documents are replaced with meaningful word embeddings, topics are modeled as multivariate Gaussian distributions over the word embeddings and topic correlations are learned among the continuous Gaussian topics. A Gibbs sampling solution with data augmentation is given to perform inference. We evaluate our model on real-world text corpora qualitatively and quantitatively.

Chapter 4 investigates how to coordinate global and local context to achieve better representations of text data. A text corpus typically contains two types of spatial context information – global context and local context. Global context carries topical information which can be utilized by topic models to discover topic structures from the text corpus, while local context can train word embeddings to capture semantic regularities reflected in the text corpus. This encourages us to exploit the useful information in both the global and the local context information. In this chapter, we propose a unified language model based on matrix factorization techniques which takes the complementary global and local context information into consideration simultaneously, and models topics and learns word embeddings collaboratively. We empirically show that by incorporating both global and local context, this collaborative model can not only

significantly improve the performance of topic discovery over the baseline topic models, but also learn better word embeddings than the baseline word embedding models.

Chapter 5 studies how to learn dynamic representations of text data based on temporal context and further facilitate the analysis of knowledge evolution. Literature based discovery (LBD) is a task that aims to uncover hidden associations between non-interacting scientific concepts by rationally connecting independent nuggets of information. In this chapter, we propose a novel dynamic Medical Subject Heading (MeSH) embedding model which is able to model the evolutionary behavior of medical concepts to uncover latent associations between them. The proposed model constructs diachronic literature data, learns dynamic representations of medical concepts and detect informative concepts. Hence, based on the dynamic MeSH embeddings, meaningful medical hypotheses can be efficiently generated. To evaluate the efficacy of the proposed model, we perform both qualitative and quantitative evaluations. The results demonstrate that leveraging the evolutionary features of MeSH concepts is an effective way of predicting novel associations.

Chapter 6 investigates how to learn time-aware representations based on the temporal context of sequential data. Epileptic seizures are a serious health problem and there is a huge population suffering from it every year. Analyzing the scalp EEG is the most common way to detect the onset of a seizure. In this chapter, we propose the context-learning based EEG analysis for seizure detection (Context-EEG) algorithm. The proposed method aims at extracting both the hidden inherent features within EEG fragments and the temporal features from EEG contexts. First, we segment the EEG signals into EEG fragments of fixed length. Second, we learn the hidden inherent features from each fragment and reduce the dimensionality of the original data. Third, we translate each EEG fragment to an EEG word so that the EEG context can provide us with temporal information. And finally, we concatenate the hidden feature and the temporal feature together to train a binary classifier. The experiment result shows the proposed model is highly effective in detecting seizures.

Chapter 7 explores how to learn domain-specific representations of text data based on domain context. MEDLINE is the primary bibliographic database maintained by National Library of Medicine (NLM). MEDLINE citations are indexed with MeSH terms. This greatly facilitates the applications of biomedical research and knowledge discovery. Currently, MeSH indexing is manually performed by human experts. To reduce the

time and monetary cost associated with manual annotation, many automatic MeSH indexing systems have been proposed to assist manual annotation. However, the existing models usually cannot extract domain-specific features and suffer from efficiency issues. In this chapter, we propose an end-to-end framework, MeSHProbeNet, which utilizes deep learning and self-attentive MeSH probes to learn domain-specific representations of biomedical articles. Each MeSH probe enables the model to extract one specific aspect of biomedical knowledge from an input article, thus comprehensive biomedical information can be extracted with different MeSH probes and interpretability can be achieved at word level. MeSH terms are finally recommended with a unified classifier, making MeSHProbeNet both time efficient and space efficient.

Chapter 8 demonstrates that even more specific representations can be learned given fine-grained domain context. In this chapter, we propose an end-to-end framework, MeSHProbeNet-P, which extends MeSHProbeNet with personalizable MeSH probes. In MeSHProbeNet-P, each MeSH probe carries certain aspects of biomedical knowledge and extracts related information from input articles. Given fine-grained domain context, MeSHProbeNet-P is able to automatically personalize/customize its MeSH probes for different input articles to ensure that the current MeSH probes best fit the current input article and the most informative features can be extracted from the article. We demonstrate the effectiveness of MeSHProbeNet-P in a real-world large-scale MeSH indexing challenge. We also provide ablation studies to show the advantages of personalizable MeSH probes.

Chapter 9 concludes the dissertation with a discussion of future research directions.

Part I

Contextual Representation Learning Based on Spatial Context

Chapter 2

Exploiting Supplementary Local Context for Topic Discovery in Short Text

2.1 Introduction

With more than five Exabytes of data being generated in less than two days [17], recent researches in Internet and social media focus on effective ways for data management and content presentation. Social networks on their part attempt to handle this by trying to provide a cohesive yet real-time view on a topic by partitioning the data into "Trending Topics" by hashtag or text mentions. However, such explicit categorization is either not possible or comes at a high cost in other domains like news titles, text advertisements, questions/tasks in crowd sourced applications, etc. To this end, topic models have proven to be a useful tool in unsupervised text analyses and pattern discovery in a corpus. Extracting meaningful topics helps us better analyze the documents, reduce the dimensionality of documents (allowing faster analyses) and is also crucial for many content analysis tasks, e.g. dynamic topic detection and topic expertise discovery [18, 19, 20, 21, 22]. However, the efficacy of conventional topic models is limited by the lack of rich context in short texts. The limitation stems from the fact that each individual document, by itself, is too short for effective topic extraction.

Conventional topic models, such as Probabilistic Latent Semantic Analysis (PLSA) [9] and Latent Dirichlet Allocation (LDA) [10], follow the bag-of-word assumption and

model documents as mixtures of latent topics, where topics are multinomial distributions over words. Bayesian methods are then employed to learn the topic distribution for each document based on the document-word frequency matrix of the corpus. However, compared with regular documents, short texts are suffering from the lack of rich context. Short texts like news titles or tweets usually span only 10-30 word long, e.g. Twitter imposes a limit of 140 characters on each tweet. From a statistical point of view, this problem will heavily limit the quality of topics extracted from short texts by conventional topic models.

To overcome the lack of context information in short text corpus and exploit external semantics, we develop a new topic model for short text using word embeddings [23] in continuous vector space. Word embeddings, also known as word vectors and distributed representations of words, have proven to be effective at capturing semantic regularities in language: words with similar semantic and syntactic attributes are projected into the same area in the vector space. More specifically, first we use Wikipedia as an external source to train word embeddings upon it. The resulting semantic regularities are then used as a supplementary information to overcome the limitation of context information in short texts. Second, in the vector space of word embeddings, we formulate topics using Gaussian distributions to handle the "continuous" space of word embeddings. The primary motivation behind this modeling is that since we are now in vector space and semantically related words are located close to each other, Gaussian distribution over the word embeddings denotes the semantic centrality. Third, instead of viewing each short text as a mixture of topics, we assume each such text focuses on only one Gaussian topic. This assumption is plausible as the size of text is in the range of 10-30 words. Fourth, considering the fact that most background words are not semantically related, we add the background mode with discrete multinomial distribution of words to complement the Gaussian topics. Thus, we are able to extract better topics from short text.

2.2 Methodology

In this section, we discuss the proposed methodology to extract high quality topics from short texts. An end-to-end framework is shown in Figure 2.1.

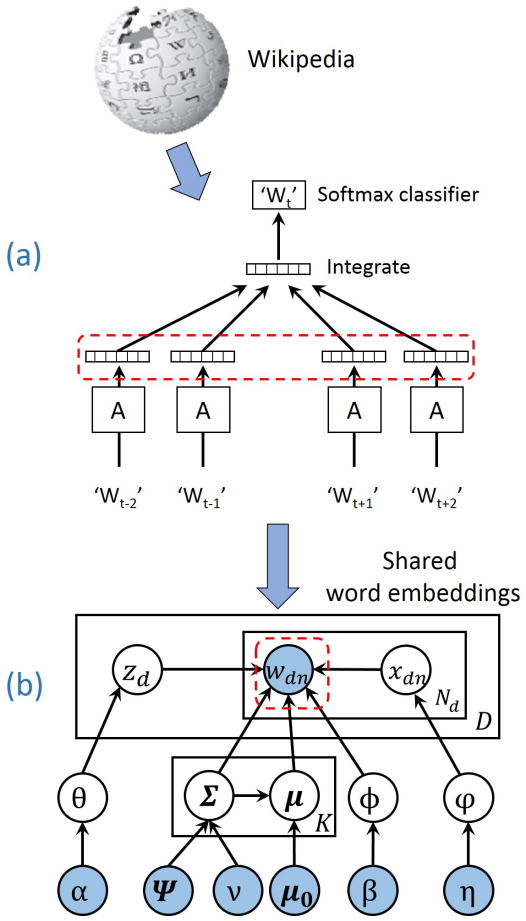


Figure 2.1: Schematic illustration of topic discovery for short texts. Part (a) represents the word embedding learning process. Part (b) represents the topic modelling in presence of word embedding for short texts.

2.2.1 Learning Word Embeddings from Wikipedia

In our approach, we learn word embeddings using Wikipedia as the external source. The motivation of using Wikipedia lies in the sheer range of topics and subjects that are covered. Extracting word embeddings from Wikipedia allows us to "enrich" the short text with additional semantics. The part (a) of Figure 2.1 illustrates the training of Continuous Bag of Words (CBOW) word embeddings using Word2Vec tool [24].

Having learnt the word embeddings, given a word w_{dn}, which is the n^{th} word in d^{th} document, we can enrich that word by replacing it with the corresponding word embedding (red blocks in Figure 2.1). The following section describes how this enrichment is

used in a generative process to extract a single topic for a given short document.

2.2.2 Strategies and Generative Process

Wikipedia word embeddings give us useful additional semantics, which is crucial due to the lack of context information in short texts. However, as the documents are now sequences of word embeddings instead of sequences of word types, conventional topic models no longer are applicable. Since the discrete word types are now replaced by continuous space of word embeddings, and those word vectors are allocated in space based on their semantics and syntax, we consider them as draws from several Gaussian distributions. Hence, each topic is characterized as a multivariate Gaussian distribution in the vector space. The choice of Gaussian distribution is justified by the observations that Euclidean distances between word embeddings correlate with their semantic similarities.

Another important observation about short texts is that each short text usually consists of only one topic instead of a combination of multiple topics. Inspired by Twitter-LDA [25], we assume each document is about one single Gaussian topic. Thus, the words in a document either belong to the document's topic or to the background mode.

However, it is not accurate to continue using word embeddings for the background mode. This is because background words are not semantically interrelated and hence, we cannot find a semantically correspondent Gaussian distribution to their physical locations in the vector space. Thus, in the background mode, we use discrete word types rather than continuous word embeddings to represent words.

More formally, a document d is construed to be of a single Gaussian topic, represented by z_d in Figure 2.1 part (b). The corresponding parameter that controls the latent topic distribution is θ and the hyper-parameter for that distribution is α. Word w_{dn} can either be a topic word or a background word, we consider both factors. For a topic, it is represented by a multivariate Gaussian distribution in the word vector space and $\boldsymbol{\mu}_k$ denotes the mean and $\boldsymbol{\Sigma}_k$ denotes the covariance for the k^{th} topic. $\boldsymbol{\Psi}$ is hyper-parameter covariance matrix and ν is the hyper-parameter denoting the initial degree of freedom. $\boldsymbol{\mu}_0$ is the hyper-parameter for mean. ϕ represents the multinomial distribution for background words for which the corresponding hyper-parameter is β. The fact that whether the word w_{dn} is a background word or not is depicted by an

indicator variable x_{dn}, whose parameter is φ representing the Bernoulli distribution. The corresponding hyper-parameter for that distribution is η. Variables in bold font mean they are either vectors or matrices. The generative process is as follows:

1. Draw $\theta \sim Dir(\alpha)$.
2. Draw $\phi \sim Dir(\beta)$.
3. Draw $\varphi \sim Dir(\eta)$.
4. For each topic $k = 1, 2, \cdots, K$:
   (a) Draw topic covariance $\boldsymbol{\Sigma}_k \sim \mathcal{W}^{-1}(\boldsymbol{\Psi}, \nu)$.
   (b) Draw topic mean $\boldsymbol{\mu}_k \sim \mathcal{N}(\boldsymbol{\mu}_0, \frac{1}{\tau}\boldsymbol{\Sigma}_k)$.
5. For each text $d = 1, 2, \cdots, D$:
   (a) Draw a topic $z_d \sim Multinomial(\theta)$.
   (b) For each word index $n = 1, 2, \cdots, N_d$:
      i. Draw a word category $x_{dn} \sim Bernoulli(\varphi)$.
      ii. Draw a word. If $x_{dn} = 1$, Draw topic word $\boldsymbol{w}_{dn} \sim \mathcal{N}(\boldsymbol{\mu}_{z_d}, \boldsymbol{\Sigma}_{z_d})$; otherwise, draw background word $w_{dn} \sim Multinomial(\phi)$.

Note that τ in step 4 (b) is a constant factor. We use the following conjugate priors: a Gaussian distribution $\mathcal{N}$ for the mean and an inverse Wishart distribution $\mathcal{W}^{-1}$ for the covariance.

2.2.3 Model Details

When $x_{dn} = 1$, the current word $\boldsymbol{w}_{dn}$ is a topic word and it corresponds to a Wikipedia word embedding; otherwise, the current word w_{dn} is a discrete background word. Hence, the conditional probability of the current word w_{dn} is:

$$p(\boldsymbol{w}_{dn}|x_{dn}, z_d, \phi, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \propto \left(f(\boldsymbol{w}_{dn}|\boldsymbol{\mu}_{z_d}, \boldsymbol{\Sigma}_{z_d})\right)^{x_{dn}} (\phi_{w_{dn}})^{1-x_{dn}},$$

where function $f(\boldsymbol{w}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is the probability density of topic k's Gaussian distribution. Thus, for document d of N_d words, the conditional probability is:

$$p(\boldsymbol{w}_d|\theta, \phi, \varphi, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_z p(z|\theta) \left(\prod_{n=1}^{N_d} p(x_{dn}|\varphi) p(\boldsymbol{w}_{dn}|x_{dn}, z, \phi, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \right). \tag{2.1}$$

Thus, for the corpora $\mathcal{D} = \{d\}_1^D$, we can obtain the overall probability $p(\mathcal{D}|\alpha, \beta, \eta, \boldsymbol{\Psi}, \nu, \boldsymbol{\mu}_0)$ by integrating out the intermediate variables. Furthermore, the objective function to minimize is the log likelihood of the corpora:

$$O = -\log p(\mathcal{D}|\alpha, \beta, \eta, \boldsymbol{\Psi}, \nu, \boldsymbol{\mu}_0). \tag{2.2}$$

2.2.4 Estimation and Parameter Inference

The observed variables are documents consisting of word types and word embeddings, and our goal is to infer the posterior distributions over the Gaussian topics and background mode along with topic assignments of words. We use Gibbs-EM to infer the parameters [26]. We begin by first fixing the other variables and derive a collapsed Gibbs sampler that samples document topic z_d document by document. The probability for sampling document topic z_d is:

$$p(z_d = k|\boldsymbol{z}_{-d}, D, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, \beta, \boldsymbol{x}) \propto (n_{-d}^k + \alpha) \cdot \prod_{n=1}^{N_d} \left(T_r(\boldsymbol{w}_{dn}|\boldsymbol{\mu}_k, \frac{\tau_k + 1}{\tau_k}\boldsymbol{\Sigma}_k) \right)^{x_{dn}}, \tag{2.3}$$

where n_{-d}^k denotes the number of times that topic k is sampled, without counting current document d. $T_r(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the multivariate Student's t-distribution for Gaussian sampling with $(r = \nu_k - dim + 1)$ being its degrees of freedom and dim being the dimensionality of word embeddings. $(\nu_k = \nu + n^k)$ and $(\tau_k = \tau + n^k)$ are the parameters of topic k, where n^k represents the total number of words that are assigned to topic k. x_{dn} is the topic/background indicator for word w_{dn}.

Then after the document topic z_d is sampled, for each word w_{dn} in document d, we sample the topic/background indicator x_{dn} according to the Bernoulli distribution:

$$\begin{aligned} &p(x_{dn}|\boldsymbol{w}, \boldsymbol{x}_{-w}, z_d = k, D, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, \beta) \\ &\qquad \propto \left(\frac{n_{x=1}^{-w} + \eta}{n_{x=1}^{-w} + n_{x=0}^{-w} + 2\eta} \cdot T_r(\boldsymbol{w}_{dn}|\boldsymbol{\mu}_k, \frac{\tau_k + 1}{\tau_k}\boldsymbol{\Sigma}_k) \right)^{x_{dn}} \\ &\qquad \cdot \left(\frac{n_{x=0}^{-w} + \eta}{n_{x=1}^{-w} + n_{x=0}^{-w} + 2\eta} \cdot \frac{n_{x=0}^{w} + \beta}{\sum_{w'=1}^{V} n_{x=0}^{w'} + V\beta} \right)^{1-x_{dn}}, \end{aligned} \tag{2.4}$$

where $n_{x=1}^{-w}$ and $n_{x=0}^{-w}$ denote the number of topic words and background words respectively, without considering the current word. $n_{x=0}^{w}$ is the number of times the current

word sampled as a background word, and V is the vocabulary size. Every time z_d or x_{dn} is re-sampled, the involved Gaussian topics would change and needs to be updated. Following the idea of [27], we can derive the updates for $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ of the posterior Gaussian distributions for topic k.

2.3 Experiments

In this section, we conduct experiments on real-world short texts to demonstrate the effectiveness of our model. This section details the dataset, the evaluation metric, baselines and the performance of our proposed model.

2.3.1 Dataset and Baselines

The dataset used for topic discovery is crawled from *abcnews*[1] . Based on the news categories in *abcnews* website, the documents in this dataset are divided into ten groups: Entertainment, Health, U.S., International, Law, Money, Politics, Sports, Technology, and Travel. In each category, there are 1000 news documents. Each document has a title and an optional description of the corresponding news article. The average length of the description, when available, is around 20 words - very short as compared to a regular document.

We use Latent Dirichlet Allocation (LDA) [10] and Gaussian-LDA [28] as the baselines to evaluate the performance of our topic discovery. Gaussian-LDA is first proposed for audio retrieval [27] and then used to leverage another kind of continuous data – word vectors to incorporate external semantics [28].

2.3.2 Experiment Setup

When learning word embeddings from Wikipedia, we set the dimensionality of word embeddings to 50, and the context window size to 12. This means when we are predicting the current word, its previous 6 words and subsequent 6 words contribute to the prediction. We train word embeddings with an iteration of 100 epochs.

As there are 10 categories in our news dataset, we are interested to see if the extracted topics can reveal a similar mixture. Hence we set the number of topics K to 10. For uniformity, all the baseline topic models are implemented with Gibbs sampling as well

[1] http://abcnews.go.com/

Table 2.1: Top 10 words in each topic for LDA.

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6	Topic 7	Topic 8	Topic 9	Topic 10
know	one	new	new	can	year	us	week	look	news
clinton	america	apple	years	episode	hotel	see	says	said	abc
police	back	today	time	full	cruise	airline	trump	first	family
need	first	facebook	day	best	vacation	3	presidential	game	latest
shooting	world	video	found	15	high	ceo	debate	season	big
year	obama	people	past	10	car	flight	john	state	star
hillary	made	1	birth	top	report	home	donald	last	get
everything	said	two	things	11	satisfaction	letter	new	win	ways
life	man	old	now	20	city	mom	president	homes	pope
inside	around	co	google	travel	four	plane	carson	4	save

Table 2.2: Top 10 words in each topic for Gaussian-LDA.

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6	Topic 7	Topic 8	Topic 9	Topic 10
presidential	back	clinton	hotel	leading	news	shooting	abc	latest	president
debate	apple	family	people	jobs	big	made	year	top	america
world	full	birth	abc	travel	ways	bringing	week	today	episode
candidate	flight	police	yaz	news	report	america	trump	pope	presidential
pope	time	life	part	homes	letter	inside	airline	found	house
york	control	man	years	latest	high	colorado	obama	long	game
time	plane	state	kids	microsoft	case	back	ceo	refugees	star
save	woman	woman	video	founder	things	found	cruise	called	season
talks	million	home	women	back	million	dead	vacation	recently	hunter
news	car	safety	rielle	top	family	police	hillary	remains	paris

Table 2.3: Top 10 words in each topic for our model.

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6	Topic 7	Topic 8	Topic 9	Topic 10
presidential	apple	star	year	america	captain	airline	episode	children	flight
clinton	world	world	game	infrastructure	control	hotel	full	cancer	police
campaign	officer	jenner	back	taskrabbit	birth	letter	15	mysterious	plane
2016	garrido	swift	google	business	wreck	satisfaction	pope	students	paris
trump	search	stars	big	company	yaz	complaint	star	doctors	attacks
candidate	failed	wars	family	microsoft	leading	cruise	shooting	rare	woman
president	mars	williams	life	jobs	pill	suite	francis	brain	refugees
debate	photo	photos	win	nokia	credit	vacation	20	price	city
hillary	shooting	bruce	abc	homeowners	identity	report	09	disease	turn
week	latest	opens	woman	myspaces	card	serving	week	symptoms	attack

and we perform 100 iterations of Gibbs sampling for all the models. We set $\eta = 20$, $\beta = 0.01$, and $\alpha = 50/K$. For the hyper parameters regarding Gaussian topics, we set prior $\boldsymbol{\mu}_0$ to the sample mean of all the word embeddings, the initial degree of freedom ν to the dimensionality of word embeddings, and assign an identity matrix to prior $\boldsymbol{\Psi}$.

As our *abcnews* dataset and Wikipedia are two different corpora, it's inevitable to encounter out-of-vocabulary words when extracting topics for *abcnews* dataset, i.e., some words in *abcnews* do not have corresponding word embeddings learnt from Wikipedia. We generate their word embeddings using the Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Psi})$.

2.3.3 Experimental Results on Topic Coherence

Usually perplexity is used as a measure to evaluate language models. But in our case, the probability of a word embedding is given by its probability density function rather than an exact probability. Furthermore, the probability of a background word is given by the discrete multinomial probability with respect to the background mode, and this disagreement between continuous probability density and discrete probability makes it incorrect and in fact infeasible to use perplexity in our analysis. Thus, we list top 10 words in each topic for LDA, Gaussian-LDA and our model on *abcnews* dataset with $K = 10$, as shown in Tables 2.1, 2.2 and 2.3. The words are ranked based on their frequency in each topic in the last round of sampling.

From Table 2.1, we can see that the topics extracted by LDA are not satisfying, and this is probably because of the limitation of document length. Only topic 6 and topic 8 are high-quality topics corresponding to Travel and Politics. Topic 1 looks to be loosely related to Law but with Politics mixed in. The other topics are not at all acceptable.

In Table 2.2, with the help of Wikipedia word embeddings, the extracted topics get more coherent. This validates the use of word embeddings in topic modelling. However, one can observe that the topics are still not as crisp as the ones we want. This is because of the fact that this model still treats the text as a mixture of many topics rather than a single topic.

Lastly we present the top words for our model - Table 2.3. As one can observe many news categories, Politics (topic 1), Entertainment (topic 3), Sports (topic 4), Technology (topic 5), Travel (topic 7), Health (topic 9) and International (topic 10), are successfully extracted with high quality keywords. It is worth noticing that Entertainment, Technology and Health have not been extracted by any of the baseline topic models,

Table 2.4: Comparison of document-topic distribution.

Model	Precision	Recall	F1
LDA	0.162	0.163	0.163
Gaussian-LDA	0.117	0.140	0.128
Our Model	**0.223**	**0.271**	**0.244**

and that our model even captures the names of entertainers in topic 3. Also we can still tell that topic 2 is somewhat related to Law, and topic 6 to Money.

2.3.4 Quality of Topic Representation of Documents

We see that the topics extracted by our model are more reasonable and have better qualities. But are the topics extracted by our model really corresponding to the coherent news categories? To answer this, we compare the category labels with the document-topic labels to see if they are consistent. The category label of each news article comes from the dataset and is used as the ground truth. The document-topic label of each news article is assigned by the models. More specifically, for LDA and Gaussian-LDA , we can assign one single topic to document d according to:

$$z_d = argmax_z p(z|d).$$

For our model, each document has only one topic according to the model assumptions. To solve the cluster matching problem, e.g., news category 1 may correspond to topic 9 instead of topic 1, we use pairwise comparison [29] to measure the consistency between news categories and extracted topic representation of documents. The pairwise comparison defined as:

$$\begin{aligned} precision(E,G) &= \frac{||pair_E \cap pair_G||}{||pair_E||}, \\ recall(E,G) &= \frac{||pair_E \cap pair_G||}{||pair_G||}, \\ F1(E,G) &= \frac{2 \times precision \times recall}{precision + recall}, \end{aligned}$$

where E and G are two clustering results corresponding to ten document-topic groups and ten ground truth categories respectively in our case, and $pair_E$ denotes the set of pairs in clustering result E. The result of this comparison is reported in Table 2.4. We

can see that, with respect to the consistency between news categories and extracted topics, our model outperforms the other baselines significantly.

2.4 Related Works

Topic models have been proposed to reveal the latent semantic structure from text corpus. Latent Semantic Analysis (LSA) [8] first tries to uncover the latent semantic information in a corpus by applying singular value decomposition to the document-term matrix. Probabilistic Latent Semantic Analysis (PLSA) [9] and Latent Dirichlet Allocation (LDA) [10] further use a hidden topic variable to capture the latent semantic structure and model documents as mixtures of topics, while topics are probability distributions over words. PLSA, LDA and their variants, such as the author topic model [30], have achieved huge success in analyzing normal texts. However, for short texts, such as tweets and news titles, conventional topic models usually don't work well due to the lack of rich context.

An intuitive way to handle this problem in short text corpus is to aggregate several short texts into one normal document based on auxiliary information before extracting topics. For instance, Weng et al. [31] utilize the user information of Twitter. They make an assumption similar to the author-topic model [30] that each user has a specific topic preference and then aggregate the tweets by the same user into one long document. Such aggregation methods can alleviate the lack of rich context problem and improve the performance of conventional topic models. However, such heuristic aggregation methods do not work in the scenarios where auxiliary information is not available. Take news titles as an example - there is no such auxiliary information as user name to utilize. Besides the assumption on user topic distribution, making assumption on the data is another way to tackle this problem. Zhao et al. [25] follow the assumption that a single tweet is usually about one single topic and further models each tweet as a variant of mixutre of unigrams. In [32], rather than each short document, they assume each sentence is about one topic.

However, these aforementioned models fail to leverage external semantics, which is quite helpful in dealing with the lack of rich context in short text corpus. Das et al. [28] first tries to combine topic modeling and word embeddings for regular texts, and further introduces a fast training method for it. Focusing on short texts, our model proposes a novel generative strategy utilizing word embeddings – modeling each short text as one

single Gaussian distribution over topic words and complementing continuous Gaussian topics with discrete multinomial background model. The word embeddings we used in our model is derived from the language models based on distributed representations of words. Those language models are mostly built on neural network structures. The distributed representation of words, i.e., word embedding, is first introduced into natural language processing by NPLM [23]. Many distributed language models with speed-up strategies, such as using tree structures, have been proposed to reduce the time complexity of NPLM [33, 24]. Mikolov et al. [24] proposed a Huffman tree based hierarchical neural network called Word2Vec, which significantly shortens the training time and is one of the most popular distributed language models currently in use.

2.5 Conclusions

In this chapter, we have proposed a topic model for short texts using word embeddings. Word embeddings learnt from external sources, such as Wikipedia, can bring supplemental semantics to short texts to overcome its lack of rich context. Hence, we model each short document as a Gaussian topic in the word embedding vector space. A short text is composed of not only topic words but also background words, we incorporate an alternative background mode to complement Gaussian topics. Considering that background words are not semantically related, background mode is implemented with discrete multinomial distribution over word types rather than in the word embedding space. The experiments validate the effectiveness of our model at discovering coherent topics from short text corpus.

Chapter 3

Exploiting Supplementary Local Context for Topic Correlations

3.1 Introduction

Conventional topic models, such as Probabilistic Latent Semantic Analysis (PLSA) [9] and Latent Dirichlet Allocation (LDA) [10], have proven to be a powerful unsupervised tool for the statistical analysis of document collections. Those methods [34], [35] follow the bag-of-word assumption and model each document as an admixture of latent topics, which are multinomial distributions over words.

A limitation of the conventional topic models is the inability to directly model correlations between topics, for instances, a document about autos is more likely to be related to motorcycles than to politics. In reality, it is natural to expect correlated latent topics in most text corpora. In order to address this limitation, the Correlated Topic Model (CTM) [36] replaces the Dirichlet prior with logistic normal distribution which allows for covariance structure among the topics.

Nowadays, the rapidly developing technique in natural language processing – word embeddings [12], [1] – provides us with the possibility to model topics and topic correlations in the continuous semantic space. Word embeddings, also known as word vectors and distributed representations of words, are real-valued continuous vectors for words, which have proven to be effective at capturing semantic regularities in language. Words with similar semantic and syntactic properties tend to be projected into nearby area in the vector space. By replacing the original discrete word types in LDA with continuous

word embeddings, Gaussian-LDA [28] has shown that the additional semantics in word embeddings can be incorporated into topic models and further enhance the performance.

The main goal of correlated topic models is to model and discover correlation between topics. And now we know that word embeddings are able to capture semantic regularities in language, and the correlations between words can be directly measured by the Euclidean distances or cosine values between the corresponding word embeddings. Moreover, semantically related words are close to each other in space and should be more likely to be grouped into the same topic. Since Gaussian distributions depict a notion of centrality in continuous space, it is a natural choice to model topics as Gaussian distributions over word embeddings in space. Therefore, the motivation of this chapter is to model topics in the word embedding space, exploit the known correlation information at word level and further improve the correlation discovery at topic level.

In this chapter, we propose the Correlated Gaussian Topic Model (CGTM) to model both topics and topic correlations in the word embedding space. More specifically, first we learn word embeddings with the help of external large unstructured text corpora to obtain additional word-level correlation information; Second, in the vector space of word embeddings, we model topics and topic correlations to exploit useful additional semantics in word embeddings, wherein each topic is represented as a Gaussian distribution over the word embeddings and topic correlations are learned among those Gaussian topics. Third, we develop a Gibbs sampling algorithm for CGTM.

To validate the efficacy of our proposed model, we evaluate our model on the 20 Newsgroups dataset and the Reuters-21578 dataset, which are well-known dataset for experiments in text mining domain. The experimental results show that our model can discover more reasonable topics and topic correlations than the baseline models.

3.2 Related Work

Correlation is an inherent property in many text corpora, for example, [37] explores the time evolution of topics and [38] analyzes the locational correlation among topics. However, due to the use of the Dirichlet prior, traditional topic models are not able to model the topic correlation directly. CTM [36] proposes to use logistic normal distribution to model the variability among topic proportions and thus learn the covariance structure of topics.

Word embeddings can capture the semantic meanings of words via low-dimensional

real-valued vectors [1], for example, vector operation vector('king') - vector('man') + vector('woman') results in a vector which is very close to vector('queen'). The concept of word embeddings was first introduced into natural language processing by Neural Probabilistic Language Model (NPLM) [12]. Due to its effectiveness and wide variety of application domains, word embeddings have garnered a great deal of attention and development [24], [2], [39], [40], [41], [42].

Since word embeddings carry additional semantics, many researchers have tried to incorporate them into topic models to improve the performance [28], [43], [44], [45]. [44] proposed Topical Word Embeddings (TWE) which combines word embeddings and topic models in a simple and effective way to achieve topical embeddings for each word. [28] uses Gaussian distributions to model topics in the word embedding space.

The aforementioned models either fail to directly model correlation among topics or fail to leverage the word-level semantics and correlations. We propose to leverage the word-level semantics and correlations within word embeddings to aid us in learning the topic-level correlations.

3.3 Learning Word Embeddings

We begin our topic discovery process with learning the word embeddings with semantic regularities. Unlike the traditional one-hot representations of words which encode each word as a binary vector of N (the size of vocabulary) digits with only one digit being 1 the others 0, the distributed representations of words encode each word as a unique real-valued vector. By mapping words into this vector space, word embeddings are able to overcome several drawbacks of the one-hot representations such as the curse of dimensionality, the out-of-vocabulary words and the lack of semantics.

In this chapter, we adopt a recently developed, very effective and efficient distributed representations of words based model called Word2Vec [1] to train word embeddings. In the learning process of Word2Vec, words with similar meanings gradually converge to nearby areas in the vector space. In this model, words in the form of word embeddings are used as input to a softmax classifier and each word is predicted based on its neighbourhood words within a certain context window.

Having learnt the word embeddings, given a word w_{dn}, which denotes the n^{th} word in d^{th} document, we can enrich that word by replacing it with the corresponding word embedding. The following section describes how this enrichment is used in a generative

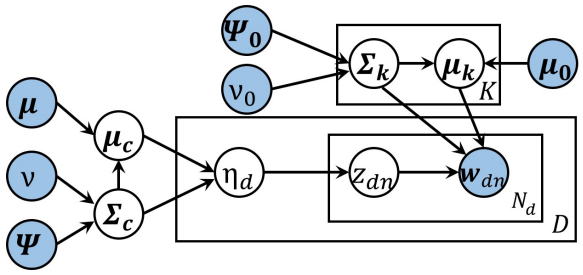


Figure 3.1: Schematic illustration of the CGTM framework.

process to model topics and topic correlations.

3.4 Generative Process

Trained word embeddings give us useful additional semantics, which helps us discover reasonable topics and topic correlations in the vector space. However, each document now is a sequence of continuous word embeddings instead of a sequence of discrete word types. Therefore, conventional topic models no longer are applicable. Since the word embeddings are located in space based on their semantics and syntax, inspired by [27] and [28], we consider them as draws from several Gaussian distributions. Hence, each topic is characterized as a multivariate Gaussian distribution in the vector space. The choice of Gaussian distribution is justified by the observations that Euclidean distances between word embeddings are consistent with their semantic similarities.

The graphical model of CGTM is shown in Figure 3.1. More formally, there are K topics and each topic is represented by a multivariate Gaussian distribution over the word embeddings in the word vector space. Let $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ denote the mean and covariance for the k^{th} Gaussian topic. Each document is a admixture of K Gaussian topics. $\boldsymbol{\eta_d}$ is a K dimensional vector where each dimension represents the weight of each topic in document d. Then the document-specific topic distribution $\boldsymbol{\theta_d}$ can be computed based on $\boldsymbol{\eta_d}$. $\boldsymbol{\mu}_c$ is the mean of $\boldsymbol{\eta}$ and $\boldsymbol{\Sigma}_c$ is the covariance of $\boldsymbol{\eta}$. By replacing the Dirichlet priors in conventional LDA with logistic normal priors, the topic correlation information is integrated into the model. $\boldsymbol{\mu}_0$, $\boldsymbol{\Sigma}_0$, ν_0, $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$ and ν are hyper parameters for Gaussian topics and logistic normal priors.

Note that variables in bold font mean they are either vectors or matrices, for example, $\boldsymbol{w}_{dn}$. The generative process is as follows:

1. Draw $\boldsymbol{\Sigma}_c \sim \mathcal{W}^{-1}(\boldsymbol{\Psi}, \nu)$.
2. Draw $\boldsymbol{\mu}_c \sim \mathcal{N}(\boldsymbol{\mu}, \frac{1}{\tau_c}\boldsymbol{\Sigma}_c)$.
3. For each Gaussian topic $k = 1, 2, \cdots, K$:
   (a) Draw topic covariance $\boldsymbol{\Sigma}_k \sim \mathcal{W}^{-1}(\boldsymbol{\Psi}_0, \nu_0)$.
   (b) Draw topic mean $\boldsymbol{\mu}_k \sim \mathcal{N}(\boldsymbol{\mu}_0, \frac{1}{\tau}\boldsymbol{\Sigma}_k)$.
4. For each document $d = 1, 2, \cdots, D$:
   (a) Draw $\boldsymbol{\eta}_d \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$.
   (b) For each word index $n = 1, 2, \cdots, N_d$:
      i. Draw a topic $z_{dn} \sim Multinomial(f(\boldsymbol{\eta}_d))$.
      ii. Draw a word $\boldsymbol{w}_{dn} \sim \mathcal{N}(\boldsymbol{\mu}_{z_{dn}}, \boldsymbol{\Sigma}_{z_{dn}})$.

where τ and τ_c are constant factors; and $f(\boldsymbol{\eta})$ is the logistic transformation:

$$f(\eta_d^k) = \theta_d^k = \frac{\exp(\eta_d^k)}{\sum_i \exp(\eta_d^i)}. \tag{3.1}$$

The following conjugate priors are utilized for topic parameters: a Gaussian distribution $\mathcal{N}$ for the mean and an inverse Wishart distribution $\mathcal{W}^{-1}$ for the covariance. However, note that there is still a non-conjugacy problem between the logistic normal distribution and multinomial distribution, and we will solve this with data augmentation technique in the following section.

3.5 Parameter Inference

The observed variables are documents consisting of word embeddings, and our goal is to infer the posterior Gaussian distribution of each topic, topic assignment of each word, and topic correlations. Given D documents and the corresponding word embeddings $\boldsymbol{w}$, the joint distribution of topic assignments $\boldsymbol{z}$ and logistic normal parameters $\boldsymbol{\eta}$ is:

$$\begin{aligned} p(\boldsymbol{z}, \{\boldsymbol{\eta}_d\}_{d=1}^D | \boldsymbol{w}) &\propto p(\boldsymbol{w}|\boldsymbol{z}) \prod_{d=1}^{D} (\prod_{n=1}^{N_d} \theta_d^{z_{dn}}) \mathcal{N}(\boldsymbol{\eta}_d | \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \\ &\propto p(\boldsymbol{w}|\boldsymbol{z}) \prod_{d=1}^{D} (\prod_{n=1}^{N_d} \frac{\exp(\eta_d^{z_{dn}})}{\sum_i^K \exp(\eta_d^i)}) \mathcal{N}(\boldsymbol{\eta}_d | \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c), \end{aligned} \tag{3.2}$$

where $p(\boldsymbol{w}|\boldsymbol{z})$ is the Gaussian probability of words $\boldsymbol{w}$ under topic assignments $\boldsymbol{z}$. Because of the choice of conjugate priors for topic parameters, those variables can be integrated out and we can efficiently re-sample topic assignment for each word. However, due to the non-conjugacy between the logistic normal and multinomial distributions, regular Gibbs sampling scheme doesn't work for the logistic normal parameters. Thus we adopt Gibbs sampling with data augmentation technique to solve this non-conjugacy problem.

3.5.1 Sampling Topic Assignments

Since the topic parameters have conjugate priors, the sampling process of topic assignments is similar to the Gibbs sampling scheme for LDA [46]. Given $\boldsymbol{\eta}$ and $\boldsymbol{z}_{-dn}$ which is the topic assignment scheme without considering the current word w_{dn}, the topic of each word is drawn iteratively as:

$$\begin{aligned} p(z_{dn} = k|\boldsymbol{z}_{-dn}, \boldsymbol{w}) &\propto p(z_{dn} = k|\boldsymbol{z}_{-dn}) p(\boldsymbol{w}_{dn}|z_{dn} = k) \\ &\propto \frac{\exp(\eta_d^k)}{\sum_i \exp(\eta_d^i)} \cdot T_r(\boldsymbol{w}_{dn}|\boldsymbol{\mu}_k, \frac{\tau_k + 1}{\tau_k}\boldsymbol{\Sigma}_k), \end{aligned} \tag{3.3}$$

where $T_r(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the multivariate Student's t-distribution for Gaussian sampling with $(r = \nu_k - dim + 1)$ being its degrees of freedom and dim being the dimensionality of word embeddings. $(\nu_k = \nu + N_k)$ and $(\tau_k = \tau + N_k)$ are the parameters of topic k, where N_k denotes the total number of words that are assigned to topic k.

3.5.2 Updating Gaussian Topics

Every time we re-sample topic assignment z_{dn}, we need to update the two involved Gaussian topics because the current word w_{dn} is either leaving or joining this Gaussian topic. Following [28], we derive the updates for $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ of the posterior Gaussian distributions for topic k:

$$\begin{aligned} \boldsymbol{\mu}_k &= \frac{\tau\boldsymbol{\mu}_0 + N_k\bar{\boldsymbol{w}}_k}{\tau_k}, \\ \boldsymbol{\Sigma}_k &= \frac{\boldsymbol{\Psi}_0 + \boldsymbol{C}_k + \tau N_k(\bar{\boldsymbol{w}}_k - \boldsymbol{\mu}_0)(\bar{\boldsymbol{w}}_k - \boldsymbol{\mu}_0)^T/\tau_k}{\nu_k - dim + 1}, \end{aligned} \tag{3.4}$$

where $\bar{\boldsymbol{w}}_k$ is the sample mean of all the word embeddings assigned to topic k, and $\boldsymbol{C}_k$ is the scaled form of sample covariance of all the word embeddings assigned to topic k.

These two intermediate variables are calculated as follows:

$$\begin{aligned}\bar{\boldsymbol{w}}_k &= \frac{\sum_{d=1}^{D}\sum_{n=1}^{N_d}\delta(z_{dn},k)\boldsymbol{w}_{dn}}{N_k},\\ \boldsymbol{C}_k &= \sum_{d=1}^{D}\sum_{n=1}^{N_d}\delta(z_{dn},k)(\boldsymbol{w}_{dn}-\bar{\boldsymbol{w}}_k)(\boldsymbol{w}_{dn}-\bar{\boldsymbol{w}}_k)^T,\end{aligned} \tag{3.5}$$

where $\delta(z_{dn},k)$ is the Kronecker delta function that $\delta(z_{dn},k)=1$ if $z_{dn}=k$, $\delta(z_{dn},k)=0$ otherwise.

3.5.3 Sampling Logistic Normal Parameters

Given topic assignments, directly sampling logistic normal parameters $\boldsymbol{\eta}$ is difficult due to non-conjugacy. To address the non-conjugacy problem between the logistic normal distribution and multinomial distribution, following [47], [48] and [49], we sample the logistic normal parameters $\boldsymbol{\eta}$ based on $\boldsymbol{z}$ with auxiliary variables. For document d, the likelihood for η_d^k conditioned on $\boldsymbol{\eta}_d^{-k}$ is:

$$l(\eta_d^k|\boldsymbol{\eta}_d^{-k}) = \prod_{n=1}^{N_d}\left(\frac{\exp(\eta_d^k)}{\sum_i\exp(\eta_d^i)}\right)^{z_{dn}^k}\left(1-\frac{\exp(\eta_d^k)}{\sum_i\exp(\eta_d^i)}\right)^{1-z_{dn}^k} = \frac{(\exp(\rho_d^k))^{C_d^k}}{(1+\exp(\rho_d^k))^{N_d}}, \tag{3.6}$$

where z_{dn}^k is the topic indicator that $z_{dn}^k=1$ if word w_{dn} is assigned to k^{th} topic, $z_{dn}^k=0$ otherwise. $\rho_d^k=\eta_d^k-\zeta_d^k$, $\zeta_d^k=\log(\sum_{j\neq k}\exp(\eta_d^j))$ and C_d^k is the number of words assigned to topic k in document d. Therefore, we obtain the posterior distribution of η_d^k proportional to multiplying the likelihood by the prior:

$$p(\eta_d^k|\boldsymbol{\eta}_d^{-k},\boldsymbol{z},\boldsymbol{w}) \propto l(\eta_d^k|\boldsymbol{\eta}_d^{-k})\mathcal{N}(\eta_d^k|\mu_d^k,\sigma_k^2). \tag{3.7}$$

For the prior part, it is a univariate Gaussian distribution conditioned on the other logistic normal parameters in the current document $\boldsymbol{\eta}_d^{-k}$. Thus, given $\boldsymbol{\eta}_d^{-k}$ and $\boldsymbol{\mu}_c$, $\boldsymbol{\Sigma}_c$ of the multivariate Gaussian distribution over $\boldsymbol{\eta}$, we have:

$$\begin{aligned}\mu_d^k &= \mu_k - \boldsymbol{\Lambda}_{kk}^{-1}\boldsymbol{\Lambda}_{k-k}(\boldsymbol{\eta}_d^{-k}-\boldsymbol{\mu}_{-k}),\\ \sigma_k^2 &= \boldsymbol{\Lambda}_{kk}^{-1},\end{aligned} \tag{3.8}$$

where $\boldsymbol{\Lambda}=\boldsymbol{\Sigma}_c^{-1}$ is the precision matrix. However, the non-conjugacy makes it difficult to directly calculate the likelihood $l(\eta_d^k|\boldsymbol{\eta}_d^{-k})$ and thus unable to directly sample η_d^k.

By introducing auxiliary Polya-Gamma variable λ_d^k [48], we are able to get around the non-conjugacy problem and the likelihood $l(\eta_d^k|\boldsymbol{\eta}_d^{-k})$ can now be expressed as:

$$l(\eta_d^k|\boldsymbol{\eta}_d^{-k}) = \frac{1}{2^{N_d}} \exp(\kappa_d^k \rho_d^k) \int_0^\infty \exp(-\frac{\lambda_d^k (\rho_d^k)^2}{2}) p(\lambda_d^k|N_d, 0) d\lambda_d^k, \tag{3.9}$$

where $\kappa_d^k = C_d^k - N_d/2$ and $p(\lambda_d^k|N_d, 0)$ is the Polya-Gamma distribution $\mathcal{PG}(N_d, 0)$. As one can observe, Equation 3.9 implies that $p(\eta_d^k|\boldsymbol{\eta}_d^{-k}, \boldsymbol{z}, \boldsymbol{w})$ is the marginal distribution of the joint distribution:

$$p(\eta_d^k, \lambda_d^k|\boldsymbol{\eta}_d^{-k}, \boldsymbol{z}, \boldsymbol{w}) \propto \frac{1}{2^{N_d}} \exp(\kappa_d^k \rho_d^k - \frac{\lambda_d^k (\rho_d^k)^2}{2}) p(\lambda_d^k|N_d, 0) \mathcal{N}(\eta_d^k|\mu_d^k, \sigma_k^2). \tag{3.10}$$

Therefore we can sample η_d^k based on the auxiliary variable λ_d^k. The sampling procedure is as follows:

- Sampling λ_d^k: according to Equation 3.10 and [48], we have the conditional distribution $p(\lambda_d^k|\boldsymbol{z}, \boldsymbol{w}, \boldsymbol{\eta}) \propto \exp(-\frac{\lambda_d^k (\rho_d^k)^2}{2}) p(\lambda_d^k|N_d, 0)$, which results in a Polya-Gamma distribution $\mathcal{PG}(N_d, \rho_d^k)$. Following the ideas in [48] and [49], Polya-Gamma variables can be drawn in O(1) time, and so a sample of λ_d^k is obtained.

- Sampling η_d^k: according to Equation 3.10, we can sample η_d^k with posterior probability:

$$p(\eta_d^k|\boldsymbol{\eta}_d^{-k}, \boldsymbol{z}, \boldsymbol{w}, \lambda) \propto \exp(\kappa_d^k \eta_d^k - \frac{\lambda_d^k (\eta_d^k)^2}{2}) \mathcal{N}(\eta_d^k|\mu_d^k, \sigma_k^2). \tag{3.11}$$

  This results in a univariate Gaussian distribution $\mathcal{N}(\gamma_d^k, (\tau_d^k)^2)$ conditioned on the auxiliary variable λ_d^k, where $\gamma_d^k = (\tau_d^k)^2(\sigma_d^{-2}\mu_d^k + \kappa_d^k + \lambda_d^k \zeta_d^k)$ and $(\tau_d^k)^2 = (\sigma_d^{-2} + \lambda_d^k)^{-1}$. Thus, given the auxiliary variable λ_d^k, η_d^k can be easily drawn from a univariate Gaussian distribution.

3.5.4 Updating Topic Correlation

Given $\{\boldsymbol{\eta}_d\}_{d=1}^D$, the logistic normal parameters $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$ are updated as:

$$\begin{aligned} \boldsymbol{\mu}_c &= \frac{\tau_c}{\tau_c + D}\boldsymbol{\mu} + \frac{D}{\tau_c + D}\bar{\boldsymbol{\eta}}, \\ \boldsymbol{\Sigma}_c &= \boldsymbol{\Psi} + Q + \frac{\tau_c D}{\tau_c + D}(\bar{\boldsymbol{\eta}} - \boldsymbol{\mu})(\bar{\boldsymbol{\eta}} - \boldsymbol{\mu})^T, \end{aligned} \tag{3.12}$$

where $\bar{\boldsymbol{\eta}}$ is the mean of $\{\boldsymbol{\eta}_d\}_{d=1}^D$, and $Q = \frac{1}{D}(\boldsymbol{\eta}_d - \bar{\boldsymbol{\eta}})(\boldsymbol{\eta}_d - \bar{\boldsymbol{\eta}})^T$.

3.6 Experiments

In this section, we carry out experiments on two real-world text collections – the 20 Newsgroups dataset[1] and the Reuters-21578 dataset[2] to demonstrate the efficacy of our proposed model. 20 Newsgroups contains approximately 20,000 text documents partitioned evenly across 20 different newsgroups. Reuters contains about 10,000 documents, but due to the imbalance of each category, only the largest 8 categories are selected in Reuters, leaving us with 7,674 documents in total. Both datasets have become popular datasets for experiments in many data mining tasks, such as text classification. Each document is associated with one single category label. For 20 Newsgroups, correlation is exhibited across different newsgroups (e.g. rec.sport.baseball and rec.sport.hockey), which makes this dataset a suitable choice to verify the effectiveness of topic correlation discovery for CGTM.

We compare CGTM with three topic modeling methods: LDA [10], CTM [36] and Gaussian-LDA [28]. CTM replaces the dirichlet prior in LDA with logistic normal distribution to capture the correlation among topic proportions. Gaussian-LDA was first proposed for audio retrieval [27] and then used to leverage word embeddings in the continuous vector space [28].

To learn high quality word embeddings, we combine the current dataset with Wikipedia as the knowledge source. The motivation of using Wikipedia as the supplemental source lies in the sheer range of topics and subjects that are covered and it allows us to enhance the semantics of word embeddings extracted from 20 Newsgroups and Reuters. In the experiment, we set the dimensionality of word embeddings to 100, and the context window size to 12. We train word embeddings for 100 epochs.

We are interested to see if the learned topics can reveal a similar mixture and correlation with the ground truth text categories. Hence we set the number of topics K to the number of categories. For uniformity, all the models are implemented with Gibbs sampling and run for 100 iterations. The Gaussian topic hyper parameter $\boldsymbol{\mu}_0$ is set to the sample mean of all the word vectors, the initial degree of freedom ν_0 to the dimensionality of word embeddings, and $\boldsymbol{\Psi}_0$ to an identity matrix.

[1] www.qwone.com/ jason/20Newsgroups/
[2] www.daviddlewis.com/resources/testcollections/reuters21578/

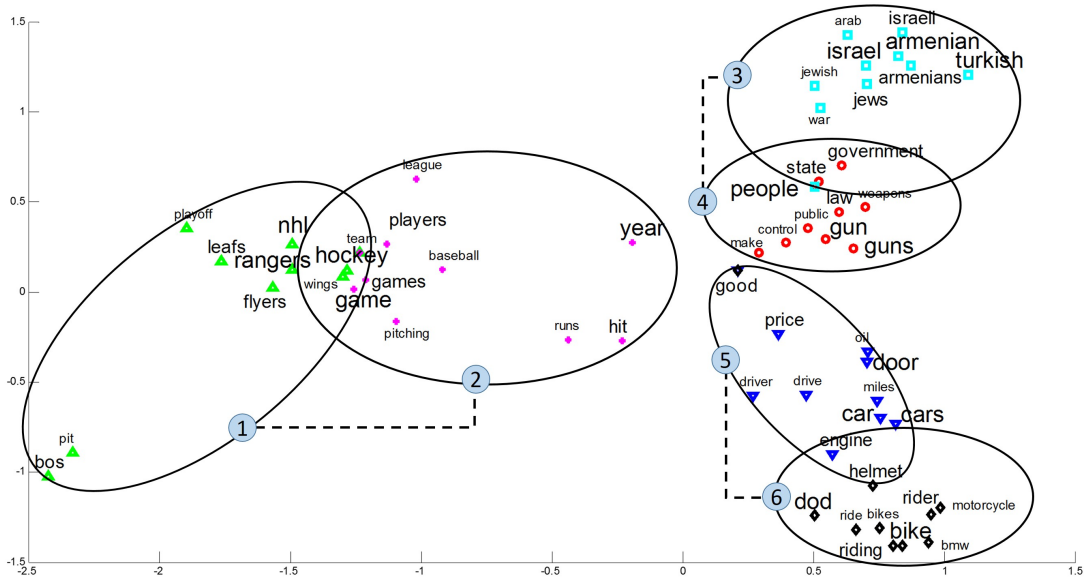


Figure 3.2: Topic words and correlations.

3.6.1 Topic Words and Correlations

To investigate the quality of topics and the topic correlations discovered by CGTM, we visualize each topic with their top words as well as the topic correlations. To make the visualization clearer, we select only 6 categories from the 20 Newsgroups dataset whose topic words and correlations can be easily recognized and defined. The selected newsgroups are "rec.autos", "rec.motorcycles", "rec.sport.baseball", "rec.sport.hockey", "talk.politics.guns", and "talk.politics.mideast". Thus, in this experiment, we set the number of topics K to 6. As Figure 3.2 shows, we display top 10 words for each topic discovered by CGTM and map the corresponding word embeddings into a two-dimensional space via Principal Component Analysis (PCA). The size of each word varies with its relative frequency in the corresponding topic. The different colors and shapes of words indicate they are from 6 different topics. Each circle depicts the Gaussian distribution for each topic. The detected topic correlation is represented as a dashed line between topics. As one can observe, all the newsgroups, Hockey (topic 1), Baseball (topic 2), Mideast (topic 3), Guns (topic 4), Autos (topic 5) and Motorcycles (topic 6), are successfully discovered with reasonable topic words.

As the ground truth labels indicate, one can easily figure that Autos is correlated with Motorcycles, Baseball is correlated with Hockey, and Guns is correlated with

Table 3.1: Comparison of topic coherence scores.

Top T words	5	10	20	50
LDA	-13.86	-64.11	-322.07	-2384.68
CTM	-13.77	-64.49	-323.71	-2395.58
Gaussian-LDA	-14.83	-66.31	-323.91	-2505.33
CGTM	**-12.37**	**-60.48**	**-317.43**	**-2362.75**

Mideast. The dashed lines in the figure denote the automatically detected topic correlations by CGTM. With the help of word embeddings and Gaussian topics, topic correlations are also correctly detected, as the dashed lines show. We can see that, since word embeddings can capture the regularities in language such as synonyms, two topics tend to be correlated if their topic word embeddings overlap in the continuous vector space. This demonstrates how the known word-level correlation information can aid us in discovering the topic-level correlations.

In this subsection, we qualitatively exhibit the effectiveness of discovering topics and topic correlations of CGTM. In the following subsections, we will quantitatively evaluate CGTM on topic coherence and topic correlation discovery.

3.6.2 Topic Coherence

In order to quantitatively assess the topic coherence, we adopt a metric called coherence score of topics proposed by [50] which is able to automatically evaluate the coherence of each discovered topic. Given a topic z and its top T words $V^z = \{v_1^z, v_2^z, ..., v_T^z\}$, the coherence score of this topic is defined as:

$$C(z; V^z) = \sum_{t=2}^{T} \sum_{l=1}^{t} \log \frac{D(v_t^z, v_l^z) + 1}{D(v_l^z)}, \tag{3.13}$$

where $D(v_l^z)$ is the document frequency of word v_l^z and $D(v_t^z, v_l^z)$ is the number of documents in which words v_t^z and v_l^z co-occurred. The coherence score follows the intuition that words from the same topic tend to co-occur in documents. This topic coherence score has been proven to be highly consistent with human coherence judgements [50].

The topic coherence result on the 20 Newsgroups dataset is reported in Table 3.1. In order to investigate the overall quality of all the discovered topics, the average coherence score is reported, which is calculated as $\bar{C} = \frac{1}{K} \sum_z C(z; V^z)$. To make this evaluation

more comprehensive, the number of topic words T ranges from 5 to 50. For all the models, the topic words are ordered by word counts in each topic. Though for Gaussian-LDA and CGTM, topic words can also be ordered by word probabilities under each Gaussian topic, we still order them by word counts, since first, the Gaussian posterior probability information has already been fully utilized in the training phase and second, this coherence score is more appropriate to measure frequent words in a topic. The result shows that the topic words discovered by our model are more coherent than the topic words discovered by the baseline models.

3.6.3 Document Topics and Topic Correlation

We see that the topics discovered by CGTM qualitatively exhibit good topic words and reasonable correlations, and CGTM also outperforms the baseline models in terms of coherence score. But are the topics discovered by our model really corresponding to the coherent news categories? If yes, it would be very convenient for us to assess the quality of the detected topic correlations, because the correlations among the ground truth newsgroups labels are well defined. For example, 20 Newsgroups categories "rec.autos" and "rec.motorcycles" are clearly correlated, and Reuters categories "money" and "trade" should also exhibit correlations. To answer this question, we compare the ground truth document labels with the document-topic labels discovered by the models to see if they are consistent. The label of each document comes from the dataset and is used as the ground truth. The document-topic label of each document is assigned by the models. More specifically, for each model, we can assign one single topic to document d according to:

$$z_d = argmax_z p(z|d).$$

So this is a clustering evaluation problem where each document is a sample. To solve the cluster matching problem, e.g., ground truth label 1 may correspond to topic 5 instead of topic 1, we adopt pairwise comparison [29] to measure the consistency between the ground truth document labels and the learned topic representation of documents.

Table 3.2: Comparison of document-topic distribution on the 20 Newsgroups dataset.

Model	Precision	Recall	F1
LDA	0.438	0.507	0.470
CTM	0.447	**0.634**	0.524
Gaussian-LDA	0.438	0.496	0.465
CGTM	**0.523**	0.623	**0.568**

Table 3.3: Comparison of document-topic distribution on the Reuters dataset.

Model	Precision	Recall	F1
LDA	0.844	0.392	0.535
CTM	0.796	**0.433**	0.561
Gaussian-LDA	0.865	0.405	0.552
CGTM	**0.870**	0.431	**0.576**

The pairwise comparison is defined as:

$$precision(E,G) = \frac{||pair_E \cap pair_G||}{||pair_E||},$$
$$recall(E,G) = \frac{||pair_E \cap pair_G||}{||pair_G||},$$
$$F1(E,G) = \frac{2 \times precision \times recall}{precision + recall},$$

where E and G are two clustering solutions corresponding to the document-topic clusters and the ground truth document labels respectively in our case, and $pair_E$ denotes the set of pairs in clustering result E, and $||pair_E||$ represents the number of instances in $pair_E$. The experimental results of document clustering on 20 Newsgroups and Reuters are reported in Table 3.2 and Table 3.3 respectively. We can see that, with respect to the consistency between ground truth document labels and discovered topics, CGTM outperforms the other baselines on both datasets.

3.7 Conclusions

In this chapter, we have proposed a correlated topic model using word embeddings. Word embeddings learnt from large, unstructured corpora, such as Wikipedia, can aid

us in modeling topics and topic correlation by bringing in additional useful semantics. The known word-level correlation information in word embeddings is passed to topic-level correlation discovery via Gaussian topics. In our case, the word embeddings are trained on the combined collections of Wikipedia and the 20 newsgroups dataset. We model each topic as a Gaussian distribution over word embeddings and directly learn topic correlations in the vector space. The experiments qualitatively show CGTM is able to learn meaningful topics and topic correlation, and quantitatively validate the effectiveness of our model in terms of topic coherence score and document clustering on two real-world datasets.

Chapter 4

Coordinating Global and Local Context

4.1 Introduction

Topic models [8, 9, 10, 25] and word embedding models [23, 51, 1] are two of the most successful and prevalent language models nowadays. They model languages from two different but complementary points of view — the global viewpoint and the local viewpoint. Topic models, such as Probabilistic Latent Semantic Analysis (PLSA) [9] and Latent Dirichlet Allocation (LDA) [10], are usually built upon the document-level global context information in a text corpus. Topic models follow the bag-of-word assumption that a document is represented as a bag of its words (disregarding grammar and even word order, but keeping multiplicity). Documents are modeled as mixtures of latent topics, where latent topics are formulated as multinomial distributions over words. Bayesian methods are then employed to infer the topic structures based on the global document-word frequency matrix of the corpus. In contrast to topic models utilizing the document-level global context information, most of the word embedding models, such as Neural Probabilistic Language Model (NPLM) [23] and Skip-Gram [1], are based on the local context information. Word embedding models follow the distributional hypothesis [7] that words occurring in similar local contexts tend to have similar syntactic and semantic properties. Semantically related words ought to be projected close to each other in the word embedding space. Word embeddings can then be constructed using internal representations from neural network architectures of local

word sequences.

While local context can help disambiguate word meanings, global context can also provide useful topical information. Therefore, it is natural to expect more sufficient input information and better performance if a language model is able to utilize these two complementary context information collaboratively. In addition, both topic structures and word embeddings can be discovered from the corpus. However, it is difficult to develop a unified language model in depth which can absorb both the idea of globality from topic models and the idea of locality from word embedding models, because topic models are usually statistical generative models while word embedding models are mostly based on artificial neural networks.

Instead of developing a unified language model, researchers tend to combine global and local context by using the pre-trained result based on one type of context information to assist in modeling language on the other type of context information [52, 53, 28, 44]. For example, Gaussian-LDA [28] uses pre-trained word embeddings learned from large external corpora such as Wikipedia and then models topics with Gaussian distributions in the word embedding space; in contrast, Topical Word Embedding (TWE) [44] uses pre-trained topic structures to learn topic embeddings and improve word embeddings.

However, there are several limitations in combining topic models and word embedding models in this separate and heuristic manner. The first limitation stems from the difference of semantics in different datasets. One popular way to model topics upon word embeddings is to replace discrete word types in the target dataset (e.g. ESPN sports news) with continuous word embeddings learned from an external corpus (e.g. Wikipedia). But the difference of semantics in ESPN sports news and Wikipedia would probably result in poor performance. The second limitation is that external corpus is not always available. In some research domains such as biomedicine, there are no such knowledge bases as comprehensive as Wikipedia. However, if we want to directly train word embeddings on the target dataset and the target dataset is relatively small, we have to face the third limitation, i.e., the lack of local context information, as training word embeddings typically requires a large amount of local contexts.

The aforementioned limitations inspire us to develop a unified language model which is able to make use of both the global and the local context information collaboratively. We propose to unify the process of modeling topics and learning word embeddings via matrix factorization, and take advantage of both the idea of globality from topic models

and the idea of locality from word embedding models. The new model is named Collaborative Language Model (CLM). In CLM, the global context information is encoded in the document-word matrix and the local context information is encoded in the word co-occurrence matrix. In addition to topic structures and word embeddings, we also introduce topic embeddings for topics and assume that the importance of a word in a topic is proportional to the inner product value of the corresponding word embedding and topic embedding. By fully exploiting the context information in a text corpus, CLM has the following advantages:

- CLM is able to discover topic structures and learn word embeddings collaboratively.
- CLM does not rely on pre-trained topic structures or pre-trained word embeddings learned from external corpora.
- When the text corpus is not large enough, with the help of global context information, CLM can overcome the lack of local context information and learn good word embeddings.
- With the help of local context information, CLM can discover more coherent latent topics.

To evaluate how well CLM discovers topics and learns word embeddings, we perform four quantitative evaluation tasks including two topic structure evaluation tasks and two word embedding evaluation tasks. We show that by taking both global and local context information into consideration, CLM outperforms the baselines on both the topic structure evaluation tasks and the word embedding evaluation tasks. We also provide qualitative assessment and case studies to explain how topic structures and word embeddings can collaboratively enhance the quality of each other.

4.2 Related Work

Topic models are a powerful unsupervised tool to reveal the latent semantic structure from a text corpus based on its global document-word context information. Latent Semantic Analysis (LSA) [8] is proposed as a dimensionality reduction technique by projecting the document-word matrix to a linear subspace with Singular Value Decomposition (SVD). PLSA [9] introduces latent variables which can be viewed as 'topics'

between documents and words, where each document is a multinomial distribution over topics and each topic is also a multinomial distribution over words. LDA [10] further extends PLSA to a complete probabilistic model by adding Dirichlet priors at the document level. Non-negative Matrix Factorization (NMF) [54] is a useful matrix decomposition technique for multivariate data and its non-negativity makes the resulting matrices easy to explain in many application domains. Ding et al. [55] have proven the equivalence between PLSA and NMF as they optimize the same objective function of the global document-word matrix.

Word embeddings, also known as word vectors and distributed representations of words, have proven to be able to capture semantic regularities in language by learning the local word co-occurrence context information. Specifically, NPLM [23] first introduces word embeddings into natural language processing. Many variants have been proposed since then to improve the efficiency of NPLM [51, 1]. In particular, the popular Continuous Bag Of Words (CBOW) and Skip-Gram models proposed by Mikolov et al. [1] are efficient to train and obtain state-of-the-art results on various linguistic tasks. The training methods of CBOW and Skip-Gram are highly popular, but not well understood until Levy et al. [56] proved that Skip-Gram with negative sampling training method is implicitly factorizing the pointwise mutual information (PMI) matrix of the local word co-occurrence patterns.

In order to make use of both the global context and the local context information, many composite models have been proposed to combine topic models and word embedding models. One common way is to use pre-trained word embeddings and replace the multinomial distribution over words with a probability function defined in the word embedding space to generate a focus word given its topic and neighboring words. Among them, Latent Feature Topic Modeling (LFTM) [57] defines the probability function as a mixture of the conventional multinomial distribution and a link function between the embeddings of the focus word and topics. TopicVec [43] adds context word embeddings to the link function in addition to the focus word embeddings and topic embeddings. Gaussian-LDA [28] models topics as Gaussian distributions over the continuous word embeddings. The other common way to combine topic models and word embedding models is to use pre-trained LDA topic structures to learn topic embeddings and assist in training word embeddings. For example, Topic2Vec [58] treats the pre-trained topic labels as special words and learns embeddings for topics by including the topic labels in the neural network architecture. Topical Word Embedding (TWE) [44] further

concatenates the topic embedding with the word embedding to form the topical word embedding for each word.

However, all of these composite models combine topic models and word embedding models in a separate and heuristic manner – they either utilize pre-trained word embeddings or pre-trained topic structures. In contrast, our CLM model proposes to make the topic model and the word embedding model work collaboratively, and fully exploit the complementary global and local context information in a text corpus. Huang et al. [42] and Le et al. [59] propose to incorporate global information to help the learning of word embeddings by assigning an embedding to each document. Their ideas can be viewed as special cases of our model, with the number of topics set to the number of documents.

4.3 Notations and Definitions

Table 4.1 shows the notations used in this chapter. We use bold uppercase letters such as $\boldsymbol{D}$ to represent matrices, bold lowercase letters such as $\boldsymbol{d}_n$ to represent vectors or embeddings, regular uppercase letters such as V to represent scalar constants, and regular lowercase letters such as d_{ij} to represent scalar variables.

Given a text corpus, its document-level global context information is encoded in the document-word matrix $\boldsymbol{D}$ and its local context information is encoded in the word co-occurrence matrix $\boldsymbol{W}$. The word co-occurrence matrix $\boldsymbol{W}$ is constructed from small fixed-sized text intervals in the documents. Each text interval is composed of a focus word and its neighboring context words falling in a fix-sized window centered at the focus word. The value of entry w_{ij} is the number of times that a context word w_j appears in word w_i's contexts.

Given the global context matrix $\boldsymbol{D}$ and the local context matrix $\boldsymbol{W}$, our goal is to discover topic structures and learn word embeddings collaboratively based on both context information.

4.4 Methodology

Our CLM model follows three basic assumptions: (1) each document focuses on only a small amount of topics and each topic assigns high probability to only a small number of words; (2) words appearing in similar local context tend to have similar syntactic and

Table 4.1: Table of notations.

Notation	Meaning
V	Vocabulary size
N	Number of documents
K	Number of topics
M	Dimensionality of the embedding space
$\boldsymbol{D} \in \mathbb{R}^{V \times N}$	Document-word matrix $[\boldsymbol{d}_1, ..., \boldsymbol{d}_N]$
$\boldsymbol{W} \in \mathbb{R}^{V \times V}$	Word co-occurrence matrix
$\boldsymbol{T} \in \mathbb{R}^{K \times V}$	Topic-word matrix $[\boldsymbol{t}_1, ..., \boldsymbol{t}_V]$
$\boldsymbol{\Theta} \in \mathbb{R}^{K \times N}$	Document-topic matrix $[\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_N]$
$\boldsymbol{A} \in \mathbb{R}^{M \times K}$	Topic embedding matrix $[\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_K]$
$\boldsymbol{B} \in \mathbb{R}^{M \times V}$	Word embedding matrix $[\boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_V]$
$\boldsymbol{C} \in \mathbb{R}^{M \times V}$	Context word embedding matrix $[\boldsymbol{c}_1, ..., \boldsymbol{c}_V]$
$\boldsymbol{d}_n$	The n^{th} document
$\boldsymbol{\theta}_n$	Topic representation for the n^{th} document
$\boldsymbol{t}_v$	Topic distribution for the v^{th} word
$\boldsymbol{\alpha}_k$	Topic embedding for the k^{th} topic
$\boldsymbol{\beta}_v$	Word embedding for the v^{th} word
$\boldsymbol{c}_v$	Context embedding for the v^{th} word
d_{ij}, w_{ij}, t_{ij}	The ij^{th} entry in matrix $\boldsymbol{D}, \boldsymbol{W}, \boldsymbol{T}$ respectively

semantic properties and should be mapped to nearby areas in the embedding space; and (3) words close to each other in the embedding space tend to have similar topic distributions and vice versa.

We will introduce our CLM model according to how CLM is formulated based on these assumptions as well as how CLM utilizes the global and the local context information.

4.4.1 Our Proposed Model

The three aforementioned assumptions correspond to three building blocks of CLM. We first introduce the three building blocks. Then we describe our proposed model CLM and its relationship with other existing composite models.

Exploiting global context information. Given the global document-word matrix $\boldsymbol{D}$, NMF decomposes it into the product of document-topic matrix $\boldsymbol{\Theta}$ and topic-word

matrix $\boldsymbol{T}$. The non-negativity of NMF ensures the explainability of document-topic distribution and topic-word distribution. The objective function to factorize $\boldsymbol{D}$ with regularization is:

$$\begin{aligned} L_{glo} &= ||\boldsymbol{D} - \boldsymbol{T}^T\boldsymbol{\Theta}||_2^2 + \lambda_s||\boldsymbol{\Theta}||_2^2 + \lambda_s||\boldsymbol{T}||_2^2, \\ &\textit{subject to}: \\ &\boldsymbol{\Theta} \geq 0 \textit{ and } \boldsymbol{T} \geq 0, \end{aligned} \tag{4.1}$$

where $||\boldsymbol{\Theta}||_2^2$ denotes the l_2 norm regularization we use on document-topic matrix $\boldsymbol{\Theta}$, $||\boldsymbol{T}||_2^2$ denotes the l_2 norm regularization we use on topic-word matrix $\boldsymbol{T}$, and λ_s is the parameter to prevent overfitting: the larger value of λ_s, the larger amount of shrinkage on $\boldsymbol{\Theta}$ and $\boldsymbol{T}$. In our model, instead of the raw frequency matrix, we use the document-word matrix with TF-IDF weights as $\boldsymbol{D}$.

Exploiting local context information. Based on the local context information, word embedding models can learn a low-dimensional representation for each word. In the Skip-Gram model [1], the objective of each training step is to predict neighboring words within a fixed window given a focus word. Stochastic gradient descent with negative sampling is a regular way to train Skip-Gram. Levy et al. [56] have proven an equivalence between Skip-Gram trained with negative sampling value of k and the factorization of the positive PMI word co-occurrence matrix shifted by $\log k$, i.e., the shifted positive pointwise mutual information matrix (SPPMI). Therefore, in our model, instead of the raw frequency matrix, we use the SPPMI word co-occurrence matrix as $\boldsymbol{W}$.

The PMI value between a pair of discrete outcomes x and y is defined as:

$$PMI(x, y) = \log \frac{P(x, y)}{P(x)P(y)}.$$

Empirically, the PMI value between a word w and its context word c can be estimated by considering the actual number of their co-occurrence times in the corpus:

$$PMI(w, c) = \log \frac{\#(w, c) \cdot E}{\#(w) \cdot \#(c)},$$

where $\#(w, c)$ is the number of times words w and c co-occur, $\#(w) = \sum_c \#(w, c)$, $\#(c) = \sum_w \#(w, c)$, and E is the total number of word-context pairs. The SPPMI matrix is then constructed as:

$$SPPMI_k(w, c) = \max(PMI(w, c) - \log k, 0).$$

Following the similar idea, CLM exploits local context information and learns word embeddings by factorizing matrix $\boldsymbol{W}$:

$$L_{loc} = ||\boldsymbol{W} - \boldsymbol{B}^T\boldsymbol{C}||_2^2 + \lambda_s||\boldsymbol{B}||_2^2 + \lambda_s||\boldsymbol{C}||_2^2. \tag{4.2}$$

Collaboration. By exploiting the global context information of a text corpus, we can discover the topic structures; and by exploiting the local context information, we can learn the word embeddings. However, these two parts should not be isolated from each other: semantically related words usually belong to similar topics and they are also close to each other in the embedding space. Hence we assume that the distances between word embeddings correlate with their topical similarities. As with exploiting the local context information, we realize this assumption by factorizing topic-word matrix $\boldsymbol{T}$ into the product of topic embedding matrix $\boldsymbol{A}$ and context word embedding matrix $\boldsymbol{C}$:

$$L_{com} = ||\boldsymbol{T} - \boldsymbol{A}^T\boldsymbol{C}||_2^2 + \lambda_s||\boldsymbol{A}||_2^2 + \lambda_s||\boldsymbol{C}||_2^2. \tag{4.3}$$

Hence the probability of word w being grouped into topic z can be measured by the inner product of the corresponding topic embedding and word embedding: $p(z|w) \propto t_{zw} = \boldsymbol{\alpha}_z^T\boldsymbol{c}_w$. Therefore, besides achieving the topic embeddings, Eq. 4.3 also regulates words with similar topic distributions to be close in the embedding space and nearby words to have similar topic distributions.

Unifying the three assumptions. Both the global context information and the local context information contain useful patterns in a text corpus. We propose to utilize both types of context information jointly, and to discover topic structures and learn word embeddings collaboratively:

$$\begin{aligned} L = &\underbrace{\lambda_d||\boldsymbol{D} - \boldsymbol{T}^T\boldsymbol{\Theta}||_2^2}_{global} + \underbrace{\lambda_w||\boldsymbol{W} - \boldsymbol{B}^T\boldsymbol{C}||_2^2}_{local} + \underbrace{||\boldsymbol{T} - \boldsymbol{A}^T\boldsymbol{C}||_2^2}_{joint} \\ &+ \underbrace{\lambda_s||\boldsymbol{\Theta}||_2^2 + \lambda_s||\boldsymbol{T}||_2^2 + \lambda_s||\boldsymbol{A}||_2^2 + \lambda_s||\boldsymbol{B}||_2^2 + \lambda_s||\boldsymbol{C}||_2^2}_{regularization}, \\ &subject\ to: \\ &\boldsymbol{\Theta} \geq 0\ and\ \boldsymbol{T} \geq 0, \end{aligned} \tag{4.4}$$

where λ_d and λ_w are the parameters controlling the weights of the global and local modeling parts. Eq. 4.4 is the objective function of our model. Topic-word distribution

matrix $\boldsymbol{T}$ is shared by both the global modeling and the joint modeling parts of the objective function. Context word embedding matrix $\boldsymbol{C}$ is shared by both the local modeling and the joint modeling parts of the objective function. Therefore, the topic structures and word embeddings we obtained must account for both the global and the local context information of a text corpus.

Relationship with other composite models. As we discussed in Section 4.2, two popular ways to combine topic models and word embedding models are 1) modeling topics based on pre-trained word embeddings and 2) learning word embeddings based on pre-trained topic structures. These models can be viewed as special cases of CLM that keep either topic structures or word embeddings fixed. More specifically, if we use pre-trained word embeddings and keep them fixed, then CLM considers only the combination of Eq. 4.1 and Eq. 4.3. This results in CLM becoming functionally equivalent to the composite models that discover topic structures with the help of pre-trained word embeddings, such as Gaussian-LDA [28] and TopicVec [43]. In contrast, if we use pre-trained topic structures and keep them fixed, then CLM considers only the combination of Eq. 4.2 and Eq. 4.3. Again CLM becomes functionally equivalent to the composite models that learn word embeddings and topic embeddings with the help of pre-trained topic structures, such as TWE [44] and Topic2Vec [58].

4.4.2 Parameter Inference

In this subsection, we will introduce how to do parameter estimation and inference for our proposed CLM model via collective matrix factorization. First the objective function of CLM in Eq. 4.4 is expanded as:

$$\begin{aligned} L =& \lambda_d \sum_{v=1,n=1}^{V,N} (d_{vn} - \boldsymbol{t}_v^T \boldsymbol{\theta}_n)^2 + \lambda_w \sum_{v=1,v'=1}^{V,V} (w_{vv'} - \boldsymbol{\beta}_v^T \boldsymbol{c}_{v'})^2 \\ &+ \sum_{k=1,v=1}^{K,V} (t_{kv} - \boldsymbol{\alpha}_k^T \boldsymbol{c}_v)^2 + \lambda_s \sum_{n=1}^{N} (\boldsymbol{\theta}_n^T \boldsymbol{\theta}_n) + \lambda_s \sum_{v=1}^{V} (\boldsymbol{t}_v^T \boldsymbol{t}_v) \\ &+ \lambda_s \sum_{k=1}^{K} (\boldsymbol{\alpha}_k^T \boldsymbol{\alpha}_k) + \lambda_s \sum_{v=1}^{V} (\boldsymbol{\beta}_v^T \boldsymbol{\beta}_v) + \lambda_s \sum_{v=1}^{V} (\boldsymbol{c}_v^T \boldsymbol{c}_v), \\ & \textit{subject to}: \\ & \boldsymbol{\Theta} \geq 0 \textit{ and } \boldsymbol{T} \geq 0. \end{aligned} \tag{4.5}$$

Then we compute the gradient of our objective function Eq. 4.5 with respect to each vector $\{\boldsymbol{\theta}_{1:N}, \boldsymbol{t}_{1:V}, \boldsymbol{\alpha}_{1:K}, \boldsymbol{\beta}_{1:V}, \boldsymbol{c}_{1:V}\}$:

$$
\begin{aligned}
\frac{\partial L}{\partial \boldsymbol{\theta}_n} =& 2\lambda_d \sum_{v=1}^{V} (d_{vn} - \boldsymbol{t}_v^T \boldsymbol{\theta}_n)(-\boldsymbol{t}_v) + 2\lambda_s \boldsymbol{\theta}_n \\
\frac{\partial L}{\partial \boldsymbol{t}_v} =& 2\lambda_d \sum_{n=1}^{N} (d_{vn} - \boldsymbol{t}_v^T \boldsymbol{\theta}_n)(-\boldsymbol{\theta}_n) + 2(\boldsymbol{t}_v - \boldsymbol{A}^T \boldsymbol{c}_v) + 2\lambda_s \boldsymbol{t}_v \\
\frac{\partial L}{\partial \boldsymbol{\alpha}_k} =& 2 \sum_{v=1}^{V} (t_{kv} - \boldsymbol{\alpha}_k^T \boldsymbol{c}_v)(-\boldsymbol{c}_v) + 2\lambda_s \boldsymbol{\alpha}_k \\
\frac{\partial L}{\partial \boldsymbol{\beta}_v} =& 2\lambda_w \sum_{v'=1}^{V} (w_{vv'} - \boldsymbol{\beta}_v^T \boldsymbol{c}_{v'})(-\boldsymbol{c}_{v'}) + 2\lambda_s \boldsymbol{\beta}_v \\
\frac{\partial L}{\partial \boldsymbol{c}_{v'}} =& 2\lambda_w \sum_{v=1}^{V} (w_{vv'} - \boldsymbol{\beta}_v^T \boldsymbol{c}_{v'})(-\boldsymbol{\beta}_v) + 2\lambda_s \boldsymbol{c}_{v'} + 2 \sum_{k=1}^{K} (t_{kv'} - \boldsymbol{\alpha}_k^T \boldsymbol{c}_{v'})(-\boldsymbol{\alpha}_k)
\end{aligned}
$$

Similar to the Alternating Least Squares (ALS) matrix factorization method, we obtain the following closed-form updates by iteratively setting the gradient to zero:

$$
\begin{aligned}
\boldsymbol{\theta}_n =& (\lambda_d \sum_{v=1}^{V} \boldsymbol{t}_v \boldsymbol{t}_v^T + \lambda_s \boldsymbol{I})^{-1} (\lambda_d \sum_{v=1}^{V} d_{vn} \boldsymbol{t}_v) \\
\boldsymbol{t}_v =& (\lambda_d \sum_{n=1}^{N} \boldsymbol{\theta}_n \boldsymbol{\theta}_n^T + (1 + \lambda_s) \boldsymbol{I})^{-1} (\lambda_d \sum_{n=1}^{N} d_{vn} \boldsymbol{\theta}_n + \boldsymbol{A}^T \boldsymbol{c}_v) \\
\boldsymbol{\alpha}_k =& (\sum_{v=1}^{V} \boldsymbol{c}_v \boldsymbol{c}_v^T + \lambda_s \boldsymbol{I})^{-1} (\sum_{v=1}^{V} t_{kv} \boldsymbol{c}_v) \\
\boldsymbol{\beta}_v =& (\sum_{v'=1}^{V} \boldsymbol{c}_{v'} \boldsymbol{c}_{v'}^T + \lambda_s \boldsymbol{I})^{-1} (\sum_{v'=1}^{V} w_{vv'} \boldsymbol{c}_{v'}) \\
\boldsymbol{c}_{v'} =& (\lambda_w \sum_{v=1}^{V} \boldsymbol{\beta}_v \boldsymbol{\beta}_v^T + \sum_{k=1}^{K} \boldsymbol{\alpha}_k \boldsymbol{\alpha}_k^T + \lambda_s \boldsymbol{I})^{-1} * (\lambda_w \sum_{v=1}^{V} w_{vv'} \boldsymbol{\beta}_v + \sum_{k=1}^{K} t_{kv'} \boldsymbol{\alpha}_k)
\end{aligned}
\tag{4.6}
$$

Note that this update does not guarantee the non-negativity of $\boldsymbol{\theta}_n$ and $\boldsymbol{t}_v$. Since our objective function is continuous, the minimum should be either at the points where the gradient is zero or on the boundary. Hence, if Eq. 4.6 assigns $\boldsymbol{\theta}_n$ and $\boldsymbol{t}_v$ negative entries, we can just set the negative entries to zeros. The main difference between our updates and ALS is that many variables are associated with more than one matrix factorization term. For example, context word embeddings $\boldsymbol{c}_v$ is associated with both local context matrix $\boldsymbol{W}$ and topic-word matrix $\boldsymbol{T}$. Iteratively performing these updates achieves a stationary point of our model's objective function L.

4.5 Experiments

We carry out experiments on two real-world text corpora to demonstrate the efficacy of our CLM model in two aspects: modeling topics and learning word embeddings. To investigate the quality of the topic structures discovered by CLM, we compare its performance with existing topic modeling methods on two topic evaluation tasks, the topic coherence evaluation task and the document classification task. To investigate the quality of the word embeddings learned by CLM, we compare its performance with existing word embedding models on two word embedding evaluation tasks, the word similarity task and the word analogy task. Moreover, we provide case studies to show the advantages of exploiting both types of context information.

The 20 Newsgroups dataset[1] and the Reuters-21578 dataset[2] are used in our experiments. 20News contains approximately 20,000 newsgroup documents evenly partitioned into 20 different categories. Reuters contains about 10,000 documents, but the numbers of documents in each category are highly imbalanced. We select only the largest 8 categories in Reuters, leaving us with 7,674 documents in total. In the preprocessing step, stop words and words with total frequency lower than 10 get removed, and all words are converted to lowercase. When constructing the local context matrix $\boldsymbol{W}$, we set the context window size to 10, i.e., 5 preceding words and 5 following words are considered as local context words for a focus word. For the parameters controlling the weights and regularization, we set $\lambda_d = 1e-2$, $\lambda_w = 2e-2$, and $\lambda_s = 1e-7$.

The source code of our implementation is available at `https://github.com/XunGuangxu/2in1`.

4.5.1 Evaluation on Topic Coherence

Baselines and experimental settings. The topic modeling methods we use as our baselines are the vanilla LDA [10], NMF [54], PLSI [9], Gaussian-LDA [28] and LFTM [57], among which Gaussian-LDA and LFTM are composite topic models that are built upon pre-trained word embeddings. For 20News and Reuters, we set the number of topics K to 20 and 8, respectively, as there are 20 newsgroups and 8 categories. We set the number of iterations to 100 for all the methods. For LDA, we set the hyperparameters alpha to $50/K$ and beta to 0.01. For the sake of fairness, the word embeddings used in

[1] http://qwone.com/ jason/20Newsgroups/
[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

Table 4.2: Topic coherence scores on 20News.

Top U words	5	10	20	50
NMF	-18.051	-85.538	-417.199	-2796.776
PLSI	-15.151	-78.597	-365.693	-2684.952
LDA	-15.308	-80.482	-368.820	-2694.437
Gaussian-LDA	-19.450	-94.523	-435.903	-3407.968
LFTM	-16.589	-78.541	-385.734	-2807.011
CLM	**-11.624**	**-60.303**	**-282.799**	**-2275.523**

Table 4.3: Topic coherence scores on Reuters.

Top U words	5	10	20	50
NMF	**-11.281**	-66.412	-335.619	-2705.525
PLSI	-13.226	-70.078	-333.570	-2767.808
LDA	-12.093	-69.806	-352.296	-2840.746
Gaussian-LDA	-24.223	-108.453	-478.433	-3688.172
LFTM	-13.268	-71.352	-369.009	-2982.395
CLM	-11.483	**-63.083**	**-313.459**	**-2683.163**

Gaussian-LDA and LFTM are trained on the same dataset using word2vec toolkit [1]. And we set the dimensionality of word embeddings to 50 for Gaussian-LDA, LFTM and CLM.

Evaluation metrics. In order to quantitatively assess the topic coherence, we adopt an automated metric, called coherence score of topics proposed by [50], which is able to automatically evaluate the coherence of each topic. Given a topic z and its top U words $V^z = \{v_1^z, v_2^z, ..., v_U^z\}$, the coherence score of this topic with respect to its top U words is defined as:

$$C(z; V^z) = \sum_{u=2}^{U} \sum_{l=1}^{u} \log \frac{D(v_u^z, v_l^z) + 1}{D(v_l^z)},$$

where $D(v_l^z)$ is the document frequency of word v_l^z and $D(v_u^z, v_l^z)$ is the number of documents in which words v_u^z and v_l^z co-occurred. The coherence score follows the intuition that top words in the same topic tend to frequently co-occur in documents. A topic coherence score closer to zero means a higher co-occurrence rate of the topic words,

indicating a more coherent topic. This topic coherence score shows high consistency with human judgements on topic qualities [50]. In order to investigate the overall quality of the discovered topic set, we use the average topic coherence score: $\bar{C} = \frac{1}{K}\sum_z C(z; V^z)$.

Experimental results. The topic coherence results of each method on 20News and Reuters are reported in Table 4.2 and Table 4.3, respectively. To make this evaluation more comprehensive, we vary the number of topic words $U = \{5, 10, 20, 50\}$. The best scores are highlighted in boldface. As generative models, PLSI and LDA achieve similar topic coherence scores. Gaussian-LDA does not perform well and this is probably because this topic coherence metric is more appropriate for measuring frequent words in a topic while Gaussian-LDA ranks words according to their Gaussian probabilities in each topic. LFTM outperforms Gaussian-LDA because LFTM takes advantage of both the conventional Dirichlet multinomial and the link function in the embedding space. As with CLM, NMF also factorizes the document-word matrix to learn topic structures, but our CLM model is able to utilize the additional semantic information in word embeddings learned from local context and this semantic information helps CLM discover more coherent topics. CLM ranks words in each topic according to their values in the topic-word matrix $\boldsymbol{T}$. We can see that CLM achieves significantly higher coherence scores than the baselines.

4.5.2 Evaluation on Document Classification

Baselines and experimental settings. In addition to the baselines used in Subsection 4.5.1, we also include TWE [44] and two Doc2Vec models [59]: PV-DBOW and PV-DM, as they can provide document-level representations for this document classification task. TWE is a composite model built upon pre-trained topic structures, so we feed the output of the vanilla LDA to TWE as the pre-trained topic structures. We keep the same experimental settings as in Subsection 4.5.1, except that, for 20News we set the number of topics to 280 for the topic models and the dimensionality of document embeddings to 280 for TWE, PV-DBOW and PV-DM, and for Reuters we set them to 110.

Evaluation metrics. In the document classification task on 20News, each newsgroup document is represented as a 280 dimensional vector. Hence, 20,000 newsgroup documents are classified into 20 classes according to their document-topic distributions or

Table 4.4: Document classification on 20News.

	Precision	Recall	F1
NMF	0.704	0.701	0.697
PLSI	0.722	0.712	0.709
LDA	0.727	0.722	0.719
Gaussian-LDA	0.309	0.265	0.227
LFTM	0.716	0.714	0.709
TWE	0.525	0.466	0.437
PV-DBOW	0.510	0.491	0.459
PV-DM	0.428	0.386	0.361
CLM	**0.825**	**0.818**	**0.816**

document embeddings. The reason why we change the number of topics from 20 to 280 is that the number of classes is already 20 and the number of features (topics) should be larger than that. Similarly, we set the number of features (topics) to 110 for Reuters. In order to evaluate the overall performance across all the document classes, we adopt the macro-averaged precision, recall and F1 measures as the evaluation metrics, as macro-averaging gives equal weight to each class.

Experimental results. Table 4.4 and Table 4.5 present the classification performance of the different methods on 20News and Reuters, respectively. The highest scores are highlighted in boldface. The document-topic representation used here corresponds to the document-topic matrix $\mathbf{\Theta}$ in our model. We can see that CLM outperforms the baselines significantly. On this task, PLSI, LDA and LFTM still obtain similar and better scores than the other baselines. As with CLM, NMF is also based on matrix factorization techniques, but NMF does not achieve as good performance as CLM due to its inability to utilize both context information. Gaussian-LDA performs considerably inferior to all other methods. By checking its output variables manually, we find that the Gaussian distributions for different topics are highly similar and hence its document-topic representations are not discriminative enough. TWE, PV-DBOW and PV-DM assign a low-dimensional embedding to each document based on the word embeddings in it, but the classification results on these document embeddings are inferior to the results on document topic proportions.

Table 4.5: Document classification on Reuters.

	Precision	Recall	F1
NMF	0.911	0.877	0.891
PLSI	0.919	0.896	0.906
LDA	0.888	0.870	0.879
Gaussian-LDA	0.462	0.315	0.353
LFTM	0.893	0.591	0.661
TWE	0.794	0.512	0.626
PV-DBOW	0.755	0.505	0.549
PV-DM	0.681	0.434	0.507
CLM	**0.944**	**0.916**	**0.929**

4.5.3 Evaluation on Word Similarity

Having shown the superiority of CLM in topic discovery, we now evaluate the quality of word embeddings learned from the 20 Newsgroups dataset by CLM in the following two tasks. As we know, training word embeddings requires a large amount of local context information to capture language regularities. Hence, Wikipedia, the largest online encyclopedia, is the most frequently used training dataset for word embeddings due to its sheer range of topics and ample local context information. However, for experiment domains involving smaller corpus size such as the 20 Newsgroups dataset, gathering the local context information is quite a challenge. We will show that our CLM model is able to overcome the challenge of lacking local context information by taking the complementary global context information into consideration.

Baselines and experimental settings. The word embedding methods we include as our baselines are the SPPMI matrix without dimensionality reduction [56], SVD of the SPPMI matrix [56], GloVe [2], CBOW [1], Skip-Gram [1], PV-DBOW [59], PV-DM [59] and TWE [44], among which TWE is a composite model that is built upon pre-trained LDA topic structures, and PV-DBOW and PV-DM take the influence of documents on word embeddings into consideration. Different from the others, GloVe constructs word co-occurrence matrix and learns word embeddings purely based on document-level global context information. For uniformity, we set the number of context window size to 10,

Table 4.6: Comparison of word similarity results.

	WS353	WS Rel	WS Sim	Men	Turk	SimLex-999	Rare
SPPMI	0.461	0.444	0.465	0.444	**0.551**	0.131	0.245
SPPMI + SVD	0.451	0.435	0.449	0.426	0.489	0.166	0.349
GloVe	0.300	0.279	0.320	0.192	0.268	0.049	0.230
Skip-Gram	0.492	0.479	0.473	0.456	0.512	0.155	0.407
CBOW	0.488	0.451	0.494	0.432	0.529	0.151	0.407
PV-DBOW	0.477	0.442	0.486	0.449	0.488	0.139	0.285
PV-DM	0.297	0.304	0.310	0.236	0.339	0.013	0.157
TWE	0.317	0.231	0.407	0.190	0.260	0.084	0.184
CLM	**0.526**	**0.486**	**0.550**	**0.477**	0.525	**0.189**	**0.411**

the number of negative samples to 5, and the dimensionality of the embedding space to 50 for all the methods. And we set the number of topics to 20 for TWE and CLM. We then perform 100 iterations of training for all the methods.

Evaluation metrics. We use several test datasets to evaluate the word pair similarities calculated by word embeddings: WordSim353 (WS353) [60] (including WordSim Relatedness (WS Sim) and WordSim Similarity (WS Rel)), MEN [61], Turk [62], SimLex-999 [63], and Rare [64]. These datasets contain word pairs associated with human-assigned similarity scores. After ranking the word pairs according to their cosine similarities in the embedding space and human-assigned similarity scores respectively, the word embeddings are evaluated by measuring the Spearman's rank correlation with the human ratings. We exclude word pairs that contain out-of-vocabulary words from the test datasets. A higher correlation value indicates it is more consistent with human judgements on word similarities.

Experimental results. The results are summarized in Table 4.6. The highest correlation scores are highlighted in boldface. Similar performances are achieved by the SPPMI matrix and SVD of SPPMI; however, the dimensionality of SPPMI word representations is the vocabulary size – 20,678 – much higher than 50 dimensions for the other methods. As with CLM, SVD also learns word embeddings by factorizing the local SPPMI matrix, but its inability to utilize the additional global topical information

Table 4.7: Comparison of word analogy results.

	Google	MSR
SPPMI	6.60%	5.40%
SPPMI + SVD	4.93%	7.32%
GloVe	2.67%	3.51%
Skip-Gram	6.62%	10.70%
CBOW	5.61%	12.00%
PV-DBOW	7.12%	11.77%
PV-DM	2.84%	7.55%
TWE	3.76%	5.38%
CLM	**8.28%**	**14.20%**

results in inferior performance to ours. Skip-Gram and CBOW yield better results than SPPMI and SVD of SPPMI. PV-DBOW performs on a par with Word2Vec models. GloVe performs inferior to the other methods. This may be due to the fact that GloVe utilizes only global context information but there is inadequate global context information to train word embeddings in the 20 Newsgroups dataset. As one can see, the SPPMI matrix obtains the best correlation score on the Turk test dataset, and CLM outperforms the baselines on all the other test datasets.

4.5.4 Evaluation on Word Analogy

Baselines and experimental settings. For the second evaluation task on the quality of word embeddings, we use the same baselines and keep the same experimental settings as in the previous word similarity task in Subsection 4.5.3.

Evaluation metrics. The word analogy task refers to questions of the form "a is to a* as b is to b*", where b* is hidden and needs to be inferred from the vocabulary. We use two test datasets for the word analogy task: MSR [65], which contains 8000 morphosyntactic analogy questions, such as "good is to better as rich is to richer", and Google [1], which contains 19544 questions, about half of the same syntactic type as in MSR, and the other half of a semantic nature, such as "king is to queen as man is to woman". We filter out questions involving out-of-vocabulary words. The hidden words

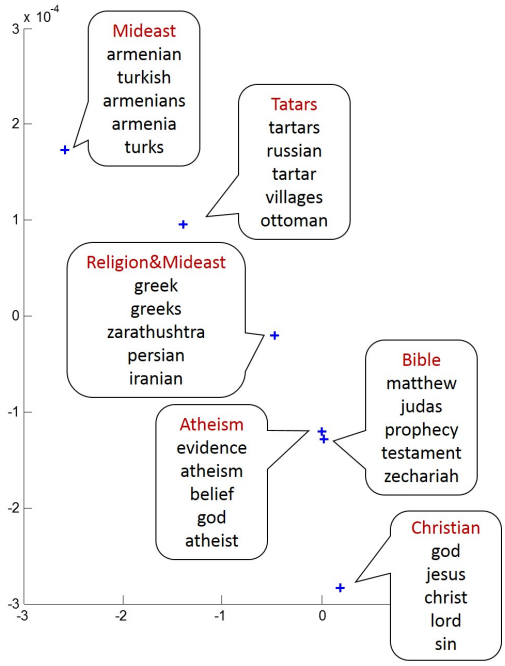


Figure 4.1: Two-dimensional PCA projection of the topic embeddings related to religions and mideast.

b* can be inferred by optimizing 3CosAdd [66]:

$$\arg\max_{b*\in V}(cos(b*, b - a + a*)).$$

The evaluation metric for the word analogy task is the percentage of questions for which the 3CosAdd result is the correct answer b*.

Experimental results. Table 4.7 shows the results on the word analogy task. As can be seen, the lack of local context information in the 20 Newsgroups dataset heavily limits the performance of the different methods on such a difficult task as word analogies. Linguistically speaking, the word analogy task relies more on contextual information from common words and auxiliary verbs to correctly infer b*. Word embeddings learned from larger dataset which provides sufficient local context such as Wikipedia can achieve better performance on this task. For word embeddings learned from the 20 Newsgroups dataset, SVD of the SPPMI matrix performs on a par with the raw SPPMI matrix. CBOW, Skip-Gram and PV-DBOW yield better results than SPPMI and SVD of SPPMI because their training procedures give more influence to frequent

Table 4.8: Case study 1.

	Coherence score	Avg cosine distance
10 random words	-171.641	0.026
NMF	-102.422	0.570
CLM	**-89.731**	**0.728**

pairs. GloVe does not perform well since it only explores global context information. TWE does not achieve good results because it is heavily limited by the sparsity issue and influenced by the pre-trained topic structures. We can still see that, by exploiting both the global and the local context information, CLM overcomes the lack of local context information and outperforms the baselines significantly.

4.5.5 Qualitative Assessment of Topic Embeddings

Besides topic structures and word embeddings, CLM can also learn topic embeddings for each topic, i.e., the topic embedding matrix $\boldsymbol{A}$. Those topic embeddings are of the same dimensionality as word embeddings. The relationships between topic embeddings and word embeddings are modeled in Eq. 4.3: the larger inner product value a word embedding and a topic embedding get, the more important that word is in the topic. After convergence, the similarities and correlations among topics are also captured in the embedding space. Figure 4.1 shows the two-dimensional PCA projection of the topic embeddings related to religions and Mideast. Each topic embedding is annotated with its topic name and top 5 words. We can observe that the semantic similarities between topics correlate with the Euclidean distances between the corresponding topic embeddings. The correlations among topics can also be captured in this embedding space. For example, Figure 4.1 illustrates how the topic of Christian transitions to the topic of Mideast through the topics of Bible and religions.

4.5.6 Case Studies

How local context information assists global context information in discovering topic structures. Having shown the superiority of CLM in discovering topic structures in Subsections 4.5.1 and 4.5.2, we now take the topic of astronomy as an example to illustrate how word embeddings can help discover more coherent topics.

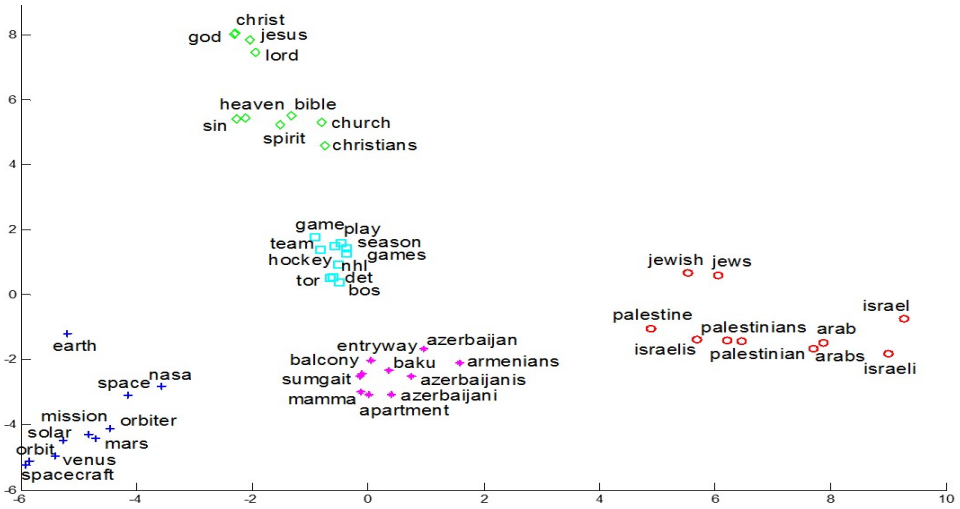


Figure 4.2: Two-dimensional PCA projection of word embedding clusters.

Word embeddings learned from local context information are able to capture semantic regularities in language: words with similar semantic properties are found to be close to each other in the embedding space. And we are encouraged to group semantically related words (words that are geographically close in the embedding space) into same topics. This intuition is illustrated in Figure 4.2: words belonging to same topics tend to locate in nearby areas.

To verify our assumption, we quantitatively show that the average cosine distance of words in a topic is consistent with the topic coherence score. As our closest competitor in topic discovery, NMF is equivalent to our CLM model without considering word embeddings. The top 10 words in the topic of astronomy discovered by CLM are {*'space', 'orbit', 'solar', 'spacecraft', 'mission', 'mars', 'earth', 'venus', 'nasa', 'orbiter'*} as shown on the bottom left corner in Figure 4.2. The top 10 words in the topic of astronomy discovered by NMF are {*'space', 'earth', 'planet', 'system', 'spacecraft', 'solar', 'venus', 'surface', 'moon', 'kilometers'*}. We then calculate the average cosine distance of words and the topic coherence score for CLM and NMF respectively. Table 4.8 justifies the consistency between the topic coherence score and the average cosine distance. Therefore, by considering the spatial information of word embeddings, more coherent topics can be discovered by CLM.

Table 4.9: Case study 2.

Word pairs	Ground truth ranking	SPPMI +SVD ranking	CLM ranking	Cosine similarity of $p(z\|w)$
king – queen	13	135	118	0.733
money – currency	7	79	41	0.851
planet – space	42.5	201	143	0.918
mile – kilometer	12	94	62	0.879
man – woman	24	70	44	0.731

How global context information assists local context information in learning word embeddings. In Subsections 4.5.3 and 4.5.4, we have shown that the word embeddings learned by CLM are closer to human judgements in terms of word similarities. We now take several word pairs in the WS353 test dataset to illustrate how global topical information can help us learn better word embeddings. We compare our rankings for these example words with the rankings of our closest competitor SPPMI+SVD which is equivalent to our CLM model without considering global topical information. As we can see in Table 4.9, due to the lack of sufficient local context information, these word pairs are not ranked properly by SPPMI+SVD. With the help of global topical information, CLM can improve the similarity ranking as words' topic distribution regulates. If two words have similar topic distributions (measured by the cosine similarity between their $p(z|w)$), such as planet and space, CLM would adjust the two corresponding word embeddings closer to each other accordingly and assign them a higher position in the similarity ranking.

4.6 Conclusions

We present a unified language model CLM based on matrix factorization techniques which is able to collaboratively discover topic structures and learn word embeddings. Moreover, building our model on both the global and the local context enables it to make use of more sufficient information. The proposed CLM model formulates documents as admixtures of topics, where each topic is a multinomial distribution over words and is influenced by word embeddings in the way that words close to each other in

the embedding space should be grouped into same topics. At the same time, CLM also assumes that words appearing in similar local contexts and having similar topic distributions tend to get mapped to nearby areas in the embedding space. Topics and words are jointly trained and embedded in the vector space that preserves semantic regularities, while sparse and interpretable document-topic distributions are achieved simultaneously. The experiments on the real-world datasets validate the effectiveness of CLM.

Part II

Contextual Representation Learning Based on Temporal Context

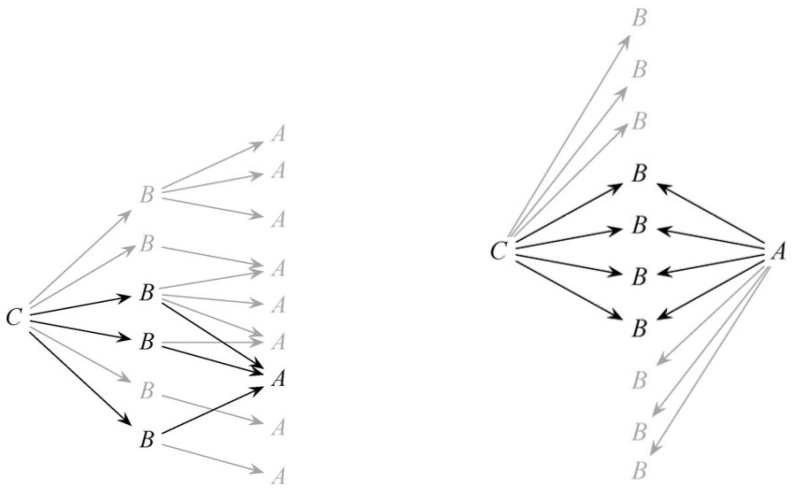
Chapter 5

Learning Semantic Evolution Based on Diachronic Literature Data

5.1 Introduction

Decades of experimentation and analysis had led to a generation of large-scale datasets in biological domain. At the same time, the technological advancements in the field of biomedical engineering have resulted in the development of many tools such as high-throughput sequencers, diagnostic imaging, etc. With the aid of these tools, researchers are now able to gather and analyze massive data to understand disease prognosis, prevention and personalized treatment options. The output of these various studies and findings are shared to the research community through publication of various scientific journals. As an example of the impact of these developments on the throughput of scientific discovery and research, consider MEDLINE, a premier bibliographic database in life sciences. With currently more than 23 million references from approximately 5,600 worldwide journals, it has seen a steady growth rate of $\sim$4% [67].

Despite the technological development that reduces the time for discovery, biomedical researchers still face a daunting task of enumerating various postulates/hypotheses based on manual inspection of evidence, which are then subsequently verified through experiments. Apart from the monetary cost for performing these experiments, researchers also have to bear the impact of possible delay in their research and its subsequent impact



(a) Open discovery. The search starts at C (e.g. disease), and results in A (drug). The intermediate B steps may represent (patho) physiological mechanisms. The black arrows indicate potentially interesting pathways of discovery, the grey ones do not qualify.

(b) Closed discovery. The search process starts simultaneously from C (e.g. disease) and A (drug), resulting in overlapping Bs (potential mechanisms). The black arrows indicate potentially interesting pathways of discovery, the grey ones spurious links.

Figure 5.1: Open and closed discovery.

on the community as a whole. Consequently, there has been a growing research interest within the computer science domain in alleviating this situation by leveraging the large body of published literature to perform text mining tasks with an aim of assisting the biomedical researchers in their research tasks. One such task is Literature based discovery (LBD), which aims to discover high quality, novel and non-trivial postulates by leveraging the already known and established scientific facts.

Originally formulated in 1986 [68], where the problem statement was to identify whether there is any relationship between "Fish Oil" and "Raynaud's Disease", over the time, LBD techniques have grown from manual inspection of causality to more sophisticated ideas involving association rules, classifiers, graph theoretics and manually curated knowledge-bases with explicitly defined semantic relationships [69, 70, 71, 72]. Apart from the development in the approaches towards the solution, the original problem statement of LBD has also evolved from a confirmatory type of problem to discovery

- often referred to as closed and open discovery respectively [73]. In closed discovery, the user provides two previously unconnected concepts as an input along with a date parameter, and the task is to identify high confident evidence (terms) that connects these inputs using all the documents published before that input date. As opposed to this, in open discovery, the user provides only one term as the input, and the task is to search and determine other related concepts that can form an indirect connection with the input term. E.g.: Given two input concepts 'Fish Oils' and 'Raynaud's Disease' and the date as 1985, closed discovery framework identifies a ranked set of connecting terms - e.g., Blood Viscosity, Platelet Aggregation, etc., by using all the documents available until the date mentioned. In the case of open discovery, only one of the two terms is given as an input - say Raynaud's Disease, and the task is to identify all the concepts that are related to it but not yet formally established. A high level view of both open and closed discovery is shown in Figure 5.1. In this work, for the sake of simplicity, we discuss our methodology for closed discovery and explain in brief on how it can be easily extended to open discovery.

Traditionally, both open and closed discovery are performed as a two-step process. The first step is responsible for enumerating all the different hypotheses and the second step ranks these hypotheses such that the reliable and novel hypotheses are ranked higher than frivolous and less confident ones. However, there are two major challenges in such a two-step approach: (a) The dense interconnection/co-occurrence between medical terms leads to exponentially large search space. (b) The ranking should not only be sensitive to the statistical significance but also be semantically meaningful. In this chapter, we try to overcome the aforementioned challenges with a novel methodology which is semantically sensitive and able to efficiently rank hypotheses.

Furthermore, it is worthwhile to note that the semantics of terms gradually evolve over time. For example, "virus" initially only meant an infectious agent that typically consists of a nucleic acid molecule in a protein coat, then as the time passed, the semantics of "virus" started getting closer to "computer". Likewise, as the research over disparate medical terms progresses their implicit semantics gradually evolves too. Thus, if we are able to capture this growing association trend between any two terms, we would then be able to rank the hypotheses based on the terms' direction of evolution. Simply put, if over a time period, the evolutionary trend between medical concepts is towards each other then there is a high likelihood that these concepts will eventually form a relationship in the future.

To capture the semantics (medical properties) of medical concepts as well as their evolutionary behaviors, we propose to formulate this problem in the embedding space. Word embeddings are well known for their capability of capturing the implicit semantics of terms [23] and have been used for a variety of association and relatedness measuring tasks [1]. Moreover, word embeddings have shown promise as a diachronic tool, but have not been systematically formulated. Thus, in this chapter, we propose the Dynamic MeSH Embedding (DME) model which uses the principles of co-occurrence and temporal smoothening to train and constrain MeSH embeddings that can be used to capture implicit semantics and track semantic evolution. Furthermore, based on the evolutionary MeSH embeddings, we are able to generate medical hypotheses that are both informative and sensitive to the semantic evolution of medical terms. In doing so, we are able to achieve better Spearman's rank correlation coefficient in the top-k confident connecting terms compared with other existing ranking methodologies. Apart from getting better results in statistical evaluation, we also perform manual verification of the top-k confident connecting terms.

Our contributions can be summarized as:

- We propose a novel dynamic MeSH embedding model for LBD, capable of modeling gradual semantic evolution of MeSH terms over time in the embedding space and inferring hidden associations.
- Our model allows us to track and visualize the evolutionary trajectories as well as research trends of various medical concepts, thus achieving fine interpretability for the generated hypotheses. Our model promotes the terms with increasing shared semantics, which tend to have a higher likelihood of forming an inter-connection in future.
- Unlike prior approaches which use complex heuristics to eliminate generic terms causing dense search space, our model is able to detect them based on their evolutionary behaviors.
- We show the effectiveness of our model in facilitating the rediscovery of five established scientific facts.

5.2 Related Work

The research area of LBD in biomedical domain came into prominence after the seminal work of Don R. Swanson in 1986 [68]. In this work, after manually inspecting MEDLINE articles pertaining to both Fish oils and Raynaud's disease respectively, he postulated that Fish oils could be a potential treatment for Raynaud's Syndrome. This was later clinically verified by Digicomo [74]. Through his work, he demonstrated that implicit "interesting" links can be found by connecting disparate research findings already present in literature.

While this initial work laid the groundwork for future research works, it needed extensive manual intervention and domain knowledge to guide the discovery process. To tackle these initial challenges, subsequent works explored in several directions to make the process more methodological and automated. Some of the works based their approach on purely distributional based approaches (term frequency, inverse document frequency, record frequency and so on) [75, 76, 77, 78, 79]. The underlying assumption is that discoveries are likely to arise if the logical fragments are either highly or rarely connected in the knowledge base. However, a critical issue with these approaches is that a high co-occurrence frequency statistic does not necessarily entail a meaningful or a novel association. Also these methods did not take into account the temporal characteristics of terms - which indicates their semantic evolution - a necessary aspect for capturing plausible hypotheses. In our proposed model, we learn the semantic evolution of concepts by incorporating their temporal features.

To address the inherent problems in distributional approaches, several relation based approaches were proposed which relied on explicit relationships (or predicates) between concepts. One pioneering example is the work of [72], where he developed an LBD system using semantic predicates (in the form of subject-verb-object) to discover indirect associations. These predicates were extracted from articles by using a Natural language processing (NLP) system known as SemRep [80]. While the relation based approaches were successful in capturing explicit relations they ignored the implicit semantic association between concepts which evolves over time. To tackle this issue, our proposed model is formulated in the low-dimensional embedding space wherein implicit semantics of terms are preserved.

More recent studies have focused on graph theoretic and supervised machine learning approaches[70, 81, 69, 71]. As graph representation provide a more natural way to

integrate heterogeneous knowledge sources into a single unified schema, it provides rich opportunities to perform various analytical tasks. However, they still suffer from scalability issues. While supervised machine learning approach has the potential to elucidate novel association, the training data required is too expensive to generate.

Semantic evolution of natural language in the embeddings space has been studied by several recent works[13, 15, 14]. Most of them train word embeddings separately on data of different time slices and then align the learned embeddings. While statistical principles of language evolution can be revealed by these model, they fail to formulate the correlation between embeddings at successive time stamps and learn smooth evolutionary trajectories.

In order to deal with the inherent issues present in prior approaches, in this work, we formulate the problem as an unsupervised learning task where we use dynamic MeSH embeddings to model smooth semantic evolution of concepts to perform knowledge discovery efficiently.

5.3 Definitions and Terminologies

5.3.1 Literature

In LBD, literature refers to a set of articles relevant to a particular subject. For instance, "Parkinson" literature refers to the set of all articles which discuss "Parkinson disease". In this regard, it is worthwhile to mention that MEDLINE is the most comprehensive source for literature collection in the biomedical domain. As full text articles are available in limited quantity, most researchers use MEDLINE title, abstract and indexing terms (MeSH terms) as a surrogate for full text articles [73, 75, 82]. Also, an advantage of using MEDLINE is that it is openly accessible and can be searched using a powerful search engine developed by National Library of Medicine (NLM), viz., PubMed[1] .

5.3.2 Concepts

Concept refers to a term or a phrase which has some biomedical importance. For example, consider a topic "Parkinson disease", the terms representing genes, proteins, drugs, symptoms and other disease related to this topic are referred to as concepts. There are many ways of extracting concepts from a text article for the purpose of LBD.

[1] `https://www.ncbi.nlm.nih.gov/pubmed`.

Some researchers [73, 76] prefer to extract biomedically meaningful concepts from free text by utilizing a controlled vocabularies such as Unified Medical Language Systems (UMLS)[80]. Alternatively, some choose to exploit Medical Subject Headings (MeSH) terms to represent documents [77, 71, 79]. In this work, we use MeSH terms to represent document. In the next section, we briefly describe MeSH terms.

5.3.3 Medical Subject Headings

Medical Subject Headings (MeSH) terms are NLM controlled thesaurus that are used to index MEDLINE articles. For instance, if an article discusses the role of fish oil in treating patients with Raynaud's disease, then the article could be indexed with MeSH terms such as "fish oil","raynaud disease","blood vessels". As MeSH terms are annotated manually to these articles by biomedical experts, it is safe to assume that if a concept is of central importance to an article, it will be assigned to that article.

5.3.4 MeSH Embeddings

MeSH embeddings, or simply MeSH vectors, are d-dimensional real-valued vectors assigned to each MeSH term in the corpus such that two vectors close to each other in the vector space denote a semantic similarity between the corresponding two MeSH terms. The idea of MeSH embeddings in this chapter is inspired by word embeddings [23, 1, 2, 52, 83, 53].

5.4 Methodology

We propose the DME model to learn the evolutionary behavior of MeSH semantics and hence help generate medical hypotheses based on their evolving semantics. DME is established on sequential text data and in every time slot DME models each MeSH term as a unique MeSH embedding. MeSH embeddings are trained to capture implicit semantics. Moreover, as our medical knowledge develops, the DME model finds that MeSH embeddings continuously drift over time in the embedding space, i.e., dynamic MeSH embeddings, allowing us to track semantic changes of MeSH terms over short and long periods of time.

More specifically, DME follows two basic assumptions: (1) The distance between two MeSH embeddings correlates with their medical similarity; (2) MeSH embeddings

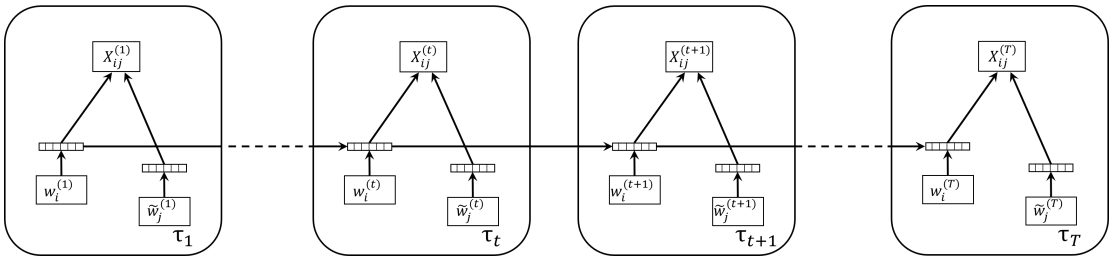


Figure 5.2: Framework of DME. T time slices of data are connected via dynamic MeSH embeddings.

evolve smoothly across time.

5.4.1 Dataset Construction

In order to generate high confident intermediary terms which connects the user input query terms in a meaningful way, we use a complete dump of MEDLINE (2016)[2] as our dataset. We split this corpus into time slices and create a co-occurrence matrix of MeSH terms for each period. We use this co-occurrence statistics to learn evolutionary characteristics of MeSH terms. More specifically, first all the articles are aggregated to the granularity of five years, e.g., 1900-1904, 1905-1909, 1910-1914 and so on. Then for each time slot t, a co-occurrence matrix $X^{(t)}$ of MeSH terms is constructed to capture the co-occurrence patterns, wherein each entry $X_{ij}^{(t)}$ denotes the number of times that the i^{th} MeSH term co-occurs with the j^{th} MeSH term in the same article.

5.4.2 Evolutionary MeSH Embeddings

Given T time-stamped medical co-occurrence matrices $\{\boldsymbol{X}^{(1)}, ..., \boldsymbol{X}^{(t)}, ..., \boldsymbol{X}^{(T)}\}$, the semantics and evolutionary patterns of each MeSH term are carried implicitly within those matrices. We describe our model according to how the semantics and evolutionary patterns of MeSH terms are learned, as well as how the two aforementioned assumptions are realized.

Static MeSH embeddings. The statistics of term occurrences in a text dataset is the primary source of information to all unsupervised embedding methods. The semantics of terms (the medical properties of MeSH terms in our case) are contained in these

[2] `https://www.nlm.nih.gov/databases/download/pubmed_medline.html`.

statistics, and our goal is to learn MeSH embeddings that can represent those medical properties. We start with learning static MeSH embeddings from an independent term-term co-occurrence matrix.

Inspired by the word embedding model GloVe [2], we assume that the medical property of a MeSH term can be described by its co-occurrence information, i.e., its context information. As an example, consider two MeSH terms $i = male$ and $j = female$, their relationship can be examined by studying the ratio of their co-occurrence probabilities with other probe terms, k: for terms k like *brain* or *carbon*, that are related to both male and female, or to neither, we expect the co-occurrence probability ratio $P(k|i)/P(k|j)$ to be close to one; for terms k more related to female than to male, say $k = pregnancy$, the probability ratio $P(k|i)/P(k|j)$ should be small; in contrast, for terms more related to male than to female, the ratio should be large. This assumption suggests that the probability ratio $P(k|i)/P(k|j)$ depends on two target terms i, j and one probe term k. By adopting the vector difference and the dot product of the MeSH embeddings, the linear structures of the embedding space can be captured and modeled via:

$$F((\boldsymbol{w}_i^{(t)} - \boldsymbol{w}_j^{(t)})^\intercal \widetilde{\boldsymbol{w}}_k^{(t)}) = \frac{P(k|i)}{P(k|j)}, \tag{5.1}$$

where $\boldsymbol{w}^{(t)} \in \mathbb{R}^d$ are MeSH embeddings at time stamp t and $\widetilde{\boldsymbol{w}}^{(t)} \in \mathbb{R}^d$ are context MeSH embeddings at time stamp t, respectively. $\boldsymbol{w}_i^{(t)}$ is used when term i works as a target term, and $\widetilde{\boldsymbol{w}}_i^{(t)}$ is used when term i works as a probe term. Given the medical term-term co-occurrence matrix at time t, $X^{(t)}$, $P(k|i)$ is empirically set as $P(k|i) = X_{ik}^{(t)}/X_i^{(t)}$, where $X_i^{(t)} = \sum_m X_{im}^{(t)}$ is the number of times any MeSH terms co-occurred with term i at time t. Thus, by taking F as the exponential function and adding biases, a simplification over Equation 5.1 is obtained:

$$\boldsymbol{w}_i^{(t)\intercal} \widetilde{\boldsymbol{w}}_k^{(t)} + b_i^{(t)} + \widetilde{b}_k^{(t)} = \log(X_{ik}^{(t)}), \tag{5.2}$$

where $b_i^{(t)}$ and $\widetilde{b}_k^{(t)}$ are biases associated with term i and k at time t. Considering that the term co-occurrence matrix $X^{(t)}$ is very sparse, static MeSH embeddings for time stamp t can be learned via a weighted least squares regression:

$$J^{(t)} = \sum_{i,j=1}^{V} f(X_{ij}^{(t)})(\boldsymbol{w}_i^{(t)\intercal} \widetilde{\boldsymbol{w}}_j^{(t)} + b_i^{(t)} + \widetilde{b}_j^{(t)} - \log(X_{ij}^{(t)}))^2, \tag{5.3}$$

where f is a weighting function for each entry in the co-occurrence matrix. As suggested

in GloVe [2], f is set as:

$$f(x) = \begin{cases} (x/x_{max})^{\alpha} & \text{if } x < x_{max} \\ 1 & \text{otherwise} \end{cases}. \tag{5.4}$$

Dynamic MeSH embeddings. The previous subsection introduces learning of static MeSH embeddings from an independent term co-occurrence matrix. Now given a time sequence of term co-occurrence matrix $\{\boldsymbol{X}^{(1)}, ..., \boldsymbol{X}^{(t)}, ..., \boldsymbol{X}^{(T)}\}$, we would like to learn dynamic MeSH embeddings which evolve smoothly as time passes and our medical knowledge progresses.

Figure 5.2 shows the framework of our DME model. The learned dynamic MeSH embeddings at time t must account for both their medical properties which are carried by the current term co-occurrence matrix and their historical evolutionary trajectories. At each time stamp t, we add a distance constraint to each MeSH term which prevents the embedding from drifting too far from its historical location:

$$O^{(t)} = \sum_{i,j=1}^{V} f(X_{ij}^{(t)}) \left((\boldsymbol{w}_i^{(t)\intercal} \widetilde{\boldsymbol{w}}_j^{(t)} + b_i^{(t)} + \widetilde{b}_j^{(t)} - \log(X_{ij}^{(t)}))^2 + \beta I^{(t)}(i) l(\boldsymbol{w}_i^{(t)}, \boldsymbol{w}_i^{(t-1)}) \right),$$

where β is the parameter controlling the damping to the historical embeddings, $I^{(t)}(i)$ is a indicator function, and $l(\boldsymbol{w}_i^{(t)}, \boldsymbol{w}_i^{(t-1)})$ measures the distance between term i's current location in the embedding space $\boldsymbol{w}_i^{(t)}$ and its historical location $\boldsymbol{w}_i^{(t-1)}$. $I^{(t)}(i)$ indicates if term i has occurred in history:

$$I^{(t)}(i) = \begin{cases} 1 & \text{if term } i \text{ has occurred before time } t \\ 0 & \text{otherwise} \end{cases}. \tag{5.5}$$

A large number of distance measurements can be used as $l(\boldsymbol{w}_i^{(t)}, \boldsymbol{w}_i^{(t-1)})$, such as cosine distance, but since we would like to learn smooth evolutionary trajectories of MeSH embeddings, we adopt the Euclidean distance between the current embeddings and historical embeddings:

$$l(\boldsymbol{w}_i^{(t)}, \boldsymbol{w}_i^{(t-1)}) = ||\boldsymbol{w}_i^{(t)} - \boldsymbol{w}_i^{(t-1)}||^2. \tag{5.6}$$

In practice, β is set to a small value, so the damping to the history is very weak. At time stamp $t = 1$, we define $I^{(0)}(i) = 0$. We put the embedding shift constraint $l(\boldsymbol{w}_i^{(t)}, \boldsymbol{w}_i^{(t-1)})$ only on MeSH embeddings, because context MeSH embeddings might

need to change its scale frequently as the scale of the co-occurrence matrices changes. Thus, the overall objective function of our DME model is as follows:

$$O = \sum_{t=1}^{T} O^{(t)} = \sum_{t=1}^{T} \sum_{i,j=1}^{V} f(X_{ij}^{(t)}) \Big((\boldsymbol{w}_i^{(t)\intercal} \widetilde{\boldsymbol{w}}_j^{(t)} + b_i^{(t)} + \widetilde{b}_j^{(t)} - \log(X_{ij}^{(t)}))^2 + \beta I^{(t)}(i) l(\boldsymbol{w}_i^{(t)}, \boldsymbol{w}_i^{(t-1)}) \Big). \tag{5.7}$$

Equation 5.7 enforces that the DME model learns dynamic MeSH embeddings which vary smoothly over time. On each occurrence of a MeSH term, its corresponding dynamic MeSH embedding is regulated not to drift too far from its historical location. Thus, the higher term frequency, the larger regulation to be stable over time. This is consistent with the law of conformity of language evolution – "rates of semantic change scale with a negative power of word frequency" [13]. DME efficiently shares information across the time domain, which allows us to feed the time-stamped data sequentially in steps. Dynamic MeSH embeddings trained by DME can both capture the implicit medical properties of each MeSH term and track their property changes.

5.4.3 Parameter Inference

We take the gradient of DME objective (Equation 5.7) with respect to each of the model parameters $\{\boldsymbol{w}_i^{(t)}, \widetilde{\boldsymbol{w}}_j^{(t)}, b_i^{(t)}, \widetilde{b}_j^{(t)}\}$ and then adopt stochastic gradient descent to update them. Thus, on each co-occurrence record, this gives us the following closed-form updates:

$$\begin{aligned}
\boldsymbol{w}_i^{(t)} &\leftarrow \boldsymbol{w}_i^{(t)} - \eta * 2f(X_{ij}^{(t)}) \Big((\boldsymbol{w}_i^{(t)\intercal} \widetilde{\boldsymbol{w}}_j^{(t)} + b_i^{(t)} + \widetilde{b}_j^{(t)} \\
&\qquad - \log(X_{ij}^{(t)})) \widetilde{\boldsymbol{w}}_j^{(t)} + \beta I^{(t)}(i) (\boldsymbol{w}_i^{(t)} - \boldsymbol{w}_i^{(t-1)}) \Big), \\
\widetilde{\boldsymbol{w}}_j^{(t)} &\leftarrow \widetilde{\boldsymbol{w}}_j^{(t)} - \eta * 2f(X_{ij}^{(t)}) \Big((\boldsymbol{w}_i^{(t)\intercal} \widetilde{\boldsymbol{w}}_j^{(t)} + b_i^{(t)} + \widetilde{b}_j^{(t)} \\
&\qquad - \log(X_{ij}^{(t)})) \boldsymbol{w}_i^{(t)} \Big), \\
b_i^{(t)} &\leftarrow b_i^{(t)} - \eta * 2f(X_{ij}^{(t)}) (\boldsymbol{w}_i^{(t)\intercal} \widetilde{\boldsymbol{w}}_j^{(t)} + b_i^{(t)} + \widetilde{b}_j^{(t)} - \log(X_{ij}^{(t)})), \\
\widetilde{b}_j^{(t)} &\leftarrow \widetilde{b}_j^{(t)} - \eta * 2f(X_{ij}^{(t)}) (\boldsymbol{w}_i^{(t)\intercal} \widetilde{\boldsymbol{w}}_j^{(t)} + b_i^{(t)} + \widetilde{b}_j^{(t)} - \log(X_{ij}^{(t)})),
\end{aligned}$$

where η is the learning rate. In Section 5.5, we will introduce how to solve LBD tasks with the evolving medical concepts – to be more specific, how to generate and rank medical hypotheses based on the dynamic MeSH embeddings.

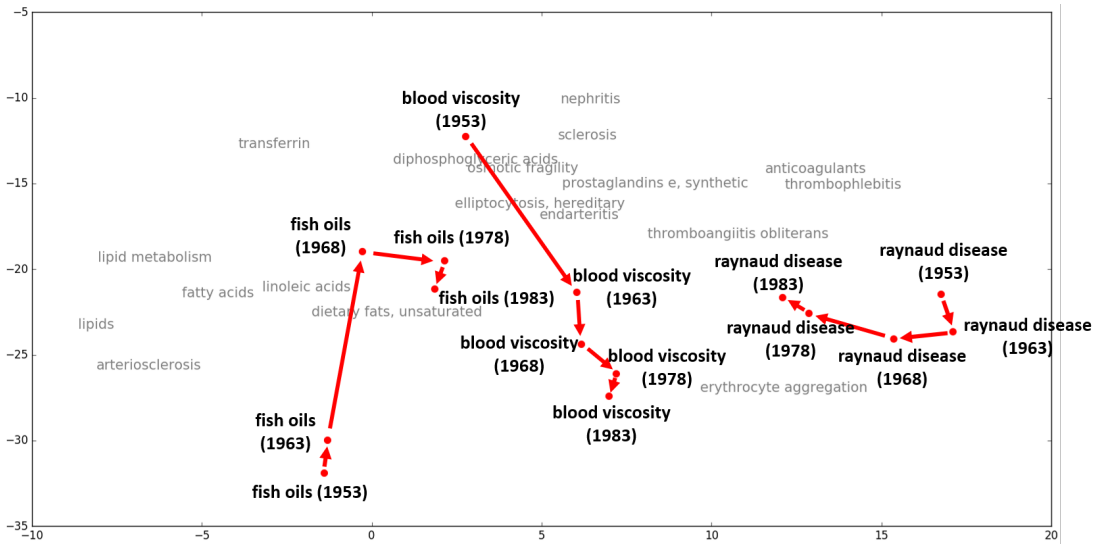


Figure 5.3: An example of the evolutionary behavior of MeSH embeddings.

5.5 Experiments

Having explained the nuances of our methodology, we now describe the evaluation and compare the results of our framework with some of the prominent existing ones. In our experiments, we set the dimensionality of MeSH embeddings $d = 200$, $\alpha = 0.75$, $\beta = 0.01$, $x_{max} = 100$, $\eta = 0.05$ and run DME for 100 iterations on each time slice.[3]

It should be noted that evaluating LBD systems is not a simple task and remains an open problem [84]. This is because of the unavailability of a comprehensive and exhaustive ground truth results, apart from lack of a systematic comparison methodology. To overcome this difficulty, replicating existing scientific discoveries has been seen as an effective evaluation approach by most LBD researchers. The pioneers in this area of study Swanson and Smalheiser applied their initial model and published several discoveries in medical domain, which were subsequently validated. Since then, these proposed terms in their discoveries have become a gold standard for evaluation. The following are the *de facto* gold test-cases that have certain expected results.

1. Fish-oil (FO) and Raynaud's Disease (RD) (1985)
2. Magnesium (MG) and Migraine Disorder (MIG) (1988)
3. Somatomedin C (IGF1) and Arginine (ARG) (1994)

[3] The source code of DME is available at `https://github.com/XunGuangxu/DME`.

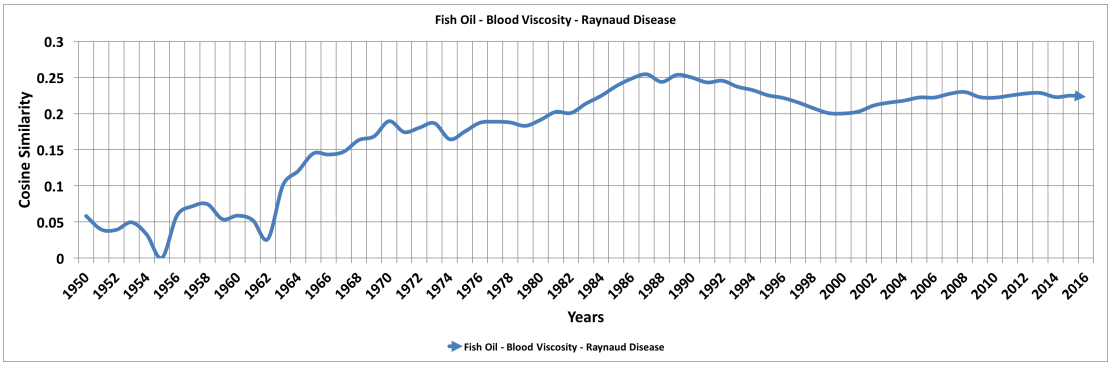


Figure 5.4: Curve of average cosine similarity between Fish Oils - Blood Viscosity - Raynaud's Disease at different time.

4. Indomethacin (INN) and Alzheimer Disease (AD) (1989)

5. Schizophrenia (SZ) and Calcium - Independent Phospholipase A2 (CI-PA2) (1997)

To assess the effectiveness of our model, we perform both evidence based as well as statistical evaluation. The evidence-based evaluation qualitatively determines the extent to which our approach is capable of rediscovering the known knowledge, while the statistical evaluation is intended to provide a quantitative understanding of overall quality of results.

Before we delve into the details of qualitative and quantitative evaluation, we discuss how our model facilitates in analyzing evolution trajectories of MeSH terms, demoting generic terms and ranking candidate intermediate terms.

5.5.1 Evolution of Medical Concepts

Evolutionary trajectories. As our medical knowledge develops, the semantics (medical properties) of MeSH terms evolve, for example, the finding of a new treatment or a new cause to a specific disease would probably result in their medical properties getting more similar. This semantic evolution is reflected as evolutionary trajectories of MeSH embeddings in the vector space. Consider the classic example of *Fish Oils (FO) – Blood Viscosity (BV) – Raynaud's Disease (RD)*, Figure 5.3 shows the two-dimensional projection of the MeSH embeddings and their evolutionary trajectories using t-Distributed Stochastic Neighbor Embedding (t-SNE) [85]. As can be observed, initially in 1953,

Table 5.1: Comparison of average cosine distance change.

MeSH Terms	Average Cosine Distance Change
humans	0.000475
animals	0.000807
female	0.000909
male	0.001940
fish oils	0.350
raynaud disease	0.311
blood viscosity	0.292
epoprostenol	0.345

all three concepts were at different positions, but, as the research over these topics increased in parallel, their implicit semantics started getting closer, making them very close to each other in 1983. This evolutionary behavior eventually in 1986 led to their co-occurrence for the first time in a research article. This growing association trend among the three medical concepts can also be quantitatively measured by the average cosine similarity in the embedding space: (cos-sim($'FO'$,$'BV'$)+cos-sim($'RD'$,$'BV'$))/2, as shown in Figure 5.4. We can see that the average cosine similarity between them gradually increased over time (from 1962-1984) and once the association was formed in 1986 the average cosine similarity between these terms reached a plateau.

Detecting generic MeSH terms. As mentioned before, one of the major challenges in LBD is to detect and differentiate generic terms from informative terms. Generic terms, such as humans and animals, frequently co-occur with a wide variety of other terms, and hence tend to have high association score with most of the terms. However, we would like to demote those generic terms when ranking hypotheses terms as they are not informative. The conventional approach to tackle this issue is to utilize certain heuristic rules such as removing the MeSH terms which appeared more than 10,000 times in MEDLINE documents [76]. However, such heuristics lack clear rationale behind them. Our DME model provides a new insight into the generality of terms. According to the law of conformity [13], the semantics of generic terms tend to remain stable from the viewpoint of evolution. This law is also confirmed by our proposed DME model in bio-medical literature, as shown in Table 5.1. Table 5.1 presents the average cosine

distance changes of four generic terms and four informative terms over time, where the average cosine distance change over time for term i till time T is calculated as:

$$\Delta(\boldsymbol{w}_i, T) = \frac{1}{N_i} \sum_{t=1}^{T-1} I^{(t)}(i) * \text{cos-dist}(\boldsymbol{w}_i^{(t)}, \boldsymbol{w}_i^{(t+1)}), \tag{5.8}$$

where N_i is the number of time slices term i occurred. As one can observe, dynamic MeSH embeddings for generic terms, such as humans, are extremely stable over time. This helps us in penalizing the ranks of terms which are fairly generic.

Ranking candidate intermediary terms. For a closed discovery task, given two previously unconnected terms i and j along with a cut-off time-stamp t, we would like to identify high confident terms k that will connect these inputs terms after t using all the documents published before t. When ranking intermediary terms k, we take into consideration three factors: (1) term k's cosine similarity with i and j at current time stamp t: k should be close to both input terms to be a bridging term; (2) the evolutionary trajectories: k is favored if there is a growing association trend between k and i, j; (3) the generality of term k: we prefer informative terms to generic terms. Therefore, the intermediary terms k are ranked according to:

$$s(k|i, j, t) = \text{sim}(\boldsymbol{w}_k^{(t)}, \boldsymbol{w}_i^{(t)}, \boldsymbol{w}_j^{(t)}) \Delta(\boldsymbol{w}_k, t) \text{trd}(\boldsymbol{w}_k, \boldsymbol{w}_i, \boldsymbol{w}_j, t), \tag{5.9}$$

where $\text{sim}(\boldsymbol{w}_k^{(t)}, \boldsymbol{w}_i^{(t)}, \boldsymbol{w}_j^{(t)})$ denotes k's cosine similarity with i and j at time t. To penalize terms which are close to only one input term but far away from the other input term, we adopt F1 cosine similarity score as $\text{sim}(\boldsymbol{w}_k^{(t)}, \boldsymbol{w}_i^{(t)}, \boldsymbol{w}_j^{(t)})$:

$$2 \frac{\text{cos-sim}(\boldsymbol{w}_k^{(t)}, \boldsymbol{w}_i^{(t)}) * \text{cos-sim}(\boldsymbol{w}_k^{(t)}, \boldsymbol{w}_j^{(t)})}{\text{cos-sim}(\boldsymbol{w}_k^{(t)}, \boldsymbol{w}_i^{(t)}) + \text{cos-sim}(\boldsymbol{w}_k^{(t)}, \boldsymbol{w}_j^{(t)})}.$$

$\Delta(\boldsymbol{w}_k, t)$ reflects the generality of k till time t as defined in Equation 5.8, and $\text{trd}(\boldsymbol{w}_k, \boldsymbol{w}_i, \boldsymbol{w}_j, t)$ is the association trend between k and i, j up untill time t, defined as:

$$\exp\left(\text{acs}(\boldsymbol{w}_k^{(t)}, \boldsymbol{w}_i^{(t)}, \boldsymbol{w}_j^{(t)}) - \text{acs}(\boldsymbol{w}_k^{(a)}, \boldsymbol{w}_i^{(a)}, \boldsymbol{w}_j^{(a)})\right),$$

where $\text{acs}(\boldsymbol{w}_k^{(t)}, \boldsymbol{w}_i^{(t)}, \boldsymbol{w}_j^{(t)})$ stands for the average cosine similarity between them, and a denotes the first time stamp they appeared. Hence, given a closed discovery task, we can rank the intermediate terms based on dynamic MeSH embeddings according to Equation 5.9.

Table 5.2: Top 15 intermediary MeSH terms for FO - RD.

	Intermediary MeSH Term	**Observation**
1	erythrocyte deformability	PMID: 4031661
2	arteriosclerosis obliterans	PMID: 2285650
3	diabetic angiopathies	False: No evidence
4	arteritis	PMID: 1033318
5	double-blind method	False: Too Generic
6	thromboxane a2	PMID: 3797213
7	vascular diseases	PMID: 3797213
8	platelet aggregation	PMID: 3797213
9	fatty acids, essential	PMID:16825676
10	hyperlipidemias	False: No evidence
11	vasodilation	PMID: 3797213
12	blood viscosity	PMID: 3797213
13	platelet aggregation inhibitors	PMID: 3797213
14	platelet function tests	False: Too Generic
15	vasodilator agents	Derivative of vasodilation

5.5.2 Evidence Based Evaluation

In evidence based evaluation, we are interested in finding whether our model can successfully rediscover already established knowledge. Also, another aspect is to examine the validity of certain other higher ranked terms that were unique to our system and not part previous LBD research. We do this by comparing with the *de facto* gold results and manually checking the relationship between the intermediary/bridging terms and the query terms in the literature. To get the medical journal for manual inspection, we formulate a boolean query (e.g. fish oils AND epoprostenol AND raynaud's disease) in Google and examine top 10 results. Tables 5.2, 5.3, 5.4, 5.5, and 5.6 present a consolidated view of our top K=15 ranked results. For each of the valid connection, we provide its corresponding PubMed identifier (PMID). We assume a connection to be a valid connection if it co-occurs together with input query terms after the cut-off date.

Fish-oil (FO) and Raynaud's Disease (RD). To reiterate, in 1986, Swanson[68] explored the research question of "role of dietary fish oils in treating patients with Raynaud's syndrome". Upon manual inspection of literature belonging to Fish oils and Raynaud's disease respectively, he found that Raynaud's disease is aggravated by high *blood viscosity*, high *platelet aggregation*, *Vasoconstriction*, and the ingestion of Fish oils

Table 5.3: Top 15 intermediary MeSH terms for MIG - MG.

	Intermediary MeSH Term	**Observation**
1	calcium	PMID: 3075738
2	nifedipine	PMID: 1071161
3	cerebrovascular disorders	PMID: 23674807
4	ischemic attack, transient	PMID: 8961243
5	potassium chloride	PMID: 4586582
6	potassium	PMID: 4586582
7	verapamil	PMID: 1071161
8	epilepsy	PMID: 3075738
9	ranitidine	False: No evidence
10	phosphorus	PMID: 2584000
11	propranolol	PMID: 3663475
12	lithium	False: Too generic
13	bipolar disorder	PMID: 2855933
14	homeostasis	PMID: 16866716
15	calcium channel blockers	PMID: 1071161

reduced these phenomena.

In table 5.2, it can be observed that we find both *platelet aggregation* and *blood viscosity* at rank 8 and 11 respectively. In this context, it is worthwhile to note that many rediscovery approaches consider it a success if they find *platelet aggregation* in their list of intermediates [70]. In addition to these important connections, other terms in our ranked set such as *'fattyacids, essential'*, *'vasodilation'* are also meaningful pathways.

Magnesium (MG) and Migraine Disorder(MIG). Swanson[86] proposed eleven pathways (intermediates) between Magnesium and Migraine Disorder. These connections are *epilepsy, serotonin, prostaglandins, platelet aggregation, calcium antagonist, type A personality, vascular tone and reactivity, calcium channel blockers, spreading cortical depression, inflammation, brain hypoxia and substance P*. Unlike the previous case, we are unable to achieve high recall. Nevertheless, we obtained important pathways such as *epilepsy*, *calcium channel blockers*, *adenosine triphosphate*, etc, in table 5.3. In this regard, it should be noted that previous research indicates this to be a difficult test case [77].

Table 5.4: Top 15 intermediary MeSH terms for IGF1 - ARG.

	Intermediary MeSH Term	**Observation**
1	somatotropin	PMID: 2406696
2	nitric oxide	PMID: 2655363
3	growth hormone-releasing hormone	PMID: 18537700
4	fibroblast growth factor 2	PMID:3543345
5	lysine	False: No evidence
6	glucagon	PMID:24582776
7	nitric oxide synthase	Derivative of nitric oxide
8	glucagon	PMID:24582776
9	growth disorders	PMID:4881955
10	epidermal growth factor	PMID:4881584
11	endothelins	PMID:4468388
12	somatostatin	PMID:18537700
13	glutamine	False: No evidence
14	cyclic gmp	PMID:11253364
15	neostigmine	False: No evidence

Somatomedin C (SMC) and Arginine (ARG). Somatomedin C (SMC) also known as Insulin-like Growth Factor I (IFG1) is a growth regulating peptide, whereas, Arginine is an important amino acid. They both were found to related to each other through the means of growth hormones such as *somatotropin* and *somatostatin*. Growth hormones tend to influence SMC and ARG in turn stimulates the secretion of growth hormones.

In our results in table 5.4, somatotropin is ranked number 1 and somatostatin is found in top K. Compared to prior approaches[82] which use ad-hoc semantic types to get these results, our model finds them in a completely automated way.

Indomethacin (INN) and Alzheimer Disease (AD). A research question of whether Alzheimer Disease (AD) - a progressive disease that destroys memory and other important functions, can be treated with an inflammatory agent - Indomethacin (INN) was explored during 1990's. Researchers reported that connections such as Acetylcholine, Membrane fluidity to be important pathways. In our results in table 5.5, similar to previous test case, Acetylcholine is ranked 1. Although Membrane fluidity was not ranked in top K, its derivatives were ranked higher. An interesting observation that we would like to discuss here is regarding the term *nitric oxide* (Rank=3). Although

Table 5.5: Top 15 intermediary MeSH terms for INN - AD.

	Intermediary MeSH Term	**Observation**
1	acetylcholine	PMID:2644496
2	amyloid	PMID:1894844
3	nitric oxide	PMID:11080519
4	cerebral cortex	PMID:1618710
5	gastric mucosa	PMID:3526946
6	cerebrovascular circulation	PMID:3656423
7	brain ischemia	PMID:4664815
8	nitroprusside	False: No evidence
9	brain chemistry	False: Too generic
10	neurotransmitter agents	PMID:12421115
11	cerebral infarction	False: No evidence
12	astrocytes	PMID:5098782
13	hippocampus	PMID:4033954
14	atropine	PMID:20980781
15	aging	PMID:12076498

not yet validated, several papers identified nitric oxide as important for understanding alzheimers [77]. Moreover, during 2000-2001, there were studies[87] showing evidence of strong influence of nitric oxide in both Alzheimer's disease and Indomethacin.

Schizophrenia (SZ) and Calcium - Independent Phospholipase A2 (CI-PA2). Schizophrenia is a disorder that affects person's ability to think, feel and concentrate. It is found that CI-PA2 is elevated in SZ patients. After combing independent works of [88] and [89], Swanson and Smalheiser postulated oxidative stress to be the key connecting term. In our results in table 5.6, we were able to cover oxidative stress indirectly through receptors, adrenergic (PMID: 3820966). Also, similar to previous test case, our top ranked term (glutamates) is found to be heavily investigated for its influences in treating Schizophrenia (PMID: 20686195) during more recent years.

5.5.3 Statistical Evaluation

In the previous subsection, we discussed how our system is able to predict novel associations much ahead of their real discovery time. While this is encouraging, one might ask, "How about the overall quality of hypotheses generated?". To measure the overall quality of ranked set, we need a ground truth. However, as mentioned before, there is

Table 5.6: Top 15 intermediary MeSH terms for SZ - CI-PA2.

	Intermediary MeSH Term	**Observation**
1	glutamates	PMID:3782191
2	calcium-binding proteins	PMID:2735787
3	mesencephalon	PMID:2735787
4	photic stimulation	False: Too generic
5	receptors, adrenergic	PMID:2836388
6	genes, immediate-early	False: Too generic
7	photosensitivity disorders	False: Too generic
8	pedigree	PMID:2665704
9	prolactin	PMID:8898352
10	4-chloromercuribenzenesulfonate	False: Inconclusive from literature
11	breast feeding	PMID:1853256
12	pituitary hormone-releasing hormones	PMID:5395093
13	synapses	PMID:22414961
14	myocardial ischemia	PMID: 4548770
15	prostaglandins	PMID:26160611

Table 5.7: Spearman's correlation for FO - RD.

Methods	Top 1505	Top 500	Top 100	Top 20
Graph	0.236	0.142	0.086	-0.266
Static	0.423	0.429	0.440	0.055
DME	**0.430**	**0.430**	**0.460**	**0.066**

no standard ground truth available, thus we generate a "supposed" ground truth based on the documents published after the cut-off date. As an example, consider test case "MG-MIG" whose cut-off year is 1988, so based on the documents in 1989-2016, the ground truth intermediate terms k are ranked according to:

$$gt(k) = \frac{\#(k, \text{“}MG\text{”}) + \#(k, \text{“}MIG\text{”})}{\#(k)}, \tag{5.10}$$

where $\#(i, j)$ is the number of times terms i and j co-occur, and $\#(i) = \sum_j \#(i, j)$. Hence, the ranked hypotheses can be evaluated by measuring the Spearman's rank correlation with the ground truth ranked set. We compare our DME model with two baselines: Static (described in Section 5.4.2) and Graph [71]. Graph is a distribution-graph theoretic approach and we made our own implementation of it. This methodology

Table 5.8: Spearman's correlation for MIG - MG.

Methods	Top 3976	Top 1500	Top 300	Top 200
Graph	0.203	0.035	-0.023	0.013
Static	0.351	0.186	0.161	0.152
DME	**0.357**	**0.201**	**0.178**	**0.174**

Table 5.9: Spearman's correlation for IGF1 - ARG.

Methods	Top 7599	Top 4000	Top 400	Top 300
Graph	0.266	0.185	0.063	0.063
Static	0.307	0.192	0.169	0.172
DME	**0.319**	**0.197**	**0.196**	**0.183**

uses a combination of graph-based global and local measures to rank the intermediary/connecting terms between a given pair of input terms. The comparison results on the 5 test cases are reported in Tables 5.7, 5.8, 5.10, 5.9 and 5.11. The first column of each table is calculated on the entire ground truth ranked set. As can be observed, DME consistently outperforms the baselines. For other prior works, as they were performed under different settings and a complete ranked set is difficult to obtain, we cannot compare our results with them.

5.5.4 Open Discovery

In this subsection, we briefly discuss on the portability of our methodology to open discovery problem. The difference between the closed discovery and the open discovery is in the absence of a "grounded" end medical concept in the latter, i.e., for a closed discovery "Is fish oil connected to Raynaud disease?", the corresponding open discovery query would be "What are all the treatments for Raynaud Disease?" It can be seen that the first query contains two grounded concepts - fish oil and Raynaud disease while the open discovery query only contains one grounded concept - Raynaud Disease. However, the open discovery query also contains meta constraint - in this case the semantic label "treatments". More formally, in biomedical knowledge the semantic label "treatment" corresponds to the semantic type *Biologically active substance*.[4] Hence, we first filter

[4] The explicit semantic types of MeSH terms can be obtained from UMLS [80].

Table 5.10: Spearman's correlation for INN - AD.

Methods	Top 5351	Top 2500	Top 500	Top 100
Graph	**0.188**	0.036	0.051	0.023
Static	0.163	0.139	0.224	0.230
DME	0.168	**0.144**	**0.239**	**0.239**

Table 5.11: Spearman's correlation for SZ - CI-PA2.

Methods	Top 519	Top 100	Top 50	Top 20
Graph	0.121	-0.244	-0.034	0.058
Static	0.317	0.362	0.176	0.202
DME	**0.327**	**0.412**	**0.247**	**0.373**

from the collection of all medical concepts to retain only those terms that satisfy the specified meta constraint. Once we have filtered and retained only relevant candidate terms, one can repeat the closed discovery process for each of the candidates. In formal words, given the input term i and a time stamp t, the candidate hypotheses are generated and ranked based on the same scheme that is used in Equation 5.9 except that the term j (denoting the end term) is ignored. In the interest of space, we provide only a summary view of the results (Table 5.12) for the first test case (FO-RD). In terms of ranking, our framework was able to identify fish oils at rank 27 out of 104. Compared to many baselines [77, 90], we were able to identify fish oil at much higher rank and that too with no manual intervention. Furthermore, some of the other terms ranked higher were also derivatives of fish oils such as '*docosahexaenoic acids*', '*eicosapentaenoic acid*', '*lipoproteins, ldl*', etc. Such terms are also deemed to be valid by researchers [82, 90, 91]. Thus, we show that our dynamic MeSH embedding based approach is eminently suitable not only for closed discovery but also could be adapted for open discovery setting.

5.6 Conclusions

In this chapter, we propose the DME model based on the evolutionary behavior of medical concepts. DME captures implicit semantic regularities of MeSH terms and tracks their semantic changes over time in the embedding space. By studying the

Table 5.12: Spearman's correlation for open discovery.

Methods	Top 1738	Top 1000	Top 800	Top 200
Static	0.127	0.011	0.031	0.027
DME	**0.189**	**0.101**	**0.081**	**0.068**

evolutionary trajectories of MeSH embeddings, informative terms can be promoted, basing upon which novel medical hypothesis can then be discovered. The methodology contributes to LBD research specifically because it uses notion of semantic evolution to facilitate making discoveries from scientific literature.

Chapter 6

Learning Time-aware Representations of Sequential Data

6.1 Introduction

An epileptic seizure is a transient aberration due to abnormally excessive or synchronous neuronal activities in the brain. The disease epilepsy is defined as an enduring predisposition in brain which generates epileptic seizures. The symptom of epilepsy can vary from uncontrolled jerking movement to as subtle as a temporary unconsciousness [92]. Frequent seizures are dangerous in which it may result in serious physical injuries and even death. According to the study, 5%-10% of the people over 80 years old have experienced the epileptic seizures for at least once. After the first experience, they would suffer from another epileptic seizure with a probability of 40%-50%. Currently about 1% of the global population are affected by epileptic seizures and at some point in time the number used to be 4% at its highest [93].

Considering the large population affected by epileptic seizure and the serious outcome caused by epileptic seizure for the patients, a device that can quickly detect the onset of seizure and deliver therapy can be of great help. In recent years, the surge in brain-computer interface (BCI) technology introduces tremendous opportunities to applying physiological signals to biomedical applications. According to previous studies, the electroencephalogram signals (EEG) are closely related to brain activities and

can be used to detect neural diseases [94, 95]. Therefore, learning the EEG signals is an efficient way to infer the onset of seizures and analyzing the seizures. While we can collect EEG signals from different parts of body, the scalp EEG is most widely adopted, which is a non-invasive, multi-channel recording of the brain's electrical activities.

The condition of seizures has proven to be closely related to the neural electrical activities which can be reflected on EEG signals. Nevertheless, there still exist several challenges in the automatic seizure detection task:

First, seizure and non-seizure states have considerable overlap in patients' EEG signals.

Second, the EEG of epilepsy patients has more than one states for both the seizure and non-seizure status and may constantly transition between them.

Third, most conventional learning models directly send all the input to the seizure classifiers without extracting features and are not able to analyze the correlation between different input data, which would result in the failure to recognize the temporal signal patterns [96, 97, 98].

Fourth, as the characteristics of seizures on EEG might vary significantly across patients, we can hardly design a general seizure detector. Because of this cross-patient variability in seizure and non-seizure activities, patient non-specific classifiers are usually not able to obtain a high accuracy and suffer from long delays in detecting the onset of a seizure. On the contrary, patient specific classifiers can exhibit impressive performance because they do not need to deal with the cross-patient variability [99].

Fifth, in practice, the automatic seizure detector should be able to detect the onset of a seizure quickly. Besides, it should also be able to handle the unbalanced training data, because seizures are rare events which results in the paucity of seizure training data.

Facing these challenges, we propose an innovative method to capture the temporal features and context information hidden in EEG data. Since the onset of a seizure is related to a sequence of EEG signals rather than the values at a certain point, temporal analysis is necessary and crucial for the seizure detection. More specifically, our model handles these challenges with the following strategies:

- We segment the EEG into small pieces of fixed length with a sliding window. By sliding the window with a fixed step length, the EEG is segmented into numerous small fragments as the "EEG words" for further analysis.

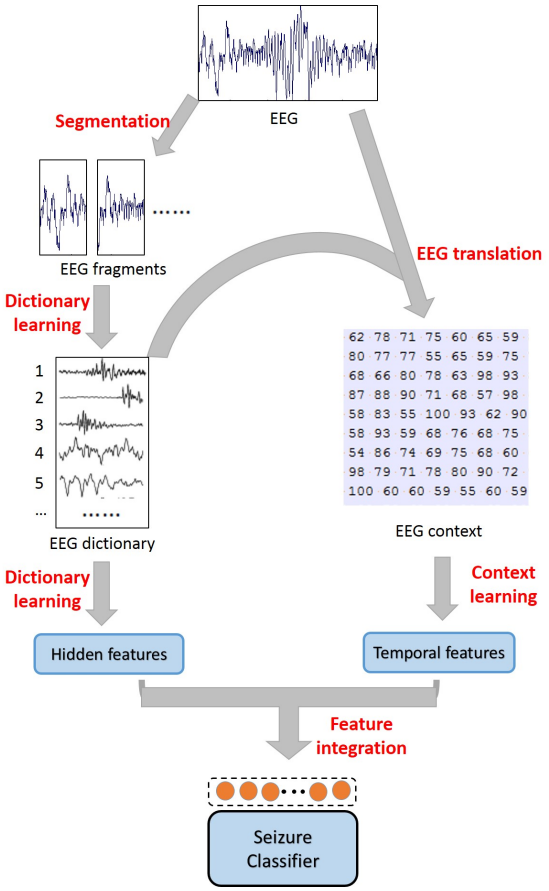
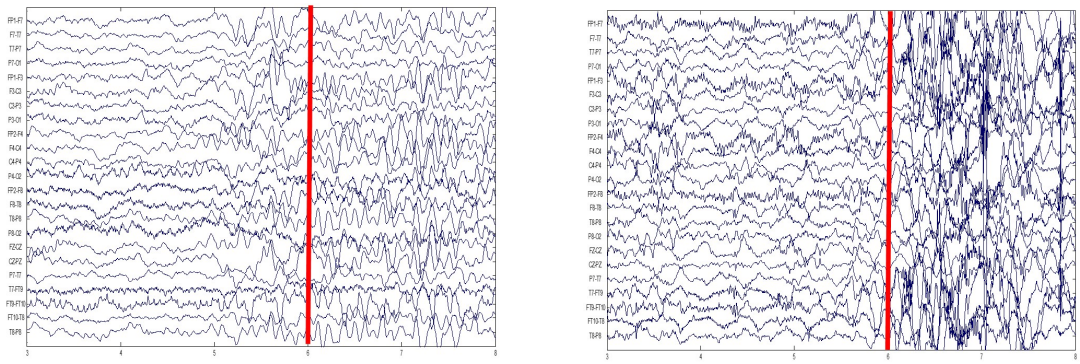


Figure 6.1: Schematic illustration of the overall framework.

- We extract the hidden inherent features within each EEG fragment. One single feature corresponds to one "EEG word" in our learned "EEG dictionary".
- We explore the temporal knowledge by learning the context information of EEG fragments. After translating the EEG fragments into "EEG words", we can infer the context knowledge for an EEG fragment.
- Finally, we combine the hidden features of each EEG fragment and the temporal knowledge together and subsequently send them to a seizure classifier. In this chapter we concentrate on the binary classification (seizure or non-seizure states) while the proposed model can be easily extended to uncover more physiological classes.

Based on the above strategies, our model is described in Figure 6.1.



(a) The scalp EEG of patient A.

(b) The scalp EEG of patient B.

Figure 6.2: The scalp EEG of two patients.

6.2 Dataset

The dataset we use is the scalp electroencephalogram collected at the Children's Hospital Boston [95]. EEG measures the electrical activities in the brain by attaching multiple electrodes to the patient's scalp. Each EEG channel records the voltage change between a specific pair of electrodes, and therefore reflects the electrical activities in the corresponding region.

This dataset consists of the EEG recording intractable seizures from pediatric subjects. 23 patients were involved in the dataset, including 5 males and 18 females from age 2 to age 22, to characterize their seizures and access the necessity of surgery for them. All the signals were recorded at 256 Hz with $16 - bit$ resolution. In most files, there are 23 EEG channels and 24 channels in a few cases.

Following the onset of a seizure, a set of EEG signals show dramatic changes from the non-seizure states. And this will assist the seizure detector in distinguishing the seizure and non-seizure states. For example, Figure 6.2a illustrates the onset of a seizure of patient A. Patient A's seizure starts at the 6th second as the red bar shows in Figure 6.2a, and then the onset of this seizure comes with the significant changes of EEG signals.

However, as we mentioned, the characteristics of seizures on EEG might vary significantly across patients, and this variability will make the seizure detection problem even more difficult. Figure 6.2b shows the onset of a seizure of patient B. Patient B's seizure also starts at the 6th second as the red bar shows in Figure 6.2b. Significant

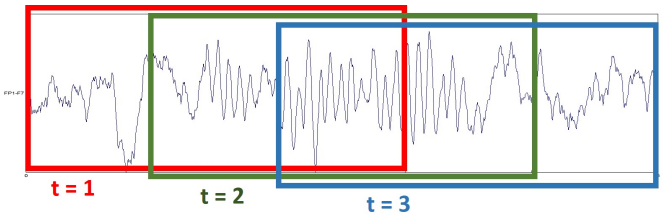


Figure 6.3: An example of EEG segmentation.

changes can still be observed between seizure and non-seizure states, but the structure of seizures on EEG differs across patients.

Sometimes, EEG signals show some certain kinds of rhythmic activities when people are excited, which are also different from the calm states. But these EEG fragments should not be confused with seizures. The ambiguity brought by these activities requires the seizure detector to learn the features of seizures.

6.3 Methodology

In this section, we propose a novel framework to extract the hidden inherent features and temporal information in EEG signals. We start by discussing the first step of our model, segmenting EEG data.

6.3.1 EEG Segmentation

Since EEG signals cannot be explicitly segmented into sub-fragments associated with physiological meanings, we segment it into several epochs of fixed length. With a sliding window of length L (for example, 3 seconds), we build our EEG fragment pool by sliding the window by 1 second at each step.

Figure 6.3 shows an example of how to segment an EEG signal into three fragments, in which the length of the sliding window is fixed to $L = 3$ seconds and the step length is 1 second.

By segmenting the EEG signals into numerous EEG fragments, we obtain an EEG pool of EEG fragments. Our further analysis and experiments are conducted on these EEG fragments.

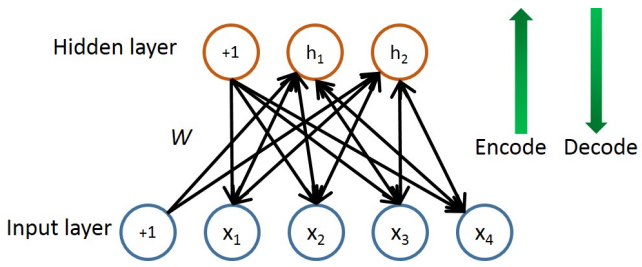


Figure 6.4: The structure of a simple autoencoder.

6.3.2 EEG Dictionary Learning

After the EEG fragment pool is built, we use a sparse autoencoder (SAE) to extract features for these fragments. Autoencoder is an unsupervised neural network based model which aims at discovering interesting structures of data by reconstructing the input [100].

In its simplest form, an autoencoder consists of two parts, an encoder and a decoder. Autoencoder can also be viewed as a technique for feature extraction and dimensionality reduction. The encoder reduces the dimensionality of the input data to obtain principal features, and the decoder is tuned to reconstruct the input data based on the output features of the encoder. Hence by minimizing the reconstruction error, we can get the optimal autoencoder whose encoder extracts features with reduced dimensionality and decoder reconstructs input data from the extracted features. In particular, when we select a linear activation function and use less hidden units than the input dimensions, the encoder works similarly with the principal component analysis (PCA) [101]. However, when a non-linear activation function is adopted, the autoencoder has proven to be capable of learning more useful features than PCA [102].

Figure 6.4 depicts the structure of an antoencoder with one hidden layer, where x is the input data, h is the hidden unit and $+1$ term is adopted to integrate the bias. The autoencoder tries to learn a hidden layer that satisfies $g(f(x)) \approx x$, where $f(x)$ extracts features from input s and $g(y)$ reconstructs the original data from the extracted features. In other words, it aims at learning a model to approximate the output with the input. The the bottom up structure denotes the process of encoding, as the left green arrow shows in Figure 6.4. The encoder corresponds to the function f that maps the input data x to the hidden layer h. The function f is defined as:

$$h = f(x) = \sigma(Wx + b_h), \tag{6.1}$$

where b_h represents the bias, W represents the weight matrix from input data to the hidden units and $\sigma(x)$ is the activation function. In our test we adopt the non-linear logistic sigmoid function, as follows:

$$\sigma(x) = \frac{1}{1 + e^{-x}}.$$

The top down structure denotes the process of decoding, as the right green arrow shows in Figure 6.4. The decoder corresponds to the function g that reconstructs the input data from the hidden features:

$$\tilde{x} = g(h) = \sigma'(W'h + b_x), \tag{6.2}$$

where b_x is the bias, W' is the weight matrix from the hidden features to the reconstruction and $\sigma'(x)$ is the activation function of the decoder. Usually the decoder adopts the same activation as the encoder for simplicity, and thus for the decoder we adopt the logistic sigmoid function as well.

As the autoencoder aims at reconstructing the input data, the cost function in terms of parameters $\theta = \{W, W', b_h, b_x\}$ is defined as:

$$J_{AE}(\theta) = \sum_x L(x, \tilde{x}) = \sum_x L(x, g(f(x))), \tag{6.3}$$

where L represents the reconstruction error, which is measured by the cross-entropy loss:

$$L(x, \tilde{x}) = -\sum_x (x \log(\tilde{x}) + (1 - x) log(1 - \tilde{x})).$$

In our case, given an 3-second EEG fragment, the input size for the autoencoder is $256\ Hz \times 3\ seconds = 768$. This may result in the number of hidden units being also very large. For each input EEG fragment, intuitively it should only activate a few of the features rather than most of the features. So when the number of hidden units is large, we can still discover some interesting structure by imposing a sparsity constraint on the hidden units [103]. Thus, we use a sparse autoencoder to deal with our EEG dictionary learning task.

A hidden unit is considered of being "inactive" when its output is close to zero, and of being 'active' when its output is close to one. Specifically the sparsity constraint makes one hidden unit inactive most of the times. For hidden unit j, we define its average activation $\hat{\rho_j}$ over all the input data x as:

$$\hat{\rho_j} = \frac{1}{N} \sum_{i=1}^{N} (f_j(x_i)),$$

where N is the size of input data, and $f_j(x_i)$ is the output of hidden unit j on the i-th input data. Then the average activation is constrained to the sparsity parameter ρ which should be quite small (for example $\rho = 0.05$). To measure the difference between ρ and $\hat{\rho}_j$, Kullback-Leibler (KL) divergence is adopted [104]:

$$KL(\rho \| \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j}.$$

This function penalizes the hidden units being active for too many times. Incorporating the sparsity constraint into the autoencoder cost function in Equation 6.3, we get the overall cost function for the sparse autoencoder:

$$J_{SAE}(\theta) = J_{AE}(\theta) + \beta \sum_{j=1}^{M} KL(\rho \| \hat{\rho}_j), \tag{6.4}$$

where J_{AE} is the cost function defined in Equation 6.3, β is the parameter that controls the weight of the sparsity constraint and M is the number of hidden units. It is noteworthy that, the average activation $\hat{\rho}_j$ is also a function of θ. We use backpropagation to update the parameters, and the gradient of the cost function is computed as:

$$\frac{\partial J_{SAE}}{\partial w(k)} = \frac{\partial J_{AE}}{\partial w(k)} + \beta(-\frac{\rho}{\hat{\rho}_k} + \frac{1-\rho}{1-\hat{\rho}_k}). \tag{6.5}$$

The sparse autoencoder we have discussed above has only one hidden layer, and in our model, we use a sparse autoencoder with two hidden layers, which is able to capture more abstract features. Each learned feature corresponds to a vocabulary in the EEG dictionary. And the EEG dictionary is constructed by decoding all the learned features.

6.3.3 EEG Sequence Translation

As the sparse autoencoder is trained, each EEG word is obtained by decoding each hidden unit and represents one basic signal type. The EEG dictionary is a set of all the EEG words, and each EEG fragment can be viewed as a combination of EEG words in the dictionary. For each EEG fragment, different features have different weights due to the different proportions of basic signal types in it. Therefore the EEG fragment can be sampled to a single EEG word according to the normalized feature weights.

Given a continuous EEG signal, we can translate it into a sequence of words by converting each EEG fragment of it into an EEG word in the dicionary. Translating continuous signal into discrete words would help us learn the temporal context information in further analysis.

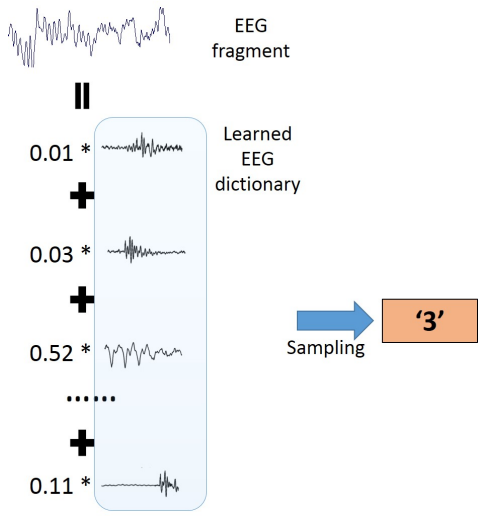


Figure 6.5: An example of EEG sequence translation.

Figure 6.5 shows an example of how to translate an EEG fragment to the corresponding EEG word based on the learned dictionary. More specifically, given an EEG fragment and the EEG dictionary learned by the sparse autoencoder, the EEG word in the dictionary corresponding to this EEG fragment is drawn from the multinomial distribution:

$$P(\epsilon_i) = \frac{h_i}{\sum_{j=1}^{M} h_j},$$

where ϵ_i is the i-th EEG word in the dictionary, h_i is the output of the i-th hidden unit on this EEG fragment input and M is the number of hidden units. Because of the sparsity constraint, most of the h_i should be close to zero, which means each EEG fragment is basically composed of a few main signal types.

6.3.4 EEG Context Learning

In order to capture the temporal features of seizures on EEG signals, we design an EEG context learning algorithm to analyze the EEG sentences.

In the previous EEG translation step, every EEG fragment is translated to an EEG word, so the continuous EEG signals are translated to EEG sentences of EEG words. In this way, we are able to learn the temporal features hidden in the context information of the EEG sentences.

The main idea of the EEG context learning algorithm is to infer the current EEG

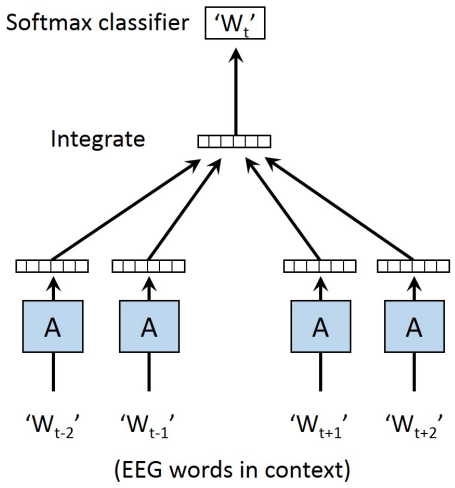


Figure 6.6: The framework of the EEG context learning algorithm.

word based on its context words. This intuition is inspired by the Continuous Bag-of-word (CBOW) model [1], where each word is represented by a vector of fixed length and words with similar semantics would be mapped to close positions in the vector space by learning the context information.

In our model, the context of an EEG word is drawn from the EEG sentence with a window of length $2k+1$, i.e., the previous k words and the following k words form the context of the current word.

Figure 6.6 shows the framework of this EEG context learning algorithm. W_t is the current word to predict, and $W_{t-2} \sim W_{t+2}$ are the context words of W_t. Each EEG word is mapped to an unique vector, represented by a column in matrix A. The integration of all the word vectors in context should lead the softmax classifier to choose the current word W_t.

More formally, our EEG sentence dataset consists of T training EEG words $a_1, a_2, ..., a_T$. We are going to predict each EEG word based on its neighborhood. So all the context EEG word vectors make a contribution to the prediction task about the current word in the context. Thus the objective of this EEG context learning algorithm is to maximize the average log probability:

$$L = \frac{1}{T} \sum_{t=k}^{T-k} \log p(a_t | a_{t-k}, a_{t-k+1}, ..., a_{t+k}), \tag{6.6}$$

where $2k+1$ is the size of the context window, i.e., when predicting every EEG word,

only its previous k words and following k words contribute to the prediction about this word as its context. The other EEG words outside this context window are not considered.

The $p(a_t|a_{t-k}, a_{t-k+1}, ..., a_{t+k})$ in Equation 6.6 is the prediction task for EEG word a_t. It is calculated by a multi-class softmax classifier, as follows:

$$p(a_t|a_{t-k}, a_{t-k+1}, ..., a_{t+k}) = \frac{e^{y_{a_t}}}{\sum_i e^{y_i}}, \tag{6.7}$$

where e^x is the exponential function and y_i is the unnormalized log probability for each output EEG word a_i. As Figure 6.6 shows, the process of prediction is based on a two-layer neural network, which means there are three steps for each prediction: first we need to project all the input words into the vector space, and second we integrate the vectors, and finally we calculate the output y. So y_{a_t} can be computed as:

$$y_{a_t} = b + Uh(a_{t-k}, a_{t-k+1}, ..., a_{t+k}; A), \tag{6.8}$$

where b and U are the parameters of the softmax classifier, and h is the integration of the context EEG word vectors extracted from matrix A. The integration is typically implemented as either average function or the concatenation.

In practice, for the sake of fast training, the softmax classifier is usually replaced by the hierarchical softmax classifier. In our model, the hierarchical softmax classifier is based on a binary Huffman tree, where the shortest path is assigned to the most frequent EEG word.

Applying hierarchical softmax classifier accelerates our model in three ways: first, according to the strategy of building the Huffman tree, frequent EEG words are assigned short codes, which means the overall accessing time is shorter; second, by representing the vocabulary with a binary tree structure, the average seeking time reduces from $O(N)$ to $O(\log(N))$, where N is the size of the vocabulary and $\log(N)$ is the height of the Huffman tree; third, by storing the EEG words in a tree structure, in each round of update, we only need to access and update the nodes on the path rather than accessing all the words in the vocabulary.

Similar with the other neural network models, the EEG context learning algorithm is trained with backpropagation. After learning EEG contexts, the EEG words with similar properties are mapped to close positions in the vector space [105]. These vectors can be used as the temporal features because in the EEG context learning process,

the order of the EEG words in an EEG sentence is considered as part of the context information.

6.3.5 Seizure Detection

The final features we use are the combination of the hidden inherent features within the EEG fragments extracted in the EEG dictionary learning process and the temporal features extracted in the EEG context learning process.

After concatenating the hidden inherent features and the temporal features, we use them together with the labels to train a support vector machine (SVM) classifier [106]. And we name our model the Context-Learning Based EEG Analysis for Seizure Detection (Context-EEG).

By incorporating the hidden inherent features and temporal features into the classifier, SVM is able to find a more distinct hyperplane between the seizure and the non-seizure EEG fragments.

6.4 Experiments

We conduct computational experiments to show the effectiveness of our model Context-EEG for detecting the onset of an epileptic seizure. To achieve this, we benchmark our model on the CHB-MIT scalp EEG dataset mentioned in Section 6.2.

6.4.1 Seizure Detection

Task and baselines. In order to design a general seizure detecting algorithm, i.e., a non-patient specific seizure detecting algorithm, we combine 4302 EEG fragments from four different patients as our experiment dataset, and randomly choose 3500 EEG fragments out of it as the training set and use the other EEG fragments as the test set. Given a piece of EEG fragment, it's a two-way classification task where the class labels are {*seizure, non-seizure*}. We measure the performance of each algorithm by the classification error rate, the receiver operating characteristic (ROC curve) and the area under curve (AUC).

Since it is a classification task, we apply several widely used classification algorithms as the baseline algorithms, including SVM and neural network (NN) [107]. For the sake

Table 6.1: The error rates of each method.

Methods	Error rate
SVM	23.43%
NN	26.22%
DSVM	29.71%
DNN	30.21%
PSVM	28.71%
PNN	26.82%
Context-EEG	**22.93%**

of fairness and to avoid the curse of dimensionality, it is necessary to reduce the dimensionality of the data before we send it to SVM and NN. So we employ the decimation process by downsampling the EEG signals, and we call the SVM with downsampling DSVM, the NN with downsampling DNN. Also, we use the principal component analysis (PCA) [108] as the data preprocessing mechanism, and we call the SVM with PCA as PSVM, the NN with PCA as PNN.

Experiment protocols. In our dataset, a seizure is usually 30∼100 seconds long and surrounded by one hour long non-seizure signals, which means seizures are rare events. Considering the rarity of seizure events, we trim our test set to balance the number of seizure fragments and non-seizure fragments to around 50-50. Otherwise, simply by labeling all the test samples as non-seizure state, a classifier can obtain an error rate as low as $30s/3600s = 0.8333\%$.

Since the original sampling rate is 256 Hz and each file contains 23 channels of EEG signals, a 3-second long EEG fragment consists of 256 $Hz \times 3$ $seconds \times 23$ $channels =$ 17664 data points. This high dimensionality problem would not only put the classifiers at the risk of the curse of dimensionality, but also consume a lot of time and space. So for the baseline methods with decimation process, we reduce their dimensions of the input data to the same dimensions as Context-EEG by PCA and downsampling.

Results. The error rates of different methods are reported in Table 6.1. We can see that our model Context-EEG outperforms the other methods by averagely 5 percent. It is worth noticing that the performances of SVM and NN decrease quite a lot as the

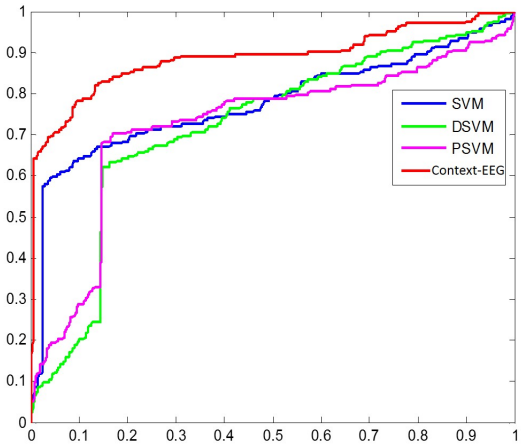


Figure 6.7: The ROC curves of the proposed model and the baselines.

dimensionality of their input data decreases. The original SVM performs well at the cost of acceptable dimensionality. However, after reducing its dimensionality to the same dimensionality as ours, PSVM and DSVM perform much worse. And the comparison result between two different decimation approaches also shows that using PCA is a better way to extract principal components of data than just simply downsampling.

Figure 6.7 and Table 6.2 show the ROC curve and the AUC of each method respectively. We can see that our model performs much better than the other methods.

In the ROC curve figure, among all the baseline methods, SVM performs the best. However, even though the dimensionality of SVM is 64 times as high as the dimensionality of the Context-EEG model, SVM still performs much worse than the Context-EEG model. And we can see that, the true positive rate of the Context-EEG model increases at a very fast speed in the beginning when the false positive rate is still close to zero, which means Context-EEG is able to capture the important features to represent and separate seizure and non-seizure data points effectively.

As shown in Table 6.2, the AUC of the Context-EEG model is higher than the other methods' AUCs by 14 percent averagely even though it has the lowest dimensionality among all the methods. As with the error rate results and the ROC curves, reducing the dimensionality of input data for SVM by downsampling and PCA has a rather big impact on the AUC, and using PCA to extract principal components is a little bit better than just simply downsampling the data.

It is worth noticing that the Context-EEG model is slow at learning but extremely

Table 6.2: The AUC of each method.

Methods	AUC
SVM	0.7764
DSVM	0.7208
PSVM	0.7232
Context-EEG	**0.8880**

fast at prediction, because once the training step is finished, the parameters will be stored and will not change anymore. When a new EEG fragment comes, the classification will be done in $O(N)$ time. So as a real-time seizure detecting algorithm, Context-EEG would be of great help for the patients in practice.

6.4.2 EEG Dictionary Learning and EEG Signal Reconstruction

The first step of our feature extraction process is to learn the hidden inherent features within each EEG fragment. In this step, we learn the hidden inherent features by setting the output of the sparse autoencoder equal to the input. Hence we can claim that the features are well learned if the features are able to precisely reconstruct the input data. Because only if the learned features contain all the crucial information of the input data, we can reconstruct the data based on the learned features.

Usually the dimensionality of the features is much lower than the dimensionality of the original data, so as we are extracting the hidden inherent features of EEG fragments, we are also reducing the dimensionality.

Figure 6.8 illustrates the EEG data reconstruction process. We can see that the reconstructed data (the red lines in the figure) is quite similar with the original data (the blue lines in the figure). The upper figure is an EEG fragment of a non-seizure state, where the EEG signal is regular and clean. And in this case, the learned features successfully reconstruct every peak and every valley of the original EEG signal, and the rebuilt value is almost the same. The lower figure is an EEG fragment of the onset of a seizure, where the second half of the EEG signal is quite intense and irregular. Despite this intensity and irregularity of seizure fragment, the original EEG signal can still be reconstructed from the learned features as the red line shows.

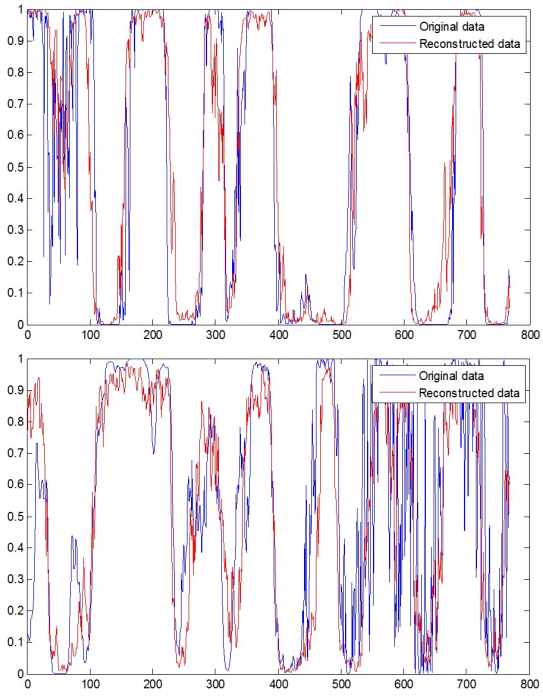


Figure 6.8: Two examples of EEG data reconstruction.

6.4.3 Parameter Sensitivity

In this part, we show the performance of the proposed method in various learning scenarios by tuning the number of hidden units M of the sparse autoencoder. In the EEG dictionary learning step, the output of the hidden layer denotes the extracted features of an EEG fragment, and each hidden unit is directly associated with each EEG word in the EEG dictionary. In practice, the number of hidden units M of the sparse autoencoder not only affects the training speed to a great extent, but also determines the dimensionality of the feature space in the classification step. Hence we conduct the parameter sensitivity experiment on the number of hidden units M.

As the input size of the sparse autoencoder is 256 $Hz \times 3$ $seconds = 768$, the number of hidden units M varies in the range of 0 and 768. So we set the number of hidden units $M = 50,\ 150,\ 250,\ 400,\ 500$, and measure the performance of Context-EEG respectively.

Figure 6.9 shows the ROC curves with different parameter settings. The proposed model gets the best performance when M is 250. Comparing to the input size 768, the dimensions are reduced effectively and 250 hidden units are enough to capture the

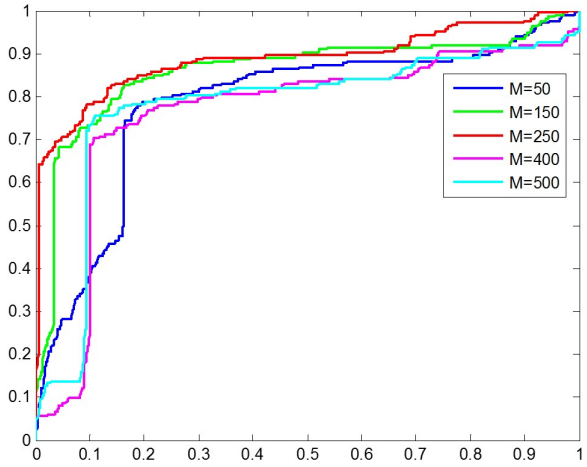


Figure 6.9: The ROC curve for the parameter sensitivity experiment.

important information of the original data. While too few hidden units might result in the proposed model being unable to extract enough features, such as $M = 50$, and too many hidden units might also put the proposed model at the risk of the curse of dimensionality, such as $M = 500$. The AUC and error rate of parameter settings also affirms this conclusion as shown in Figure 6.10. When the number of hidden units is too large or too small, the performance of Context-EEG decreases somewhat, but we can observe that it still outperforms the baseline methods.

6.5 Conclusions

In this chapter, we design and evaluate the context-learning based EEG analysis for seizure detection model (Context-EEG) that utilizes the scalp EEG to detect the onset of a seizure. The proposed model is a general, non-patient specific model which is capable of extracting both the hidden inherent features and the temporal features for the EEG signals. The hidden inherent features are extracted from each EEG fragment internally by a sparse autoencoder and the temporal features of an EEG fragment are extracted in its EEG context by the EEG context learning method. When detecting seizure with respect to a given EEG fragment, not only its internal hidden features but also the temporal features make a contribution to the classification task.

The proposed method has been tested on the CHB-MIT scalp EEG dataset and compared with several baseline methods. In general, the results show the effectiveness

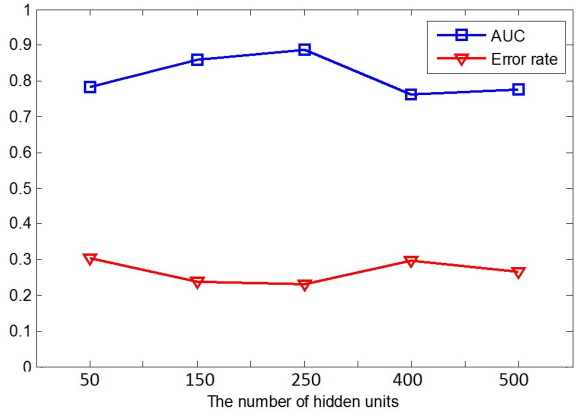


Figure 6.10: The AUC and error rate of Context-EEG with different parameter settings.

and superiority of the proposed model in detecting epileptic seizures. Since the proposed model is very fast at testing, once we obtain the trained model, we can detect the onset of a seizure in real time.

Part III

Contextual Representation Learning Based on Domain Context

Chapter 7

Extracting Biomedical Features Using Domain Attentions

7.1 Introduction

MEDLINE[1] , the primary component of PubMed[2] , is a bibliographic database maintained by U.S. National Library of Medicine (NLM). As the online counterpart to MEDLARS (MEDical Literature Analysis and Retrieval System), MEDLINE currently covers more than 5,200 worldwide journals, and contains more than 24 million references to journal articles in life sciences with a concentration on biomedicine. A distinctive feature of MEDLINE citations is that they are indexed with NLM Medical Subject Headings (MeSH)[3] . The MeSH thesaurus is a controlled vocabulary curated by the NLM experts and used for indexing, cataloging and searching for biomedical articles and information [109, 110]. Thus accurate MeSH indexing greatly facilitates biomedical research and knowledge discovery [111, 112, 113].

Currently, MeSH indexing for MEDLINE is mainly performed by the human experts in NLM. They have to go through the full text of each biomedical article to assign suitable MeSH terms. This ensures high accuracy of MeSH indexing but inevitably renders it very expensive. It is estimated that the average cost of annotating one biomedical article is around $9.4 [114]. More than 813,500 citations were added to MEDLINE in the year of 2017, and this number is rapidly increasing by the year. Apart

[1] https://www.nlm.nih.gov/bsd/medline.html
[2] https://www.ncbi.nlm.nih.gov/pubmed/
[3] https://www.nlm.nih.gov/mesh/meshhome.html

from the huge monetary cost, manual MeSH indexing could also cause a possible delay before a newly published biomedical article gets annotated. This presents a challenge to the NLM experts to annotate biomedical articles efficiently and promptly.

Therefore, a system that can automatically annotate biomedical articles with relevant MeSH terms or assist human experts could be of great help. To this end, NLM has developed Medical Text Indexer (MTI) [115, 114, 116]. MTI takes the title and abstract of an article as the input and outputs relevant MeSH terms. MTI mainly consists of two modules: MetaMap Indexing (MMI) and PubMed-Related Citations (PRC). MetaMap [117] is a software tool to extract biomedical concepts from the text. MMI recommends MeSH terms based on the biomedical concepts discovered by MetaMap. PRC recommends MeSH terms by looking at the MeSH annotations of similar citations in MEDLINE found by the PubMed-Related Articles (PRA) algorithm [118]. The two sets of MeSH terms are combined to generate the final list of MeSH recommendations.

In order to continue to advance the development of MeSH indexing systems, the BioASQ challenge[4] on biomedical semantic indexing and question answering is held every year since 2013 [119]. One of the two BioASQ tasks is to annotate new MEDLINE documents with relevant MeSH terms before MEDLINE curators annotate them manually. As new manual annotations become available, they are used to evaluate the performance of participating systems. Many new MeSH indexing systems have been proposed since then, e.g., MetaLabeler [120], MeSHLabeler [121] and DeepMeSH [122]. MetaLabeler trains an independent binary classifier for each MeSH term; MeSHLabeler proposes to integrate MetaLabeler with multiple evidence such as similar publications and term frequencies; and DeepMeSH is an improved version of MeSHLabeler by incorporating deep semantics in the word embedding space [1, 123, 124]. They also have another classifier to determine the number of MeSH terms to recommend.

Formally speaking, MeSH indexing is a multi-label classification task, where each MeSH term can be regarded as a class label and each article can be labeled with multiple MeSH terms. Compared with regular multi-label classification problems, the large size of MeSH vocabulary and the imbalanced nature of different MeSH terms pose more challenges to the MeSH indexing problem. Currently there are more than 28,000 distinct MeSH terms and new MeSH terms are added to the vocabulary every year. The most frequent MeSH term "humans" appears around 8,000,000 times in MEDLINE citations, while there are hundreds of infrequent terms that appear less than 10 times.

[4] http://bioasq.org/

These challenges have been taken into consideration by the previous researchers when designing their MeSH indexing systems. However, there are some other challenges and limitations that previous systems seem to have overlooked. First, the biomedical articles are sequences in nature, but most previous systems are based on models that cannot be easily used for sequential modeling in an end-to-end fashion, such as K-Nearest-Neighbors (KNN) and Support Vector Machine (SVM). Second, most previous systems train independent classifiers for each MeSH term, resulting in extremely long training time, high disk usage and inability to collaboratively train the classifier and exploit the correlation between different MeSH terms at the same time. Third, every time a new biomedical article is added, the previous MeSH indexing systems need to find similar articles from the MEDLINE database. In other words, millions of MEDLINE articles have to be stored with the system and a thorough search has to be done for each indexing. This further exacerbates the time and space consumption for the existing systems.

Deep learning is a family of machine learning methods that employ multiple processing layers to learn representations of data with multiple levels of abstraction [125]. Attention mechanism [126, 127] including self-attention [6] enables deep learning models to selectively pay attention to different parts of the input and provides interpretability. Deep learning and attention mechanism have improved the state-of-the-art in many research fields such as machine translation [126] and text classification [6].

Inspired by the aforementioned challenges and the rapid development of deep learning techniques, we propose an end-to-end deep framework for this multi-label classification task. We propose to train a unified classifier instead of a large number of independent classifiers, thus the efficiency is improved and the correlation between different MeSH terms can be learned simultaneously. More specifically, the new framework is a self-attentive deep neural network classifier. The proposed model contains three major components: a bidirectional Recurrent Neural Network (RNN), a number of self-attentive MeSH probes and a multi-view neural classifier. The proposed model is able to extract different aspects of biomedical knowledge from an input article. RNNs are naturally suitable for sequential text data, and by mapping the input text into the embedding space, RNNs can benefit from word embeddings that carry semantic regularities [23, 1, 53, 83]. By feeding RNN hidden states to self-attentive MeSH probes, each article can be converted into a fixed-dimension domain-specific feature matrix. The multi-view neural classifier is a unified multi-label classifier that considers the extracted feature

from the input text, the journal information as well as the correlation between different MeSH terms. The new framework is named MeSHProbeNet (in the 2018 BioASQ challenge, we used the name xgx for our system). To sum up, MeSHProbeNet has the following advantages:

- MeSHProbeNet is an end-to-end framework that does not rely on any other existing MeSH indexing systems or software tools.
- MeSHProbeNet is a unified multi-label classifier, thus very efficient in terms of training time consumption and disk usage for this large-scale MeSH indexing task.
- The bidirectional RNN of MeSHProbeNet is able to make use of the word embedding semantics and capture the context-dependent information via sequence modeling.
- The MeSH probes on top of the RNN allow us to extract different aspects of biomedical knowledge from the input article and represent it as a fixed-dimension feature matrix.
- The multi-view classifier considers both the extracted features and the journal information.
- MeSHProbeNet, as a unified multi-label classifier, simultaneously exploits the correlation between different MeSH terms as it is being trained.

The efficacy of MeSHProbeNet was demonstrated in Task A of the 2018 BioASQ challenge. We also provide an interpretability visualization of the MeSH probes to show how the proposed model selectively pays attention to different parts of the input article and how different aspects of biomedical knowledge are extracted by the MeSH probes. We also perform an ablation study of MeSHProbe to show the importance of MeSH probes.

7.2 Methodology

The overview of our proposed MeSHProbeNet model is shown in Figure 7.1. MeSHProbeNet is a self-attentive deep neural network, which is able to predict a set of MeSH terms for a biomedical article based on its textual content and journal information.

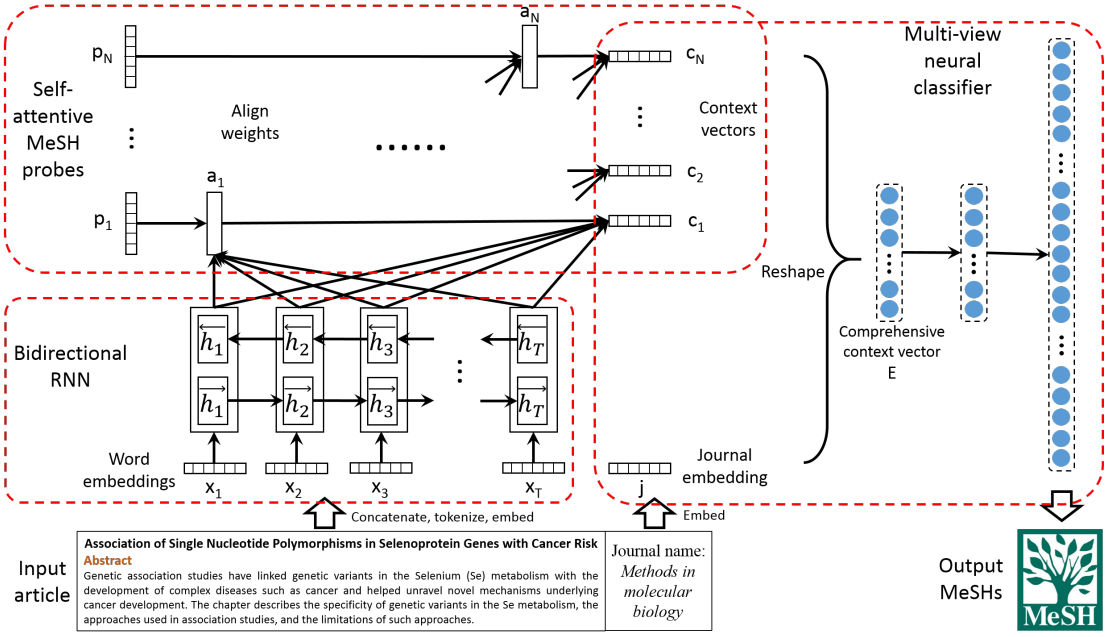


Figure 7.1: The framework of MeSHProbeNet.

The textual content of a biomedical article includes the title, abstract and body (in the challenge dataset, only the title and abstract are available). The journal information refers to the name of the journal it was published in.

Briefly speaking, MeSHProbeNet consists of three main components. The first component is a bidirectional RNN on the textual contents of biomedical articles. The second component is a set of self-attentive MeSH probes, which are responsible for extracting useful information from the RNN hidden states and converting articles of various lengths into fixed-dimension feature matrices. The third component is a multi-view neural classifier which combines the extracted textual information with the journal information, and generates a set of relevant MeSH terms.

We will introduce our model according to how to convert the textual contents into fixed-dimension matrices and how to recommend MeSH terms based on the combined information.

7.2.1 Bidirectional RNN

The bidirectional RNN reads the textual contents of a biomedical article, i.e., the concatenation of the title and the abstract, and generates a hidden state for each word in

the textual contents, as shown in the bottom left part of Figure 7.1. RNNs model texts in a sequential fashion and are able to capture the dependency between adjacent words. Long Short-Term Memory (LSTM) [128] and Gated Recurrent Unit (GRU) [129] have proven to be more effective in modeling long sequences than the vanilla RNN [130]. In MeSHProbeNet, we use a bidirectional GRU, as GRUs are simpler and perform on par with LSTMs. Suppose we have a sequential text which has T words as the input, i.e., the concatenation of the title and the abstract in our case. The first step is to represent the text as a sequence of T word embeddings:

$$\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_t, ..., \boldsymbol{x}_T\},$$

where $\boldsymbol{x}_t$ is a D_w dimensional real-valued vector, denoting the embedding for the t^{th} word in the input article. Thus a biomedical article can be represented as a T-by-D_w matrix, which is the concatenation of all the word embeddings in it. Then we feed article embedding matrix $\boldsymbol{X}$ to the bidirectional GRU:

$$\overrightarrow{\boldsymbol{h}_t} = \overrightarrow{GRU}(\boldsymbol{x}_t, \overrightarrow{\boldsymbol{h}_{t-1}}),$$
$$\overleftarrow{\boldsymbol{h}_t} = \overleftarrow{GRU}(\boldsymbol{x}_t, \overleftarrow{\boldsymbol{h}_{t+1}}),$$

where $\overrightarrow{\boldsymbol{h}_t}$ and $\overleftarrow{\boldsymbol{h}_t}$ are two U dimensional real-valued vectors, standing for the hidden states for the t^{th} word in normal direction and reverse direction, respectively. By concatenating $\overrightarrow{\boldsymbol{h}_t}$ and $\overleftarrow{\boldsymbol{h}_t}$, we derive a $2U$ dimensional hidden state $\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}_t}, \overleftarrow{\boldsymbol{h}_t}]$ which includes both the normal direction sequential information and the reverse direction sequential information at time stamp t. Hence, the hidden states of the input article can be represented as a T-by-$2U$ matrix:

$$\boldsymbol{H} = [\boldsymbol{h}_1; \boldsymbol{h}_2; ...; \boldsymbol{h}_t; ...; \boldsymbol{h}_T].$$

7.2.2 Self-attentive MeSH Probes

One simple way to obtain the summary of the input article is to use the last hidden states of the bidirectional GRU: $[\overrightarrow{\boldsymbol{h}_t}; \overleftarrow{\boldsymbol{h}_1}]$. Although GRUs have proven to be more effective at modeling long sequences than the vanilla RNNs, their performances on really long sequences are still limited, such as the entire title and abstract text in our case. Hence, we propose to use a self-attentive MeSH probe mechanism to extract comprehensive

aspects of biomedical information from the input article. Each MeSH probe carries one aspect of biomedical knowledge, and only pays attention to the RNN hidden states that contain related information. For instance, a MeSH probe that carries disease related knowledge is able to selectively extract the RNN hidden states that are related to disease. Specifically, one MeSH probe generates a weight vector for the RNN hidden states and multiply the RNN hidden states with the weight vector. Therefore, the resulting weighted RNN hidden state can be regarded as a summation of the input biomedical article with respect to the biomedical knowledge carried by the MeSH probe. With the help of the MeSH probe, biomedical articles of different lengths can be represented as a fixed-length vector containing related information. In fact, we can have multiple MeSH probes to cover multiple aspects of biomedical knowledge. Hence, given a certain number of MeSH probes, we can obtain a fixed-dimension output matrix that carries corresponding biomedical knowledge extracted from the input article.

More specifically, a MeSH probe is an inherent vector of MeSHProbeNet, which is associated with one specific aspect of biomedical knowledge. As with the GRU hidden state, the dimension of a MeSH probe is also $2U$. The goal of a MeSH probe is to extract related biomedical information from the input article and output a fixed-length vector. We achieve that by calculating a weighted combination of the T GRU hidden states. In particular, given MeSH probe $\boldsymbol{p}_n$, we first take all the GRU hidden states $\boldsymbol{H}$ as the input and then compute a normalized weight vector $\boldsymbol{\alpha}_n$:

$$\boldsymbol{\alpha}_n = \text{softmax}(\boldsymbol{p}_n \boldsymbol{H}^T).$$

Hence, $\boldsymbol{\alpha}_n$ is a 1-by-T vector where element α_{nt} indicates the weight for the t^{th} GRU hidden state and all the weights sum up to 1:

$$\alpha_{nt} = \frac{\exp(\boldsymbol{p}_n \cdot \boldsymbol{h}_t)}{\sum_{t'=1}^{T} \exp(\boldsymbol{p}_n \cdot \boldsymbol{h}_{t'})}.$$

By taking the inner product between MeSH probe $\boldsymbol{p}_n$ and each GRU hidden state, MeSH probe $\boldsymbol{p}_n$ assigns higher weights and pays more attention to the hidden states that carry related biomedical knowledge. Then we can use the weighted summation of the GRU hidden states according to the weights in $\boldsymbol{\alpha}_n$ to represent the input article, denoted as context vector $\boldsymbol{c}_n$:

$$\boldsymbol{c}_n = \boldsymbol{\alpha}_n \boldsymbol{H} = \sum_{t=1}^{T} \alpha_{nt} \cdot \boldsymbol{h}_t.$$

Context vector $\boldsymbol{c}_n$ is a $2U$ dimensional vector, which pays attention only to the parts of the input article related to MeSH probe $\boldsymbol{p}_n$. However, for a research article, one MeSH probe is normally insufficient as there are multiple aspects in it. For example, a research article about Alzheimer's disease is probably also related to aging and treatments. Therefore, to get a more comprehensive representation of the input article, we need multiple MeSH probes to pay attention to different aspects of the article, for instance, one probe for disease, one probe for treatments, another probe for anatomy, and so on. As illustrated by the top left part of Figure 7.1, if we want to examine N different aspects of the input article, N MeSH probes are required:

$$\boldsymbol{P} = [\boldsymbol{p}_1; \boldsymbol{p}_2; ...; \boldsymbol{p}_N],$$

where $\boldsymbol{P}$ is a N-by-$2U$ matrix composed of N different MeSH probes. Accordingly, we can obtain a N-by-T weight matrix A, where each row $\boldsymbol{\alpha}_n$ denotes the weight vector with respect to each MeSH probe $\boldsymbol{p}_n$:

$$\boldsymbol{A} = \text{softmax}(\boldsymbol{P}\boldsymbol{H}^T),$$

where the softmax function is performed along the second dimension of the input. Hence, with the help of multiple MeSH probes, we are able to extract different aspects of biomedical knowledge from the input article, and represent it with a N-by-$2U$ context matrix $\boldsymbol{C}$:

$$\boldsymbol{C} = \boldsymbol{A}\boldsymbol{H}.$$

7.2.3 Multi-view Neural Classifier

With the help of the bidirectional RNN and the MeSH probes, now we are able to convert a biomedical article of arbitrary length to a fixed-dimension context matrix, where each row represents one particular aspect of the input article. In fact, for each input article, we also have its journal information in addition to the textual content. This journal information is quite useful, as biomedical journals typically have a definite research topic and focus on a specific research domain. Therefore, it is natural to expect that research

papers published in the same journal tend to be annotated with MeSH terms related to the journal's research focus. To take the journal information into consideration, our multi-view neural classifier has a journal embedding module, where each journal name can be converted to a unique vector of length D_j. Thus, by reshaping the extracted context matrix $\boldsymbol{C}$ to a vector and concatenating it with the journal embedding, we are able to obtain a context vector of length $N * 2U + D_j$ that carries all the available information of the input article: the title, the abstract and the journal information. We denote this comprehensive context vector by $\boldsymbol{E}$.

Our task is to annotate a biomedical article with suitable MeSH terms. Hence, having extracted comprehensive context vector $\boldsymbol{E}$ from the input article, what we need to do next is to learn a function f that maps context vector $\boldsymbol{E}$ to V conditional probability distributions, where V is the size of the MeSH vocabulary. The output of f is a vector whose i^{th} element estimates the probability that the i^{th} MeSH term should be assigned to the current article:

$$P(m_i = 1|\boldsymbol{E}) = f(i, \boldsymbol{E}),$$

where m_i denotes the i^{th} MeSH term in the MeSH vocabulary. Function f could be implemented by a feed forward neural network. We employ a three layer neural network, whose first layer is the input context vector $\boldsymbol{E}$, second layer is the hidden layer with ReLU activation and third layer is the output layer. More precisely, the multi-layer neural network calculates the following function, with a sigmoid output layer to guarantee each output neuron being a probability in the range of $[0, 1]$:

$$f(\boldsymbol{E}) = \sigma(\boldsymbol{W}_2\text{ReLU}(\boldsymbol{W}_1\boldsymbol{E} + \boldsymbol{b}_1) + \boldsymbol{b}_2), \tag{7.1}$$

where $\sigma(\cdot)$ is the element-wise sigmoid function, $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are the weight matrices for each layer, and $\boldsymbol{b}_1$, $\boldsymbol{b}_2$ are the biases. During training, each biomedical article comes with several manually annotated MeSH terms. So it can be regarded as a multi-label classification task, where the ground truth label is a V-length binary vector whose i^{th} element is set to 1 if the i^{th} MeSH term is assigned to the current article and set to 0 otherwise. We represent this ground truth vector by g. Therefore, given a biomedical article k, the objective is to minimize the following binary cross entropy loss:

$$L_k = -\sum_{i=1}^{V}(g[i] \cdot \log(f(i, \boldsymbol{E})) + (1 - g[i]) \cdot \log(1 - f(i, \boldsymbol{E}))).$$

Let K be the total number of articles in the training dataset, then the overall training objective is:

$$L = \sum_{k=1}^{K} L_k. \tag{7.2}$$

Note that unlike most previous works that train a binary classifier for each MeSH term separately, we train a unified multi-label classifier that considers all the MeSH terms simultaneously. The advantages of training a unified multi-label classifier are manifold. First, the efficiency for both training and predicting can be drastically improved by learning a unified classifier as there are more than 28,000 distinct MeSH terms. Second, by learning a unified classifier, the semantics of the word embeddings and journal embeddings can be shared by all MeSH terms. Third, the correlation between different MeSH terms is automatically exploited and carried by neural network weights $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$. If one MeSH term frequently co-occurs with other MeSH terms, for example, "Alzheimer disease" is often accompanied by "aged, 80 and over", this co-occurrence will influence the corresponding neurons in $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ simultaneously, and thus the correlation and dependency relationship can be captured.

Infrequent MeSH terms also benefit from this unified architecture. Hundreds of infrequent terms appear in less than 10 articles. Therefore, if an independent classifier is trained for each infrequent term, the classifier inevitably suffers from the lack of training data and would encounter tons of out-of-vocabulary words during prediction. By sharing parameters across all MeSH terms, such as word embeddings and weight matrices, the unified classifier is able to tackle the problem of lacking training data and the out-of-vocabulary problem for infrequent MeSH terms. In addition, infrequent terms can further take advantages of the correlation information in the unified classifier, especially if an infrequent term always co-occurs with some specific frequent terms.

The free parameters of the whole model are the word embeddings, the GRU weight matrix, the GRU bias, the MeSH probes, the journal embeddings, the fully connected neural network weight matrices and biases. Let θ denote the overall free parameter set. Then training can be achieved by looking for θ that minimizes the training corpus binary cross entropy loss in Eq. 7.2 via stochastic gradient descent. Stochastic gradient descent iteratively updates the free parameters after feeding the k^{th} article of the training corpus:

$$\theta \leftarrow \theta - \eta \frac{\partial L_k}{\partial \theta},$$

where η is the learning rate.

In the prediction phase, there are two approaches to determine the final MeSH terms based on the output of function f in Eq. 7.1. One approach is to find the optimal thresholds for each MeSH term on a held-out validation set. The other approach is to learn another neural network to predict the number of related MeSH terms given a biomedical article. In practice, we adopt the first approach in the prediction phase, as it is more efficient and intuitive.

7.3 Experiments

We carry out experiments on the large-scale MeSH indexing task to demonstrate the efficacy of our MeSHProbeNet model. To illustrate how MeSHProbeNet extracts different aspects of biomedical knowledge from the input articles, we visualize MeSH probes and their attentions on different parts of the input sequence. To investigate the quality of the MeSH terms recommended by MeSHProbeNet, we participated in the 2018 BioASQ challenge and compare its performance with several state-of-the-art MeSH indexing systems, including MTI and DeepMeSH. Our system won the first place in the third batch of the challenge.[5]

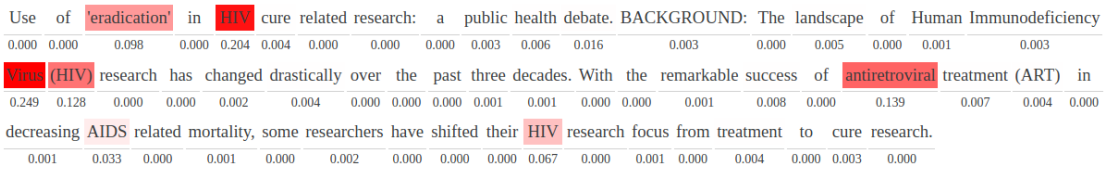
7.3.1 Dataset and Experimental Settings

The training dataset is downloaded from the challenge webpage[6] . It contains 13,486,072 biomedical articles which are annotated with relevant MeSH terms by the PubMed human experts. On average, 12.69 MeSH terms are assigned to each article. In total, 28,340 distinct MeSH terms are covered by the training dataset. For each article in the training dataset, we have the unique identifier of the article (PMID), the title of the article, the abstract of the article, the year the article was published, the journal the article was published in and a set of MeSH terms assigned to the article.

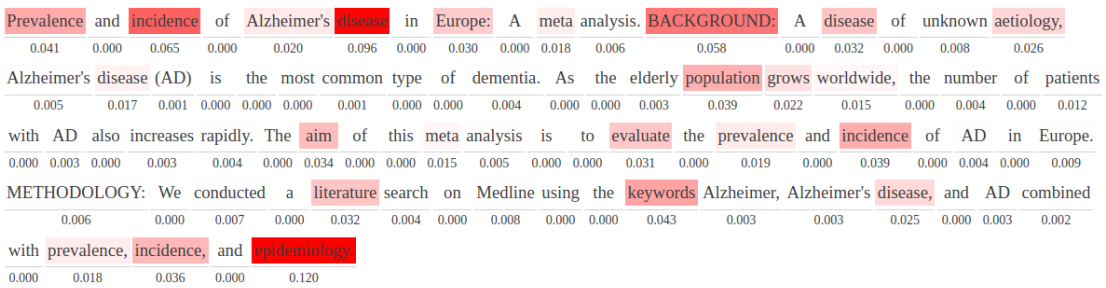
In the preprocessing step, all non-alphanumeric characters, stop words and words

[5] The source code of MeSHProbeNet is available at `https://github.com/XunGuangxu/MeSHProbeNet`.

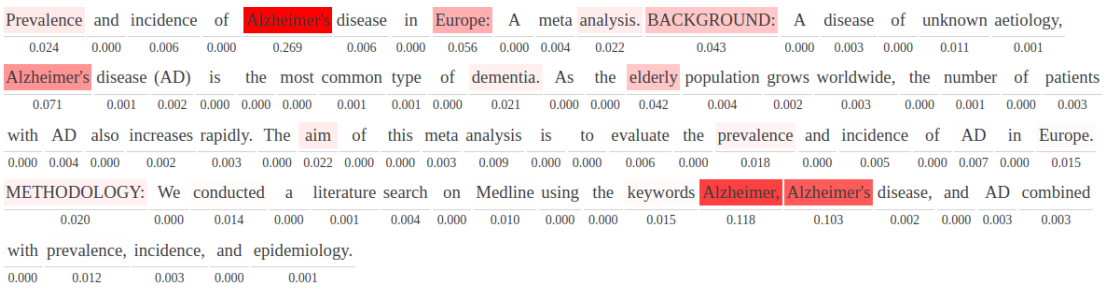
[6] http://participants-area.bioasq.org/general_information/Task6a/



(a) MeSH probe No.2 extracts disease related information from article 29439706.



(b) MeSH probe No.2 extracts disease related information from article 27130306.



(c) MeSH probe No.11 extracts Alzheimer's related information from article 27130306.

Figure 7.2: MeSH probe interpretability visualization.

with a total frequency lower than 10 are removed, and all words are converted to lowercase. The dimensionalities of word embeddings and journal embeddings are set to 250 and 100, respectively. The number of GRU layers is set to 2. The size of the GRU hidden unit is set to 200 per direction, thus 400 for a bidirectional unit. The dimensionality of MeSH probes is also set to 400 accordingly. The number of different MeSH probes that the model contains is 25. The multi-view neural classifier has a hidden layer of 10000 units. We deploy 0.5 dropout, 5e-10 L2 regularization and snapshot ensemble [131] to prevent over-fitting. The learning rate for stochastic gradient descent is set to 0.0005 and we also clip the gradients whose values are larger than 5.

7.3.2 MeSH Probe Visualization

Interpretability is one of the advantages of MeSHProbeNet. For the users of automatic MeSH indexing models, a good model should not only be accurate, but also be able to tell them which parts of the input support the recommended MeSH terms. For instance, the human indexers can achieve higher annotation efficiency with the help of interpretable MeSH indexing models, as this interpretability of automatic MeSH indexing models can provide them with evidence for adding or deleting a recommended MeSH term.

The interpretability of MeSHProbeNet can be achieved through examining the attention weight matrix $\boldsymbol{A}$. Each row $\boldsymbol{a}_n$ in attention weight matrix $\boldsymbol{A}$ represents the weight vector with respect to MeSH probe $\boldsymbol{p}_n$. Each element in weight vector $\boldsymbol{a}_n$ corresponds to how much attention MeSH probe $\boldsymbol{p}_n$ pays to each GRU hidden state and each word. Thus we can visualize the attention by drawing a heat map of the weight vector.

It is worth mentioning that another advantage of MeSHProbeNet is its unsupervised nature: the MeSH probes are learned in a completely unsupervised fashion. The training objective function drives the MeSH probes to extract comprehensive aspects of biomedical knowledge with each probe focusing on one specific aspect. In other words, we do not need any prior knowledge, external knowledge or human guidance for the MeSH probes. The probes are automatically learned and are able to capture biomedical semantics during training and provide interpretability.

We select two articles from the last test set of the 2018 BioASQ challenge, whose PMIDs are "29439706" and "27130306", to visualize MeSH probes and show the interpretability in Figure 7.2. For article 29439706, the ground truth MeSH terms assigned by human curators are "biomedical research", "disease eradication", "HIV infections", "humans", "public health", and "terminology as topic"; and the MeSH terms assigned by MeSHProbeNet are "humans", "HIV infections", "research", "disease eradication", "public health", "AIDS vaccines", "HIV-1" and "anti-HIV agents". For article 27130306, the ground truth MeSH terms assigned by human curators are "Alzheimer disease", "Bayes theorem", "Europe", "humans", "incidence" and "prevalence"; and the MeSH terms assigned by MeSHProbeNet are "prevalence", "humans", "male", "female", "Alzheimer disease", "aged", "aged, 80 and over", "incidence", "Bayes theorem" and "Europe".

We first demonstrate how MeSH probe No.2 extracts disease related information

from different articles in Figures 7.2a and 7.2b. The values below each word denote the normalized weights. We can see that MeSH probe No.2 pays more attention to words like "HIV", "virus" and "disease". Some words such as "incidence" and "background" also have high attention weights. This is because of the sequential nature of RNNs and the system recognizes those words as related words in the context of "disease". Then in Figures 7.2b and 7.2c, we demonstrate how two different MeSH probes extract two different aspects of biomedical knowledge from the same article. As we just mentioned, in Figure 7.2b MeSH probe No.2 extracts disease related information. While in Figure 7.2c, MeSH probe No.11 extracts Alzheimer's related information. One can observe that in this article, MeSH probe No.2 is sensitive to words like "disease" and "epidemiology", while MeSH probe No.11 is sensitive to words like "Alzheimer's" and "elderly".

7.3.3 Evaluation Metrics

In order to evaluate MeSH indexing performance, two sets of measures are used, one flat and one hierarchical.

The flat measures consist of accuracy and two sets of F-measure based metrics: Macro F-Measure (MaF) and Micro F-Measure (MiF). Accuracy represents the fraction of correct predictions. MaF, Macro Precision (MaP) and Macro Recall (MaR) give equal weight to each MeSH class. Frequent MeSH terms and infrequent MeSH terms are equally important. Thus MaP and MaR are calculated as the average precision and recall over all the MeSH classes. MiF, Micro Precision (MiP) and Micro Recall (MiR) aggregate the contributions of all MeSH classes to compute the average metric. Frequent MeSH terms therefore have higher weights than infrequent MeSH terms. We can see that different F-Measures have different focus, for example, MiF focuses more on the frequent MeSH terms, while MaF treats all MeSH terms equally regardless of their frequencies. Since the BioASQ challenge evaluates the systems based on MiF, we will also take MiF as our major measure.

The MeSH vocabulary is organized in a hierarchical structure. Thus hierarchical measures are also used to evaluate the performance, including Hierarchical Precision (HiP), Hierarchical Recall (HiR), Hierarchical F-Measure (HiF), Lowest Common Ancestor Precision (LCA-P), Lowest Common Ancestor Recall (LCA-R) and Lowest Common Ancestor F-measure (LCA-F) [132].

Table 7.1: Comparison results based on the flat measures.

Models	MiP	MiR	MiF	MaP	MaR	MaF	Acc
Access Inn MAIstro	0.2351	0.3423	0.2788	0.3942	0.4641	0.3905	0.1669
MeSHmallow	0.3798	0.2707	0.3161	0.1333	0.0049	0.0037	0.1915
UMass Amherst T2T	0.5239	0.4759	0.4988	0.4179	0.2526	0.2481	0.3392
iria	0.4654	0.5792	0.5161	0.4271	0.4658	0.4147	0.3525
MTIFL	0.6730	0.5977	0.6332	0.6377	0.5622	0.5408	0.4759
MTI	0.6475	0.6473	0.6474	0.6086	**0.6084**	0.5667	0.4911
AttentionMeSH	0.6833	0.6447	0.6635	0.6178	0.4943	0.4827	0.4982
DeepMeSH	0.6761	0.6517	0.6637	0.6352	0.5455	0.5281	0.5020
MeSHProbeNet	**0.7172**	**0.6611**	**0.6880**	**0.6782**	0.5804	**0.5671**	**0.5310**

7.3.4 Experimental Results

We show the comparison result of the proposed MeSHProbeNet model with the default MTI, MTI First Line indexing (MTIFL) [115], DeepMeSH [122], AttentionMeSH [133], iria [134], UMass Amherst T2T, MeSHmallow and Access Inn MAIstro on the last test set of the 2018 BioASQ challenge. There are 15 test sets in total (one test set per week during the challenge) and the complete results are available on the challenge webpage[7] (please note that we used the name xgx in the challenge). The main difference between MTI and MTIFL is that MTIFL has higher precision by limiting its recommendation to a smaller number of MeSH terms, while MTI balances precision and recall, and achieves better F-measure.

The comparison results based on the flat measures of each model are reported in Table 7.1. The challenge allows each model to make at most 5 attempts to try out different settings, such as different initializations and parameters, as a significance test. Our model consistently achieves the best performance. To conserve space, we only show the best performance score of each model here. Interested readers may refer to the complete result on the challenge website. The best scores are highlighted in boldface in Table 7.1. Compared with MTI, MTIFL has higher precision but lower recall, resulting in low F-measures. DeepMeSH outperforms MTI in terms of MiF score but its MaF score is not as good as MTI's, which means DeepMeSH pays more attention to the frequent MeSH terms such as "humans", "animals", "male" and "female". We can

[7] http://participants-area.bioasq.org/results/6a/

Table 7.2: Comparison results based on the hierarchical measures.

Models	LCA-P	LCA-R	LCA-F	HiP	HiR	HiF
Access Inn MAIstro	0.2722	0.3615	0.2964	0.4696	0.5921	0.5043
MeSHmallow	0.4000	0.2369	0.2871	0.5633	0.3287	0.3967
UMass Amherst T2T	0.4818	0.4087	0.4276	0.7094	0.5961	0.6262
iria	0.4251	0.4902	0.4443	0.6174	0.7290	0.6536
MTIFL	0.5662	0.5014	0.5172	0.7964	0.7186	0.7373
MTI	0.5510	0.5415	0.5325	0.7703	0.7647	0.7514
AttentionMesh	0.5627	0.5235	0.5290	0.7902	0.7396	0.7472
DeepMeSH	0.5643	0.5364	0.5366	0.7899	0.7555	0.7560
MeSHProbeNet	**0.5901**	**0.5561**	**0.5596**	**0.8123**	**0.7714**	**0.7760**

observe that MeSHProbeNet achieves the highest scores in all F-Measures and accuracy. Since MeSHProbeNet is able to capture the correlation between different MeSH terms and MeSH indexing for infrequent terms can benefit from this correlation information, MeSHProbeNet gains both the best MiF and the best MaF scores.

The comparison results based on the hierarchical measures of each model are reported in Table 7.2. As with the flat measure result, we also only show the best performance score of each MeSH indexing model. The best scores are highlighted in boldface. The hierarchical measures are calculated based on the hierarchical structure of the MeSH vocabulary, thus the semantic distance between MeSH terms is under consideration. As with their performances on the flat measures, MTI achieves higher F-Measures than MTIFL and DeepMeSH outperforms both of them. We can see that MeSHProbeNet obtains the highest scores in all measures.

7.3.5 Ablation Studies on MeSH Probes

We have demonstrated strong empirical results of MeSHProbeNet. Now we perform ablation experiments in order to better understand the importance of the self-attentive MeSH probes. Since the 2018 BioASQ challenge is closed and the challenge test sets are currently not available, we split the dataset into training and test sets. The test set contains 7,000 articles and is used to evaluate the ablation models. All the models are trained on this new training set.

Table 7.3: Ablation results based on the flat measures.

Models	MiP	MiR	MiF	MaP	MaR	MaF	Acc
bi-GRU	0.6691	0.6243	0.6459	0.6228	0.4997	0.4937	0.4801
MeSHProbeNet-5	0.6978	0.6511	0.6736	0.6500	0.5609	0.5485	0.5124
MeSHProbeNet-15	0.7072	0.6617	0.6837	0.6675	0.5792	0.5670	0.5243
MeSHProbeNet-25	**0.7094**	**0.6643**	**0.6861**	**0.6732**	**0.5846**	**0.5706**	**0.5276**

Table 7.4: Ablation results based on the hierarchical measures.

Models	LCA-P	LCA-R	LCA-F	HiP	HiR	HiF
bi-GRU	0.5610	0.5111	0.5200	0.7935	0.7250	0.7380
MeSHProbeNet-5	0.5767	0.5341	0.5400	0.8064	0.7487	0.7583
MeSHProbeNet-15	0.5844	0.5434	0.5487	0.8118	0.7575	0.7660
MeSHProbeNet-25	**0.5859**	**0.5465**	**0.5511**	**0.8129**	**0.7606**	**0.7681**

To show the effect of MeSH probes, we include in the comparison bi-GRU, which directly feeds the GRU output to the multi-view neural classifier and uses no MeSH probes. To show the influence of different numbers of MeSH probes, MeSHProbeNet models with 5, 15 and 25 MeSH probes are also included in the comparison, among which the MeSHProbeNet-25 model has the same amount of MeSH probes as the model we used in the challenge. All the other parameters, such as the embedding dimension and the number of GRU layers, are the same as the challenge model for each model.

The ablation results based on the flat measures and the hierarchical measures are reported in Table 7.3 and Table 7.4, respectively. The best scores are highlighted in boldface. One can observe that the self-attentive MeSH probe mechanism significantly improves the performance. Adding more MeSH probes is also helpful, although the improvement per added MeSH probe becomes less and less significant as the number of MeSH probes gets higher. Adding more probes will also increase the computation cost and disk usage of the model.

7.3.6 Computational Efficiency

The training of MeSHProbeNet on the entire MEDLINE database can be finished within 24 hours with one NVIDIA TITAN Xp GPU. Given a new test set of 10,000 articles,

the prediction takes less than 1 minute. Compared with other state-of-the-art MeSH indexing models, for example, DeepMeSH needs 1 week to train on 1 million articles and AttentionMeSH needs 4 days to train on 3 million articles with 2 GPUs, this improved training efficiency of MeSHProbeNet allows us to exploit the entire database of more than 13 million annotated articles. Moreover, since MeSHProbeNet does not need to store any article information to perform KNN to find similar articles in the database, nor does it need to train separate classifiers for more than 28,000 MeSH terms, the disk usage of MeSHProbeNet is just about 1 GB.

7.4 Conclusions

We present an end-to-end MeSH indexing model MeSHProbeNet. MeSHProbeNet participated in the 2018 BioASQ challenge and achieved the best performance in the latest batch. MeSHProbeNet is a self-attentive deep neural network classifier, which is able to extract different aspects of biomedical knowledge from an input article with different MeSH probes, and generate MeSH recommendations based on the extracted features, journal information and MeSH correlations. The experimental results demonstrate the effectiveness of MeSHProbeNet on both frequent and infrequent MeSH terms.

Chapter 8

Customizable Domain Attentions for Accurate Feature Extraction

8.1 Introduction

Centuries of research and experimentation have led to a generation of large-scale literature in the biomedical domain. For example, MEDLINE/PubMed[1] is a biomedical database maintained by U.S. National Library of Medicine (NLM). It currently contains more than 24 million biomedical journal citations from more than 5,200 worldwide journals. To improve large-scale biomedical text retrieval and facilitate biomedical research [135, 111, 112, 113], MEDLINE articles are indexed with the Medical Subject Headings (MeSH) vocabulary, a vocabulary curated by the NLM experts.

Currently, MeSH indexing is performed manually by human experts. They examine the full body of each biomedical article and annotate it with suitable MeSH terms. This manual MeSH indexing has high accuracy but inevitably comes at a high price. It is estimated that on average annotating one article in MEDLINE costs around $9.4 [114] and in 2017 more than 813,500 citations were added to MEDLINE. In addition to the monetary cost, it is also time consuming for the human experts to annotate a newly published article. Thus, it would be very helpful to develop an automatic system capable of annotating biomedical articles with MeSH terms or assisting the human indexers in doing so. From the viewpoint of machine learning, automatic MeSH indexing is a large-scale multi-label classification problem with extremely imbalanced classes. The MeSH

[1] https://www.nlm.nih.gov/bsd/medline.html

vocabulary is large in order to cover all possible aspects of domain knowledge, and the frequency of different terms vary drastically.

Many automatic MeSH indexing systems have been proposed for this imbalanced large-scale multi-label classification task, such as Medical Text Indexer system (MTI) [115, 114], MeSHLabeler [121], DeepMeSH [122], AttentionMeSH [133] and MeSHProbeNet [136]. MeSHLabeler, DeepMeSH and AttentionMeSH train an independent binary classifier for each MeSH term, making the whole system large and inefficient. This further results in the inability of the system to make use of all the annotated articles, for instance, MEDLINE contains more than 12 million annotated articles, while DeepMeSH is only trained on 1 million articles. In addition, these systems rely on the results of other existing systems, for examples, K-Nearest-Neighbors (KNN) is utilized to find similar articles from literature and generate a candidate list of MeSH recommendations. This further exacerbates the time and space consumption for the system, as KNN has to go through the entire biomedical literature for each training and test article. Moreover, this independent training process also heavily limits the power of the system to exploit the correlation between different MeSH terms. Although, MeSHProbeNet is an end-to-end model and achieves high training efficiency using self-attentive MeSH probes, it can only extract general information from input articles as those probes are universal, leaving the informative topic specific information underexploited.

The aforementioned challenges and limitations inspire us to develop a new end-to-end model for this imbalanced large-scale multi-label classification task. The proposed model extends our previous MeSHProbeNet model with personalizable/cusomizable MeSH probes, and is named MeSHProbeNet-P, where P stands for personalization. As with MeSHProbeNet, it is also a unified self-contained classifier, but the personalizable MeSH probes enable it to extract both general and topic-related biomedical knowledge from biomedical articles. Specifically, MeSHProbeNet-P utilizes deep learning and attention mechanism to label biomedical articles. In MeSHProbeNet-P, different aspects of biomedical information are extracted from the input article with different MeSH probes. Each MeSH probe only pays attention to certain aspects of biomedical information. In general, we have three types of MeSH probes in MeSHProbeNet-P: one type of general MeSH probes and two types of topic-specific MeSH probes. The general MeSH probes are responsible for extracting general biomedical information, such as diseases and treatment, and are used in the same way for all input articles, whereas the topic-specific MeSH probes are responsible for extracting biomedical information related to

the topic of the current input article, and are tailor-made for each input article, for example, the diabetes topic-specific MeSH probe is suitable for articles about diabetes. MeSHProbeNet-P automatically customizes MeSH probes for different input articles. In practice, we use a combination of general MeSH probes and topic-specific MeSH probes in order to extract both general and specific biomedical information from the input articles. As a unified classifier, MeSHProbeNet-P is able to annotate all MeSH terms at once and exploit the correlation between different terms. In addition, MeSHProbeNet-P has high efficiency, allowing it to be trained on all of the annotated biomedical articles.

We demonstrate the effectiveness of our proposed model on the BioASQ[2] MeSH indexing challenge benchmark, in comparison with the state-of-the-art models. To sum up, the main advantages of MeSHProbeNet-P are as follows:

- MeSHProbeNet-P achieves the best performance on the challenge test set.
- MeSHProbeNet-P is an end-to-end model that does not need any intermediary results from other algorithms and trains efficiently on GPUs.
- MeSHProbeNet-P contains general MeSH probes and topic-specific MeSH probes, which are able to automatically extract general and topic-specific biomedical information from the input articles.
- MeSHProbeNet-P can automatically customize the MeSH probes to make the model best fit each different input article.
- MeSHProbeNet-P is a unified multi-label classifier, which facilitates the learning of term correlations and improves model efficiency.

8.2 Related Work

MeSH indexing greatly facilitates information retrieval and research in the domain of biomedicine. To this end, NLM has developed MTI [115, 114], a software tool to annotate biomedical articles with MeSH terms, and has been aiding the NLM human MeSH indexers since 2002. MTI takes the title and abstract of a biomedical article, and combines MetaMap Indexing (MMI) and PubMed-Related Citations (PRC) to make MeSH recommendations. MMI recommends MeSH terms based on the UMLS concepts

[2] http://bioasq.org/

extracted by MetaMap. PRC recommends MeSH terms by integrating the MeSH annotations of similar articles in MEDLINE found by PubMed-Related Articles (PRA) [118].

To encourage worldwide researchers to design new effective MeSH indexing models and advance this research domain, the BioASQ challenge is held every year since 2013 [119], presenting a real-world large-scale MeSH indexing benchmark. In the MeSH indexing task of BioASQ, participating models are required to annotate new MEDLINE articles with relevant MeSH terms, before human indexers annotate them manually. As the manual annotations for the new articles become available, they are used as ground truth to evaluate the performance of participating models. Many efficacious systems have emerged from the challenge, for example, MetaLabeler [120], MeSHLabeler [121], DeepMeSH [122] and MeSHProbeNet [136] are the challenge winners in recent years.

MetaLabeler views the challenge as a classification problem and trains an independent binary classifier for each MeSH term. MeSHLabeler integrates MetaLabeler with other evidence, such as similar publications found by KNN and term frequencies, to generate a ranked list of candidate MeSH terms. DeepMeSH further introduces word embeddings [23, 1] and the deep semantics carried by word embeddings to MeSHLabeler. Inspired by the rapid development of deep learning and attention techniques [125, 137, 53, 83, 6, 138, 126, 127, 123], MeSHProbeNet is proposed to extract biomedical information from input articles with self-attentive MeSH probes.

However, these models either suffer from low model efficiency or can only extract general biomedical information. In this chapter, we design an efficient end-to-end deep MeSH indexing model with personalizable MeSH probes. The MeSH probes can be customized for different biomedical articles based on the topics of the articles. Therefore, both general and topic-related biomedical knowledge can be extracted. Subsequently, a multi-view classifier is utilized to make use of both the textual content and the journal information of a given article. Multi-view classifiers have proven to be effective at improving model performance and generality [139, 140, 141]. In addition, MeSHProbeNet-P is also able to exploit the correlation between biomedical labels at the same time as the classifier is being trained.

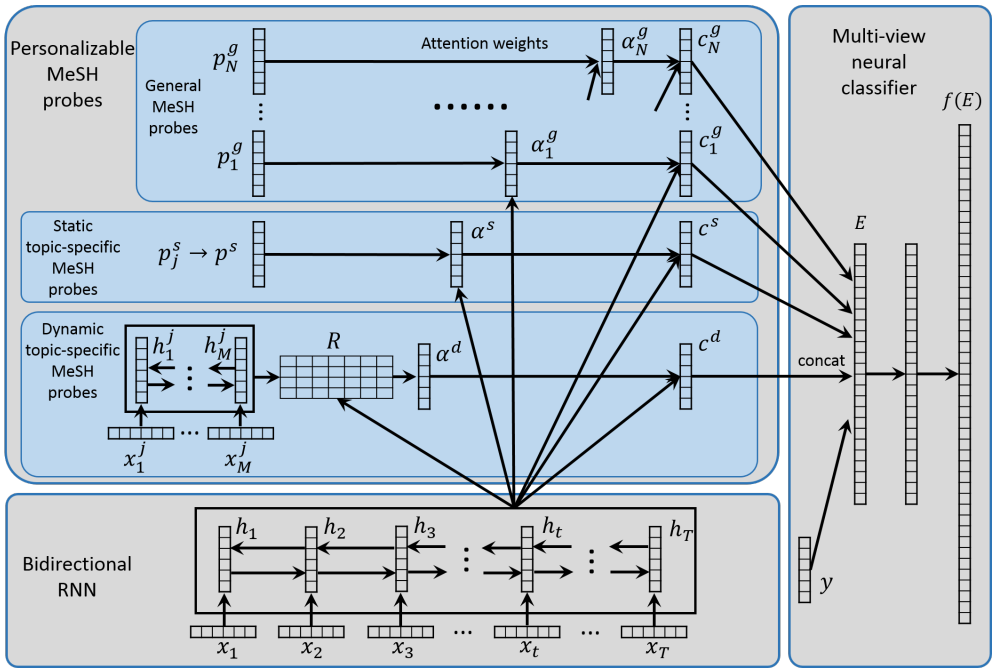


Figure 8.1: The framework of MeSHProbeNet-P.

8.3 Methodology

Figure 8.1 shows the overview of our proposed model MeSHProbeNet-P. MeSHProbeNet-P is a unified deep classifier with personalizable MeSH probes. As with the format of most biomedical articles, the input of MeSHProbeNet-P consists of the textual content of the article and the name of the journal or conference it is published in. The output is a set of MeSH recommendations for this article.

The textual content of a biomedical article refers to its title, abstract and main body, and carries the major information of the article, whereas the journal name of a biomedical article indicates the topic information of the article, as every journal has a definite focus, for example, *Brain Research* is an international journal devoted to fundamental research in the brain sciences. This topic information of each biomedical article helps MeSHProbeNet-P personalize the MeSH probes accordingly.

Briefly speaking, the proposed MeSHProbeNet-P model consists of three major components: a bidirectional Recurrent Neural Network (RNN), a set of personalizable MeSH probes and a multi-view neural classifier. The bidirectional RNN takes the article textual content as its input. The personalizable MeSH probes extracts useful information

from the RNN hidden states.A biomedical article of arbitrary length can be converted to a set of fixed-length feature vectors by the RNN and MeSH probes. The multi-view classifier takes the extracted feature vectors and recommends a set of MeSH terms.

8.3.1 Bidirectional RNN

The bidirectional RNN reads the textual content of a biomedical article and generates a hidden state for each word in the textual contents, as shown in the bottom left part of Figure 8.1. RNNs take words in textual content in sequential order and are able to capture the dependency between adjacent words. Specifically, MeSHProbeNet-P uses a bidirectional GRU (Gated Recurrent Unit) [129] instead of a vanilla RNN, as GRU has proven to be more effective in modeling long sequences than vanilla RNN [130]. Thus, given the textual content of an article containing T words, MeSHProbeNet-P first represents the text as a sequence of T word embeddings through a lookup table:

$$\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_t, ..., \boldsymbol{x}_T\},$$

where $\boldsymbol{x}_t$ is a D_w dimensional real-valued vector, denoting the embedding for the t^{th} word in the input article. Thus an article can be represented as a T-by-D_w matrix, which is the concatenation of all the word embeddings in it. Then we feed article embedding matrix $\boldsymbol{X}$ to the bidirectional GRU:

$$\overrightarrow{\boldsymbol{h}_t} = \overrightarrow{GRU}(\boldsymbol{x}_t, \overrightarrow{\boldsymbol{h}_{t-1}}),$$
$$\overleftarrow{\boldsymbol{h}_t} = \overleftarrow{GRU}(\boldsymbol{x}_t, \overleftarrow{\boldsymbol{h}_{t+1}}),$$

where $\overrightarrow{\boldsymbol{h}_t}$ and $\overleftarrow{\boldsymbol{h}_t}$ are two U dimensional real-valued vectors, standing for the hidden states for the t^{th} word in normal direction and reverse direction, respectively. By concatenating $\overrightarrow{\boldsymbol{h}_t}$ and $\overleftarrow{\boldsymbol{h}_t}$, we derive a $2U$ dimensional hidden state $\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}_t}, \overleftarrow{\boldsymbol{h}_t}]$ which includes both the normal direction sequential information and the reverse direction sequential information at time stamp t. Hence, the hidden states of the input article can be represented as a T-by-$2U$ matrix:

$$\boldsymbol{H} = [\boldsymbol{h}_1; \boldsymbol{h}_2; ...; \boldsymbol{h}_t; ...; \boldsymbol{h}_T].$$

8.3.2 Personalizable MeSH Probes

One simple way to obtain the summary of the input article is to use the last hidden states of the bidirectional GRU: $[\overrightarrow{\boldsymbol{h}_T}; \overleftarrow{\boldsymbol{h}_1}]$. Although GRUs have proven to be more effective in modeling long sequences than the vanilla RNNs, their performances on really long sequences are still limited, such as the entire textual content of an article in our case. Hence, we propose to use personalizable MeSH probes to extract useful biomedical information from the input article. Each MeSH probe carries certain aspects of biomedical knowledge, and only pays attention to the RNN hidden states that contain related information.Technically, a MeSH probe generates a weight vector whose elements sum up to 1 for the RNN hidden states. The higher the weight value, the more related the RNN hidden state is to the MeSH probe, and the more attention is paid to the hidden state. Then, based on the weight vector, a MeSH probe can output a weighted RNN hidden state which can be viewed as a summation of the input biomedical article with respect to the biomedical knowledge carried by the MeSH probe. At the same time, this MeSH probe allows MeSHProbeNet-P to represent an input article of arbitrary length with one fixed-length vector. In fact, we can have multiple MeSH probes to cover multiple aspects of biomedical knowledge. Hence, given a set of personalizable MeSH probes, a set of fixed-length context vectors that carry corresponding biomedical information of the input article can be obtained.

MeSHProbeNet-P has three types of MeSH probes: one type of general MeSH probes which are sensitive to general biomedical knowledge and two types of topic-specific MeSH probes which are sensitive to the topic of the current input article. MeSHProbeNet-P is called personalizable because it is able to automatically generate the most suitable set of MeSH probes for each input article based on its topic. The different types of MeSH probes are presented in details below.

General MeSH Probes. General MeSH probes carry general biomedical information and work in the same self-attentive way for all articles. They are inherent vectors of the model parameters, each carrying one general aspect of biomedical knowledge. As with the GRU hidden state, the dimension of a general MeSH probe is also $2U$. Given general MeSH probe $\boldsymbol{p}_n^g$, where superscript g stands for general and subscript n is the MeSH probe index, we first take all the GRU hidden states $\boldsymbol{H}$ and then compute a normalized weight vector $\boldsymbol{\alpha}_n^g$:

$$\boldsymbol{\alpha}_n^g = \text{softmax}(\boldsymbol{p}_n^g \boldsymbol{H}^T),$$

where $\boldsymbol{\alpha}_n^g$ is a 1-by-T vector where element α_{nt}^g indicates the weight for the t^{th} GRU hidden state and all the weights sum up to 1:

$$\alpha_{nt}^g = \frac{\exp(\boldsymbol{p}_n^g \cdot \boldsymbol{h}_t)}{\sum_{t'=1}^{T} \exp(\boldsymbol{p}_n^g \cdot \boldsymbol{h}_{t'})}.$$

By taking the inner product between MeSH probe $\boldsymbol{p}_n^g$ and each GRU hidden state, MeSH probe $\boldsymbol{p}_n^g$ assigns higher weights and pays more attention to the hidden states that carry related biomedical knowledge. Then we can use the weighted summation of the GRU hidden states according to the weights in $\boldsymbol{\alpha}_n^g$ to represent the input article, denoted as general context vector $\boldsymbol{c}_n^g$:

$$\boldsymbol{c}_n^g = \boldsymbol{\alpha}_n^g \boldsymbol{H} = \sum_{t=1}^{T} \alpha_{nt}^g \cdot \boldsymbol{h}_t.$$

General context vector $\boldsymbol{c}_n^g$ is a $2U$ dimensional vector, which consists of only the parts of the input article related to general MeSH probe $\boldsymbol{p}_n^g$. In fact, a research article can be associated with multiple general aspects of biomedical knowledge. For example, a research article about some disease probably also talks about its treatment. Therefore, in order to extract multiple general aspects of biomedical information from an input article, we need multiple general MeSH probes, for instance, one MeSH probe for disease, another for treatment, and so on. Thus, with N general MeSH probes $\{\boldsymbol{p}_1^g, \boldsymbol{p}_2^g, ..., \boldsymbol{p}_N^g\}$, we can obtain N general context vectors as an N-by-$2U$ matrix:

$$\{\boldsymbol{c}_1^g, \boldsymbol{c}_2^g, ..., \boldsymbol{c}_N^g\}. \tag{8.1}$$

Static Topic-specific MeSH Probes. In practice, rather than general biomedical knowledge, such as disease and treatment, research articles are usually more focused on specific biomedical knowledge, such as diabetes and brain research. However, limited by the model size, we can only have a handful of general MeSH probes that carry the most general biomedical knowledge shared by all training articles. Considering the sheer range of specific biomedical knowledge, it is impossible for MeSHProbeNet-P to have one general MeSH probe for every specific aspect of biomedical knowledge. Thus,

we propose two types of topic-specific MeSH probes for specific biomedical knowledge: static topic-specific MeSH probes and dynamic topic-specific MeSH probes.

Static topic-specific MeSH probes are a large set of vectors of the model parameters, each carrying one specific aspect of biomedical knowledge. But only one static topic-specific MeSH probe is activated at a time based on the topic of the input article. The topic of the input article is indicated by the journal name it is published in, as every biomedical journal has a definite research topic. For example, the international journal *Brain Research* is focused on the brain sciences. Hence, given a dataset containing articles from N_j journals, MeSHProbeNet-P has N_j corresponding static topic-specific MeSH probes $\{\boldsymbol{p}_1^s, \boldsymbol{p}_2^s, ..., \boldsymbol{p}_{N_j}^s\}$, whose dimension is also $2U$ like general MeSH probes and superscript s stands for static. Thus, given a biomedical article published in journal j, MeSHProbeNet-P will select a static topic-specific MeSH probe for the input article based on the journal information:

$$\boldsymbol{p}^s = \boldsymbol{p}_j^s.$$

Accordingly, we can obtain a normalized weight vector $\boldsymbol{\alpha}^s$ over the GRU hidden states with respect to static topic-specific MeSH probe $\boldsymbol{p}^s$:

$$\boldsymbol{\alpha}^s = \text{softmax}(\boldsymbol{p}^s \boldsymbol{H}^T).$$

This allows MeSHProbeNet-P to output a topic-specific context vector $\boldsymbol{c}^s$, by calculating the weighted summation of the GRU hidden states according to the weights in $\boldsymbol{\alpha}^s$:

$$\boldsymbol{c}^s = \boldsymbol{\alpha}^s \boldsymbol{H}. \tag{8.2}$$

Different from general context vectors, $\boldsymbol{c}^s$ is a single vector that contains the specific aspect of biomedical knowledge related to the topic of the input article.

Dynamic Topic-specific MeSH Probes. Static topic-specific MeSH probes assume each research journal has a distinct research topic and model the journal topic attention at phrase level. Sometimes it is also helpful to model the journal topic attention at word level. For example, different journals may have similar or overlapped research topics, such as *Brain Research*, *Experimental Brain Research* and *Behavioural Brain Research*. Modeling the topic-specific attentions at word level can help us discover

similar research topics among similar journals. Therefore, MeSHProbeNet-P also has dynamic topic-specific MeSH probes, which are personalized at journal word level to incorporate semantics in journal names and dynamically calculate the attention weights.

Specifically, dynamic topic-specific MeSH probes take the journal name j of an input article as a text sequence. Suppose the journal name is M word long. Then journal name j can be represented as a sequence of M word embeddings:

$$\boldsymbol{X}^j = \{\boldsymbol{x}_1^j, \boldsymbol{x}_2^j, ..., \boldsymbol{x}_m^j, ..., \boldsymbol{x}_M^j\}.$$

As with the textual content of the input article, a bidirectional GRU is used to capture the dependency between adjacent journal name words:

$$\overrightarrow{\boldsymbol{h}_m^j} = \overrightarrow{GRU}(\boldsymbol{x}_m^j, \overrightarrow{\boldsymbol{h}_{m-1}^j}),$$
$$\overleftarrow{\boldsymbol{h}_m^j} = \overleftarrow{GRU}(\boldsymbol{x}_m^j, \overleftarrow{\boldsymbol{h}_{m+1}^j}).$$

In order to take advantage of shared semantics, dynamic topic-specific MeSH probes use the same word embeddings and GRU as the textual context in Section 8.3.1. Therefore, the dimension of $\boldsymbol{x}_m^j$ is also D_w. The GRU hidden state for the m^{th} word in journal name j is denoted as $\boldsymbol{h}_m^j = [\overrightarrow{\boldsymbol{h}_m^j}, \overleftarrow{\boldsymbol{h}_m^j}]$. Hence, the hidden states of journal name j can be represented as a M-by-$2U$ matrix $\boldsymbol{H}^j = [\boldsymbol{h}_1^j; \boldsymbol{h}_2^j; ...; \boldsymbol{h}_m^j; ...; \boldsymbol{h}_M^j]$.

Then we can obtain an M-by-T relatedness matrix between the journal name GRU hidden states and the textual content GRU hidden states:

$$\boldsymbol{R} = \boldsymbol{H}^j \boldsymbol{H}^T,$$

where each element $r_{mt} = \boldsymbol{h}_m^j \cdot \boldsymbol{h}_t$ is the inner product between journal name hidden state $\boldsymbol{h}_m^j$ and textual content hidden state $\boldsymbol{h}_t$, indicating the relatedness between the m^{th} journal name word and the t^{th} content word. Next, a max operation is applied to $\boldsymbol{R}$ along its first dimension, so that a content word is considered topic related as long as it is related to one of the journal name words:

$$z_t = \max_{1 \leq m \leq M} r_{mt}.$$

Hence, we get a 1-by-T dynamic relatedness vector $\boldsymbol{z}$. After that, we normalize $\boldsymbol{z}$ to obtain dynamic attention weight vector $\boldsymbol{\alpha}^d$, where superscript d stands for dynamic:

$$\boldsymbol{\alpha}^d = \text{softmax}(\boldsymbol{z}).$$

Accordingly, MeSHProbeNet-P can output a dynamic topic-specific context vector $\boldsymbol{c}^d$ based on attention weight $\boldsymbol{\alpha}^d$ and the textual content GRU hidden states:

$$\boldsymbol{c}^d = \boldsymbol{\alpha}^d \boldsymbol{H}. \tag{8.3}$$

The static topic-specific MeSH probes are personalized at phrase level, while the dynamic topic-specific MeSH probes are personalized at word level. The former is more straightforward, while the latter is more fine-grained. It is helpful for MeSHProbeNet-P to contain both types of topic-specific probes.

8.3.3 Multi-view Neural Classifier

Given a biomedical article, MeSHProbeNet-P first generates a set of personalized MeSH probes based on the research topic of its journal: N general MeSH probes, one static topic-specific MeSH probe and one dynamic topic-specific MeSH probe. Then the input article is converted to a set of fixed-length context vectors regardless of the article length. These context vectors are then fed to the multi-view neural classifier. In addition, in order to directly utilize the journal information of each input article, the multi-view neural classifier also has a journal embedding module, where each journal name is represented as a unique vector of length D_j. Hence, the input to the multi-view neural classifier is the concatenation of journal embedding $\boldsymbol{y}$ and the context vectors derived in Eq. 8.1, 8.2 and 8.3, denoted by comprehensive context vector $\boldsymbol{E}$:

$$\boldsymbol{E} = \text{concat}(\boldsymbol{c}_1^g, \boldsymbol{c}_2^g, ..., \boldsymbol{c}_N^g, \boldsymbol{c}^s, \boldsymbol{c}^d, \boldsymbol{y}).$$

$\boldsymbol{E}$ is a $(N+2) * 2U + D_j$ dimensional vector that carries all extracted features of an input article. The multi-view neural classifier implements a function f that maps comprehensive context vector $\boldsymbol{E}$ to V conditional probability distributions, where V is the size of the MeSH vocabulary. The output of f is a vector whose i^{th} element $f(i, \boldsymbol{E})$ estimates the probability that the i^{th} MeSH term should be assigned to the current article:

$$P(MeSH_i = 1|\boldsymbol{E}) = f(i, \boldsymbol{E}),$$

where $MeSH_i$ denotes the i^{th} MeSH term in the MeSH vocabulary. In practice, a three layer neural network is adopted as function f:

$$f(\boldsymbol{E}) = \sigma(\boldsymbol{W}_2 \text{ReLU}(\boldsymbol{W}_1 \boldsymbol{E} + \boldsymbol{b}_1) + \boldsymbol{b}_2), \tag{8.4}$$

where $\sigma(\cdot)$ is the element-wise sigmoid function to ensure that each output neuron is a probability in the range of $[0, 1]$, $\boldsymbol{W}_1$, $\boldsymbol{W}_2$ are the weight matrices for each layer, and $\boldsymbol{b}_1$, $\boldsymbol{b}_2$ are the biases. Thus, the training objective is to minimize the binary cross entropy loss between the ground truth labels and the labeling probabilities in Eq. 8.4.

Unlike most previous large-scale biomedical text labeling models with separate binary classifiers for MeSH terms, MeSHProbeNet-P has a unified multi-label classifier for all the MeSH terms. This not only greatly improves the model efficiency as the size V of MeSH terms are huge, but also allows the semantics of the word embeddings and MeSH probes to be shared across all MeSH terms. Moreover, the correlation between different MeSH terms can be captured by classifier weight matrices.

In the prediction phase, we can obtain the final labeling recommendations based on the output of function f in Eq. 8.4 by learning the optimal thresholds for each MeSH term on a held-out validation set.

8.4 Experiments

We demonstrate the effectiveness of our MeSHProbeNet-P model on a real-world MeSH indexing task. We participated in the 2019 BioASQ challenge and compare its performance with several state-of-the-art MeSH indexing systems, including MTI and DeepMeSH. Our system won the first place in the first batch of the challenge. To illustrate how the personalization of MeSH probes affects the performance, we also provide ablation studies on the personalizable MeSH probes.

8.4.1 Dataset and Experimental Settings

The training dataset is downloadable on the challenge webpage[3] . It contains 14,200,259 biomedical articles which are annotated manually by the NLM human experts. On average, 12.69 MeSH terms are assigned to each article. In total, 28,863 distinct MeSH terms are covered by the training dataset. For each article in the training dataset, we

[3] http://participants-area.bioasq.org/general_information/

have the unique identifier of the article (PMID), the title of the article, the abstract of the article, the year the article was published, the journal name the article was published in and a set of MeSH terms assigned to the article. Hence, in here the textual content of a biomedical article refers to the concatenation of its title and abstract.

In the preprocessing step, all non-alphanumeric characters, stop words and words with a total frequency lower than 10 are removed, and all words are converted to lowercase. The dimensionalities of word embeddings and journal embeddings are set to 350 and 100, respectively. The number of GRU layers is set to 2. The size of the GRU hidden unit is set to 350 per direction, thus 700 for a bidirectional unit. The dimensionality of MeSH probes is also set to 700 accordingly. The default MeSHProbeNet-P is equipped with 5 general MeSH probes, 1 static topic-specific MeSH probe and 1 dynamic topic specific MeSH probe. The multi-view neural classifier has a hidden layer of 10000 units. We deploy 0.1 dropout, 5e-10 L2 regularization and snapshot ensemble [131] to prevent over-fitting. The learning rate for stochastic gradient descent is set to 0.0005 and we also clip the gradients whose values are larger than 1. All parameters including embeddings are randomly initialized.

8.4.2 Evaluation Metrics

Two sets of evaluations metrics are used to evaluate the MeSH indexing performance: the flat measures and the hierarchical measures.

The flat measures consist of accuracy and two sets of F-measure based metrics: Macro F-Measure (MaF) and Micro F-Measure (MiF). Accuracy represents the fraction of correct predictions. Macro measures give equal weight to all MeSH classes. Frequent MeSH terms and infrequent MeSH terms are equally important. Thus Macro Precision (MaP) and Macro Recall (MaR) are calculated as the average precision and recall over all MeSH classes. MaF is then computed as the harmonic mean of MaP and MaR. Micro measures aggregate all test cases and treat each test case equally. Frequent MeSH terms therefore have higher weights than infrequent MeSH terms. We can see that different F-Measures have different focuses, for example, MiF focuses more on the frequent MeSH terms, while MaF treats all MeSH classes equally regardless of their frequencies. As with the BioASQ challenge evaluation, we will also take MiF as our major measure.

Table 8.1: Comparison results based on the flat measures.

Models	MiP	MiR	MiF	MaP	MaR	MaF	Acc
MTIFL	0.6610	0.5960	0.6268	0.6284	0.5552	0.5320	0.4640
MTI	0.6328	0.6468	0.6397	0.5937	0.5953	0.5524	0.4797
ceb	0.6768	0.6280	0.6515	0.6125	0.5045	0.4967	0.4873
DeepMeSH	**0.7319**	0.6056	0.6628	**0.6886**	0.5137	0.5139	0.5053
MeSHProbeNet-P	0.7090	**0.6624**	**0.6849**	0.6728	**0.5670**	**0.5556**	**0.5264**

Table 8.2: Comparison results based on the hierarchical measures.

Models	LCA-P	LCA-R	LCA-F	HiP	HiR	HiF
MTIFL	0.5508	0.4909	0.5035	0.7889	0.7105	0.7277
MTI	0.5358	0.5333	0.5207	0.7601	0.7567	0.7408
ceb	0.5624	0.5130	0.5228	0.8013	0.7231	0.7420
DeepMeSH	**0.5905**	0.5015	0.5298	**0.8304**	0.7120	0.7507
MeSHProbeNet-P	0.5746	**0.5587**	**0.5535**	0.8011	**0.7761**	**0.7725**

Since the MeSH vocabulary is organized in a hierarchical structure, the hierarchical measures are also used to evaluate the performance, including Hierarchical Precision (HiP), Hierarchical Recall (HiR), Hierarchical F-Measure (HiF), Lowest Common Ancestor Precision (LCA-P), Lowest Common Ancestor Recall (LCA-R) and Lowest Common Ancestor F-measure (LCA-F) [132].

8.4.3 Experimental Results

We show the comparison results of our model with the state-of-the-art models, including ceb, the NLM default MTI, MTI First Line indexing (MTIFL) [115] and DeepMeSH [122] on the first test set which contains 7194 articles and the complete results are available on the challenge webpage[4] .

The comparison results of each model are reported in Table 8.1. The best scores are highlighted in boldface. Between the NLM official baselines, MTIFL focuses more on the precision score than the recall, resulting in lower F-measures than MTI. DeepMeSH utilizes the deep doc2vec technique [59] and the tf-idf representations. It achieves higher

[4] http://participants-area.bioasq.org/results/7a/

MiF scores than MTI, but its MaF score is lower than MTI, which means it pays more attention to frequent MeSH terms such as "humans" and "animals". It can be observed that MeSHProbeNet-P achieves the highest scores in all F-measures. The MeSH probes are able to extract comprehensive feature vectors to improve MeSH indexing performance. It is worth mentioning that, compared with the best performing baseline DeepMeSH, MeSHProbeNet-P achieves a 3.3% improvement in MiF and an 8.1% improvement in MaF, indicating that MeSHProbeNet-P achieves a much larger improvement on infrequent MeSH terms. That is because the personalizable MeSH probes are able to extract topic specific features for more specific and less frequent terms. In addition, the multi-view classifier of MeSHProbeNet-P contains the MeSH correlation information which can further benefit the labeling performance for infrequent MeSH terms, therefore MeSHProbeNet-P also gains the best MaF score.

The comparison results based on the hierarchical measures of each model are reported in Table 8.2. As with the flat measure result, the best scores are also highlighted in boldface. The hierarchical measures are calculated based on the hierarchical structure of the MeSH thesaurus and MeSH terms are not treated as independent from each other, thus the hierarchical measures consider the semantic distance between MeSH terms. The performances of each model in terms of the hierarchical measures are similar with their flat measure performances: MTI achieves higher F-Measures than MTIFL; ceb and DeepMeSH outperform both of the official MTIFL and MTI; and MeSHProbeNet-P still obtains the highest scores in all F-measures.

Another advantage of MeSHProbeNet-P is its end-to-end nature: it does not need any prior knowledge, external knowledge or human guidance for the training of the MeSH probes. The probes are able to automatically learn biomedical semantics during training.

8.4.4 Ablation Studies on Personalizable MeSH Probes

We have demonstrated strong empirical results of MeSHProbeNet-P on the challenge test set. In order to further investigate the importance of the personalizable MeSH probes and the effect of different types of MeSH probes, we perform ablation studies of MeSHProbeNet-P. Since the 2019 challenge test set is not publicly available yet, the ablation study is conducted on the 2017 challenge test set of 6661 articles, and accordingly the training set is also changed to the 2017 version, which contains 12,834,585 articles.

Table 8.3: Ablation results based on the flat measures.

MeSHProbeNet-P	MiP	MiR	MiF	MaP	MaR	MaF	Acc
No Attn	0.6728	0.6216	0.6462	0.6180	0.5002	0.4951	0.4809
1 General Attn	0.6759	0.6221	0.6479	0.6302	0.4987	0.4940	0.4829
1 Static Attn	0.6763	0.6291	0.6519	0.6327	0.5057	0.5027	0.4868
1 Dynamic Attn	0.6819	0.6284	0.6541	0.6346	0.5071	0.5030	0.4887
5 General Attns	0.7075	0.6544	0.6799	0.6993	0.5707	0.5693	0.5188
7 General Attns	0.7034	0.6638	0.6830	0.6648	0.5856	0.5710	0.5234
Default	**0.7147**	**0.6691**	**0.6911**	**0.6814**	**0.5931**	**0.5782**	**0.5338**

Table 8.4: Ablation study results based on the hierarchical measures.

MeSHProbeNet-P	LCA-P	LCA-R	LCA-F	HiP	HiR	HiF
No Attn	0.5632	0.5125	0.5217	0.7971	0.7255	0.7399
1 General Attn	0.5658	0.5141	0.5239	0.7985	0.7276	0.7419
1 Static Attn	0.5663	0.5174	0.5267	0.7958	0.7315	0.7440
1 Dynamic Attn	0.5666	0.5149	0.5250	0.8007	0.7264	0.7426
5 General Attns	0.5877	0.5382	0.5484	0.8176	0.7504	0.7655
7 General Attns	0.5798	0.5458	0.5480	0.8081	0.7610	0.7661
Default	**0.5898**	**0.5512**	**0.5555**	**0.8168**	**0.7649**	**0.7724**

One of our core claims is that the MeSH probes can extract comprehensive features from an input article, and this is crucial for MeSH indexing. To give evidence for this claim, we include in comparison the MeSHProbeNet-P model without attentions, which directly feeds the GRU output to the multi-view classifier and uses no MeSH probes, denoted by MeSHProbeNet-P (No Attn).

To study the difference between the three types of MeSH probes, we include in comparison three single attention models: MeSHProbeNet-P with 1 general MeSH probe, denoted by MeSHProbeNet-P (1 General Attn); MeSHProbeNet-P with 1 static topic-specific MeSH probe, denoted by MeSHProbeNet-P (1 Static Attn); and MeSHProbeNet-P with 1 dynamic topic-specific MeSH probe, denoted by MeSHProbeNet-P (1 Dynamic Attn).

We also emphasize that the personalizability of MeSH probes can improve the model performance. Thus, we compare the performance of MeSHProbeNet-P with

non-personalized MeSH probes and the performance of MeSHProbeNet-P with personalized MeSH probes. Please note that, MeSHProbeNet-P with non-personalized MeSH probes falls back to the vanilla MeSHProbeNet [136]. For the sake of fair comparison, both models have 7 MeSH probes: MeSHProbeNet-P with non-personalized MeSH probes contains 7 general MeSH probes, denoted by MeSHProbeNet-P (7 General Attns); MeSHProbeNet-P with personalized MeSH probes contains 5 general MeSH probes, 1 static topic-specific MeSH probe and 1 dynamic topic-specific MeSH probe, denoted by MeSHProbeNet-P (Default). We call it default because MeSHProbeNet-P (Default) is the same model as used in Section 8.4.3.

In addition, to inspect the influence of different numbers of MeSH probes, MeSHProbeNet-P with 1, 5 and 7 general MeSH probes are also included in the comparison, denoted by MeSHProbeNet-P (1 General Attn), MeSHProbeNet-P (5 General Attns) and MeSHProbeNet-P (7 General Attns). All the other parameters, such as the embedding dimension and the number of GRU layers, are the same as the challenge model in Section 8.4.3 for each model.

The analysis results based on the flat measures and the hierarchical measures are reported in Table 8.3 and Table 8.4, respectively. The best scores are highlighted in boldface. The importance of MeSH probes is shown in the comparison between the MeSHProbeNet-P models with and without MeSH probes. The performance is significantly improved by the attentive MeSH probes, especially by the MeSHProbeNet-P models with multiple MeSH probes. MeSHProbeNet-P (1 General Attn) and MeSHProbeNet-P (No Attn) have similar performance, as they both try to use a $2U$ dimensional vector to represent the entire input article. In the comparison among single probe models, MeSHProbeNet-P (1 General Attn), MeSHProbeNet-P (1 Static Attn) and MeSHProbeNet-P (1 Dynamic Attn), we can observe that topic-specific MeSH probes outperform general MeSH probes, as they directly extract the information related to the topic of the input article. The comparison between MeSHProbeNet-P (7 General Attns) and MeSHProbeNet-P (Default) demonstrates the effectiveness of personalizable MeSH probes. With personalized MeSH probes, MeSHProbeNet-P (Default) can extract both general and specific biomedical knowledge, and the performance is further improved. In addition, based on the comparison between MeSHProbeNet-P (1 General Attn), MeSHProbeNet-P (5 General Attns) and MeSHProbeNet-P (7 General Attns), we can observe that increasing the number of MeSH probes is also helpful, although the improvement per added MeSH probe becomes less and less significant as the number of

MeSH probes gets larger. However, increasing the number of general probes will also increase the model size and time consumption. Therefore, in the experiments, we use 5 as the default setting for the number of general probes, which is a good trade-off point between efficiency and effectiveness. It is also worth mentioning that an ensemble of MeSHProbeNet-P and MTI can further improve the MiF score to 0.6965.

8.5 Conclusions

We present a novel end-to-end large-scale MeSH indexing model MeSHProbeNet-P. MeSHProbeNet-P is a deep neural network classifier with personalizable MeSH probes, which is able to customize its MeSH probes for every different input article automatically. With customized MeSH probes, both general and specific biomedical knowledge can be extracted from an input article, and MeSH terms are then labeled by a unified multi-view neural classifier. The experimental results on the BioASQ challenge dataset demonstrate the effectiveness of MeSHProbeNet-P.

Chapter 9

Conclusions and Future Directions

In this dissertation, three types of context information in text data and their corresponding contextual representation learning models are studied. Context information not only allows us to learn representations with coherent syntax and semantics from text data, but also is easy and convenient to collect.

For contextual representation learning methods based on spatial context, we propose a series of models that utilize global context and local context to learn topic structures and word embeddings. In those models, we focus on exploiting global context and local context collaboratively, and further enhance the quality of representations that were originally learned on a single type of spatial context information.

For contextual representation learning methods based on temporal context, we propose two models on diachronic literary data and time series data, respectively. The former constructs temporal context by splitting literary data into time slices and learns coherent dynamic representations of text objects, which further enables us to study language and knowledge evolution. The latter learns temporal features of EEG time series using contextual representation learning, allowing us to detect the onset of seizures in real time.

For contextual representation learning methods based on domain context, we propose to learn representations of text data based on its domain context. We first propose to extract features from input documents using self-attentive probes. The self-attentive probes are essentially memory cells that memorize the domain information of the task.

Thus, the learned representations are guaranteed to be domain-specific. We further propose to customize the self-attentive probe if given more fine-grained domain context information. The customized probes are more domain-specific and space-efficient.

Among all the advantages of using context information to learn representations of text data, the most outstanding one is the ease to collect and construct the context data. We are in the era of big data and text data is being generated at an increasing speed. Invaluable knowledge and language regularities are contained in the innumerable text data, but it is infeasible to manually label all the data. Being easy and convenient to construct, context information provides a way to learn representations from large-scale text data with zero or very little human supervision. Therefore, one future research direction for text-based contextual representation learning is about new ways to construct the context information, i.e., what assumptions we can make about the context information to reflect human language regularities. A good example of contextual assumptions is the distributional hypothesis [7] which makes the assumption that "words that are used and occur in the same contexts tend to purport similar meanings". It allows us to establish language models based on the local spatial context of text data and further learn the distributed representations of words. It would be exciting to see new contextual representation learning assumptions narrowing the gap between human understandings and computer understandings of languages. Another future research direction is to devise more sophisticated and effective model architectures to realize the contextual assumptions. For example, although NPLM [12], Word2Vec [1] and BERT [5] are all based on the distributional hypothesis, each of the models was able to significantly advance the research domain with a different architecture. A more sophisticated architecture also means that more complicated assumptions can be made about the context information of text data.

References

[1] 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*, pages 3111–3119, 2013.

[2] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[3] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[4] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations, 2018.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[6] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017.

[7] Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

[8] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391, 1990.

[9] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[10] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[11] Geoffrey E Hinton. Learning distributed representations of concepts. In *Proceedings of the eighth annual conference of the cognitive science society*, volume 1, page 12. Amherst, MA, 1986.

[12] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

[13] William L. Hamilton, Jure Leskovec, and Dan Jurafsky. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016.

[14] Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. Statistically significant detection of linguistic change. In *Proceedings of the 24th International Conference on World Wide Web*, pages 625–635. ACM, 2015.

[15] Robert Bamler and Stephan Mandt. Dynamic word embeddings via skip-gram filtering. *arXiv preprint arXiv:1702.08359*, 2017.

[16] Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. Dynamic word embeddings for evolving semantic discovery. In *Proceedings of the eleventh acm international conference on web search and data mining*, pages 673–681, 2018.

[17] John Gantz and David Reinsel. The digital universe decade-are you ready. *IDC iView*, 2010.

[18] Kathy Lee, Diana Palsetia, Ramanathan Narayanan, Md Mostofa Ali Patwary, Ankit Agrawal, and Alok Choudhary. Twitter trending topic classification. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 251–258. IEEE, 2011.

[19] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat Thalmann. Who, where, when and what: discover spatio-temporal topics for twitter users. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 605–613. ACM, 2013.

[20] Weiwei Cui, Shixia Liu, Li Tan, Conglei Shi, Yangqiu Song, Zekai J Gao, Huamin Qu, and Xin Tong. Textflow: Towards better understanding of evolving topics in text. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2412–2421, 2011.

[21] Mengen Chen, Xiaoming Jin, and Dou Shen. Short text classification improved by learning multi-granularity topics. In *IJCAI*, pages 1776–1781. Citeseer, 2011.

[22] Qiming Diao, Jing Jiang, Feida Zhu, and Ee-Peng Lim. Finding bursty topics from microblogs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 536–544. Association for Computational Linguistics, 2012.

[23] Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer, 2006.

[24] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[25] 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 *Advances in Information Retrieval*, pages 338–349. Springer, 2011.

[26] Hanna M Wallach. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd international conference on Machine learning*, pages 977–984. ACM, 2006.

[27] Pengfei Hu, Wenju Liu, Wei Jiang, and Zhanlei Yang. Latent topic model based on gaussian-lda for audio retrieval. In *Pattern Recognition*, pages 556–563. Springer, 2012.

[28] Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian lda for topic models with word embeddings. In *ACL (1)*, pages 795–804, 2015.

[29] David Menestrina, Steven Euijong Whang, and Hector Garcia-Molina. Evaluating entity resolution results. *Proceedings of the VLDB Endowment*, 3(1-2):208–219, 2010.

[30] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494. AUAI Press, 2004.

[31] 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. ACM, 2010.

[32] Amit Gruber, Yair Weiss, and Michal Rosen-Zvi. Hidden topic markov models. In *International conference on artificial intelligence and statistics*, pages 163–170, 2007.

[33] Nemanja Djuric, Hao Wu, Vladan Radosavljevic, Mihajlo Grbovic, and Narayan Bhamidipati. Hierarchical neural language models for joint representation of streaming documents and their content. In *Proceedings of the 24th International Conference on World Wide Web*, pages 248–255. International World Wide Web Conferences Steering Committee, 2015.

[34] Jun Zhu, Amr Ahmed, and Eric P. Xing. Medlda: maximum margin supervised topic models. *Journal of Machine Learning Research*, 13:2237–2278, 2012.

[35] Jun Zhu, Ning Chen, Hugh Perkins, and Bo Zhang. Gibbs max-margin topic models with data augmentation. *Journal of Machine Learning Research*, 15(1):1073–1110, 2014.

[36] David Blei and John Lafferty. Correlated topic models. *Advances in neural information processing systems*, 18:147, 2006.

[37] 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.

[38] Qiaozhu Mei, Deng Cai, Duo Zhang, and ChengXiang Zhai. Topic modeling with network regularization. In *Proceedings of the 17th international conference on World Wide Web*, pages 101–110. ACM, 2008.

[39] Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pages 246–252. Citeseer, 2005.

[40] 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. ACM, 2008.

[41] Andriy Mnih and Geoffrey E Hinton. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088, 2009.

[42] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *ACL (1)*, pages 873–882, 2012.

[43] 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 (ACL)*, 2016.

[44] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Topical word embeddings. In *AAAI*, pages 2418–2424, 2015.

[45] Shaohua Li, Jun Zhu, and Chunyan Miao. Psdvec: A toolbox for incremental and scalable word embedding. *Neurocomputing*, 237:405–409, 2017.

[46] Tom Griffiths. Gibbs sampling in the generative model of latent dirichlet allocation. *Unpublished note at http://citeseerx.ist.psu.edu*, 2002.

[47] Chris C Holmes, Leonhard Held, et al. Bayesian auxiliary variable models for binary and multinomial regression. *Bayesian analysis*, 1(1):145–168, 2006.

[48] Nicholas G Polson, James G Scott, and Jesse Windle. Bayesian inference for logistic models using pólya–gamma latent variables. *Journal of the American statistical Association*, 108(504):1339–1349, 2013.

[49] Jianfei Chen, Jun Zhu, Zi Wang, Xun Zheng, and Bo Zhang. Scalable inference for logistic-normal topic models. In *Advances in Neural Information Processing Systems*, pages 2445–2453, 2013.

[50] David Mimno, Hanna M Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 262–272. Association for Computational Linguistics, 2011.

[51] 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.

[52] Guangxu Xun, Vishrawas Gopalakrishnan, Fenglong Ma, Yaliang Li, Jing Gao, and Aidong Zhang. Topic discovery for short texts using word embeddings. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 1299–1304. IEEE, 2016.

[53] Guangxu Xun, Yaliang Li, Wayne Xin Zhao, Jing Gao, and Aidong Zhang. A correlated topic model using word embeddings. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017.

[54] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[55] Chris H. Q. Ding, Tao Li, and Wei Peng. Nonnegative matrix factorization and probabilistic latent semantic indexing: Equivalence chi-square statistic, and a hybrid method. In *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA*, pages 342–347. AAAI Press, 2006.

[56] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185, 2014.

[57] 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, 2015.

[58] Liqiang Niu, Xinyu Dai, Jianbing Zhang, and Jiajun Chen. Topic2vec: learning distributed representations of topics. In *Asian Language Processing (IALP), 2015 International Conference on*, pages 193–196. IEEE, 2015.

[59] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. In *ICML*, volume 14, pages 1188–1196, 2014.

[60] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM, 2001.

[61] Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 136–145. Association for Computational Linguistics, 2012.

[62] Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th international conference on World wide web*, pages 337–346. ACM, 2011.

[63] Felix Hill, Roi Reichart, and Anna Korhonen. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 2016.

[64] Thang Luong, Richard Socher, and Christopher D Manning. Better word representations with recursive neural networks for morphology. In *CoNLL*, pages 104–113, 2013.

[65] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Hlt-naacl*, volume 13, pages 746–751, 2013.

[66] Omer Levy, Yoav Goldberg, and Israel Ramat-Gan. Linguistic regularities in sparse and explicit word representations. In *CoNLL*, pages 171–180, 2014.

[67] Zhiyong Lu. Pubmed and beyond: a survey of web tools for searching biomedical literature. *Database*, 2011:baq036, 2011.

[68] Don R Swanson. Fish oil, raynaud's syndrome, and undiscovered public knowledge. *Perspectives in biology and medicine*, 30(1):7–18, 1986.

[69] D. Weissenborn, M. Schroeder, and G. Tsatsaronis. Discovering relations between indirectly connected biomedical concepts. *J Biomed Semantics*, 6:28, 2015.

[70] D. Cameron, R. Kavuluru, T. C. Rindflesch, A. P. Sheth, K. Thirunarayan, and O. Bodenreider. Context-driven automatic subgraph creation for literature-based discovery. *J Biomed Inform*, 54:141–57, Apr 2015.

[71] V. Gopalakrishnan, K. Jha, A. Zhang, and W. Jin. Generating hypothesis: Using global and local features in graph to discover new knowledge from medical literature. In *BICOB*, pages 23–30, 2016.

[72] D. Hristovski, C. Friedman, T. C. Rindflesch, and B. Peterlin. Exploiting semantic relations for literature-based discovery. *AMIA Annu Symp Proc*, pages 349–53, 2006.

[73] Marc Weeber, Henny Klein, Lolkje de Jong-van den Berg, Rein Vos, et al. Using concepts in literature-based discovery: Simulating swanson's Raynaud–Fish oil and Migraine–magnesium discoveries. *J. Assoc. Inf. Sci. Technol.*, 52(7):548–57, 2001.

[74] Ralph A DiGiacomo, Joel M Kremer, and Dhiraj M Shah. Fish-oil dietary supplementation in patients with raynaud's phenomenon: a double-blind, controlled, prospective study. *The American journal of medicine*, 86(2):158–164, 1989.

[75] Michael D Gordon and Robert K Lindsay. Toward discovery support systems: A replication, re-examination, and extension of swanson's work on literature-based discovery of a connection between raynaud's and fish oil. *Journal of the American Society for Information Science*, 47(2):116–128, 1996.

[76] Wanda Pratt and Meliha Yetisgen-Yildiz. Litlinker: capturing connections across the biomedical literature. In *Proceedings of the 2nd international conference on Knowledge capture*, pages 105–12, 2003.

[77] P. Srinivasan and B. Libbus. Mining MEDLINE for implicit links between dietary substances and diseases. *Bioinformatics*, 20 Suppl 1:i290–96, Aug 2004.

[78] Jonathan D Wren. Extending the mutual information measure to rank inferred literature relationships. *BMC bioinformatics*, 5(1):145, 2004.

[79] Kishlay Jha and Wei Jin. Mining novel knowledge from biomedical literature using statistical measures and domain knowledge. In *Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 317–26, 2016.

[80] T. C. Rindflesch and M. Fiszman. The interaction of domain knowledge and linguistic structure in natural language processing: interpreting hypernymic propositions in biomedical text. *J Biomed Inform*, 36(6):462–77, Dec 2003.

[81] B. Wilkowski, M. Fiszman, C. M. Miller, D. Hristovski, S. Arabandi, G. Rosemblat, and T. C. Rindflesch. Graph-based methods for discovery browsing with semantic predications. *AMIA Annu Symp Proc*, 2011:1514–23, 2011.

[82] Padmini Srinivasan. Text mining: Generating hypotheses from medline. *J. Assoc. Inf. Sci. Technol.*, 55(5):396–413, 2004.

[83] Guangxu Xun, Yaliang Li, Jing Gao, and Aidong Zhang. Collaboratively improving topic discovery and word embeddings by coordinating global and local contexts. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 535–543. ACM, 2017.

[84] Meliha Yetisgen-Yildiz and Wanda Pratt. A new evaluation methodology for literature-based discovery systems. *Journal of biomedical informatics*, 42(4):633–643, 2009.

[85] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[86] Don R Swanson. Migraine and magnesium: eleven neglected connections. *Perspectives in biology and medicine*, 31(4):526–557, 1988.

[87] Ronen Feldman and Haym Hirsh. Exploiting background information in knowledge discovery from text. *Journal of Intelligent Information Systems*, 9(1):83–97, 1997.

[88] Brian M Ross, Craig Hudson, Jeffrey Erlich, Jerry J Warsh, and Stephen J Kish. Increased phospholipid breakdown in schizophrenia: evidence for the involvement of a calcium-independent phospholipase a2. *Archives of general psychiatry*, 54(5):487–494, 1997.

[89] Chia-Feng Kuo, Shun Cheng, and John R Burgess. Deficiency of vitamin e and selenium enhances calcium-independent phospholipase a2 activity in rat lung and liver. *The journal of nutrition*, 125(6):1419, 1995.

[90] Xiaohua Hu, Xiaodan Zhang, Illhoi Yoo, Xiaofeng Wang, and Jiali Feng. Mining hidden connections among biomedical concepts from disjoint biomedical literature sets through semantic-based association rule. *International Journal of Intelligent Systems*, 25(2):207–23, 2010.

[91] Xiaohua Hu, Xiaodan Zhang, Illhoi Yoo, and Yanqing Zhang. A semantic approach for mining hidden links from complementary and non-interactive biomedical literature. In *SDM*, pages 200–09, 2006.

[92] Robert S Fisher, Walter van Emde Boas, Warren Blume, Christian Elger, Pierre Genton, Phillip Lee, and Jerome Engel. Epileptic seizures and epilepsy: definitions proposed by the international league against epilepsy (ilae) and the international bureau for epilepsy (ibe). *Epilepsia*, 46(4):470–472, 2005.

[93] Jessica A Wilden and Aaron A Cohen-Gadol. Evaluation of first nonfebrile seizures. *Am Fam Physician*, 86(4):334–40, 2012.

[94] Robert Ryan Clancy and Agustin Legido. Postnatal epilepsy after eeg-confirmed neonatal seizures. *Epilepsia*, 32(1):69–76, 1991.

[95] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.

[96] Kang Li, Xiaoyi Li, Yuan Zhang, and Aidong Zhang. Affective state recognition from eeg with deep belief networks. In *Bioinformatics and Biomedicine (BIBM), 2013 IEEE International Conference on*, pages 305–310. IEEE, 2013.

[97] Alexander J Casson, David C Yates, John S Duncan, Esther Rodriguez-Villegas, et al. Wearable electroencephalography. *Engineering in Medicine and Biology Magazine, IEEE*, 29(3):44–56, 2010.

[98] Xiaowei Jia, Kang Li, Xiaoyi Li, and Aidong Zhang. A novel semi-supervised deep learning framework for affective state recognition on eeg signals. In *Bioinformatics and Bioengineering (BIBE), 2014 IEEE International Conference on*, pages 30–37. IEEE, 2014.

[99] Ali Hossam Shoeb. *Application of machine learning to epileptic seizure onset detection and treatment*. PhD thesis, Massachusetts Institute of Technology, 2009.

[100] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5:3, 1988.

[101] Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58, 1989.

[102] Nathalie Japkowicz, Stephen Jose Hanson, Mark Gluck, et al. Nonlinear autoassociation is not equivalent to pca. *Neural computation*, 12(3):531–545, 2000.

[103] Bruno A Olshausen et al. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

[104] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, pages 79–86, 1951.

[105] Guangxu Xun, Yujiu Yang, Liangwei Wang, and Wenhuang Liu. Latent community discovery with network regularization for core actors clustering. In *COLING (Posters)*, pages 1351–1360, 2012.

[106] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[107] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[108] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[109] NCBI Resource Coordinators. Database resources of the national center for biotechnology information. *Nucleic acids research*, 44(Database issue):D7, 2016.

[110] Stuart J Nelson, Michael Schopen, Allan G Savage, Jacque-Lynne A Schulman, and Natalie Arluk. The mesh translation maintenance system: structure, interface design, and implementation. In *Medinfo*, pages 67–69, 2004.

[111] Vishrawas Gopalakrishnan, Kishlay Jha, Guangxu Xun, Hung Q Ngo, and Aidong Zhang. Towards self-learning based hypotheses generation in biomedical text domain. *Bioinformatics*, 34(12):2103–2115, 2017.

[112] Guangxu Xun, Kishlay Jha, Vishrawas Gopalakrishnan, Yaliang Li, and Aidong Zhang. Generating medical hypotheses based on evolutionary medical concepts. In *Data Mining (ICDM), 2017 IEEE International Conference on*, pages 535–544. IEEE, 2017.

[113] Kishlay Jha, Guangxu Xun, Vishrawas Gopalakrishnan, and Aidong Zhang. Augmenting word embeddings through external knowledge-base for biomedical application. In *Big Data (Big Data), 2017 IEEE International Conference on*, pages 1965–1974. IEEE, 2017.

[114] James G. Mork, Antonio Jimeno-Yepes, and Alan R. Aronson. The NLM medical text indexer system for indexing biomedical literature. In Axel-Cyrille Ngonga Ngomo and George Paliouras, editors, *Proceedings of the first Workshop on Bio-Medical Semantic Indexing and Question Answering, a Post-Conference Workshop of Conference and Labs of the Evaluation Forum 2013 (CLEF 2013) , Valencia, Spain, September 27th, 2013.*, volume 1094 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2013.

[115] Alan R. Aronson, James G. Mork, Clifford W. Gay, Susanne M. Humphrey, and Willie J. Rogers. The NLM indexing initiative's medical text indexer. In Marius Fieschi, Enrico W. Coiera, and Jack Yu-Chan Li, editors, *MEDINFO 2004 - Proceedings of the 11th World Congress on Medical Informatics, San Francisco, California, USA, September 7-11, 2004*, volume 107 of *Studies in Health Technology and Informatics*, pages 268–272. IOS Press, 2004.

[116] James G. Mork, Dina Demner-Fushman, Susan Schmidt, and Alan R. Aronson. Recent enhancements to the NLM medical text indexer. In Linda Cappellato,

Nicola Ferro, Martin Halvey, and Wessel Kraaij, editors, *Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014.*, volume 1180 of *CEUR Workshop Proceedings*, pages 1328–1336. CEUR-WS.org, 2014.

[117] Alan R. Aronson and François-Michel Lang. An overview of metamap: historical perspective and recent advances. *JAMIA*, 17(3):229–236, 2010.

[118] Jimmy J. Lin and W. John Wilbur. Pubmed related articles: a probabilistic topic-based model for content similarity. *BMC Bioinformatics*, 8, 2007.

[119] George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artieres, Axel Ngonga, Norman Heino, Eric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics*, 16:138, 2015.

[120] Lei Tang, Suju Rajan, and Vijay K Narayanan. Large scale multi-label classification via metalabeler. In *Proceedings of the 18th international conference on World wide web*, pages 211–220. ACM, 2009.

[121] Ke Liu, Shengwen Peng, Junqiu Wu, ChengXiang Zhai, Hiroshi Mamitsuka, and Shanfeng Zhu. Meshlabeler: improving the accuracy of large-scale mesh indexing by integrating diverse evidence. *Bioinformatics*, 31(12):339–347, 2015.

[122] Shengwen Peng, Ronghui You, Hongning Wang, Chengxiang Zhai, Hiroshi Mamitsuka, and Shanfeng Zhu. Deepmesh: deep semantic representation for improving large-scale mesh indexing. *Bioinformatics*, 32(12):70–79, 2016.

[123] Ye Yuan, Guangxu Xun, Qiuling Suo, Kebin Jia, and Aidong Zhang. Wave2vec: Learning deep representations for biosignals. In *Data Mining (ICDM), 2017 IEEE International Conference on*, pages 1159–1164. IEEE, 2017.

[124] Ye Yuan, Guangxu Xun, Fenglong Ma, Yaqing Wang, Nan Du, Kebin Jia, Lu Su, and Aidong Zhang. Muvan: A multi-view attention network for multivariate

temporal data. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, pages 717–726. IEEE Computer Society, 2018.

[125] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[126] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[127] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[128] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[129] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014*, pages 103–111, 2014.

[130] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[131] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger. Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*, 2017.

[132] Aris Kosmopoulos, Ioannis Partalas, Eric Gaussier, Georgios Paliouras, and Ion Androutsopoulos. Evaluation measures for hierarchical classification: a unified view and novel approaches. *Data Mining and Knowledge Discovery*, 29(3):820–865, 2015.

[133] Qiao Jin, Bhuwan Dhingra, William Cohen, and Xinghua Lu. Attentionmesh: Simple, effective and interpretable automatic mesh indexer. In *Proceedings of the*

6th BioASQ Workshop A challenge on large-scale biomedical semantic indexing and question answering, pages 47–56, 2018.

[134] Francisco J Ribadas, Luis M De Campos, Vıctor M Darriba, and Alfonso E Romero. Cole and utai participation at the 2014 bioasq semantic indexing challenge. In *Proceedings of the CLEF BioASQ Workshop*, pages 1361–1374. Citeseer, 2014.

[135] Shanfeng Zhu, Jia Zeng, and Hiroshi Mamitsuka. Enhancing medline document clustering by incorporating mesh semantic similarity. *Bioinformatics*, 25(15):1944–1951, 2009.

[136] Guangxu Xun, Kishlay Jha, Ye Yuan, Yaqing Wang, and Aidong Zhang. Meshprobenet: A self-attentive probe net for mesh indexing. *Bioinformatics*, 32(12):70–79, 2016.

[137] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[138] Xin Li, Lidong Bing, Wai Lam, and Bei Shi. Transformation networks for target-oriented sentiment classification. *arXiv preprint arXiv:1805.01086*, 2018.

[139] Lecheng Zheng, Yu Cheng, and Jingrui He. Deep multimodality model for multi-task multi-view learning. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 10–18. SIAM, 2019.

[140] Yao Zhou and Jingrui He. A randomized approach for crowdsourcing in the presence of multiple views. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 685–694. IEEE, 2017.

[141] Yanjie Fu, Junming Liu, Xiaolin Li, and Hui Xiong. A multi-label multi-view learning framework for in-app service usage analysis. *ACM Trans. Intell. Syst. Technol.*, 9(4), January 2018.