11-16-2018

Deep Learning for Learning Representation and Its Application to Natural Language Processing

Shekoofeh Mokhtari
Florida International University, smokh004@fiu.edu

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

DEEP LEARNING FOR LEARNING REPRESENTATION, AND ITS APPLICATION TO NATURAL LANGUAGE PROCESSING

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

by

Shekoofeh Mokhtari

2019

To: Dean John L. Volakis
College of Engineering and Computing

This dissertation, written by Shekoofeh Mokhtari, and entitled Deep Learning for Learning Representation, and Its Application to Natural Language Processing, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Sundaraja Sitharama Iyengar

Shu-Ching Chen

Jaime Leonardo Bobadilla

Debra VanderMeer

Ning Xie, Major Professor

Date of Defense: November 16, 2018

The dissertation of Shekoofeh Mokhtari is approved.

Dean John L. Volakis
College of Engineering and Computing

Andrés G. Gil
Vice President for Research and Economic Development
and Dean of the University Graduate School

Florida International University, 2019

DEDICATION

I dedicate this dissertation work to my beloved family, especially my parents. Without their patience, understanding, support, or love, the completion of this work would not have been possible.

ACKNOWLEDGMENTS

It is the support from many people that brings me with the possibility to complete the dissertation for concluding my Ph.D. study. First and foremost, I would like to express my sincerest thanks and appreciation to my advisor Dr. Ning Xie, for his inimitable support, meticulous guidance, insightful advice, and getting me immersed with an excellent research atmosphere. Second, I want to extend my gratitude to my late co-advisor, Dr. Tao Li, who has not only provided valuable guidance for me doing research project, but also given me constructive suggestion in developing my Ph.D. career. Third, my thanks goes to all my dissertation committee members: Dr. S.S. Iyengar, Dr. Jaime Leonardo Bobadilla, and Dr. Shu-ching Chen, for their helpful advices, insightful comments on my dissertation research and future research career plans. Fourth, Id like to thank all my mentors including Dr. Reza Amini, Dr. Si-Qing Chen , Dr. Benjamin Yao for my three internships: once at IPSOft and twice at Microsoft. The patient guidance and help from them let me accumulate valuable experience and benefit me a lot in my Ph.D. study. Moreover, I would like extend my thanks to our department staff for assisting me with the administrative tasks necessary during my doctoral study: Olga Carbonell, Carlos Cabrera, Steven Luis, Luis Rivera, etc. Additionally, its extremely fortunate for me to join Knowledge Discovery and Research Group (KDRG) where I have built up my research experience and enhanced my knowledge. I am very grateful to all my colleagues of KDRG including Dr. chunqiu Zeng, Dr. Wuabi Zhou , Dr. Wei Xue, Dr. Longhui Zhang, Qing Wang, Wentao Wang, Ramesh Baral, Xiaolong Zhu and Boyuan Guan. Both valuable discussion and helpful suggestion from them continuously enlighten me with new insights into my research problems. Finally, I would like to express my utmost gratitude to my parents and family, whose endless love and understanding are with

me in whatever I pursue. Without the unlimited support from them, I would never go through any tough times in my life.

ABSTRACT OF THE DISSERTATION

DEEP LEARNING FOR LEARNING REPRESENTATION, AND ITS APPLICATION TO NATURAL LANGUAGE PROCESSING

by

Shekoofeh Mokhtari

Florida International University, 2019

Miami, Florida

Professor Ning Xie, Major Professor

As the web evolves even faster than expected, the exponential growth of data becomes overwhelming. Textual data is being generated at an ever-increasing pace via emails, documents on the web, tweets, online user reviews, blogs, and so on. As the amount of unstructured text data grows, so does the need for intelligently processing and understanding it. The focus of this dissertation is on developing learning models that automatically induce representations of human language to solve higher level language tasks.

In contrast to most conventional learning techniques, which employ certain shallow-structured learning architectures, deep learning is a newly developed machine learning technique which uses supervised and/or unsupervised strategies to automatically learn hierarchical representations in deep architectures, and has been employed in varied tasks such as classification or regression. Deep learning was inspired by biological observations on human brain mechanisms for processing natural signals, and has attracted tremendous attention of both academia and industry in recent years due to its state-of-the-art performance in many research domains such as computer vision, speech recognition, and natural language processing.

This dissertation focuses on how to represent the unstructured text data and how to model it with deep learning models in different natural language processing

applications such as sequence tagging, sentiment analysis, semantic similarity and etc. Specifically, my dissertation addresses the following research topics:

- In Chapter 3, we examine one of the fundamental problems in NLP, text classification, by leveraging contextual information [MLX18a];
- In Chapter 4, we propose a unified framework for generating an informative map from review corpus [MLX18b];
- Chapter 5 discusses the tagging address queries in map search [Mok18]. This research was performed in collaboration with Microsoft; and
- In Chapter 6, we discuss an ongoing research work in neural language sentence matching problem. We are working on extending this work to a recommendation system.

TABLE OF CONTENTS

CHAPTER PAGE

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background

As the web evolves even faster than expected, the exponential growth of users becomes overwhelming. The total number of internet users increased from 16 million in December of 1995 to 4,157 million users in December 2017, which consist of 54.4 % population of the world [1]. As the number of users increases, so does the amount of data.

Data and more specifically textual data is being generated at an ever-increasing pace via emails, documents on the web, tweets, online user reviews, blogs, logs, news and so on by users from all around the world. The size of digital data is expected to double every two years, a 50-fold growth from 2010 to 2020. A research study published from IDC in 2014 claimed that 90% of generated data is unstructured [2].

As the amount of unstructured text data grows, so does the need for intelligently processing and understanding it. My overall research goal is to develop learning models that can automatically induce representations of human language to solve higher level language tasks.

In contrast to most conventional learning techniques, which employ certain shallow-structured learning architectures, deep learning is a newly developed machine learning technique that uses supervised and/or unsupervised strategies to automatically learn hierarchical representations in deep architectures for classification [BB00, BC+08]. Deep learning was inspired by biological observations on hu-

[1] https://www.internetworldstats.com/emarketing.htm

[2] https://www.businesswire.com/news/home/20140715005986/en/New-IDC-Study-Uncovers-Practices-Unlocking-Hidden

man brain mechanisms for processing natural signals, and has attracted tremendous attention from the academic community in recent years due to its state-of-the-art performance in many research domains such as computer vision [KSH12, HSK+12], speech recognition [HDY+12], and natural language processing [CWB+11].

1.2 Motivation and Problem Statement

Building and designing intelligent systems which can understand and process human languages as well as humans or even better than them is the ultimate goal of natural language processing. The technology behind all the virtual assistants such as Siri, Google Now, Cortana and Alexa are all competing with each other to exhibit intelligent communication with humans and at the final path pass the Turing test.

The focus of my thesis is on understanding short text data with deep learning techniques. I chose three applications of NLP which consist of understanding and tagging queries in map search, document classification which narrows down to sentiment classification and paraphrase identification or semantic similarity of short text data.

These three tasks are all essential components of understanding human languages which are required for communicating like a human.

The rest of the Section 1.2 briefly discusses the challenges associated with short text data and the main reasons for choosing the neural network to address those challenges.

1.2.1 Short Text Data

All the research topics in my dissertation deal with short text documents. Short text documents are unlike traditional long text documents, due to their characteristics

in terms of length and conciseness. Short text data has some unique characteristics that make them more challenging to process and understand.

- Firstly, short texts such as search queries, do not always follow the syntax structure of a written language, which makes traditional NLP methods, such as syntactic parsing, less effective on processing and understanding short text data.
- Second, short texts contain a limited number of words/characters. For instance, tweets are limited to have only 140 characters. Majority of search queries have 3 words or less [3].
- Third, they are mostly generated based on colloquial phrases so they are usually noisy and contains typos, misspelling errors and abbreviations.

The above characteristics of short text data make them difficult to understand and process. Driven by the challenges above, automatic and efficient techniques for understanding and processing short text data are pressingly demanded.

In recent years, deep learning has acquired great interest to address the different challenges in NLP [CW08, dSG14, SLMN11]. These neural network based models are employed for generating good representation from text data for each specific task.

1.2.2 Deep Learning

Deep learning is the foundation of a recent breakthrough in different research areas [HZRS16, CVMG+14, GMH13]. The fundamentals of deep learning techniques such as backpropagation through time (BPTT), long short-term memory (LSTM) date

[3] $https : //www.slideshare.net/randfish/keepng - up - wth - seo - n - 2017 - beyond/11 - Keyword_Length_of_Search_Queries1Word$

back to decades ago in the 1980s. Recently, they are popular again because of Graphical Processing Units (GPU) which can run experiments much faster than before and also the availability of a huge amount of data. (About 90% of the world's data has been generated over the past two years.[4])

Scales drive deep learning progress. The scale could refer to either the amount of data or the size of the neural network. As shown in Figure 1.1, the algorithm's performance improves after increasing the amount of data for both traditional learning algorithms and neural networks. The performance of traditional learning algorithms such as a support vector machine (SVM) and regression improves for a while as you add more data but after a while the performance plateaus. However, if we train a very large neural network performance keeps getting better and better while the amount of data is increasing.

For hitting high performance, we need two elements. First, we should be able to train a large enough neural network to take advantage of the availability of a very large amount of data and the second is having the huge amount of data. Hence, scale drives deep learning progress, and scale refers to both size of the neural network with a lot of hidden layers and parameters and as well as the size of the data [Shr17].

In the following sections, the contributions of my dissertation along these research directions are briefly presented.

1.3 Contributions

My dissertation addresses the challenges relevant to the research topics outlined above, by designing and developing deep learning based techniques, with the purpose of understanding textual data. Primarily, the main outcomes of my dissertation are

[4]https://www.sciencedaily.com/releases/2013/05/130522085217.html

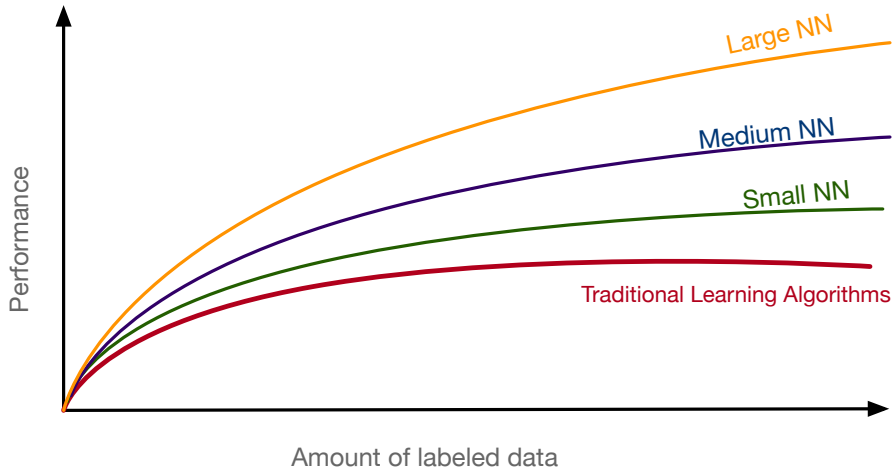


Figure 1.1: Scales drive deep learning progress [Shr17]

highlighted as follows: (1) A general contextual guided hierarchical model for fine-grained sentiment classification of short text data such as reviews by utilizing both content data in review text and the context knowledge from the user and product; (2) A unified framework for visualization and creation of a map to have a holistic view of text data; (3) An end-to-end system for parsing and tagging address queries in a map search; (4) An intelligent document recommendation system for active users. The detailed contributions for three research directions are provided in the remainder of Section 1.3.

1.3.1 Review Mining

The revolution in e-commerce and online shopping has facilitated people's ability to express their opinions and hands-on experiences of products and services. Customers usually share their experience through reviews, comments or ratings on online shopping websites. Reviews contain information that benefits both consumers and producers.

They are favorable to consumers since the reviews from actual buyers can provide a more objective, comprehensive, and precise description of a product. Such reviews are highly influential in shaping the shopping behaviors of potential buyers.

The reviews can also be influential in strategic decision making such as improvement in the quality of service, product manufacturing, sales, and marketing. In the competitive business market, the reviews can also be beneficial to customer satisfaction.

There is an immense number of online user reviews for some favorite products. For instance, some famous books at Amazon [5] have more than 20,000 reviews, and even some restaurants in Yelp [6] have more than 10 thousand reviews!

On the one hand, having an abundant number of reviews is desirable for users because a large review corpus is more likely to offer more comprehensive and accurate assessments of the item. On the other hand, the sheer amount of reviews consume more time and might not be self-intuitive. Some reviews are rather lengthy and flooded with personal stories which are irrelevant to the product.

It is therefore time-consuming for a potential buyer to sift through the texts and make an informed decision. The biased view of a product might influence the users who consult few reviews and might not be able to gauge the overall opinions for the product.

Finally, the large amount of reviews also makes it hard for business personnel to keep track of customer opinions which can be used as a means for product development and quality assurance. As many products are sold from different venues, it is desirable to develop tools that collect, compile and extract useful information from user reviews and provide it as a simple bird's-eye view picture.

[5] www.amazon.com

[6] www.yelp.com

Customers usually read reviews on their smart phones for immediate decisions like choosing a nearby restaurant. Small screen display will also make the process of extracting a digest summary out of reviews more challenging. The huge amounts of lengthy reviews spread across different online stores' websites might not be efficiently utilized in reasonable amount of time.

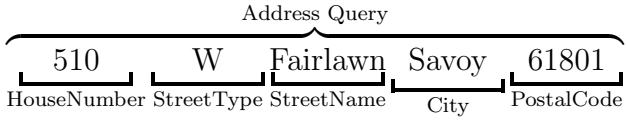
To address the above mentioned concerns, we propose a framework which extracts opinions from a set of reviews, and displays them in an easy to comprehend manner. Our contributions related to this research direction are elaborately discussed in Chapters 3 and 4. Our contributions related to this research direction are summarized as below.

1. Proposing a novel neural network architecture for text classification.
2. Utilizing contextual information and incorporating them into text classification.
3. Generating a structured map for presenting nugget of information in a useful and informative manner.
4. Consequently, the arising classification problem is solved by a hierarchical neural model. An extensive empirical study over an available public dataset was conducted to validate the effectiveness and efficiency of our method.

1.3.2 Tagging Address Queries

Map search is a major vertical in all popular search engines. It also plays an important role in personal assistants on mobile, home or desktop devices. A significant fraction of map search traffic is comprised of "address queries" — queries where either the entire query or some terms in it refer to an address or part of an address (road segment, intersection etc.).

Address parser (AP) is a fundamental annotating process for geocoding service [FS69]. AP aims to predict a label for each element of address query. A simple example of parsing address query is provided below.



In chapter 5, we demonstrate that correctly understanding and tagging address queries is critical for map search engines to fulfill them. We describe several recurrent sequence architectures for tagging such queries. We compare their performance on two subcategories of address queries — single entity (aka *single point*) addresses and multi entity (aka *multi point*) addresses, and finish by providing guidance on the best practices when dealing with each of these subcategories. Our contributions related to this research direction are summarized as below.

1. Proposing a hierarchical neural model for address parsing which mimics the structure of address queries (query from the word, word from the character).
2. We thoroughly examine the address data queries and identify two different patterns in address queries and generate corresponding synthetic queries based on them.
3. An extensive empirical study over real address queries from Bing map logs and synthetic data are conducted to validate the effectiveness and efficiency of our proposed model. A case study from real scenario shows the usefulness of our proposed method in practice.

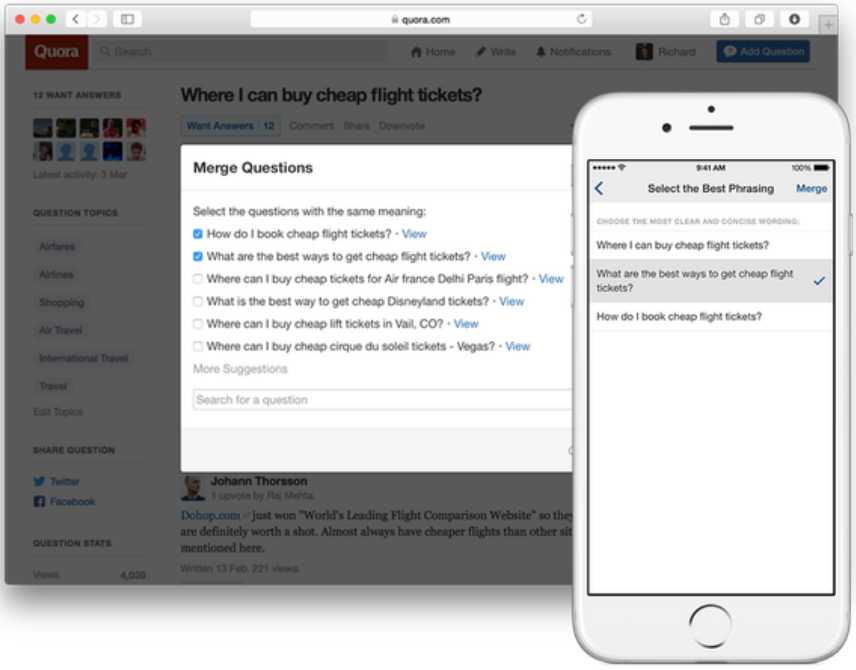


Figure 1.2: An example of application of semantic textual similarity which is in question-answer forms such as Quora [Sri].

1.3.3 Semantic Textual Similarity (STS)

Measuring the semantic similarity between text snippets sentences, paragraphs and documents plays an important component in various tasks such as information retrieval [DDF+90], machine translation [MLS13], text summarization [MCS+06] and ambiguity problem in NLP [Res99]. However, the number of tasks that require computing the similarity between text segments are growing significantly which includes in personal digital assistants (PDAs), image search, query search and so many other applications [MDM07, LZLM07]. A real application of semantic textual similarity is on question-answering forums such as Quora or Stack Overflow (See Figure 1.2). Choosing the best phrasing of asked questions in QA forums will be effectively help the intent of the question which result in finding the best response. Another application of STS is within digital assisstants such as Siri, Cortana, Alexa, Google

Figure 1.3: A sample conversation with Alexa (Amazon virtual assistant).

Home and etc — See Figure 1.3. Our related contribution to this research area are listed below.

1. Considering the structure format of text data and not having enough and large corpus of labeled data, we first formulate the problem with unsupervised method such as auto-encoder.
2. Conducting benchmark experiments on publicly available dataset to compare the performance of popular embedding to choose among them.
3. Proposing a neural network model for paraphrase identification which employ semantic similarity measures of text vector representations as well.
4. Extensive empirical studies on real and synthetic data are conducted to demonstrate the effectiveness and the efficiency of the proposed method.

1.4 Summary and Roadmap

In order to facilitate the reading and understanding the research problems discussed, the organization of this dissertation is outlined as follows. First, we briefly presents the preliminaries and related work of the aforementioned three research directions in Chapter 2. To be continue, we study the problems related to these research directions in Chapter 3, Chapter 4, Chapter 5 and Chapter 6, respectively.

Particularly, in Chapter 3, the text classification problem is studied, where both the content and context knowledge are utilized. In Chapter 4, we focus on visualization framework (*RevMap*) of the review corpus for each item, where the *RevMap* facilitate the browsing through thousens of reviews. In Chapter 5, we study the problem about how to tag each word of address queries in map search. Those tags help the geocoding systems to return more accurate results. Chapter 6 discuss the problem of semantic similarity of textual data and identifying whether questions have the same intent. Finally, in Chapter 7, we conclude the work of this dissertation and discuss the future work along our research.

CHAPTER 2

PRELIMINARIES AND RELATED WORK

This dissertation studies the concrete problems along the aforementioned three research directions in natural language processing applications and the corresponding solutions are exhaustively discussed as well. In this chapter, we highlight existing literature studies that are related to our work in this dissertation. In particular, Section 2.1 reviews the existing work related to the problem determination as well as the relevant techniques such as text classification, sentiment classification, text visualization. Section 2.2 introduces different types of address queries used in the map search, and further describes the corresponding methodologies to understand these queries. Section 2.3 presents existing literature of semantic similarity of text documents.

2.1 Related Work on Review Mining

To summarize opinions or select a subset from reviews in an accurate way, many methods have been studied. In the following of this section, we will first introduce some of the existing work on review mining such as review selection, review summarization and review assessment and ranking.

Review Selection Review selection problem chooses a subset from the large corpus of reviews. There are different metrics to consider in the selection process. Yu and Zhang [YZHS13] consider selecting a subset of reviews which consider attribute coverage and opinion diversity while providing high rate quality reviews.The drawback of this approach is they do not consider to select high-quality reviews. Lappas and Gunopuls [LG10] approach select the representative set with the ma-

jority opinion on each feature. The shortcoming of this approach is that it neglects the minority opinion, without considering how important it is. Lappas and Crovella [LCT12] present an approach for selecting a characteristic set of reviews which provide a good formalization of a review selection problem. They also prove that CRS problem is not only NP-hard but also NP-hard to approximate. They measure the closeness of selected subset with the target vector via L_2 norm differences. They also apply different algorithms like greedy, Integer-regression and a randomized one to a large corpus of reviews and then compare the results of them for different size of the selected subset. One of the bugs of their solution is that they ignore to consider the quality of reviews as well. The selected subset needs to contain high-quality content. Tsaparas and Ntoulas [TNT11] adopted a generic framework to review the subset selection problem that covers the different attributes of the product with high-quality content that also contain different viewpoints. Their formulation provides the subset with at least one positive and at least one negative opinion of the covered feature. The drawback of their method is that it does not consider the proportion of positive or negative opinions of a feature. In another word, They ignore the distribution of opinion on features of the product. Which may give the wrong idea or impression to users.

Review Summarization Review summarization extracts the statistical description of the corpus of reviews and provides a summary of general ideas. Review summarization also has its own drawbacks, it actually sacrifices the immediacy and narrative structure of reviews. There have been several works on automatic summarization; most of them focus on extraction-based summarization. Hu and Liu [HL04], who first came up with review summarization solution; analyze the sentences sentimentally and consider the opinion sentences as positive or negative and

then a statistical summary of the number of positive or negative sentences for a user. They do not consider opinions as biased which is one of the bugs in their job.

Review Assessment and Ranking. Salient research studies devoted to reviewing assessment and ranking. A rank is assigned to each review to define the quality rate of reviews. Reviews are being ranked according to their quality rating.[,,] In this method, they do not consider the quality rating of a selected group of reviews. The main disadvantage of these methods is in the selection process. High rated reviews do not reflect the distribution of entire corpus of reviews and these selected subsets could include repeated opinions on the same feature. Our method dominates these shortcomings by considering other criteria for selecting reviews.

2.1.1 Text Classification

Text classification have gained significant popularity over the past few years within the natural language processing. It has wide range of application such as sentiment analysis [Kim14, PL+08, MDP+11], topic modeling [WM12], email categorization and email spam filtering [SDHH98, CC05, CCM04, LK97, SDHH98] and language modeling [PC98, LM14, MSC+13, BDVJ03].

Current approaches for various aspects of text classification can be divided into two groups, one uses traditional NLP techniques and the recent methods which utilize deep neural network models. The approach taken in this work belongs to the second group.

Traditional Methodologies In traditional NLP methods, one technique that attracts great attention recently is word2vec [MSC+13], in which Mikolov et al.

introduce building representation of words into fixed length vector trained on the large corpus.

Much work has been done for training neural networks on text documents. Neural networks were first introduced for text representation by Bengio [BDVJ03]. More recently, deep learning methods have been extensively studied for text classifications [ZZL15, Kim14, dSG14, LXLZ15, SM15]. Convolution neural network was first used for sentence classification in [Kim14]. A deep model of character level convolution is presented in [ZZL15]. While the basic idea in these studies [ZZL15] and [Kim14] are pretty similar, with the main difference between these two is the number of layers and embedding channels. Temporal ConvNet model in [ZZL15] has 9 layers consist of 6 convolutions and 3 fully connected layers. Also the input text in this model is treated as a sequence of characters. The model in [Kim14] is much simpler with only one convolution layer and one fully connected layer. It also treats the input as a sequence of words instead of characters.

There is also some other similar work for learning representation of tweets for hashtag prediction [DZF+16] and for learning general purpose embedding [VVR16].

2.1.2 Sentiment Classification

The focus of my research is on the task of sentiment classification. In a sentiment classification task, we are looking to predict the expressed sentiment of each document.

Content based Sentiment Classification

Salient number of studies focus only on the content of documents for sentiment classification. Existing approaches for content-based sentiment classification can

be divided into two categories, one uses traditional NLP techniques and the other utilizes deep neural network models.

Neural Network based Methods: Neural networks for text representation were first introduced by Bengio [BDVJ03]. Recently, deep learning methods have been extensively studied for sentiment classification [Kim14, dSG14].

Context based Sentiment Classification

More recently methods leveraged contextual information as well as the content of the document for classification.

A user modeling neural network was presented by Tang et al. (2015), [TQLY15]. They employed user/product preference matrices in order to create personalized word embedding and employ in CNN models. A context-sensitive neural network has proposed by Ren et al.(2016), where they integrate the content with the contextual feature in the form of word embeddings vectors [RZZJ16]. (Chen et al.) proposed neural sentiment classification (NSC) model with the user and product attentions in both word and semantic levels [CST+16]. Seo et al. built vector representation of user and items by using attention based convolutional neural network [SHYL17].

2.1.3 Text Visualization

Mining customer reviews include review selection, summarization, review assessment and ranking, and visualization. We explored some of the studies that focused on review visualization.

Some of the existing studies [WWL+10], [CISSW06], [LHC05] applied visual analytics to present the extracted features of reviews. Chen et al. [CISSW06] used graphs and trees to visualize the conflicting opinions in texts. Liu et al. [LHC05]

proposed a framework with a visual component for analysis and comparison of consumer opinions on competitive products. The OpinionSeer framework from Wu et al. [WWL+10] provided an interactive system for visual analysis of hotel customers' feedbacks. Their model focused on multi-dimensional visual representation which encoded the uncertainty information. They used feature based opinion mining and combined opinions with subjective logic and visualized the output in a radial layout.

Our proposed framework not only extracts opinions from reviews but also explicitly reflects temporal and statistical data. Our model has two significant differences from OpinionSeer [WWL+10]. Firstly, our fine-grained opinion mining is based on the hierarchical neural network model. Secondly, we generate a structured map to display extracted feature in an evolutionary timeline.

2.2 Related Work on Tagging Address Queries

It is crucial to tag each token of address queries correctly. There have been a few strands of works focusing on the category of address queries [WHY+16, LKWS14, CCLZ02, BDS01, SZ14, SMFCM18, RDA+17]. There are two general approach for tagging address queries in map search. The first trend approach the problem as toponym matching which is rely on pairing queries/strings that are The problem can be viewed as toponym matching or sequence tagging. [SMFCM18]

however, traditional rule-based address parser [CCLZ02] require domain knowledge and have been shown to be limited in classification accuracy. Probabilistic methods such as [WHY+16, LKWS14] which are based on HMM and CRF have been developed to improve the rule-based methods but they have some difficulties dealing with rich and complex features. In recent years RNN based models have shown state of the art results on sequence tagging problems [MH16]. In this work,

after formalizing the address tagging problem, we outline several recurrent architectures suitable for modeling it (Section 5.3). We analyze their performance on two structurally different query patterns that we find prevalent in map search logs(§5.5). We conclude with guidance on the practices for choosing a suitable architecture when working with such patterns (Section 5.7).

2.3 Related Work on Semantic Textual Similarity

Matching two potentially heterogeneous language sentences is essential in many natural language processing applications such as paraphrase identification (PI), question answering (QA) and information retrieval (IR) [XJC08, BGWB14, HLLC14].

Natural language sentence matching (NLSM) is the task of comparing two sentences and analyzing the relationship between them. Sentences have complicated structures, both sequential and hierarchical, Hence, it is essential to understand their structure first and then compare and find the sentence matching.

Sentence matching is of central importance to many natural language tasks such as paraphrase identification (PI), image retrieval and etc [CF87]. One of the well studied and useful application in this field is paraphrase identification. Majority of the existing methods formulate this problem as a binary classification task. Given a pair of input text data and classify whether they convey the same meaning/concept or not. We will discuss some recent studies in this research direction below.

Paraphrase identification with noisy pretraining [TDT+17] They employed decomposable attention model[PTDU16] with the model trained on noisy dataset. Instead of using pertained word embedding, they applied sum of embedding character n-grams.

BiMPM: Bilateral Multi-Perspective Matching for Natural Language Sentences In this study [WHF17] , they used joint feature models which use the cross sentence feature or attention from one sentence to another.

Facebook researchers in [CKS+17], demonstrated the effectiveness of the natural language inference (NLI) for transfer learning to various natural language processing tasks and computer vision. They studied learning universal representations of sentences trained on a large corpora and afterwards transfer to the other tasks such as entailment and semantic relatedness, caption-image retrieval, paraphrase detection, binary and multi-class classification. They studied whether a supervised learning can be leveraged instead of learning sentence encoder in an unsupervised manner [BDVJ03, CWB+11, MSC+13, PSM14].

2.4 Summary

This chapter highlights the existing works in the literature, which are strongly related to the three research directions of my dissertation, i.e., document classification, tagging address queries in map search and intelligent document recommendation system with focus on semantic similarity. For each research direction, both the related approaches and evaluation metrics are exhaustively surveyed.

CHAPTER 3

SENTIMENT CLASSIFICATION

3.1 Introduction

Enabling a computer to precisely understand a document so that it can predict the sentiment is crucial, yet the unsolved goal of natural language processing (NLP)! There has been a great deal of effort spent on developing methodologies for text classification and, subsequently in sentiment classification.

The majority of existing methods classify text documents only based on the semantic of text data and ignore relevant contextual information. There are a few studies which take this data into account, but they suffer from model complexity. Thus, it's essential to have to provide an efficient architecture for document classification which leverages both content and context data. [MLX18a]

3.1.1 Background

The task of text classification have gained significant popularity over the past few years in NLP. It has a wide range of applications such as sentiment analysis [Kim14, PL+08, MDP+11], topic modeling [WM12] and spam detection [SDHH98]. In this thesis, we focus on the task of sentiment classification in which accurate prediction of the expressed sentiment of each document is desired.

Motivated by impressive success of neural network in different disciplines such as computer vision [KSH12], speech recognition [GMH13] and natural language processing [Kim14], we propose a hierarchical neural network based model for contextual sentiment classification. Our approach is partly inspired by the success of utilizing image metadata in computer vision, which shows promising results for different

tasks: image annotation and image search [ZHH+10, GJL+13], and multi-modal learning representation of image and tags [SS12]. In this chapter, we introduce an attention-based hierarchical neural network method for sentiment classification by taking contextual information into account.

The rest of this chapter is organized as follows. We first formulate the sentiment classification problems to be studied in this paper. Then, after a brief review of general DNN-based text classification method, our approach with detailed descriptions of the techniques that incorporate contextual information into the neural network architecture is presented. Finally, experimental results with various evaluations are discussed.

3.2 Problem Description

The fine-grained document classification is a classic classification problem, in which we are given a document, in the form of a sequence of tokens $w = (w_1, w_2, ..., w_n)$, and we are required to find the best sentiment class matches this document y. In this specific task, we two class and fine-grained! In this section, we formulate and describe the notations that will be used throughout the paper.

Preliminaries We denote R as a collection of reviews,
$R = \{r_1, r_2, ..., r_n\}$. Each review as shown in figure 3.1 is associated with several attributes such as content(R_t), reviewer(R_u), subject(R_p), helpful rate(R_h), date(R_d) and score(R_s); $r = \{R_t, R_u, R_p, R_h, R_d, R_s\}$.

- **Content** (R_t) is the textual feedback a reviewer left for a subject.

- **Reviewer/User** (R_u) is the actual individual who left the review. Each distinct reviewer in our datasets is represented by a unique random (hashed) string.
- **Subject** (R_p) is either a product/item on Amazon or a business on Yelp dataset.
- **Date** (R_d) is the time that the review is written, which is further divided into three fields: day, month and year.
- **helpful Rate** (R_h) is a number which measure the rating given by other users to the review on one or several topics. On Amazon, this is the percentage of the users who found this review helpful, therefore $h_{amazon} \in [0, 1]$. On Yelp, review rate has three fields: usefulness, coolness, and funniness. Each field of the review rate is represented as an integer number, hence $h_{yelp} = \{uf, c, f\}$.
- **Sentiment Score** (R_s) is the quantitative number that a reviewer assigned an overall rating of the subject. Score is usually a number in range of 1 to 5, $s \in S = \{1, 2, 3, 4, 5\}$, in which 5 is the best feedback and 1 is the poorest. In some variations of this classification problem, score domain (S_D) is polarized to two values: negative and positive.

All of the discussed features are highlighted in Figure 3.1. The main goal of this study is to generate a general-purpose learning of representation of reviews that can be applied in a broad range of classification tasks.

Notation and symbols applied are summarized in the following table.

3.3 Our End-to-End System

The overall architecture of our hierarchical attention model is shown in Figure 3.2. It consists of four main modules: content embedding layer, context embedding layer,

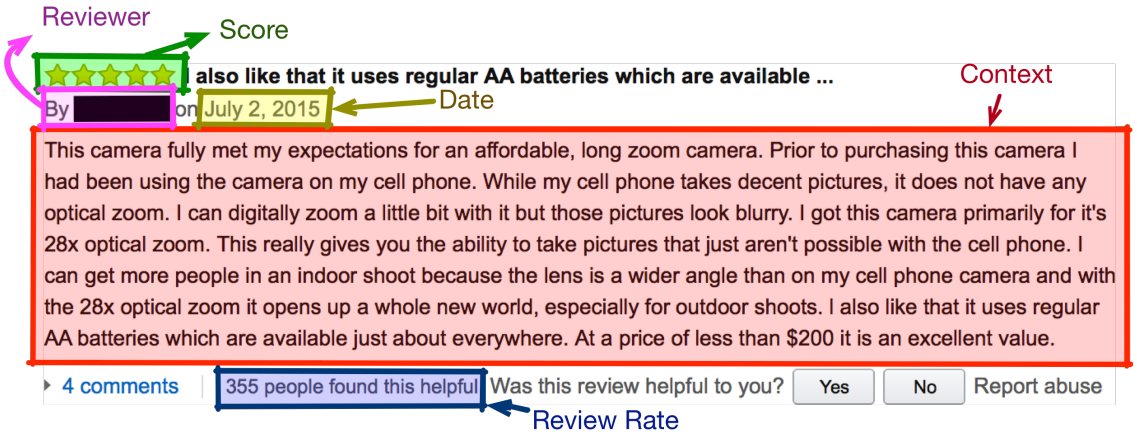


Figure 3.1: A sample review from Amazon

attention layer and the output layer. We describe the details of each component in the following sections.

3.3.1 Content Embedding Layer

This layer is computing features from content (review text) at different levels of granularity. We use a hierarchical neural model to capture two levels of syntactic and semantics representation of the content. Word level solely does not work well for informal text, misspelling words and out of vocabulary (OOV) words. Therefore, our model combines the benefits of character level and word level to learn better representation of informal content of reviews. Our content embedding layer utilize character-level convolutional neural [ZZL15] and bi-directional gated recurrent neural network [CGCB14] for word level embedding. Then, the concatenation of the character embedding and word embedding layer are fed to the highway network [SGS15].

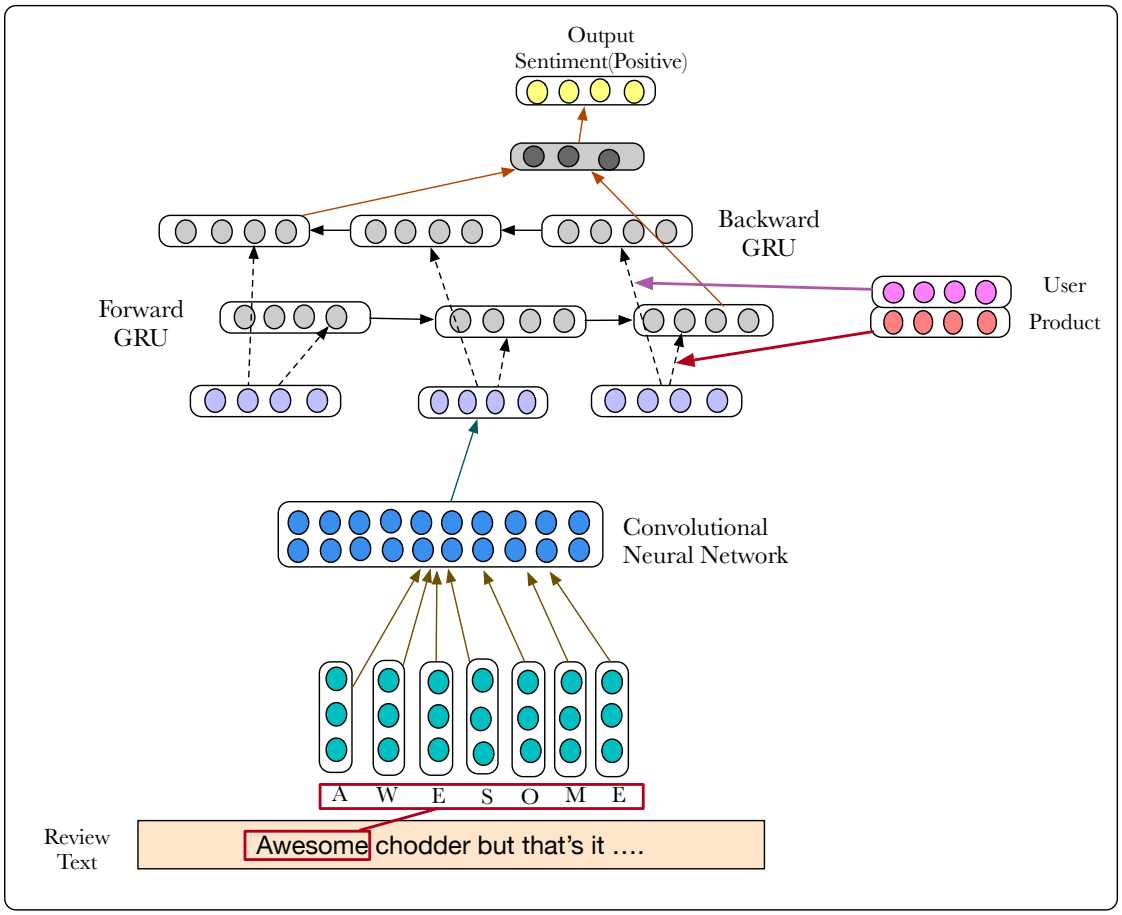


Figure 3.2: Our hierarchical contextual sentiment classification (CSC) model

Table 3.1: Notion used in this article

Symbols	Description
d	Date
b	Bias
s	Score
w	Filter
p	Subject
t	Context
m	Metadata
r_i	Review i
u	Reviewer ID
h	Helpful rate
C_i	Character i
c_i	Feature map i
V	coefficient of text data
V'	coefficient of meta data
R	Collection of reviews
M	Convoluted vector metadata
T	Convoluted vector text data

Character Encoder Review content data (R_t) is treated as a sequence of words $[W_1, W_2, ..., W_{|R|}]$ and each word is a sequence of characters $[C_1, C_2, ..., C_{|W|}]$ and this layer is responsible for projecting each word to a fixed length vector encoding. As illustrated in Figure 3.2, we used convolutional to capture morphological information from characters of words. A convolution layer takes sequence of characters (represented as embeddings) as input and outputs a representation that learn local n-gram features.

Input characters of each word are converted into two dimension matrix with dimension of (first L characters) * (Character embedding length (N)). Where L is the length that will set empirically and N is the character embedding length. Longer reviews are truncated and shorter ones are zero padded so that all the input will have the same length. Word matrix $w_i \in \mathbb{R}^{L \times N}$ fed into the convolution layer to pass through different filter. We applied n group of kernels with varying width size

in order to capture n-gram features and concatenating the extracted features.

$$W_c = C^{f_1} \oplus C^{f_2} \oplus ... \oplus C^{f_n} \tag{3.1}$$

Pooling layer is right after convolution which select specific features from feature map. Features can be selected by using MAX or MIN or AVG. operation. We use max pooling layer in order to extract the most import feature in our model.

$$W_i = [c_1, ..., c_{|W_i|}] \tag{3.2}$$

$$\hat{W}_i = \max\{W_i\} \tag{3.3}$$

We applied a dropout layer [SHK+14] before passing character embeddings input to convolution layers.

Word Encoder Given a review with sequence of words, we map each word to a vector space by using a pre-trained word embedding. We used Glove [PSM14] with dimension d_w for our model. The outputs of character encoder (R_{W_c}) and word encoder (R_{W_w}) concatenated through highway network [SGS15].

$$x = [R_{w_w}; R_{w_c}] \tag{3.4}$$

$$y = H(x, W_H).T(x, W_T) + x.(1 - T(x, W_T)) \tag{3.5}$$

Afterwards, learning representation of review text (R_t) fed to the bidirectional grated recurrent units (GRU)[CGCB14]. GRUs are variants of recurrent neural network (RNN) which are designed to cope with gradient vanishing/exploding problems. We chose GRU instead of long short term memory (LSTM), since it has fewer parameters and therefore less expensive computationally. Basically, a GRU unit is composed of two multiplicative gates which monitor the proportions of information to update and to pass on to the next unit. The formulas to update a GRU unit at time t are:

$$z_t = \sigma(x_t W^z + U^z h_{t-1}) \tag{3.6}$$

$$r_t = \sigma(x_t W^r + U^r h_{t-1}) \tag{3.7}$$

$$h'_t = \tanh(W x_t + r_t \odot U h_{t-1}) \tag{3.8}$$

$$h_t = (z_t \odot h_{t-1} + (1 - z_t) \odot h_t) \tag{3.9}$$

Where σ is sigmoid function and $\odot$ stands for element-wise multiplication. x_t and h_t are the input vector and hidden units at time t, respectively. U^r, U^z denote the weight matrices for the hidden unit h_{t-1}. W^r, W^z are the weight matrices of different gates for input x_t.

3.3.2 Context Embedding Layer

In this layer, we utilizes contextual information available from user (R_u) and product (R_p) to refine the embedding of the content. We project user and product context data to vector space and couple them to produce one single contextual vector by passing through two layer highway neural network.

$$Context = [R_u; R_p] \tag{3.10}$$

3.3.3 Attention Layer

Attention mechanism is motivated by human visual attention. It's been widely used with neural networks specially in image recognition and more recently in NLP tasks. In this module, the objective is to compare the content embedding and contextual embedding, select the nugget of information that is relevant to context. We calculate a probability distribution α based on the relevance between each word in the content (w_i) and the context C. The output vector (O_{att}) is the weighted sum of all the words

in the document.

$$O_{att} = \sum_{i=1}^{L} \alpha_i W_i \tag{3.11}$$

where α_i is the weight of each word w_i. O_{att} produces a set of context-aware feature vectors for each word in the content.

3.3.4 Output Layer

In the output layer, we have a fully connected network and softmax layer. After all the layer mentioned above, the general review embedding is passed through a fully connected layer in which neurons have full connection to all activation in the previous layer. The output from previous layers is flattened to a dense vector and passed to a fully connected softmax layer. In softmax layer, the probability distribution over the classes calculated as follows:

$$p(s = j|r) = \frac{\mathrm{e}^{r^T \theta_j}}{\sum_{k=1}^{K} \mathrm{e}^{r^T \theta_k}} \tag{3.12}$$

where θ_k is the weight vector for class of score k. r is the review dense feature embedding which passed through different layers. We are optimizing the cross entropy loss function as the training goal between predicted and true label associated with each review.

$$\begin{aligned} \mathcal{C} &= -\log \Pi_{i=1}^{N} p(s_i|\mathbf{t_i}, \mathbf{m_i}) + \lambda \|\theta\|_2^2 \\ &= -\sum_{i=1}^{N} [s_i \log(\mathbf{z_i}) + (1 - s_i) log(1 - \mathbf{z_i})] + \lambda \|\theta\|_2^2 \end{aligned} \tag{3.13}$$

Where t_i and m_i is the content data and context data of the review. z is the output from the softmax layer and θ contains all parameters optimized by the network including filter weights and biases of the convolution layers and biases and weights of fully connected layer and softmax, λ is a coefficient of L2 regularization.

3.4 Experiments

In this section, we introduce the experimental setup, empirical evaluation and results on the sentiment classification task. Our work focus is on sentiment classification of online user reviews.

3.4.1 Data

Our experiments are performed with data from two publicly available datasets, yelp academic dataset challenge (2013, 2014, 2015) and Amazon published by [ML13]. These datasets includes all the review characteristics required for our experiments, such as content, sentiment score and product properties. Integrating contextual information will enable the model to learn about users behavior and products as well. For instance, if a product is highly rated, one expects that most of the reviews about it to be positive.

Yelp: The yelp academic dataset is publicly available online[1]. This dataset has $4.1M$ reviews and $947K$ tips by $1M$ users for $144K$ businesses. $1.1M$ business attributes, e.g., hours, parking availability, ambience. Aggregated check-ins over time for each of the 125K businesses $200,000$ pictures from the included businesses.

Amazon: We conduct experiments on SNAP Amazon dataset which was crawled from Amazon between May 1996 and July 2014, and includes 142.8 million reviews in total [MPL15, MTSvdH15]. The entire dataset is very large and unbalanced. We chose our data only from the categories such as Books, Electronics, Movies and TV, CDs and Vinyl.

[1]https://www.yelp.com/dataset/challenge

Table 3.2: Data statistics of Yelp and Amazon datasets. Datasets are from [ZZL15]

Dataset	**Yelp P**	**Yelp F**	**Amazon P**	**Amazon F**
# Train	$560K$	$650K$	$3600K$	$3000K$
# Test	$38K$	$50K$	$400K$	$650K$
# of classes	2	5	2	5

In order to compare our performance with baselines discussed in 3.4.2, we also run experiments with data in [ZZL15]. However we are unable to use this dataset for our experiments since the contextual information was not provided. Therefore, we randomly select our data from Yelp and Amazon datasets and conduct our experiments.

Previous works have demonstrated the effectiveness of data augmentation in classification task with deep learning. Data augmentation is another method which can be used to reduce over-fitting of the models by replicating and increasing the amount of training data. In our case, we replaced some of the words in reviews with their synonyms and reproduced them. These synonyms are obtained from WordNet [Mil95] which contains words grouped together based on their meanings. The statistics of the datasets are depicted in Table 3.2.

Data Preprocessing We defined character set including English alphabets, digits and some of the special characters. Any character which is not in our defined character set such as emotion, HTML tag, etc. is removed from review text data. We also eliminate the English stop words in reviews. Text data varies on length of the input. We consider a fixed length for all the reviews in training and test datasets. Longer reviews are truncated and shorter ones are zero-padded so that all reviews in the dataset have the same length. For Yelp, the lengths of reviews are in the range of [0, 32952] with mean value 540. For Amazon, the lengths of reviews

are in the range of [1, 5000] with mean value 621. In our experiments, we use 1014 characters for review length in Yelp and Amazon.

3.4.2 Baseline Methods

We compare our model with several baselines approaches including both content and context based approaches; consisting of traditional methods such as Bag-of-words (BOW), SVM and neural network like LSTM, character-level CNN, word-level CNN and etc.

BOW (Bag of Words) generates text representation based on the occurrence of words within a document.

Support Vector Machine (SVM) support vector machine classifies document by using some features such as unigrams and bigrams.

Paragraph Vector proposes the PVDM for document sentiment classification [LM14].

Word CNN word based convolutional neural network which is proposed in [Kim14, KGB14]. They trained on top of pre-trained word vectors for sentence-level classification tasks.

Char CNN character based convolutional neural network which is proposed in [ZZL15].

LSTM word-level long short term memory by using Word2vec [GL14] with 300 embedding dimension for embedding the words [HS97].

FastText compact architecture which build upon product quantization to store word embedding [JGB+16].

User Product Neural Network (UPNN) modifies the word embedding with preference matrix of users and products[TQL15]

NSC model documents through hierarchical structure, word-level, sentence-level and document-level. They adopt LSTM with user product attention to capture crucial semantic components for document representation[CST+16].

3.4.3 Experimental Setup

We implement our model with Tensorflow 0.11[AAB+16]. All the experiments are performed on a single machine with *Intel Xeon CPU E5-2660*, 64GB of memory, and GeForce GTX TITAN GPU.

We train all models for 10 epochs and choose the model that has the best performance in validation dataset. We apply L_2 regularization to the weight matrices and apply dropout [SHK+14] with $p = 0.2$ for the hidden layers. We optimize using Adam method [KB14] with learning rate 10^{-3} and initialize the weight parameters using the same method as in [GB10].

Tuning Hyper-Parameters. For training our model, we eliminate the less frequent words from datasets ($|v| < 5$). We initialized the word embedding in word encoder by 100-dimensional pre-trained Glove word embedding.
The hidden size for GRU set to $h = 200$ and for convolution, we used filters with length 2,3,4. The parameters for attention were initialized with a uniform distribution between (0.01, 0.01). We also choose the samples with mini-batch of size 32.

class 1	class 2	class 3	class 4	class 5
$300K$	$200K$	$300K$	$600K$	$1M$
7	10	7	3	2
$2M$	$2M$	$2M$	$1.8M$	$2M$

Table 3.3: Yelp Data distribution over 5 score classes

class 1	class 2	class 3	class 4	class 5
$2.5M$	$2.0M$	$3.8M$	$8.3M$	$24M$
10	11	6	3	1
$25M$	$24.8M$	$23M$	$25M$	$24.3M$

Table 3.4: Amazon Data distribution over 5 score classes

We tuned the hyper parameters on validation dataset by random search. We run our model up to 50 epochs and then we choose the model which achieves the best accuracy on the validation dataset. We run our models 5 times independently with different random seeds and report average performance across the runs. We also report ensemble results which average the prediction probabilities of the 5 models.

Data Augmentation One of the challenges we faced was the skewness of collected data. The distribution of reviews over different score classes on Yelp and Amazon dataset is unbalanced; see e.g. Figure 3.3a for the distribution of reviews over 5-classes in our Yelp dataset. However, to improve the prediction accuracy we need to have almost equal number of reviews among all classes. If we trim the dataset in accordance with the class with minimum number of reviews, more than half of the original data would have been discarded. Data augmentation, in our context, refers to replicating reviews in classes that fall short of data, and replacing some of the words in the replicated ones with their synonyms. These synonyms are obtained from WordNet [Mil95, Fel98] which contains words grouped together based on their meanings.

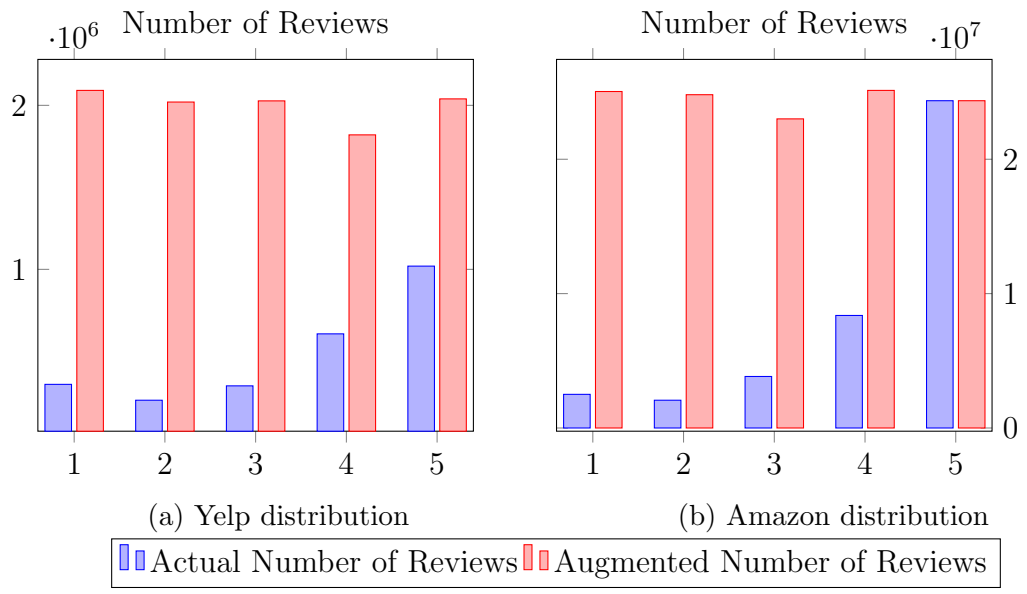


Figure 3.3: Distribution of yelp and Amazon data over 5 classes- The blue column is the real number of reviews for each class and the orange column is after data augmentation

In order to compare our results with those in [ZZL15] and [JGB 16], we conduct experiments with dataset used in [ZZL15]. However we are unable to use their dataset exactly because the metadata of their dataset are not available. We randomly select the same number of samples from Yelp and Amazon datasets in [ZZL15] and perform experiment with this randomly selected data.

3.4.4 Results

The results of our model and comparing with baselines on Yelp 2013 and 2014 are summarized in Table 3.5. The results on Yelp and Amazon dataset from [ZZL15] and with augmented data are shown in Table 3.6. We evaluated the performance of our model with baselines based on accuracy and root mean square error ($RMSE$) in order to measure the deviation between ground truth (S_i) and the predicted

Table 3.5: Classification accuracy (Acc) and RMSE on Yelp 13 and Yelp 14 corpus. Results marked with † are reported from [TQL15] and those marked with ‡ are from [CST+16]. The best result in each group depicted in bold format.

Model	**2013-Acc**	**2013-RMSE**	**2014-Acc**	**2014-RMSE**
Majority †	0.411	1.060	0.392	1.097
Trigram †	0.569	0.814	0.577	0.804
Text Feature †	0.556	0.845	0.572	0.800
Avg Wordvec +SVM †	0.526	0.898	0.530	0.893
SSWE +SVM †	0.549	0.849	0.557	0.851
Paragraph Vector†	0.554	0.832	0.564	0.802
RNTN +Recurrent†	0.574	0.804	0.582	0.821
UPNN(CNN w/0 UP) †	0.577	0.812	0.585	0.808
NSC ‡	0.627	0.701	0.637	**0.686**
NSC + LA ‡	0.631	0.706	0.630	0.715
Our Model(CSC)	**0.638**	**0.699**	**0.641**	**0.688**
Trigram + UPF †	0.570	0.803	0.576	0.789
TextFeature + UPF †	0.561	0.822	0.579	0.791
JMARS †	N/A	0.985	N/A	0.999
UPNN(CNN) ‡	0.596	0.784	0.608	0.764
UPNN(NSC) ‡	0.631	0.762	N/A	N/A

sentiment score (Z_i).

$$RMSE = \sqrt{\left(\frac{\Sigma_{i=1}^{n}(Z_i - S_i)^2}{N}\right)} \tag{3.14}$$

3.5 Data Analysis

In this section, we conduct an analysis of the review datasets in order to find the better design the model for the problem.

The distribution of number of review for some categories of Amazon products such as electronics, movies, books and Yelp data illustrated in Figure 3.4. As shown, Majority number of products have less than 10 reviews and there are limited number of products with high number of review.

After looking closer into data, we noticed that there are some reviews with same text but different rating. An example of such reviews are shown in Table 3.7. An

Table 3.6: Classification accuracy on Amazon and Yelp corpus which used in [ZZL15]. Numbers are in percentage.

Model	**Amazon F**	**Amazon P**	**Yelp F**	**Yelp P**
n-gram	54.27	92.02	56.26	95.64
n-gram (TF-IDF)	52.44	91.54	54.8	95.44
LSTM	59.47	94.34	58.17	94.74
Char-CNN	59.43	93.9	62.05	94.58
FastText.zip	59.9	94.5	63.6	95.6
OurModel(CSC)	**60.11**	**95.1**	**64.03**	93.1
OurModel(Augmented Data)	N/A	N/A	**65.82**	**96.2**

Table 3.7: Selected reviews which have same text data but different ratings

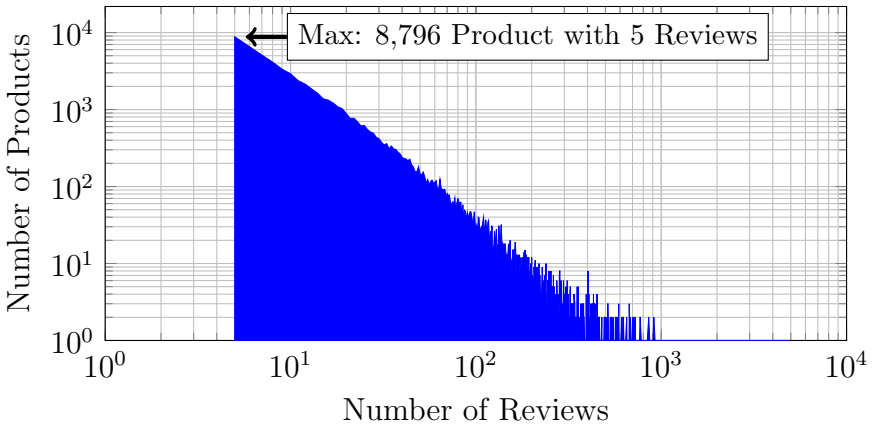
Review	**Rating**
good stuff	4,5
They are usually very friendly and the restaurant is a very nice environment. However, the last couple of times I went in, it was very busy and the service and food was not that great.	3, 5
Fire Grilled Chicken, done whichever way you like: burrito, tacos, salad, etc. They have healthy "Under 500 calorie" menu options. Individual combos for $7-$9 and Family Meals $20-$30. Salsa bar with four fresh salsa options.	3,4

example review such as good stuff can be rated with 4 or 5 star. For such reviews especially those with short text data, more data is needed to predict the sentiment accurately.

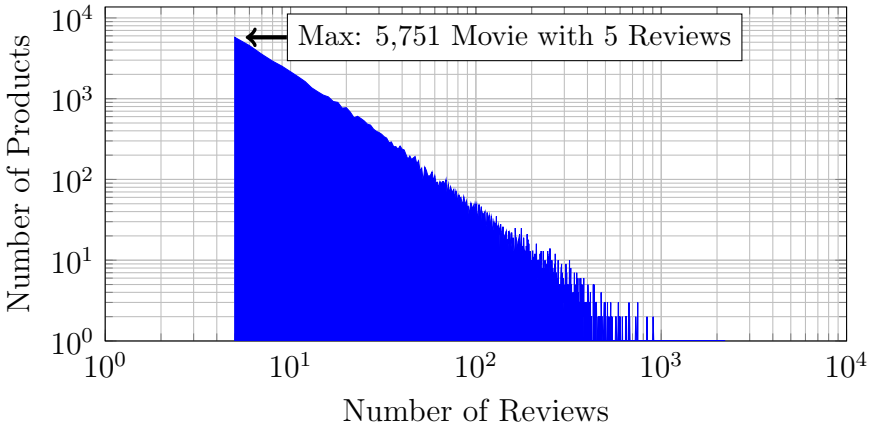
Another example which that the same text data has been rated differently are spams! Although Yelp and Amazon have strong spam filtering detection, still some of them appear in the academic dataset.

3.6 Summary

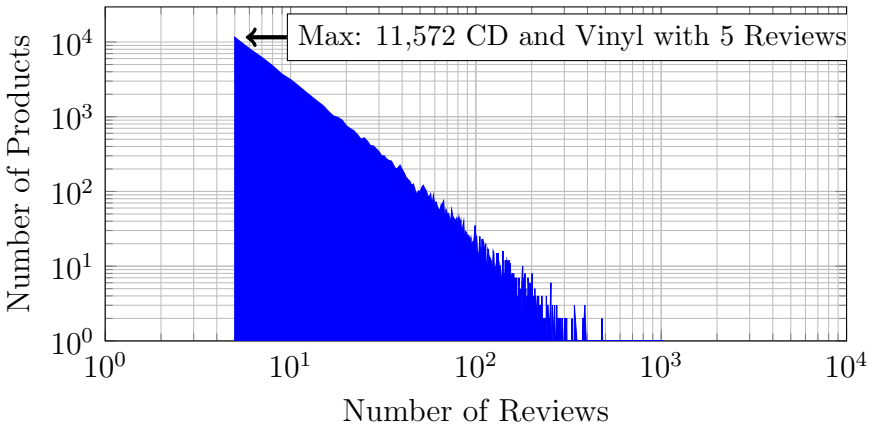
This chapter provides an empirical study on the effectiveness of contextual information for text classification. We compared our approach with a large number of traditional methods and deep learning models on user reviews corpus from Yelp and Amazon. On one hand, experiments and deep analysis shows the effectiveness of our model and employing contextual data. On the other hand, there are many factors affecting the performance in comparison, such as dataset size, types of context information, number of samples from each user or products.



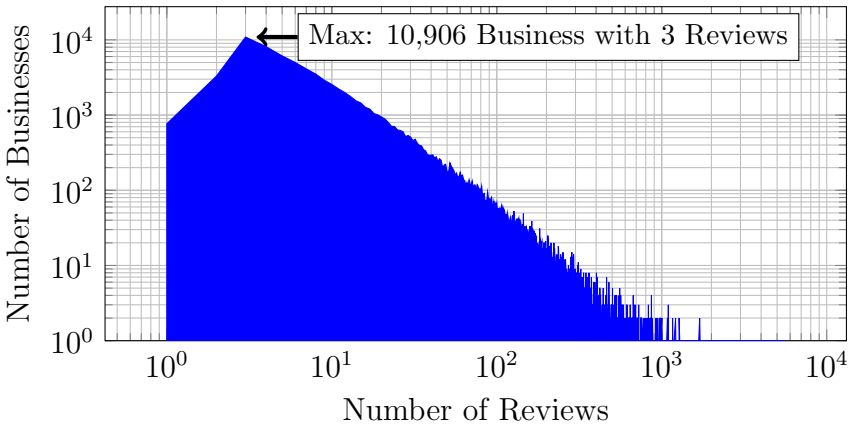
(a) Amazon Electronics review corpus



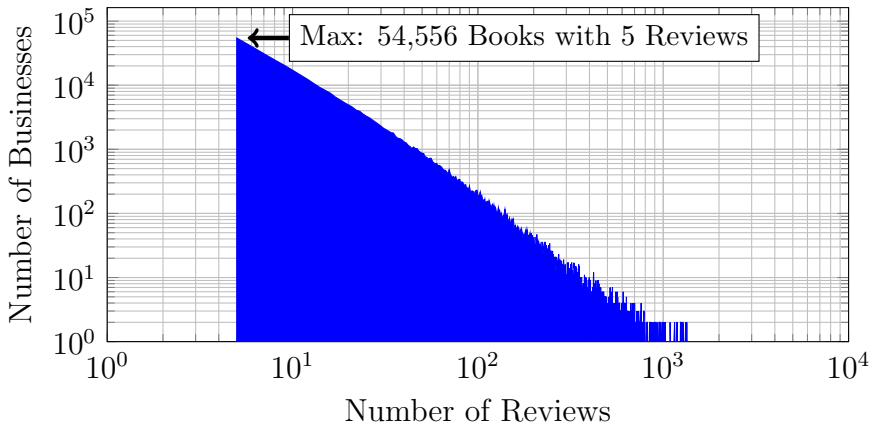
(b) Amazon Movie Review Corpus



(c) Amazon CD and Vinyl review corpus



(d) Yelp Academic Review Corpus



(e) Amazon Book Review Corpus

Figure 3.4: Frequency Number of Reviews

CHAPTER 4

REVIEW MAP VISUALIZATION

4.1 Motivation

Our daily digital life is surrounded by algorithmically selected content such as recommendations, reviews and news feed. The process of extracting useful knowledge out of large corpus have a great impact on shaping the users life and experience. As the data becomes abundant, it becomes increasingly difficult to extract beneficial knowledge out of large data corpus.

In this chapter, we propose a unified framework to display a holistic view of online user review corpus. The framework provides a visual summary of reviews considering the features of the related product and the evolution of reviews along the time. To this end, we present a novel approach, named RevMap, which first extracts features from reviews. Empirical analysis and extensive case studies on a large corpus of Amazon reviews demonstrate the efficacy of our proposed framework.[MLX18b]

4.2 Introduction

The revolution in e-commerce and online shopping have facilitated people to express their opinions and hands-on experiences on products or services. The user reviews from actual buyers can provide more objective, comprehensive, and precise description of a product. Such reviews are highly influential in shaping the shopping behaviors of potential buyers. The reviews can also be influential in strategic decision making for the service quality improvement, product manufacturing, and sales and marketing. In the competitive business market, the reviews can also be very effective in customer satisfaction.

There is an immense number of online user reviews for some popular products. On the one hand, having an abundant number of reviews is desirable for users because large review corpus is more likely to offer more comprehensive and accurate assessments of the item. On the other hand, the sheer amount of reviews consume more time and might not be self-intuitive. Some reviews are rather lengthy and flooded with personal stories which are irrelevant to the product.

It is therefore time-consuming for a potential buyer to sift through the texts and make an informed decision. The biased view of a product might influence the users who consult few reviews and might not be able to gauge the overall opinions for the product. Finally, the large amount of reviews also makes it hard for business personnel to keep track of customer opinions which can be used as a means for product development and quality assurance. As many products are sold from different venues, it is desirable to develop tools that collect, compile and extract useful information from user reviews and provide it as a simple bird's-eye view picture.

The enormous amounts of lengthy reviews spread across different online stores' websites might not be efficiently utilized in the reasonable amount of time. To address the concerns as mentioned earlier, we propose a framework which extracts opinions from a set of reviews and displays them in an easy to comprehend manner.

4.3 Proposed Framework

In this section, we describe the proposed framework. The high-level overview of the proposed unified framework is illustrated in Figure. **??**.

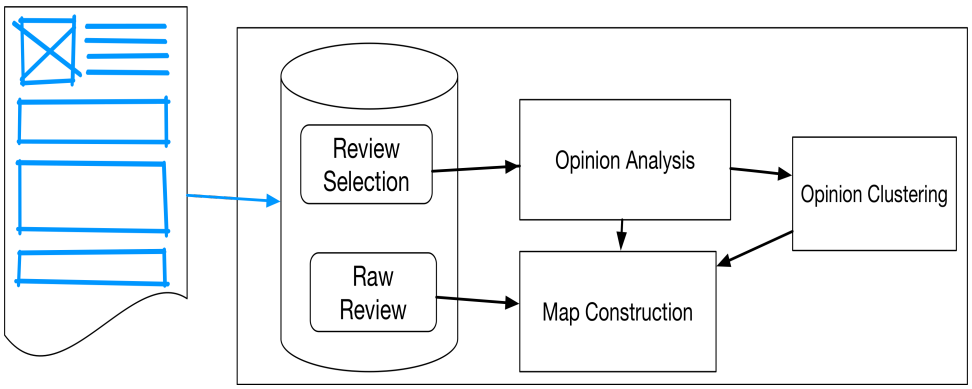


Figure 4.1: System Architecture

4.3.1 Review Selection

For every item, the representative set of reviews are selected. The review selection can be made using this method [LCT12]. Lappas and Crovella in [LCT12] aim to select a characteristic set of reviews which is "close" to the original set of reviews, where they measure the closeness between two sets of reviews using L_2 norm in the vector space.

4.3.2 Opinion Analysis

This module extracts opinions from the review summary. It applies the character based *Bi-directional Gated Recurrent Unit* [CVMBB14] neural network for learning the review embedding and computing distribution probabilities over opinions by projecting this embedding to a softmax output layer. A blueprint of our model is illustrated in Figure 4.2. The review context is treated as a sequence of characters $[C_1, C_2, ..., C_{|R|}]$ and one-hot vector encoding embeds each character.

$$R = C_1 + C_2 + ... + C_{|R|} \tag{4.1}$$

The encoder consists of a forward-GRU and a backward-GRU. Both of them have the same architecture, except the backward-GRU processes the sequence in reverse

order. Each of the GRU units processes these vectors sequentially. Given the initial state h_0, the sequence $h_1, h_2, ...h_m$ is computed as follows:

$$z^{(t)} = \sigma(W^{(z)}x^{(t)} + U^{(r)}h^{(t-1)} + b_z) \tag{4.2}$$

$$r^{(t)} = \sigma(W^{(r)}x^{(t)} + U^{(r)}h^{(t-1)} + b_r) \tag{4.3}$$

$$\tilde{h}^{(t)} = \tanh(W^{(h)}x^{(t)} + r^{(t)} \circ Uh^{(t-1)} + b_h) \tag{4.4}$$

$$\tilde{h}^{(t)} = (1 - z^{(t)}) \circ \tilde{h}^{(t)} + z^{(t)} \circ h^{(t-1)} \tag{4.5}$$

The gates are updated using Equation 4.2 and are reset using Equation 4.3. The new memory $\tilde{h}^{(t)}$ is the consolidation of the input character of review $x^{(t)}$ with past hidden state $h^{(t-1)}$. In above equations, the terms W and b refer to the weight and bias respectively. The final state from the forward GRU and the backward GRU are combined and passed through a fully connected layer. The review embedding is calculated by combining the final state from forward GRU (h_m^f) and backward GRU (h_0^b):

$$r = W^f h_m^f + W^b h_0^h \tag{4.6}$$

Softmax

The output from previous layers is flattened to a dense vector and passed to a fully connected softmax layer. In the softmax layer, the probability distribution over the k classes is calculated as follows:

$$p(y_t = k|r) = \frac{\exp(w_k^T r + b_k)}{\sum_{k=1}^{K} \exp(w_k^T r + b_k)} \tag{4.7}$$

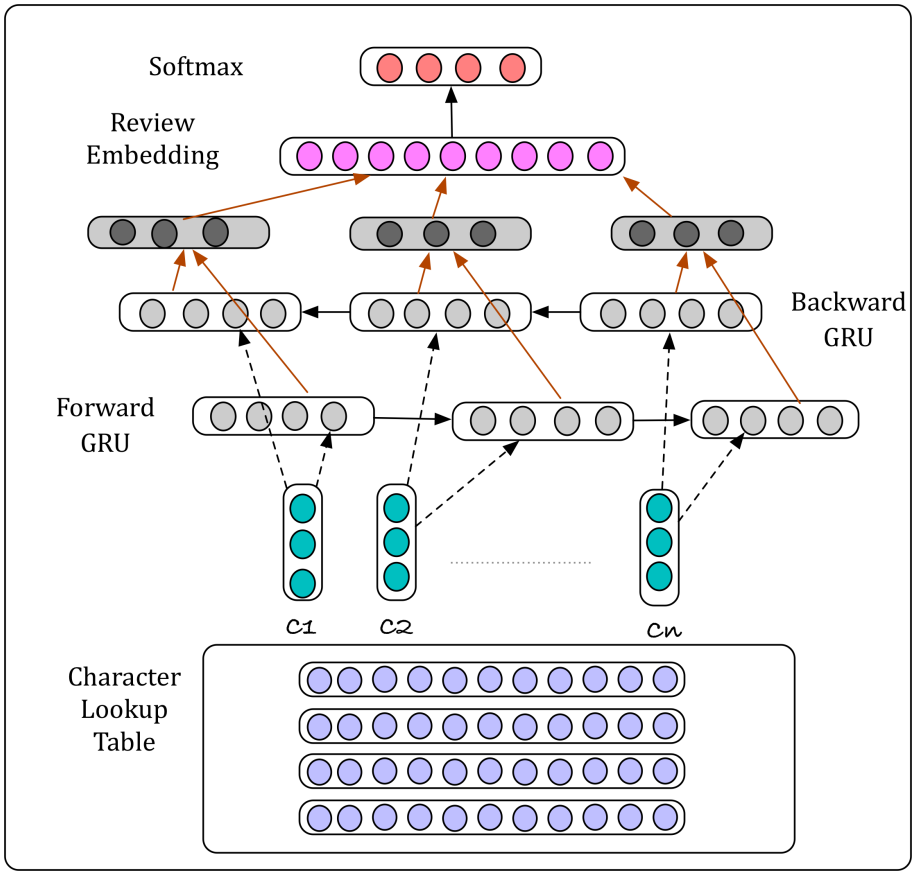


Figure 4.2: Bidirectional Gated Recurrent Unit

where r is the review embedding which is fed to the softmax layer. We also optimize the cross entropy loss function between predicted and true opinions associated with each review.

$$C = \frac{1}{B} \sum_{i=1}^{B} \sum_{j=1}^{L} -r_{ij} \log(p_{ij}) \tag{4.8}$$

where B is the batch size, L is the number of classes, r_{ij} is the ground truth, and $p_{i,j}$ is the probability of review i having the opinion j.

After extracting the features, we discover the paradigmatic relation among features. A paradigmatic relation is a particular part of word association mining and analysis. By definition, two words are paradigmatically related if they share a similar context. In order to discover such a relation, we look at the context of each

word and then compute the similarity of those contexts. Highly similar word pairs can be assumed to have paradigmatic relations. There are many different ways to implement this general idea. We adopt the BM25 retrieval model for paradigmatic relation mining.

Let's consider r_1 and r_2 two reviews which is represented in vector space model. Their definition in VSM is as follows:

$$r_1 = (x_1, x_2, ..x_n) \tag{4.9}$$

$$BM25(w_i, r_1) = \frac{(k+1)c(w_i, r_1)}{c(w_i, r_1) + k(1 - b + b * |r_1|/avr_1)} \tag{4.10}$$

$$x_i = \frac{BM25(w_i, r_1)}{\sum_{j=1}^{N} BM25(w_j, r_1)} \tag{4.11}$$

Then,

$$sim(r_1, r_2) = \sum_{i=1}^{N} IDF(w_i) x_i y_i \tag{4.12}$$

Figure 4.3 shows an output example of feature extraction from digital camera's reviews. For example, Image and picture have paradigmatic relation. We find features with paradigmatic relation to reduce the number of features of a product in map. Since we could not cover all of the features in the map, reduced number of feature helps to cover more features and increase diversity.

4.3.3 Map Construction

The proposed map consists of a set of nodes and connecting lines in an evolutionary timeline. A node represents an aspect based opinion, and a line connects a sequence of opinions. Each line follows a coherent narrative thread, and different lines focus on distinct views of the story. The map structure can be easily adapted to modern

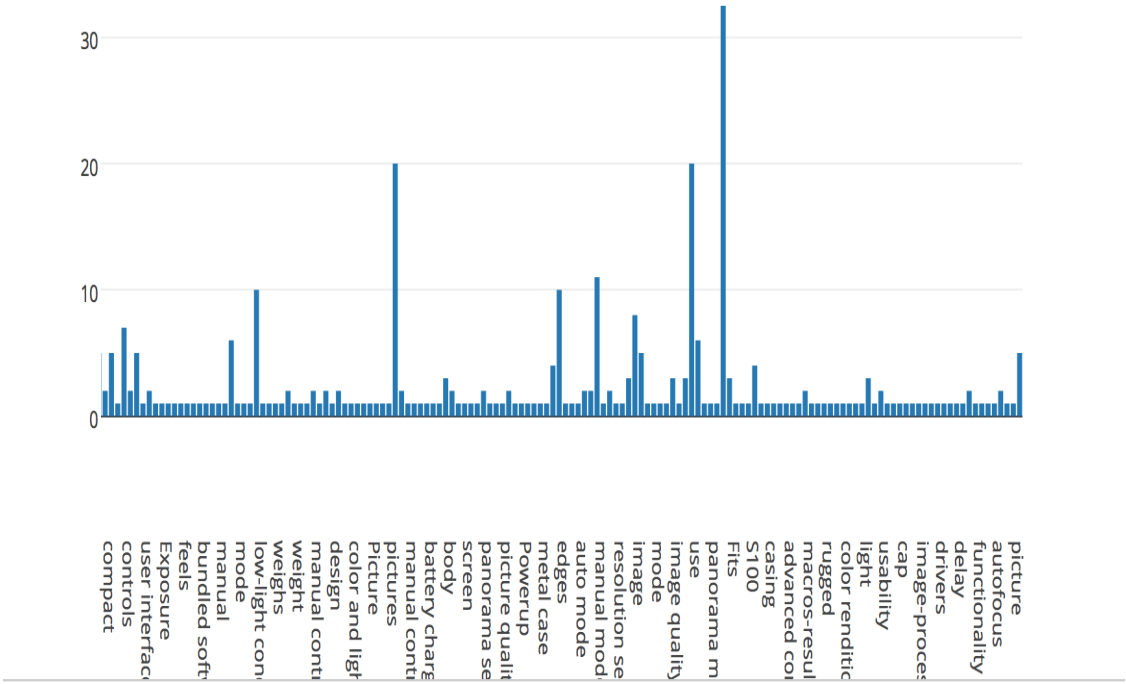


Figure 4.3: Distribution of Extracted Features for a Digital Camera

visualization techniques which can provide a holistic and self-intuitive explanation of the reviews. The map structure adapts most of the essential specifications which are relevant to the opinion.

Opinion Sentiment Orientation

The color of the opinion nodes gives the holistic interpretation at first glance. The colors define the aspect-based orientation of each opinion. We use the green color for the positive sentiment, red color for the negative sentiment, and yellow for the neutral ones.

Frequency

The frequency of the positive and negative comments on the aspects can have a significant impact on the potential buyers. The map structure can define the frequency of each aspect based opinion to address this need. As illustrated in Figure 4.4, the frequency is defined in the vertical axis and ranges from minimum to the maximum frequency count for all of the selected opinions.

Evolutionary Time line

The customer's comments on a product can have some temporal trend (e.g., getting better with time). We assume that the posted date of a review gives the more reliable measure of the trend. For instance, a review posted four years back and focused on the cost of shipping service might not be relevant in the current context of better or even shipping service. The generated map also helps to track the changes in the product features over the time. The proposed framework chronologically presents the selected review/summary for each time slot as shown in Figure 4.4.

Review Rate

Review Rate (h) is a rating measure on one or several topics of a review. On Amazon, this is the percentage of the users who found this review helpful. On Yelp, the review rate has three fields: usefulness, coolness, and funniness. The rating indicates the importance of discussed opinions in a review. This can be realized using the following relation:

$$D_{node} \propto H_{opinion} \tag{4.13}$$

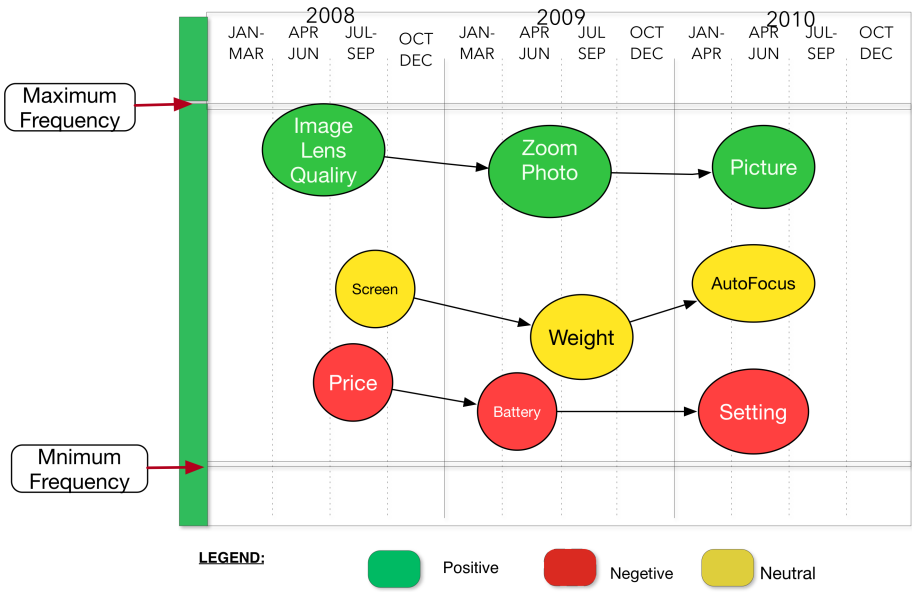


Figure 4.4: *RevMap* for Digital Camera Reviews

, where D is the diameter of the circle (an opinion node) and H is the helpfulness rating of a review.

Connection of Nodes

Coherence and self-explanatory representation of the maps is one of the major goals of our framework. This goal is achieved by the connection between the closely related nodes (for instance, Image quality and picture) via the evolutionary timeline. We use the Okapi BM25 retrieval model[RWJ+95] for paradigmatic relation mining to measure the relevance of nodes and connect them.

The framework generates a visual summary (illustrated in Figure 4.4) based on the corpus of reviews.

4.4 Experimental Setup

4.4.1 Data

We selected MPQA 1.2 corpus [WWC05] as the evaluation dataset. It was publicly available and had 535 news articles. There were 11,111 sentences annotated with both Direct Subjective Expression (DSE)s and Expressive Subjective Elements (ESE)s at the phrase level. We randomly selected 80 percent of the data as the training data, 10 percent as the validation data, and the remaining 10 percent as the testing data.

4.4.2 Character Set Embedding

The character set included the English alphabets, digits, and some of the special characters. There were 70 characters in total which are listed below:

```
abcdefghijklmnopqrstuvwxyz0123456789
-,;.!?:’’’/\|_@#$%ˆ&*˜‘+-=<>()[]{}
```

The characters in reviews were encoded using one-hot vector $x_i \in \{0,1\}^{70}$.

We applied L_2 regularization to the weighted matrices and applied the method of Dropout [SHK+14] with $p = 0.5$ for the hidden layers. The weight parameters were initialized using the same approach as in [GB10]. The optimization was done with the Adam optimization method [KB14] with a learning rate of 10^{-3}. We trained all the models for ten epochs and selected the model that did best in the validation set.

Table 4.1: Experimental Evaluation for ESE Extraction

Model	**Proportional Overlap**
Shallow RNN	51.34
Deep RNN	51.13
Our Model	53.45

4.5 Results and Discussions

The experiments were conducted for opinion mining, and the results were compared with shallow and deep RNN [IC14] models as shown in Table 4.1. We used soft notation proportional overlap [JM10] for performance evaluation.

The generated map (see Figure 4.4) can be more informative because it can convey the crucial information such as opinions, sentiment, and helpful rate of reviews. Additionally, it can also provide that information which was not intuitive in the original reviews. For instance, some of the valuable information such as the frequency of each opinion, and trend of the opinion in reviews are self-explanatory in the generated map and can be easily adapted for a useful review summarization. The generated map can quickly convey the statistical information of a substantial review corpus which is not possible with the traditional approaches of review selection and review summarization.

4.6 Summary

In this chapter, we presented *RevMap*, a novel method for generating the visualized summary of reviews using a character-level bi-directional GRU encoder-decoder architecture. To the best of our knowledge, our work is the first attempt to address the problem of creating a holistic view of reviews.

CHAPTER 5

TAGGING ADDRESS QUERIES IN MAP SEARCH

5.1 Introduction

Map search has become an integral part of our everyday experience - users search for locations to visit or just for information. On mobile devices, Maps are among the most frequently downloaded and used apps [Com17]. One of the main categories of queries to map search is "address queries". We use this term loosely to refer to a broad set of address patterns - cases where either the entire query or part of it contains address reference. The address reference itself can constitute a complete or *partial* reference to a *point address*[Mok18]. Some common address query patterns which users issue and expect map search to resolve are:

- {*123 Main St, San Francisco, CA 94105*} — complete point address query.
- {*30 Rockefeller Plaza*}, {*350 5th Ave, New York*} — partial queries, with city, post code or other information missing.
- {*Pennsylvania Ave Washington DC*}, {*Fremont, Seattle*} — road or neighborhood queries, missing exact address.
- {*Quai Branly et Avenue de la Bourdonnais*} — intersection queries.
- {*Bakery near Castro Street, Mountain View*} — business or place with road as location reference.

Address queries are fulfilled by a special search engine known as *geocoder* [BET+15]. Similar to web search engines, geocoders map the terms of the query to certain documents, in this case, known as *geo entities* which are subsequently formatted and returned as results with their associated latitude and longitude. Unlike web search,

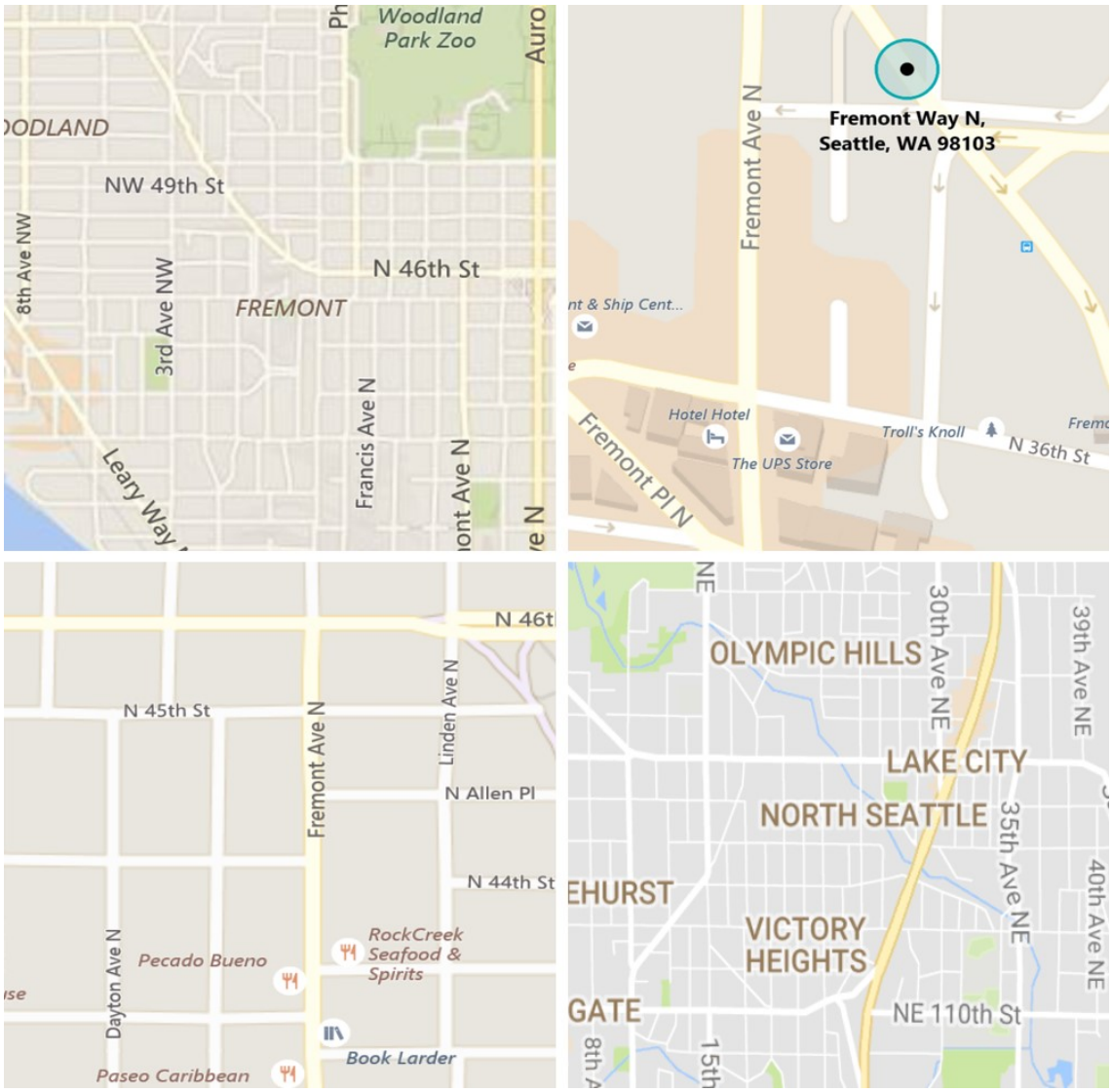


Figure 5.1: Depending on the query interpretation the geocoder may infer different results. *Top*: Possible result for the query is {*Fremont Seattle*}. *Bottom*: Possible result for the query is {*Fremont Ave North Seattle*}

however, geocoders are expected to return very limited (usually only one) results with very high precision. The cost to users of returning a bad result can be very high. For example, they may use the result as driving direction and end up in a wrong place and incur time or other loss. Therefore having an accurate interpretation of address queries is of critical importance to geocoding and map search in general.

Tagging terms in an address query correctly turn out to be a challenging problem. On the one hand, it depends heavily on the user language and market; on the other hand, even in the same language, there might be different query interpretations that lead to different geocoded results. The interpretations will also impact how confident is the geocoder in the identified result [BET+15]. This is important because geocoding confidence is often used by consumer system, e.g., navigation systems, to decide whether to accept the result or to inform the user that no location was found matching their query which leads to decrease in recall and poor user experience. As an illustration, consider the following two queries:

Query 1 {*Fremont Seattle*}. There are several possible annotations for this query: 1) Neighborhood: Fremont, City: Seattle; 2) Street name: Fremont, City: Seattle; 3) Business: Fremont (e.g., colloquial for Fremont Brewery), City: Seattle. Depending on which interpretation we assume most likely, the geocoder may infer different results with different confidence (in the second interpretation the confidence may be lower as there are multiple equally likely candidates for the road "Fremont" — Fremont Way, Fremont Pl, and Fremont Ave) — see Fig 5.1 top.

Query 2 {*Fremont Ave North Seattle*}. Interpretations: 1) Street: Fremont Ave North, City: Seattle; 2) Street: Fremont Ave, Neighborhood (colloquial): North Seattle (see Fig 5.1 bottom). In the second interpretation, the geocoder will not

find a road with such name in such neighborhood and may simply return the up-hierarchy result of North Seattle. Hence, it is central to tag each token of address queries correctly.

In this chapter, we give an overview of address queries, and the formal definition of tagging address queries in Section 5.3. We then outline the different model architectures for our problem in Section **??**. Finally, we discuss the main results and experiments of this problem in Section 5.5.

5.2 Address Parser Training

Address parser training consists of five important steps listed below. The detailed description for each step listed below. The main challenge for training an address parser is data. Collecting the data and labeling them is really time-consuming. In this chapter, we only focus on address queries in united states in English (en-US). The process of address parser training is depicted in Figure 5.2.

- Data Collection: in this step, we need to generate synthetic data for training data and also labeling data collection for validation data.
- Data Cleaning: Normalization of the collected data and mapping data to the labeled address parser tags. There are no tagged address queries; therefore, we need to have human judges to label them. Even after labeling by a human, data is still noisy. Each query labeled by more than one judge to have more accurate tags/labels for each address queries.
- Model Training: After all the data engineering work, training the model on the collected and cleaned data.

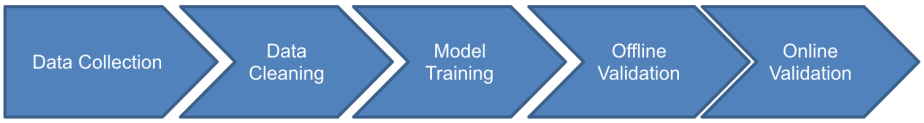


Figure 5.2: Address Parser Training Process

- Offline Validation: validating the trained model on new data and analysis of the model using human labeled data. It needs iterating on features and data gaps to find the best model.
- Online Validation: validating the model offline is not enough. The true validation is when a model is on the production system and validate it with real queries.

5.3 Tagging Address Queries

By analyzing large data sets with address queries from an industrial map search engine, we observe that certain aspects of the problem make correctly tagging such queries a non-trivial task. The following is a partial list of the challenges:

- Data is unstructured (or semi-structured), with irregularities or omissions;
- Data is noisy and may have typos and abbreviations;
- Tagging is market[1] dependent, with a large number of tags present in each market (see Table 5.1 for en-US);
- Small number of head terms and a large number of tail terms;

[1]A market is defined by language and country, e.g., fr-CA are French queries issued in Canada.

- Data is sparse: unlike web search queries, most address queries appear only once in the logs.

The above issues are further exacerbated by the shift that conversational interfaces in personal assistants (e.g., Cortana, Google Now, Siri, Alexa, etc.) allow users to submit increasingly longer and more colloquial queries. So not only the queries are sparse and form a long tail, but also contain multiple irrelevant terms which need to be identified and tagged correctly.

To build a system that learns to interpret correctly address queries, we first focus on understanding how many distinct geo-entities are present in them. In doing so, we identify two types of queries: Single point (SP) — queries that contain a single entity (point), and Multipoint (MP) — queries that contain references to multiple geo-entities (points). We now go into more details for each of these categories.

For convenience, in Table 5.1 we summarize some of the tags that our models are trained to identify. The table only shows a limited number of tags. In en-US addresses, judges identified more than 20 tags.

5.3.1 Single Point Queries

Single point address queries have a standard format which is mostly used by national postal service of each country. The format of the address queries greatly depends on the country. Figure 5.3 shows an example of an SP address query for the United States. The query contains only one geo-entity, in this case, a fully-qualified complete address point.

Abbr.	Tag Description
HN	House Number
SBT	Sub address Type (Building, Tower)
BN	Building Name
SD	Street Direction
ST	Street Type (Ave, St, etc.)
SN	Street Name
CI	City
CO	Country
ST	State name
N	Neighborhood (Kirkland)
ZP	Zip Code
OT	Occupancy Type (Floor, Suit, Apt)
SP	Separator (near, by, in , etc.)
B	Business (Starbucks,Walgreen's, etc.)
UNK	Unknown (Not related to address)

Table 5.1: Abbreviation and description of existing tags in both multi point and single point address queries.

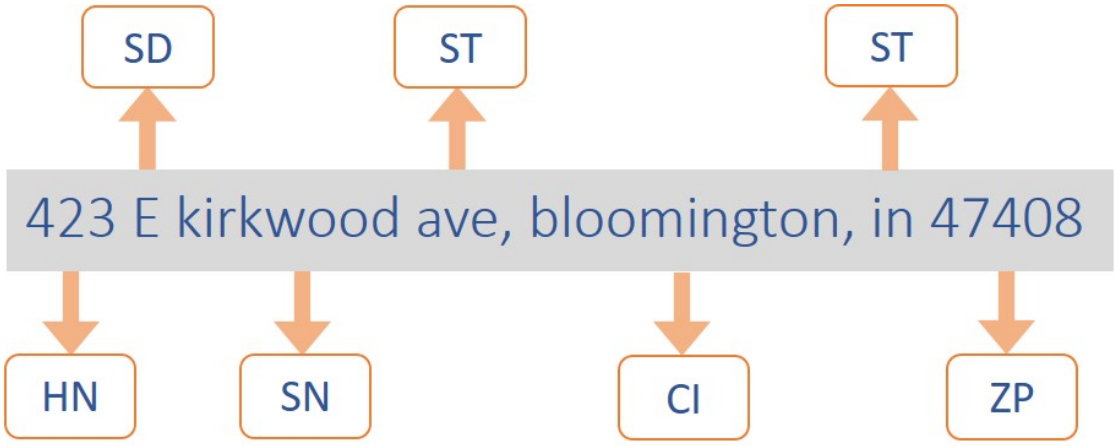


Figure 5.3: Example of a Single Pointer query. The result (and the query) identifies one address point entity. One or more terms may be missing from the query.

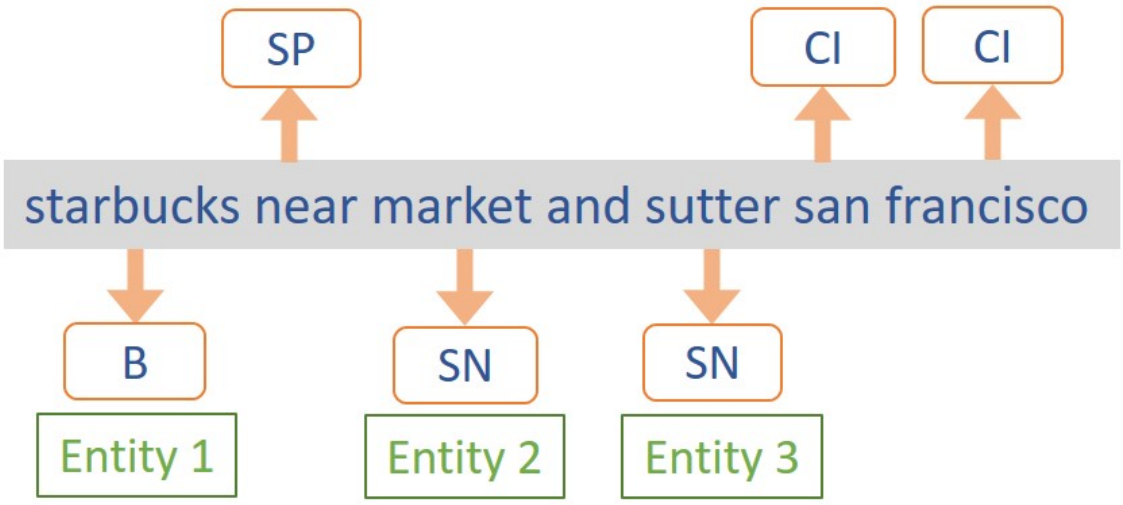


Figure 5.4: Example of a Multi Pointer query. The *where* part after the separator can be a full or a partial address, or it can contain multiple entities itself.

5.3.2 Multi Point Queries

MP queries contain terms that identify multiple geo-entities. Some of them define the expected result and others define points of reference. For example, many MP queries follow the pattern:

[what][separator][where]

in which the "[where] part" is used as a reference. Referential queries are a very common way through which users specify addresses in some countries, for instance, India [BET+15]. Figure 5.4 shows one such MP query where a user is asking about a particular "Starbucks" (entity 1 — business) that is close to an intersection (reference point). The intersection is comprised of two geo-entities — "Market Str" (entity 2) and "Sutter Str" (entity 3) in San Francisco. The intersection is another common MP query pattern.

5.4 Sequence Tagging Architectures

Sequence tagging is a well-studied task in NLP including named entity recognition (NER), chunking, and part of speech tagging (POS). Most of the existing approaches are probabilistic in nature such as Hidden Markov models (HMM), Maximum entropy Markov models (MEMM) [MFP00], and Conditional Random Fields (CRF) [LMP01]. There are several neural network based approaches to address the sequence tagging task [CWB+11, MH16]. In this section, we will briefly describe the neural network architectures that have been used for sequence tagging tasks.

In this work we experimented with three categories of architectures:

One-Directional This general structure consists of (i) an embedding layer, (ii) a forward recurrent cell, and (iii) a fully connected layer [Kaw08]. For the recurrent cell, we have two choices which result in the following different architectures:

- **Forward-RNN:** The one-directional architecture that uses the vanilla RNN as its recurrent cell.
- **Forward-LSTM:** The one-directional architecture that uses the LSTM as its recurrent cell.

Bi-Directional Our general architecture for bi-directional structure consists of (i) an embedding layers of words, (ii) a forward recurrent cell applied to the input sequence, (iii) a backward recurrent cell applied to the input sequence, and (iv) a fully connected layer applied to the concatenation of the forward and backward recurrent cells [Kaw08]. Similar to the one-directional structure, we have two choices for the recurrent cells, and each gives rise to a different architecture:

- **Bi-RNN:** The bi-directional architecture that uses the vanilla RNN as its recurrent cell [Pin87].

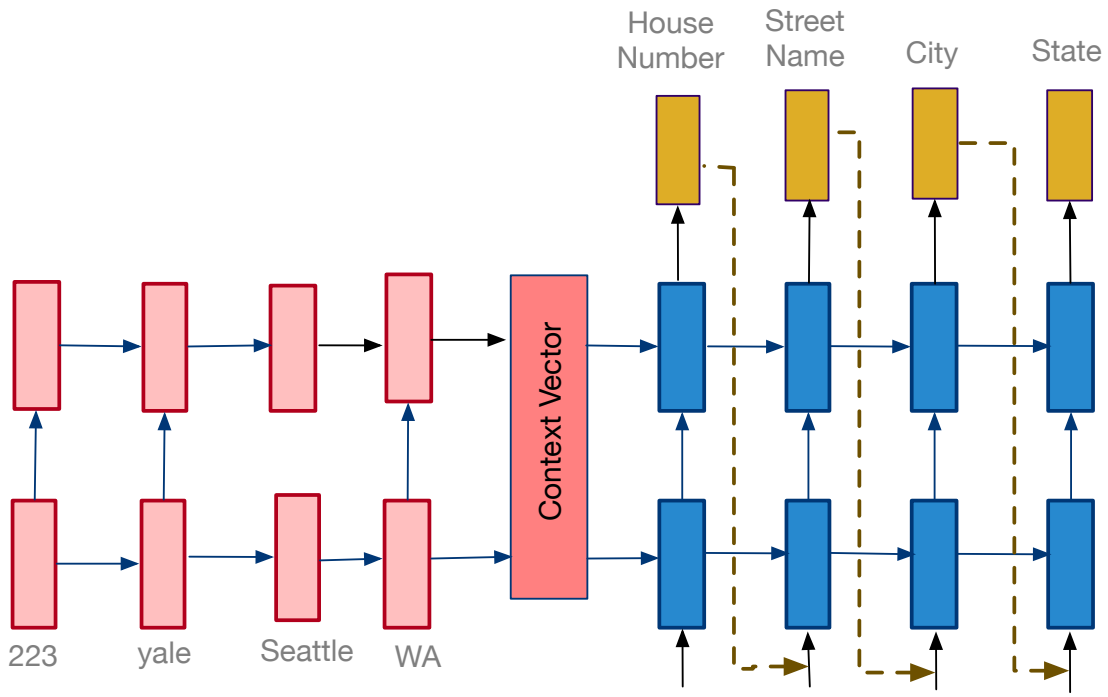


Figure 5.5: Sequence to Sequence model architecture for sequence tagging task. The light red boxes are LSTM encoder and The blue boxes are LSTM decoder.

- **Bi-LSTM:** The bi-directional architecture that uses the LSTM as its recurrent cell [HS97].

Sequence-to-sequence We also try the sequence-to-sequence model introduced in [HS97] that has shown great success in neural machine translation (NMT), speech recognition, and text summarization [SVL14].

Hierarchical Model We propose a hierarchical neural network for parsing address queries in the map search. Given a query containing address data, our model employs gated recurrent units on both character and word levels to encode morphology and context information and applies a conditional random field layer to predict the corresponding label for each token.

Our end-to-end neural network model has two layers for embedding input: character-level and word-level. A character level convolution neural network [ZZL15] takes a sequence of characters (represented as embedding) as input and outputs a representation that encodes the morphological information at the character level. A word-level layer subsequently combines the character-level feature representation and a word embedding and further incorporates the contextual information to output a new feature representation. The feature representation output by the word-level layer is fed to a conditional random field (CRF) layer that outputs the label sequence. The first layer is based on temporal convolution network, and the word level layer is based on bidirectional long short term memory (LSTM). Our neural network architecture is based on the sequence tagging model proposed by in [MH16]. A general overview of our architecture is illustrated in Figure 5.6.

A conditional random field is simply a conditional distribution $p(y|x)$ with an associated graphical structure. Because the model is conditional, dependencies among the input variables, x do not need to be explicitly represented, offering the use of rich, global features of the input.

Two characteristics in this architecture make it works really well for address queries. First, the hierarchical structure exploits both character and word level information from address text data. Second, the two-level structure captures well the dependency between tokens in the address query: the LSTM layer handles longer dependencies and CRF layer deals with shorter dependencies. The combination of these two layers makes the system accurately choose the suitable tag.

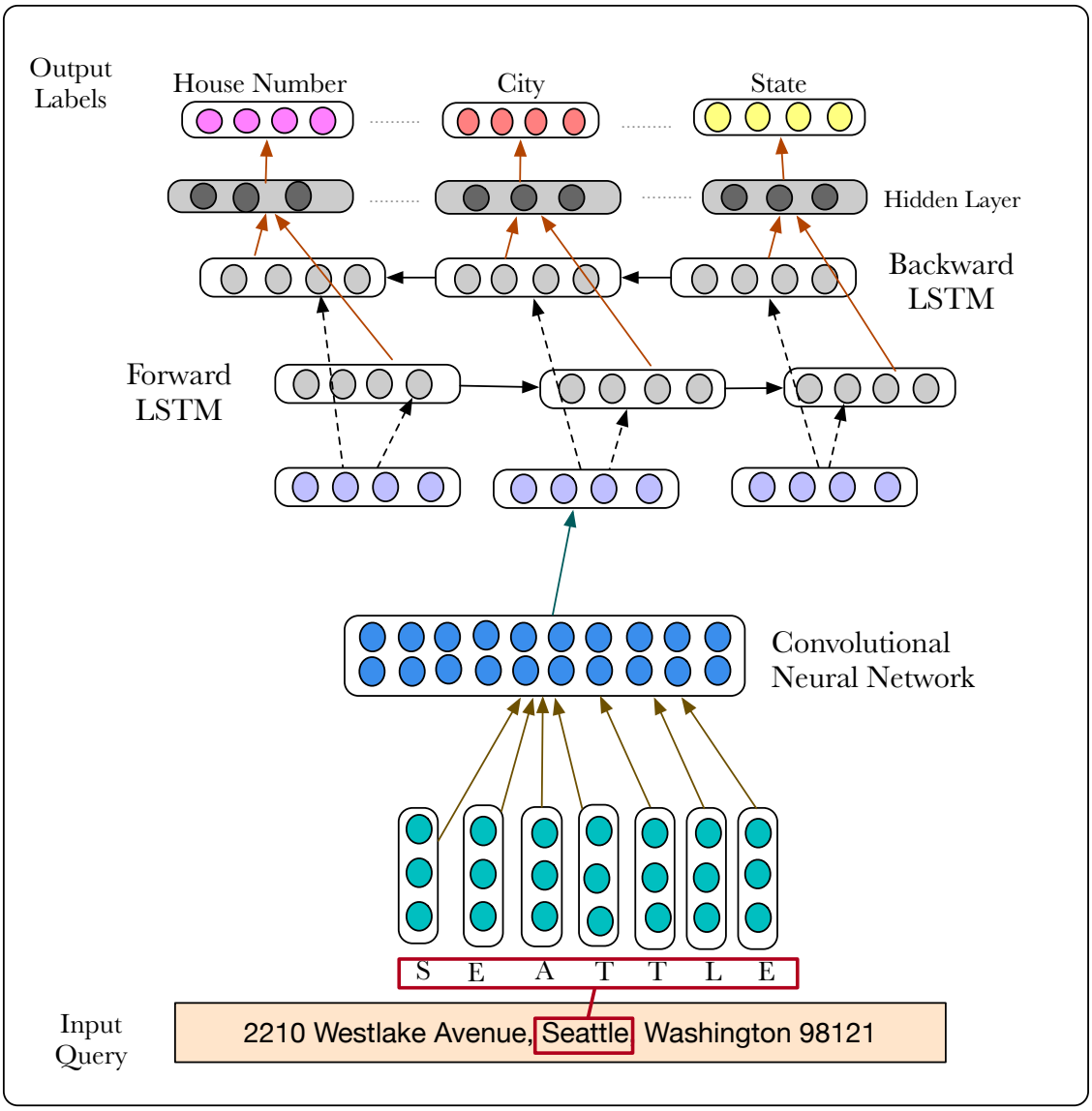


Figure 5.6: Our hierarchical neural network architecture for parsing address queries

	en-US(SP)	en-US(SP & MP)	Yelp SP	Yelp MP
#Train	180K	770K	100K	1M
#Validation	12K	180K	2.5K	100k
#Test	42K	50K	4K	200K

Table 5.2: Data statistics of the real and synthetic datasets.

5.5 Experiments

To train the models, we implement all the models in Microsoft cognitive toolkit (CNTK) 2.0[2]. All the experiments are performed on a single GPU machine. We run all the models up to 256 epochs and select the model that achieves the best accuracy on the validation set. We use hidden size $h = 128$ and optimization is carried out using *Adam*, with a fixed learning rate of 0.1. In training our neural networks, we only keep the frequent $|V| > 5$ words and map all other words to a *UNK* token. In order to deal with the segmentation problems in queries such as *Las Vegas*, we transform them into BIO encoding.

5.5.1 Data Collection

We conduct all the experiments on both en-US real and synthetic data. The synthetic data sets have been generated from the sheer volume of local business information available online in Yelp academic data set[3]. Yelp dataset includes information about $156k$ local businesses from 11 metropolitan areas across four countries. For our experiment, we only keep the united states addresses which contain about $100K$ local businesses. We create a single point (SP), and multipoint (MP) address queries from local business information such as address, neighborhood. Table 5.2 provides some statistics on the two datasets.

[2]https://github.com/Microsoft/CNTK

[3]https://www.yelp.com/dataset/challenge

Real Data. These queries are collected from logs and labeled by human judges as SP and MP queries. We create two separate datasets including *en-US (SP)* which purely contains SP queries and *en-US (SP & MP)* which consists of both single and multi-point queries in order to mimic the real queries in the map search.

Yelp Single Point (SP). To generate synthetic single point address queries, we extract the address of all available businesses in United States and employ Parserator [4] for parsing unstructured address strings into address components.

Yelp Multi-Point (MP). To generate the Yelp MP queries we follow the above mentioned common MP pattern:

[what][separator][where]

Let us term the entities in the [where] part *primary entities* and in the [what] part *secondary entities*. The secondary entities mostly involve business name or business category from the address fields of Yelp, and the primary entities are neighborhood, city, and businesses or roads near the secondary entities. We employ different patterns for generating MP queries such as business near the road, a business near business, a business near a place, a road near a road, etc. Then, we apply some perturbation techniques in order to make the generated query looks like the real data. Data and more details will be publicly available.

We also prepare a list of the separators, including near, by, around, in, etc. We then shuffle the entire generated data and divide them into train, development, and test dataset of sizes 98 We added some noise and removed some percentage of each token form entities with specific probability from each entity or connector. For instance, “peet's coffee shop” could be “peet's coffee” or “peet's”.

[4]https://parserator.datamade.us/usaddress

Model	en-US(SP)	en-US(SP & MP)	Yelp(SP)	Yelp(MP)
F-RNN	89.46	72.16	96.97	98.24
Bi-RNN	98.48	96.14	**98.44**	99.68
F-LSTM	90.09	73.02	97.09	98.23
Bi-LSTM	98.77	96.69	98.39	**99.69**
Seq2Seq	**99.17**	**97.50**	98.22	**99.69**

Table 5.3: Per query tagging test accuracy of all models on Yelp and en-US datasets, where en-US is a dataset with real English address queries from United States, and Yelp(SP) and Yelp(MP) are datasets with generated address queries from Yelp dataset.

5.6 Evaluations and Results

Since geocoders require highly accurate tags for address queries, we evaluate the performance of models per query and not per entity. We compare the performance of RNN based model such as RNN, Bi-RNN, the bi-direction RNN; LSTM, Bi-LSTM, and sequence to sequence model. We carefully generated address queries from Yelp dataset for our experiments, one purely single point queries, and the other one only multi-point queries. Our experiments lead to novel insights and practical advice for building and extending tagging address queries. The sequence to sequence models performs well on real data (en-US) according to the results in Table 5.3. For pure single point queries (Yelp SP), Bi-RNN performs slightly better than sequence to sequence model.

We do not have access to Bing address query data for evaluating our hierarchical model, and we can only evaluate its performance on generated Yelp address queries. The results are summarized in 5.3. Table 5.5.

Model	Query: "223 yale seattle"
F-LSTM	{*HouseNumber*, *StreetName*, *StreetName*}
Bi-LSTM	{*HouseNumber*, *Unknown*, *City*}
Correct Tagging	{*HouseNumber*, *StreetName*, *City*}

Table 5.4: An example of tagging address query with trained model on en-US dataset.

Model	Yelp(SP)	Yelp(MP)
RNN	96.97	98.24
Bi-RNN	98.44	99.68
F-LSTM	97.09	98.23
Bi-LSTM	98.39	**99.69**
Seq2Seq	98.22	**99.69**
Our Model	**98.99**	**99.69**

Table 5.5: Per query tagging test accuracy of all models on Yelp datasets. Yelp(SP) and Yelp(MP) are generated address queries from Yelp dataset.

5.7 Summary

In this chapter, we investigate applying hierarchical neural network models for parsing and tagging address queries in the map search. The hierarchical structure is used because address queries are noisy in nature and contain typos and misspelling. We examine strictly real queries and categorize them into two different groups: single point queries and multipoint queries. The insight of having two different types of address queries helps for the design of the general model. We conducted experiments with both types of queries to examine which model is suitable for what kind of the queries. Since real-world example queries are a mixture of both single point and multipoint queries, we need to design a system that works well for all queries. We then investigate several RNN based models for this task. Our experiments show that our hierarchical model performs well on generated address queries from Yelp academic dataset. Our extensive experiments with real data from Bing Map reflects the excellent performance of the sequence to sequence model. For future work, we

are going to also run experiments with other markets and will seek a global model which can detect and label queries from the various market.

CHAPTER 6

LEARNING SIMILARITY OF SHORT TEXT PAIRS

6.1 Introduction

This chapter studies the linguistic or semantic similarity among text documents to which some academic works referred to as *paraphrase identification*. Learning semantic similarities is fundamental to many natural language processing tasks such as machine translation [MLS13], question answering [JCL05, XZS16], and query document pair [NFH+96]. Driving semantic similarity has a broad range of applications such as question-and-answering forums [CWL+08], personal digital assistants [BCG+17] and document retrieval in recommendation tasks or search engine [PB07, BYHM04]. For instance, in question-and-answer (QA) forums which are ubiquitous on the Web, there are vast numbers of duplicate questions. Detecting the repetitive questions and consolidating their response will increases the efficiency of the forums. Another example is finding the user expertise in the recommendation tasks. To learn about each user expertise, we can compare the similarity between the current item and the past items that the user clicked or bought [TPN+17].

In the most scenarios of learning semantic similarity, the data is not labeled and labeling a large amount of data needs human judges which is both expensive and domain dependent.

In this chapter, we will first present background concepts such as the semantic similarity measurements and formal definitions in Section 6.2.1. Then we describe the details of our model and the evaluation process in public data sets in Section 6.5. Finally, we compare the performance of the trained and tuned model on benchmark datasets in Section 6.8.

6.2 Background

6.2.1 Similarity Measurements

Similarity or distance measurements reflect the likeliness of two objects, or in some cases, it is the separation from the target object [Hua08, GF13]. We will stick to the closeness between objects in this study and will briefly review a few of the common measurements such as Manhattan distance, euclidean distance and cosine similarity.

Manhattan Distance

Manhattan distance has been named because of grid structure in the Manhattan metropolitan area. The Manhattan distance is also known as L1 distance or L1 norm, which is the distance between two points measured along their axes. It can also be easily generalized to higher dimensions. Following formula is for calculating the Manhattan distance between two vectors A and B (or points) over their dimensions.

$$\sum_{i=1}^{n} |A_i - B_i| \tag{6.1}$$

Euclidean Distance

Euclidean distance (L2 distance) is the stand-alone distance metric in geometry. It is the most popular distance metrics in machine learning methods and also applied as the default metric in the K-means algorithm and frequently used metric in the text clustering methods.

$$\sqrt{\sum_{i=1}^{n} (A_i - B_i)^2} \tag{6.2}$$

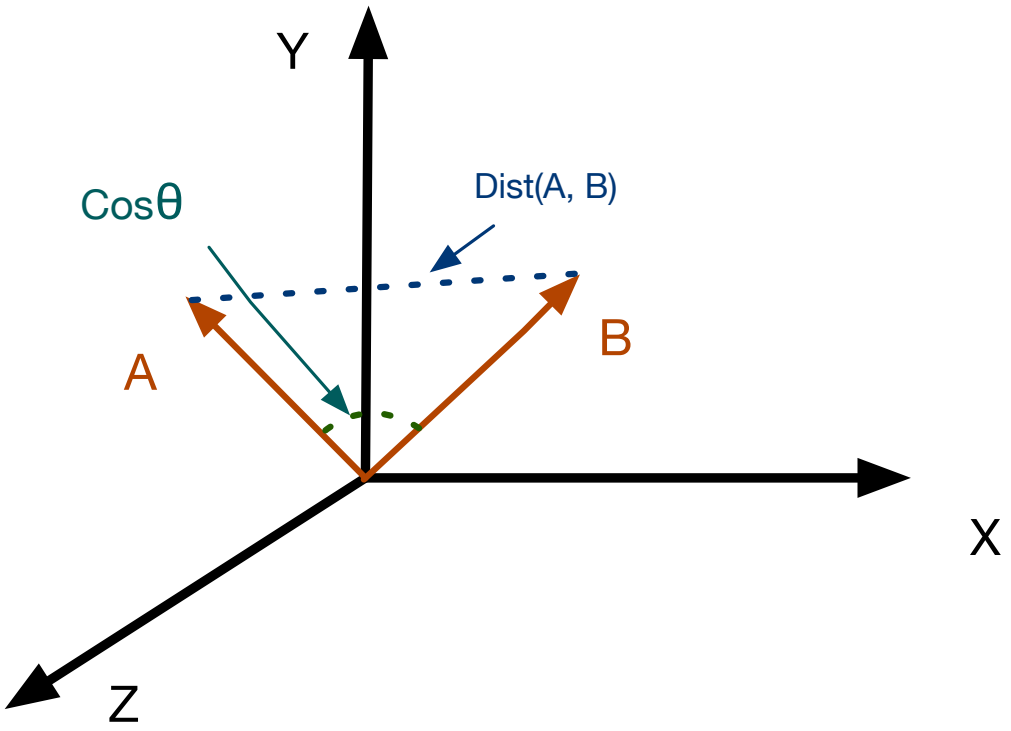


Figure 6.1: The difference between cosine and euclidean similarity in vector space.

Cosine Similarity Measure

Cosine similarity is generally for measuring the distance or correlation between two non-zero vectors. It measures the distance by computing the cosine of the angle between vectors. Therefore, it is suitable when the magnitude or weight of vectors does not matter. The difference between cosine similarity and Euclidean distance in vector space illustrated in Figure 6.1. The formula in equation 6.3 produces a single scalar that quantifies the relatedness of two words or text snippet vectors. Cosine similarity is one of the most popular measure applied to text documents information retrieval applications [BY99] and clustering [LA99].

$$\cos\theta = \frac{A \cdot B}{|(A)|\,|(B)|} \tag{6.3}$$

6.3 Semantic Text Similarity

There are two main methods for approaching the text similarity problem. The first method will see the problem as a binary classification in which for each pair of question we will train a binary classifier and classify it as either positive or negative. Positive means that each sentence of the pair is conveying the same meaning, and the negative is the other way around it. The main idea of this approach has come from Severyn's paper in which they are training convolution neural network to rank the short text pairs [SM15]. We modify their ranking model and loss function to make it applicable for our semantic text similarity problem. More detailed is described in section 6.4.

The second approach is based on the similarity metrics calculation among documents, questions and more generally text data. To be able to apply similarity metrics that have been discussed in 6.2.1, we have to find a way to convert text data such as "Life is beautiful " to numbers or vectors. There any various ways to convert them to numerical values, we will explain some of the popular one in the section **??**. After learning proper representation of text data, we would be able to employ the similarity metrics. Among all the existing semantic similarity measure, we employ the cosine similarity to our problem, which is mostly used in the text similarity cases. Additionally, it is well suited to our problem since the latent representation may have been calculated differently and might not have the same dimension or magnitude. The latent representation (embedding) explored in section 6.5.

6.4 Binary Classification for Semantic Text Similarity

The basic notion behind this model is to learn the similarity function between the given pairs of sentences or question during training phase instead of using existing

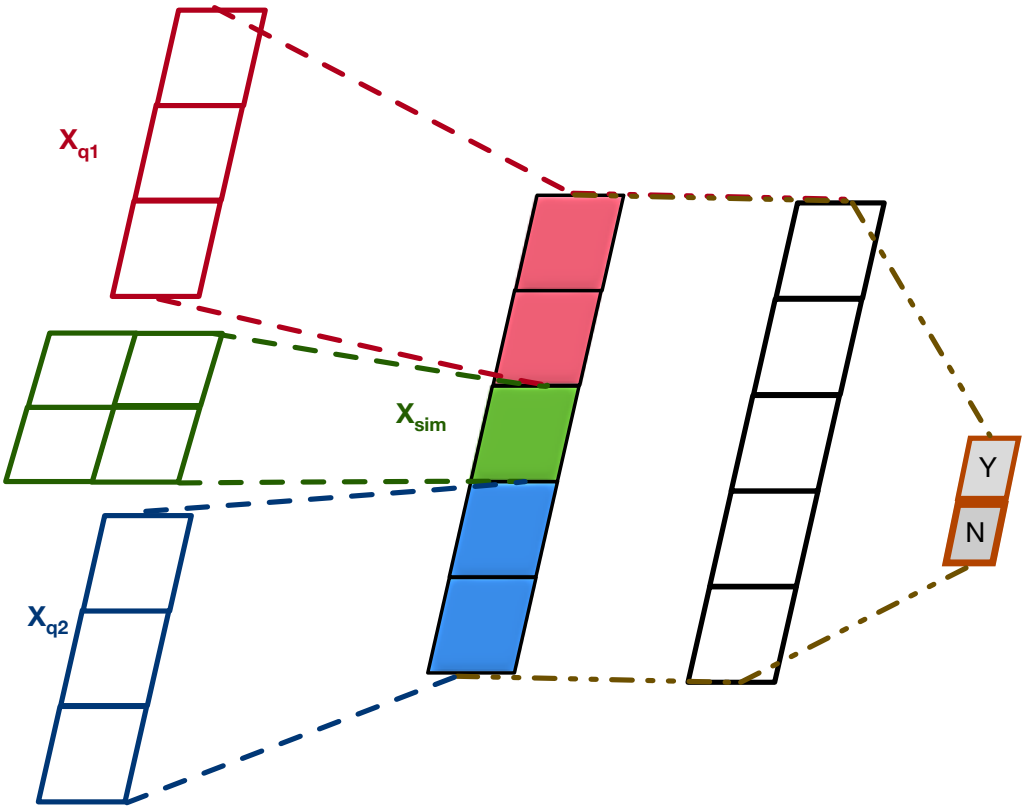


Figure 6.2: This architecture is used for learning similarity function among two given pair of sentences/ questions [SM15]. X_q1 is the vector representation for the first question/sentence and X_q2 is the same for the second one. X_sim is the similarity function that is learned through backpropagation.

similarity metrics. The structure of this model is depicted in Figure 6.2. As shown in this architecture, first the feature extracted form the given text data through convolution filters and the most important extracted features have been chosen with max pooling layers. The output of each extracted feature vectors concatenated with the output of similarity function. In order to improve the results, we also added the similarity metric value to the final representation such as cosine similarity, Manhattan and Euclidean distances between feature vectors. The final concatenated output will be passed to the softmax layer for binary classification.

6.5 Representation Learning

We employed the general model of the auto encoder architecture for learning a generic representation of the input data. The traditional Auto encoder first introduced by Hinton [RHW85]. The model is mostly used for unsupervised machine learning problems by learning to reconstruct input using backpropagation . A The detailed architecture is depicted in the following sections.

6.5.1 AutoEncoder

Autoencoders are a specific type of the neural network that been used for unsupervised machine learning tasks, and they only use x_t as inputs for learning. They extract meaningful features from the data, and also they leverage the availability of unlabeled data. They have been used for dimension-laity reduction [GBC16, HZ94, Lec87]. There are two main components in autoencoder: encoder, decoder. The only constrained in autoencoder is that the output should be similar to the input. The main object of the neural network is to minimize reconstruction loss. The encoder is responsible for encoding input data into a latent representation, which usually has a lower dimension than the input. The decoder is also a neural network which generally has the same architecture as an encoder which takes the feature vector from the encoder, and gives the best closest match to the actual input or intended output. The architecture is depicted in Figure 6.3.

AutoEnocder vs PCA

Principal component analysis (PCA) from theoretical aspect is similar to AutoEncoder (AE). Both of them are trying to minimize the reconstruction error. Employing neural network enable us to benefit from non-linearity transformation by using

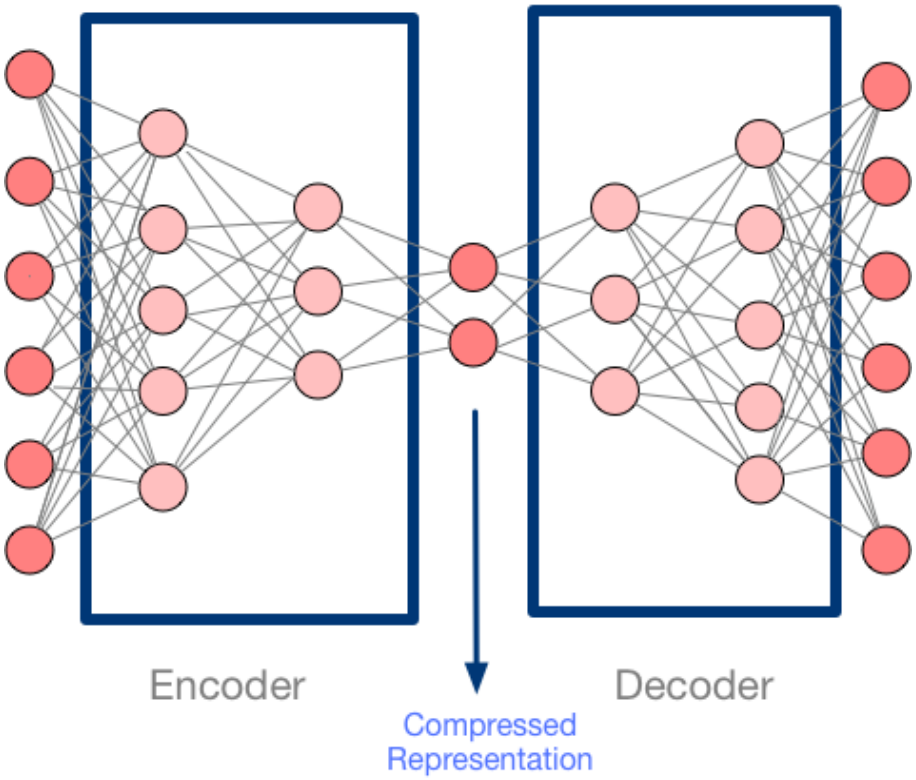


Figure 6.3: AutoEncoder

non-linear activation function such as ReLU (rectified linear unit) or tanh (hyperbolic tangent). These activation functions introduce non-linearity in our encoder while PCA only represents a linear transformation. More than nonlinear transformation, PCA will result in slower processing with an increasing amount of features comparing to AutoEncoder.

6.6 Embedding

Embedding is a general term for language representation by mapping each word to a real-value vector. These vectors treated as input features in a broad spectrum of applications such as text classification [LXLZ15, TQM15, Kim14], information retrieval (IR) [GRMJ15, PDS+16], query understanding [DMC16], image retrieval [CCS+04].

6.6.1 Word2Vec

Google researchers proposed word2Vec, which represents the words as low-dimensional real-value vectors, and the vectors are obtained with the model training process. It contains word vectors for a vocabulary of 3 million words trained on around 100 billion words from the google news dataset. [GL14, MSC+13, MCCD13]

Word2vec described two different model architectures to compute numerical representation (embedding) of words: CBOW (Continuous Bag-of-Words) model and Skip-Gram model. The proposed two kinds of algorithms which could be applied both in CBOW and Skip-Gram: Hierarchical Softmax and negative sampling. The hierarchical softmax is a key technique which improves the performance of Word2vec significantly. Negative sampling is an alternative approach to Hierarchical Softmax

in Word2vec. Negative sampling chooses some negative examples, while the current word serves as a positive example. [Ron14]

6.6.2 Global Vectors for Word Representation (GloVe)

Glove generates word vector representation based on co-occurrence statistics of words in the corpus in an unsupervised manner [1]. It has been trained on Wikipedia, Gigawords, a combination of Wikipedia and Gigawords and words which common crawl from the web. [PSM14]

6.6.3 Deep Structured Semantic Model (DSSM)

Microsoft researchers studied a deep neural network based model for projecting text queries into a continues semantic space and modeling semantic similarity between two text queries (aka Sent2Vec)[PDS+16].

DSSM has been widely applied in a different domain such as information retrieval (IR), ad relevance, question answering, and image captioning. [HHG+13, SHG+14, FGI+15, GHYD14]

DSSM projects the raw text into continuous semantic space. Semantic space is a virtual space where each sentence will be projected into the vector in this space based on the semantic meaning. It provides a useful framework for problems like web search where both query and web documents mapped to a semantic space. Hence, the semantically matching documents for the query can be retrieved.

[1] https://nlp.stanford.edu/projects/glove/

6.6.4 FastText

Facebook researchers proposed an efficient baseline for text classification, Fastext, which is on par with the deep learning classifier in terms of accuracy[JGBM16, BGJM16]. FastText is an approach based on the skip-gram model, where each word is represented as a bag of character n-grams to capture the local word order. The numerical vector representation is associated with each character n-gram; words being represented as the sum of these representations.

6.7 Latent Semantic Analysis (LSA)

Latent Semantic Indexing (LSI) called Latent Semantic Analysis (LSA) is one of the fundamental techniques in natural language understanding for representing the contextual meaning of words by statistical computations applied to a large corpus of text. [Dum04]

LSA uses algebraic technique Singular Value Decomposition (SVD), which is similar to principal component analysis (PCA) [MR93, WEG87]. A truncated singular value decomposition is used to estimate the structure in word usage across documents.

LSA applies SVD to the term-document matrix of the corpus in order to reduce this sparse, high-dimensional matrix to a denser, lower-dimensional matrix whose dimensions correspond to the latent topics in the corpus.

6.8 Experimental Setup

We conduct extensive experiments on publicly available data listed in Section 6.8.1 to compare the performance of different embedding and the proposed architecture

Embedding Models	Training Data	Dimension	Vocabulary
TFIDF + LSA (Baseline)	N/A	300	N/A
Word2vec pre-training model	Google New dataset	300	300M
Glove pre-trained model	Wikipedia + Gigaword	300	400K
FastText pre-trained model	Wikipedia + news	300	100K
MSR DSSM pre-train model	Bing search CTR	300	Not Revealed

Table 6.1: Word Embedding Models Experimented

in Section 6.4. We also compared it with the baseline model using the pioneering embedding techniques LSA. We then experiment and compare performance across multiple state-of-art pre-trained embedding models using multiple semantic benchmark datasets in different domains. Table 6.1 lists the details of the models we have experimented. Section 6.8.1 lists the details of the public benchmark dataset we used.

6.8.1 Data

We extensively conduct our experiments on four publicly available datasets for semantic similarity task consist of Quora question pair similarity dataset, MSR-paraphrase, SICK, and SemEval. We used a public benchmarked dataset to validate our model.

1. Paraphrase Adversaries from Word Scrambling (PAWS): Google AI language researchers recently release a paraphrase dataset which contain both paraphrase and non-paraphrase pair with with high lexical overlap [ZBH19].
2. MSR Paraphrase Dataset: Dataset consists of 5801 pairs of sentences of 18 months from thousands of news sources on the web. Each pair is annotated for semantic equivalence by human judges[2].

[2]https://www.microsoft.com/en-us/download/details.aspx?id=52398

Data	# Q-pairs	# U Q1	# U Q2	# U Q
Train	384,347	279,518	287,628	517,970
Dev	9,999	9,667	9,677	19,056
Test	9,999	9,696	9,674	19,081
Quora	404,351	290,473	299,204	537,390

Table 6.2: Data statistics of the Quora dataset. U refers to unique number of question. The first column refers to the total number of questions. The second and third columns are the unique number of question as first pair and second pair, respectively. The last column is the total number of unique questions in the whole dataset. The last row summarizes the same statistics for the entire Quora dataset.

3. Quora Question Pairs Dataset: Data set consist of over 400,000 of actual question pairs from Quora. Each pair is annotated by human judges for semantic equivalence. Used in Kaggle competition for identifying question pairs of the same intent.

4. SemEval Dataset: SemEval (Semantic Evaluation) is an ongoing series of evaluations of computational semantic analysis systems. Dataset consists of 11,500 pairs of sentences annotated for semantic relatedness from 2012 to 2016.

5. SICK (Sentences Involving Compositional Knowledge) dataset: Dataset consists of about 10,000 English sentence pairs. Sentences were derived from 8K Image Flicker data set and SemEval-2012. Each sentence pair is annotated semantic relatedness and for the entailment relation by crowdsourcing.

We deeply look into data, and the statistics of data analysis is provided in Table 6.2. There is a total number of 404,351 pairs of questions in Quora dataset [3]. We used the same split for train, dev and test dataset as [WHF17]. [4] These statistics will be beneficial for choosing the best sampling method.

[3]https://data.quora.com/First-Quora-Dataset-Release- Question- Pairs.

[4]This split is available at https://zhiguowang.github.io.

6.8.2 Text Data Preprocessing

The universal fundamental principle among all natural language processing tasks is data preprocessing. Text data is a sequence. At the very low-level representation, it is a sequence of bytes or characters, or as higher levels, text can be treated as a sequence of words, phrases, named entities, sentences, paragraphs or bigger chunks.[GBVS15, KSNM03, HXTS16]

Text preprocessing consists of several steps: tokenization, lemmatization, stemming, removing stop words, etc.

The entire process of splitting text into meaningful tokens called tokenization. A token is an independent unit for further semantic preprocessing, and it can be word, sentence, paragraph. Most commonly used tokenization uses white space, punctuation, or grammar rules for separating tokens.

After having tokens, we may need to have the same token for various forms of the word. The two steps in token normalization are stemming and lemmatization. Stemming is the set of rules or heuristics for removing and replacing the end of the word (suffixes) to get the root form of the word. Lemmatization uses vocabulary and morphological analysis to return the base or dictionary form of the word. There are numerous steps of normalization which however are not less important to the overall text data cleaning process that listed below:

- removing text file headers, footers
- removing HTML, XML, etc. markup and metadata
- extracting valuable data from other formats, such as JSON
- set all characters to lowercase
- remove numbers (or convert numbers to textual representations)

Figure 6.4: Text Preprocessing Pipeline

- remove punctuation (generally part of tokenization, but still worth keeping in mind at this stage, even as confirmation)
- strip white space (also generally part of the tokenization)
- remove default stop words (general English stop words)

Stop words refers to words which are filtered out since these words contribute little to the overall meaning. Generally, they are the most common words in a language such as "the," "and," and "a."

Before generating embedding features, we applied a set of a suite of text-preprocessing steps using NLP technology on all questions and text segments to remove noise and reduce dimension, as shown in Figure 6.4.

6.9 Evaluations and Results

We conduct the experiment for evaluation the two discussed model in Section 6.3. We used the cosine similarity between autoencoder's compressed representation of each text segments to measure the similarity between them. Figure 6.6 shows the performance comparison of the above-listed word embedding models and datasets. Autoencoder model used for embedding have the overall best performance in F1-

Methods	Accuracy
Siamese-CNN	79.60
Multi-Perspective-CNN	81.38
Siamese-LSTM	82.58
Multi-Perspective-LSTM	83.21
L.D.C.	85.55
BiMPM	88.17

Table 6.3: Accuracy of paraphrase identification on Quora dataset

measure. Cosine similarity is computed as the measure between each pair of document's embedding.

Among the pre-trained embedding, Microsoft researcher's DSSM (Deep Semantic Similarity Model) performs better than Glove and Word2vec. We evaluate the same model and embedding on SICK and SemEval datasets. We used the Pearson Correlation Coefficient to measure the similarity. The most common measure of correlation in statistics is the Pearson Correlation. The full name is the Pearson Product Moment Correlation (PPMC). It shows the linear relationship between two sets of data. Figures 6.7, 6.8 illustrate the comparison for these two datasets.

6.10 Summary

This chapter elaborately discusses the neural sentence matching problem in natural language processing. We investigate the autoencoder model for text similarity problem and compare the performance with the popular pre-training embedding model. We observed that leaning embedding in each specific domain works better than transfer learning.

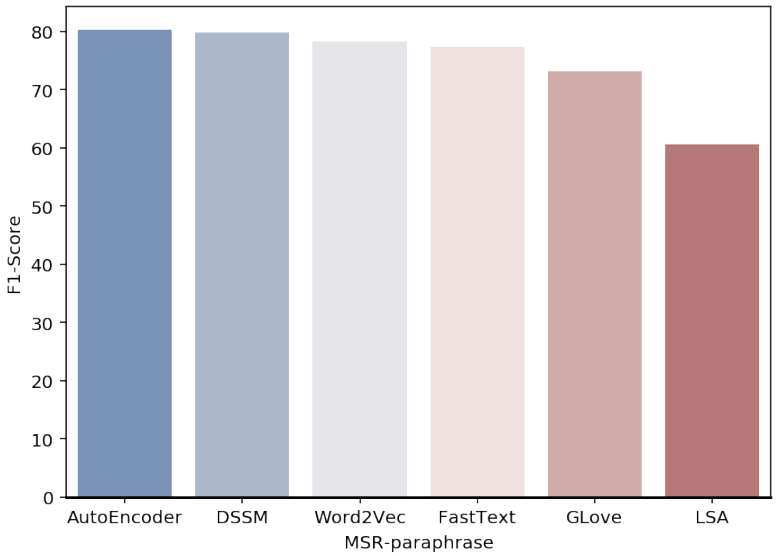


Figure 6.5: Performance of different embedding and auto encoder model on the MSR-paraphrase dataset

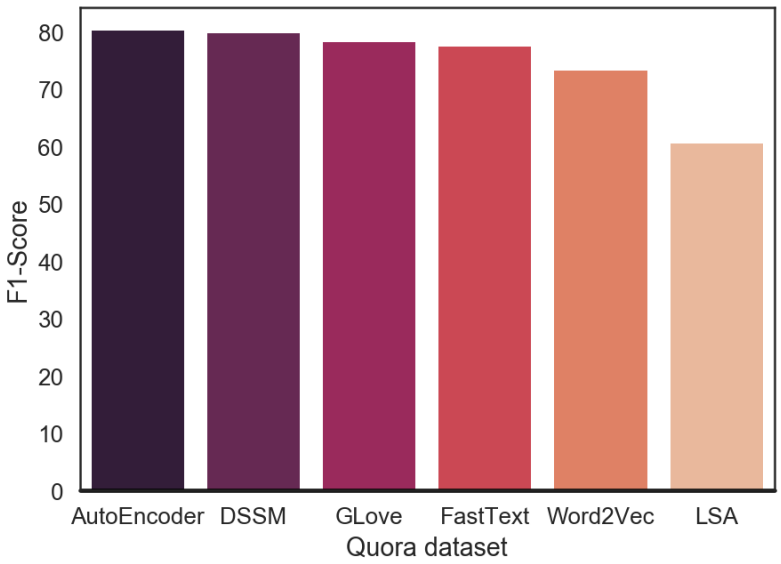


Figure 6.6: Performance of different embedding and auto encoder model on Quora dataset

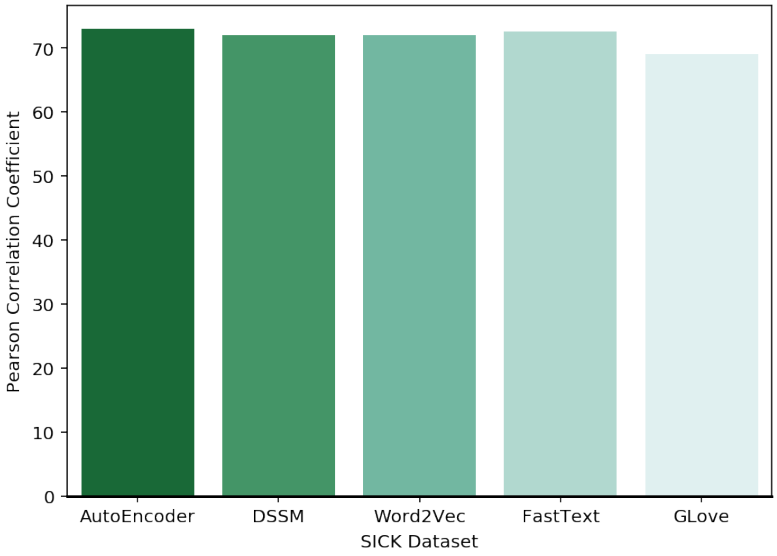


Figure 6.7: Performance of different embedding and auto encoder model on SICK dataset

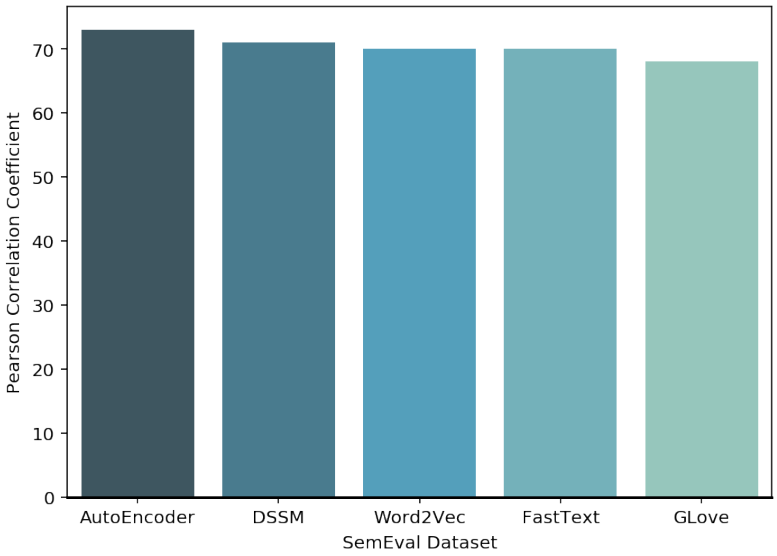


Figure 6.8: Performance of different embedding and auto encoder model on SemEval dataset

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

We are surrounded by text data, from social media to news articles to machine logs. Therefore, it is essential to have an automatic system to process and understand these text data. In previous chapters, we discussed research studies where we applied deep learning techniques to various text mining problems. The three main research directions are highlighted as below:

1. Classifying the documents and more specifically find the sentiment class and visualizing the result;
2. Automatically understanding and tagging each token of the address queries in map search;
3. Neural language sentence matching or more specifically paraphrase identification.

To follow up on the work in my dissertation, some future work along the three directions are provided.

- We focus on context neural sentiment analysis and how to visualize the digest of review corpora. In the future, we will extend these two models to recommend reviews based on user interest and purchase pattern. In addition, more temporal data can be incorporated into the framework to improve the performance of the system and benefit from this valuable knowledge in review corpora in the best possible way.
- For tagging address queries in map search; we are looking to also run experiments with other markets and also checking other complicated neural network architecture to improve the results. We are looking to have one global address

parser which can handle detecting the language and tagging of the queries based on location. Another factor to be applied in the model is to leverage the current location of the user or in general user search pattern to rank the desired results higher.

- In the future, we are looking to apply these text semantic similarity to looking for similar items in recommendation problems or to discover user expertise.

BIBLIOGRAPHY

[AAB+16] Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[BB00] Yoshua Bengio and Samy Bengio. Modeling high-dimensional discrete data with multi-layer neural networks. In *Advances in Neural Information Processing Systems*, pages 400–406, 2000.

[BC+08] Y-lan Boureau, Yann L Cun, et al. Sparse feature learning for deep belief networks. In *Advances in neural information processing systems*, pages 1185–1192, 2008.

[BCG+17] Petr Babkin, Md Faisal Mahbub Chowdhury, Alfio Gliozzo, Martin Hirzel, and Avraham Shinnar. Bootstrapping chatbots for novel domains. In *Workshop at NIPS on Learning with Limited Labeled Data (LLD). https://lld-workshop. github. io/papers/LLD_2017_paper_10. pdf*, 2017.

[BDS01] Vinayak Borkar, Kaustubh Deshmukh, and Sunita Sarawagi. Automatic segmentation of text into structured records. In *ACM SIGMOD Record*, volume 30, pages 175–186. ACM, 2001.

[BDVJ03] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *journal of machine learning research*, 3(Feb):1137–1155, 2003.

[BET+15] Pavel Berkhin, Michael R. Evans, Florin Teodorescu, Wei Wu, and Dragomir Yankov. A new approach to geocoding: Binggc. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '15, pages 7:1–7:10, 2015.

[BGJM16] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[BGWB14] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259, 2014.

[BY99] Ricardo Baeza-Yates. Berthier ribeiro-neto. *Modern Information Retrieval*, 1999.

[BYHM04] Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. Query recommendation using query logs in search engines. In *International Conference on Extending Database Technology*, pages 588–596. Springer, 2004.

[CC05] Vitor R Carvalho and William W Cohen. On the collective classification of email speech acts. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 345–352. ACM, 2005.

[CCLZ02] Tim Churches, Peter Christen, Kim Lim, and Justin Xi Zhu. Preparation of name and address data for record linkage using hidden markov models. *BMC Medical Informatics and Decision Making*, 2(1):9, 2002.

[CCM04] William W Cohen, Vitor R Carvalho, and Tom M Mitchell. Learning to classify email into"speech acts". In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004.

[CCS+04] Tatiana AS Coelho, Pável Pereira Calado, Lamarque Vieira Souza, Berthier Ribeiro-Neto, and Richard Muntz. Image retrieval using multiple evidence ranking. *IEEE Transactions on Knowledge and Data Engineering*, 16(4):408–417, 2004.

[CF87] Stephen Crain and Janet Dean Fodor. Sentence matching and overgeneration. *Cognition*, 26(2):123–169, 1987.

[CGCB14] 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.

[CISSW06] Chaomei Chen, Fidelia Ibekwe-SanJuan, Eric SanJuan, and Chris Weaver. Visual analysis of conflicting opinions. In *Visual Analytics Science And Technology, 2006 IEEE Symposium On*, pages 59–66. IEEE, 2006.

[CKS+17] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.

[Com17] Comscore. Mobile app report, 2017.

[CST+16] Huimin Chen, Maosong Sun, Cunchao Tu, Yankai Lin, and Zhiyuan Liu. Neural sentiment classification with user and product attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1650–1659, 2016.

[CVMBB14] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[CVMG+14] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[CW08] 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.

[CWB+11] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011.

[CWL+08] Gao Cong, Long Wang, Chin-Yew Lin, Young-In Song, and Yueheng Sun. Finding question-answer pairs from online forums. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 467–474. ACM, 2008.

[DDF+90] 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):391–407, 1990.

[DMC16] Fernando Diaz, Bhaskar Mitra, and Nick Craswell. Query expansion with locally-trained word embeddings. *arXiv preprint arXiv:1605.07891*, 2016.

[dSG14] Cícero Nogueira dos Santos and Maira Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, pages 69–78, 2014.

[Dum04] Susan T Dumais. Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230, 2004.

[DZF+16] Bhuwan Dhingra, Zhong Zhou, Dylan Fitzpatrick, Michael Muehl, and William W Cohen. Tweet2vec: Character-based distributed representations for social media. *arXiv preprint arXiv:1605.03481*, 2016.

[Fel98] Christiane Fellbaum. *WordNet*. Wiley Online Library, 1998.

[FGI+15] Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. From captions to visual concepts and back. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1473–1482, 2015.

[FS69] Ivan P Fellegi and Alan B Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969.

[GB10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. *2015*, 2016.

[GBVS15] Dan Gillick, Cliff Brunk, Oriol Vinyals, and Amarnag Subramanya. Multilingual language processing from bytes. *arXiv preprint arXiv:1512.00103*, 2015.

[GF13] Wael H Gomaa and Aly A Fahmy. A survey of text similarity approaches. *International Journal of Computer Applications*, 68(13):13–18, 2013.

[GHYD14] Jianfeng Gao, Xiaodong He, Wen-tau Yih, and Li Deng. Learning continuous phrase representations for translation modeling. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 699–709, 2014.

[GJL+13] Yunchao Gong, Yangqing Jia, Thomas Leung, Alexander Toshev, and Sergey Ioffe. Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894*, 2013.

[GL14] Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

[GMH13] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE, 2013.

[GRMJ15] Debasis Ganguly, Dwaipayan Roy, Mandar Mitra, and Gareth JF Jones. Word embedding based generalized language model for information retrieval. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 795–798. ACM, 2015.

[HDY+12] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[HHG+13] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2333–2338. ACM, 2013.

[HL04] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, 2004.

[HLLC14] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*, pages 2042–2050, 2014.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[HSK+12] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[Hua08] Anna Huang. Similarity measures for text document clustering. In *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand*, pages 49–56, 2008.

[HXTS16] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. A joint many-task model: Growing a neural network for multiple nlp tasks. *arXiv preprint arXiv:1611.01587*, 2016.

[HZ94] Geoffrey E Hinton and Richard S Zemel. Autoencoders, minimum description length and helmholtz free energy. In *Advances in neural information processing systems*, pages 3–10, 1994.

[HZRS16] 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.

[IC14] Ozan Irsoy and Claire Cardie. Opinion mining with deep recurrent neural networks. In *EMNLP*, pages 720–728, 2014.

[JCL05] Jiwoon Jeon, W Bruce Croft, and Joon Ho Lee. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 84–90. ACM, 2005.

[JGB+16] Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, 2016.

[JGBM16] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[JM10] Richard Johansson and Alessandro Moschitti. Syntactic and semantic structure for opinion expression detection. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 67–76. Association for Computational Linguistics, 2010.

[Kaw08] Kazuya Kawakami. *Supervised Sequence Labelling with Recurrent Neural Networks*. PhD thesis, Ph. D. thesis, Technical University of Munich, 2008.

[KB14] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KGB14] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[Kim14] Yoon Kim. Convolutional neural networks for sentence classification. 2014.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[KSNM03] Dan Klein, Joseph Smarr, Huy Nguyen, and Christopher D Manning. Named entity recognition with character-level models. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 180–183. Association for Computational Linguistics, 2003.

[LA99] Bjornar Larsen and Chinatsu Aone. Fast and effective text mining using linear-time document clustering. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 16–22. ACM, 1999.

[LCT12] Theodoros Lappas, Mark Crovella, and Evimaria Terzi. Selecting a characteristic set of reviews. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 832–840. ACM, 2012.

[Lec87] Yann Lecun. Phd thesis: Modeles connexionnistes de l'apprentissage (connectionist learning models). 1987.

[LG10] Theodoros Lappas and Dimitrios Gunopulos. Efficient confident search in large review corpora. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 195–210. Springer, 2010.

[LHC05] Bing Liu, Minqing Hu, and Junsheng Cheng. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of the 14th international conference on World Wide Web*, pages 342–351. ACM, 2005.

[LK97] David D Lewis and Kimberly A Knowles. Threading electronic mail: A preliminary study. *Information processing & management*, 33(2):209–217, 1997.

[LKWS14] Xiang Li, Hakan Kardes, Xin Wang, and Ang Sun. Hmm-based address parsing with massive synthetic training data generation. In *Proceedings of the 4th International Workshop on Location and the Web*, pages 33–36. ACM, 2014.

[LM14] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. In *ICML*, volume 14, pages 1188–1196, 2014.

[LMP01] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[LXLZ15] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In *AAAI*, pages 2267–2273, 2015.

[LZLM07] Ying Liu, Dengsheng Zhang, Guojun Lu, and Wei-Ying Ma. A survey of content-based image retrieval with high-level semantics. *Pattern recognition*, 40(1):262–282, 2007.

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[MCS+06] Rada Mihalcea, Courtney Corley, Carlo Strapparava, et al. Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI*, volume 6, pages 775–780, 2006.

[MDM07] Donald Metzler, Susan Dumais, and Christopher Meek. Similarity measures for short segments of text. In *European conference on information retrieval*, pages 16–27. Springer, 2007.

[MDP+11] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics, 2011.

[MFP00] Andrew McCallum, Dayne Freitag, and Fernando CN Pereira. Maximum entropy markov models for information extraction and segmentation. In *Icml*, volume 17, pages 591–598, 2000.

[MH16] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*, 2016.

[Mil95] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[ML13] Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172. ACM, 2013.

[MLS13] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013.

[MLX18a] S. Mokhtari, T. Li, and N. Xie. Context-sensitive neural sentiment classification. In *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 293–299, July 2018.

[MLX18b] S. Mokhtari, T. Li, and N. Xie. Revmap: A visualized framework for holistic view of reviews. In *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 240–243, July 2018.

[Mok18] Shekoofeh Mokhtari. Hierarchical neural model fortagging address queries in map search, 2018.

[MPL15] Julian McAuley, Rahul Pandey, and Jure Leskovec. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2015.

[MR93] Andrzej Mackiewicz and Waldemar Ratajczak. Principal components analysis (pca). *Computers and Geosciences*, 19:303–342, 1993.

[MSC+13] 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.

[MTSvdH15] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52. ACM, 2015.

[NFH+96] Lucy Terry Nowell, Robert K France, Deborah Hix, Lenwood S Heath, and Edward A Fox. Visualizing search results: some alternatives to query-document similarity. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 67–75. ACM, 1996.

[PB07] Michael J Pazzani and Daniel Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer, 2007.

[PC98] Jay M Ponte and W Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281. ACM, 1998.

[PDS+16] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(4):694–707, 2016.

[Pin87] Fernando J Pineda. Generalization of back-propagation to recurrent neural networks. *Physical review letters*, 59(19):2229, 1987.

[PL+08] Bo Pang, Lillian Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[PTDU16] Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*, 2016.

[RDA+17] Sina Rashidian, Xinyu Dong, Amogh Avadhani, Prachi Poddar, and Fusheng Wang. Effective scalable and integrative geocoding for massive address datasets. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 26. ACM, 2017.

[Res99] Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of artificial intelligence research*, 11:95–130, 1999.

[RHW85] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[Ron14] Xin Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.

[RWJ+95] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. Okapi at trec-3. *Nist Special Publication Sp*, 109:109, 1995.

[RZZJ16] Yafeng Ren, Yue Zhang, Meishan Zhang, and Donghong Ji. Context-sensitive twitter sentiment classification using neural network. In *AAAI*, pages 215–221, 2016.

[SDHH98] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 workshop*, volume 62, pages 98–105, 1998.

[SGS15] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[SHG+14] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 373–374. ACM, 2014.

[SHK+14] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[Shr17] Ryan Shrott. Deep Learning Specialization by Andrew Ng - 21 lessons learned, 2017.

[SHYL17] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. Representation learning of users and items for review rating prediction using attention-based convolutional neural network. In *3rd International Workshop on Machine Learning Methods for Recommender Systems (MLRec)(SDM17)*, 2017.

[SLMN11] Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136, 2011.

[SM15] Aliaksei Severyn and Alessandro Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 373–382. ACM, 2015.

[SMFCM18] Rui Santos, Patricia Murrieta-Flores, Pável Calado, and Bruno Martins. Toponym matching through deep neural networks. *International Journal of Geographical Information Science*, 32(2):324–348, 2018.

[Sri] Shubhankar Srivastava. Identifying duplicate questions on quora.

[SS12] Nitish Srivastava and Ruslan R Salakhutdinov. Multimodal learning with deep boltzmann machines. In *Advances in neural information processing systems*, pages 2222–2230, 2012.

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[SZ14] Cicero D Santos and Bianca Zadrozny. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1818–1826, 2014.

[TDT+17] Gaurav Singh Tomar, Thyago Duque, Oscar Täckström, Jakob Uszkoreit, and Dipanjan Das. Neural paraphrase identification of questions with noisy pretraining. *arXiv preprint arXiv:1704.04565*, 2017.

[TNT11] Panayiotis Tsaparas, Alexandros Ntoulas, and Evimaria Terzi. Selecting a comprehensive set of reviews. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–176. ACM, 2011.

[TPN+17] Sandeep Tata, Alexandrin Popescul, Marc Najork, Mike Colagrosso, Julian Gibbons, Alan Green, Alexandre Mah, Michael Smith, Divanshu Garg, Cayden Meyer, et al. Quick access: Building a smart experience for google drive. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1643–1651. ACM, 2017.

[TQL15] Duyu Tang, Bing Qin, and Ting Liu. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1422–1432, 2015.

[TQLY15] Duyu Tang, Bing Qin, Ting Liu, and Yuekui Yang. User modeling with neural network for review rating prediction. In *IJCAI*, pages 1340–1346, 2015.

[TQM15] Jian Tang, Meng Qu, and Qiaozhu Mei. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1165–1174. ACM, 2015.

[VVR16] Soroush Vosoughi, Prashanth Vijayaraghavan, and Deb Roy. Tweet2vec: Learning tweet embeddings using character-level cnn-lstm encoder-decoder. 2016.

[WEG87] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[WHF17] Zhiguo Wang, Wael Hamza, and Radu Florian. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*, 2017.

[WHY+16] Minlue Wang, Valeriia Haberland, Amos Yeo, Andrew Martin, John Howroyd, and J Mark Bishop. A probabilistic address parser using conditional random fields and stochastic regular grammar. In *Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on*, pages 225–232. IEEE, 2016.

[WM12] Sida Wang and Christopher D Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 90–94. Association for Computational Linguistics, 2012.

[WWC05] Janyce Wiebe, Theresa Wilson, and Claire Cardie. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2):165–210, 2005.

[WWL+10] Yingcai Wu, Furu Wei, Shixia Liu, Norman Au, Weiwei Cui, Hong Zhou, and Huamin Qu. Opinionseer: interactive visualization of hotel customer feedback. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):1109–1118, 2010.

[XJC08] Xiaobing Xue, Jiwoon Jeon, and W Bruce Croft. Retrieval models for question and answer archives. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 475–482. ACM, 2008.

[XZS16] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604*, 2016.

[YZHS13] Wenzhe Yu, Rong Zhang, Xiaofeng He, and Chaofeng Sha. Selecting a diversified set of reviews. In *Asia-Pacific Web Conference*, pages 721–733. Springer, 2013.

[ZBH19] Yuan Zhang, Jason Baldridge, and Luheng He. Paws: Paraphrase adversaries from word scrambling. *arXiv preprint arXiv:1904.01130*, 2019.

[ZHH+10] Shaoting Zhang, Junzhou Huang, Yuchi Huang, Yang Yu, Hongsheng Li, and Dimitris N Metaxas. Automatic image annotation using group sparsity. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3312–3319. IEEE, 2010.

[ZZL15] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

VITA

SHEKOOFEH MOKHTARI

2014-2018	Ph.D., Computer Science Florida International University,
Miami, Florida	
2013-2014	M.S., Computer Science Florida International University,
Miami, Florida	
2005–2010	B.S., Computer Science University of Isfahan, Isfahan, IRAN

PUBLICATIONS

Shekoofeh Mokhtari, Ahmad Mahmoudy, Dragomir Yankov, Ning Xie, *Tagging Address Queries in Map Search*, in Proceedings of the 33rd Innovative Applications of Artificial Intelligence (IAAI), 2019.

Shekoofeh Mokhtari, Tao Li, Ning Xie, *Context-Sensitive Neural Sentiment Classification*, in Proceedings of the 19th IEEE International Conference on Information Reuse and Integration (IRI), 2018.

Shekoofeh Mokhtari, Tao Li, Ning Xie, *RevMap: A Visualized Framework For Holistic View of Reviews*, in Proceedings of the 19th IEEE International Conference on Information Reuse and Integration (IRI), 2018.

Shekoofeh Mokhtari, Dragomir Yankov, Ning Xie, *Hierarchical Neural Model for Tagging Address Queries in Map Search*, in Proceedings of the 2nd Widening Natural Language Processing Workshop (WiNLP), 2018.

Shekoofeh Mokhtari, Tao Li, *Tagging Address Queries in Map Search*,Women in Machine Learning Workshop (WiML), 2017.

Chunqiu Zeng, Qing Wang, Shekoofeh Mokhtari, Tao Li, *Online Context-Aware Recommendation with Time Varying Multi-Armed Bandit*, in Proceedings of the 22nd annual ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2016.

Kianoosh G Boroojeni, Shekoufeh Mokhtari, M Hadi Amini, SS Iyengar, *Optimal two-tier forecasting power generation model in smart grids*, in Proceeding of the International Journal of Information Processing (IJIP), 2015.

Kianoosh G Boroojeni. Boroojeni, Shekoofeh Mokhtari, S. S. Iyengar, *A Hybrid Model for Forecasting Power Demand and Generation in Smart Grids*, In ICCN Proceedings, pages 1-9, 2014.