A Study of Deep Learning for Sentiment Classification on Social Media

NGUYEN THANH HUY

Japan Advanced Institute of Science and Technology

Doctoral Dissertation

A Study of Deep Learning for Sentiment Classification on Social Media

NGUYEN THANH HUY

Supervisor: Associate Professor NGUYEN LE MINH

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
Information Science

March, 2019

Abstract

Sentiment classification on Twitter social networking has been becoming popular in recent years. People express their opinions and feeling about everything on Twitter social networking. These opinions and feeling can be used as useful information for decision making. For example, customers want to know the opinions of other users about a product before making a purchasing decision. Companies want to know the feedback of consumers about a product or the aspects of the product to improve the quality of that product. Therefore, sentiment analysis is playing a big role in the real world and become one of trending research topics in natural language processing. Some previous studies showed the satisfying results of sentiment classification by using traditional machine learning models or lexicon-based approaches. However, these results are on traditional social networks such as forum and review, where texts/ documents are formal, long and easily to interpret. It is still hard to analyze the sentiments of tweets. Tweets are very short and contain many noises (e.g., slang, informal expression, emoticons, mistyping and many words that have no in a dictionary). Traditional methods can not achieve good performance due to the unique characteristics of Twitter social networking. Moreover, most of the traditional methods require laborious feature engineering that is difficult to extract for a specific domain. On the other hand, existing sentiment analysis approaches mainly focus on measuring the sentiment of individual words without considering the semantics of a word and the relationship between words.

In this thesis, we research and develop deep learning methods to classify the sentiment polarities of tweets on Twitter micro-blogging. We not only focus on classifying the sentiment polarity of each tweet by considering textual information but also considering the aspects of each tweet. Three main sentiment analysis tasks are considered (1) Tweet-level sentiment analysis. We introduce a deep learning approach that models the different characteristics (flavor-features) of each word and tries to incorporate them into the deep neural network in order to extract correct sentiment contextual words. Four flavor-features (Word embeddings, Dependency-based word embeddings, Lexicon embeddings, and Character attention embeddings) provide real-valued hints to alleviate the data sparseness and improve sentiment classification performance. Specifically, the data sparseness is reduced by the following two methods. First, we perform data processing and apply semantic rules to deal with noise, negation and specific PoS particles in tweets. Second, we develop the multiple perspectives of each word upon word embeddings for the deep neural network to modeling the structure of tweets. (2) Aspect-level sentiment classification. We propose methods to incorporate aspect information into deep neural networks by using the advantages of multiple attention mechanisms, iterative attention mechanism. In this task, the sentiment lexicon feature is still interpolated into feature vectors and is studied the effect of classifying the sentiment polarities of aspects. (3) Multitask-based aspect-level sentiment classification. We introduce a multi-task learning approach which combines multiple inputs to address the drawbacks of aspect-level data. The multi-task learning called transfer learning allows the model to learn interactive knowledge between many tasks in order to deal with the difficulty in aspect-level data is that existing public data for this task are small which largely limits to the effectiveness of deep learning models. The sentiment lexicon is still considered as a flavor-feature to highlight the importance of aspects and their contexts.

The proposed methods are effective and significantly improve the performance compared to the baselines and the-state-of-the-art models.

Keywords: Tweet-level Sentiment Analysis, Aspect-level Sentiment Analysis, Twitter Social Networking, Deep Learning, Multi-task Learning.

Acknowledgments

I would like to thank Associate Professor. NGUYEN, Le Minh who supported my work and helped me get results of better quality. I am also grateful to the members of my committee for their patience and support in overcoming numerous obstacles I have been facing through my research

I would like to thank Professor. Kiyoaki Shirai, my minor research project advisor. The discussions with him widens my research point of views and inspire me with a handful of ideas.

I am grateful to Professor. Satoshi Tojo who showed me the science world and encourages me to become a scientist.

I would like to thank my excellent laboratory members for their feedback, cooperation and of course friendship. In addition I would like to express my gratitude to the staff of Japan Advanced Institute of Science and Technology for their supports.

Last but not the least, I would like to thank my family: my parents, my brother and to my wife and my son for supporting me spiritually throughout writing this thesis and my life in general.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, the overview of sentiment analysis and the benefits of sentiment analysis in the real world are introduced. Additionally, we formulate the challenges of social networks which sentiment analysis models are applied to deal with and how we achieve the objective. In summary, Section 1.1 describes the background and motivation of sentiment analysis, Section 1.2 draws up the challenges of sentiment analysis on social networking, Section 1.3 expresses the research objective which need to be achieved through research methodologies described in Section 1.4.

1.1 Background and Motivation

Sentiment analysis or opinion mining is the computational study of people's opinions, sentiments, emotions, appraisals, attitudes towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes. There are some differences between sentiment and opinion is that sentiment is defined as an attitude, thought, or judgment prompted by feeling, whereas opinion is defined as a view, judgment, or appraisal formed in mind about a particular matter. The definitions indicate that an opinion is more of a person's detailed view about something, whereas a sentiment is more of a feeling. Formally, we can define a sentiment is a quintuple as follows:

$$(e_i, a_{ij}, s_{ijkl}, h_k, t_l) \tag{1.1}$$

Where e_i is the name of an entity, a_{ij} is an aspect of e_i, s_{ijkl} is the sentiment on the aspect a_{ij} of the entity e_i, h_k denotes the opinion holder, and t_l is the time when the opinion is expressed by h_k. The sentiment s_{ijkl} is positive, negative, or neutral, or expressed with different strength/intensity levels, such as the 1-5 stars

system used by most review websites (e.g., Amazon[1]).

Recent years have witnessed the development of digital forms such as reviews, forum discussions, blogs, and micro-blogs, we have a massive volume of opinionated data recorded. As such, sentiment analysis has grown to be one of the most active research areas in natural language processing (NLP) and had importance to business and society. Nowadays, sentiment analysis is a trending topic and has gained even more value with the advent of social networks. Their great diffusion and their role in modern society represent one of the most exciting novelties in recent years. However, sentiment analysis on social networking consists of many challenges. The first challenge mainly focuses on the constant evolution of the language used online in user-generated contents: the words that surround us every day influence the words we use. Because the language used in social networks for us to communicate with each other tends to be more malleable than formal writing, the combination of informal, personal communication, and the mass audience afforded by social networks is a recipe for rapid change. The sentiment analysis needs to adapt to it or be adapted by researchers. However, being able to solve these problems requires robust natural language processing and linguistics skills. Another challenge relates to the social networking natures, in which definition is the dynamic, heterogeneous, domain-mixing environment and the entities involved are connected. Specifically, social websites allow users to post anything that they want without restrictions and the users often connect in large networked environments. Therefore, investigating the sentiment on social networking is difficult. One of the popular social networks which exist these problems is Twitter social networking. Analyzing the sentiment of tweets is still difficult because the tweets are very short and contain slang, informal expressions, emoticons and many words not found in a dictionary.

Twitter social networking[2] is a famous micro-blogging and accessible through the website interface, SMS, or mobile devices. *80%* users are active through mobiles and express their opinions/ sentiments every day. Investigating the sentiment polarity of user data on Twitter social networking have become popular in recent years and become to be an important research direction of sentiment analysis. For example, companies want to know the opinion of customers about their products or a person can notify an essential event to people and listens to people about this event. Therefore, micro-blogging is a useful resource which can be extracted as useful information.

The purpose of sentiment analysis is to define automatic tools which able to extract sentiment information from texts in natural language towards to create structure and actionable knowledge to be used by a decision support system. The

[1]http://amazon.co.jp
[2]http://www.twitter.com

traditional approaches such as traditional machine learning and lexicon-based approaches mainly focus on predicting the sentiment of document data such as product, movie reviews and achieved the successful results such as the works of [Hu and Liu, 2004], [Aue and Gamon, 2005], [Pang and Lee, 2008] [Go et al., 2009], [Kumar and Sebastian, 2012], [Mohammad et al., 2013] and [Kiritchenko et al., 2014a]. Existing approaches mainly focus on the classification where textual content is tackled without considering the semantic information of the textual information. These traditional approaches usually require laborious feature engineering which is difficult to identify. The laborious feature engineering decides the classification performance of the traditional classifiers.

Thanks to the rapid development of deep learning networks, the problems described above can be addressed. In this thesis, we develop methods based on deep neural networks to classify the sentiment on micro-blogging. We identify the characteristics of Twitter social networking and model them upon word embeddings in order to cast flavor features for tweets. These flavor features are incorporated into neural networks to capture the relations of tweets in order to produce more reliable and robust results. This step is considered as the sentiment summarization. Subsequently, we are diving into the detail of each tweet to capture the sentiment polarity of each aspect of the tweet. Aspect-level sentiment analysis yields very fine-grained sentiment information which can be useful for applications in various domains. For example, in reality, when a company evaluates the quality of a product, they often look at the overall sentiment of tweets on Micro-blogging first. However, if they want to improve the quality of the product, they must consider each aspect of the product.

To deal with this, we consider the specific parts of a tweet instead of considering the whole tweet. For example, the sentence *"The battery life is too short, the iPhone screen is good.* In the sentence, the information regarding the sentiment polarity of the aspect *"battery life* is the sub-sentence *"the battery life is too short* only. On the other hand, there is a remaining challenge that each word of an aspect may have different contributions. For example, the aspect *"iPhone screen"*, the word *"screen"* is more important than the word *"iPhone"* in its context. Inspire by this problem, we propose multiple attention mechanisms to attend the important parts of an aspect and its context. The purpose of the multiple attention mechanisms is to learn to generate a context vector for each output time step. In other words, a deep learning model tries to learn what to attend based on an input sentence and what it has produced so far. The multiple attention mechanisms are intra-attention and interactive attention mechanisms in which intra-attention called self-attention tries to capture the importance of an aspect, while interactive attention interactively assigns an attention score to the relationship between the aspect context vector and each context words. In order words, the interactive-

attention incorporates the aspect information into the learning model to adaptively learn to focus on the correct sentiment words towards the given aspect term.

Twitter social networking is used as a representative case study in this thesis because of the following principal reasons. First, Twitter social networking is one of the popular social networks containing large data. Second, many previous works used Twitter as the case study which conducting the experiments on the Twitter data. However, Twitter social networking has recently limited to crawling data due to the privacy setting. This lead to public data for aspect-level sentiment classification task is small which largely limits to the effectiveness of deep learning models. To tackle this shortcoming, we propose transfer approaches to allow the model to integrate interactive knowledge from annotated and un-annotated corpora being much less expensive for improving the performance of aspect-level sentiment classification. In this approach, we develop multi-task learning which allows the model to learn interacting knowledge between tasks.

We investigate the methods and evaluate the effectiveness of our proposed models in multiple level sentiment classification tasks: (1) tweet-level sentiment classification is to summarize the sentiment polarity of a tweet. (2) aspect-level sentiment classification is to analyze the sentiment orientation of each aspect of a tweet (3) multitask-based sentiment analysis is to utilize sharing knowledge for aspect-level sentiment classification.

1.2 Problem Statement

Twitter micro-blogging service contains many challenges due to the ubiquitous nature of Twitter social networking. The texts of Twitter social networking is short text messages in term of noise, relevance, emotion, folksonomy, and slang. Therefore, these unique properties of Twitter social networking cause the difficulties of predicting sentiment on micro-blogging. There are some challenges that we have to face on social networking raised by the following reasons:

Noisy Data

Depend on the purpose of Twitter micro-blogging, the users are forced to write their messages within a limited space. Twitter social networking requires 140 characters only. Moreover, because it is a user's private space, the language used by the users are very informal. The users create their own words: *spelling shortcuts*, *punctuation*, *emoticons*, *misspellings*, *slang*, *new words*, *URLs*, *genre-specific terminology* and *abbreviations*. Therefore, the messages of Twitter social networking contain many slang, informal expressions, emotions, mistyping and many words not found in a dictionary, and even hashtags.

On the other hand, the users of Twitter social networking tend to use many emoticons to express their opinions. Some previous works using traditional machine learning such as the work of [Go et al., 2009] consider the emoticons (e.g., *":("*, *":("*) as noisy data. Indeed, the users commonly express their feeling opposite to the emoticons, especially in the sarcasm case. These characteristics cause noisy data to tweet-level sentiment analysis.

Lack of Data

Unlike other social media websites such as *reddit* forum where users express their opinions into specific topics and domains, Twitter social networking allows the users to freely express their opinion to any topics and domains without restriction. As such, the sentiment polarities of words are dependent on the targets or domains. Moreover, the users may express many aspects or targets in one tweet, even though the length of tweets is limit. For example, the tweet *"the iPhone screen is good, but the battery life is to short"*. This requires we not only take into consideration for the tweet level but also the aspect level and cause difficulties to annotate training data.

On the other hand, there has been no any system to gather data on Twitter social networking recently for specific domains due to noisy data and lack of specific domains. This causes very time consuming to collect training data for every target domain. Moreover, classifiers trained on a specific domain is poor performance when applied to other domains. Additionally, identifying good laborious feature engineering for traditional machine learning is difficult due to noisy data. Therefore, a machine learning without any laborious feature engineering is desirable.

Personalized Style

Because of arbitrariness in posting messages on Twitter social networking, the users tend to use their own personalized sentiment words when expressing their opinion. For example, the word *"good"* may be a strong feeling for one user, but it is a bit feeling for another one. Therefore, the user-sentiment consistency should be considered.

1.3 Research Objective

To capture the users' sentiment expressed in micro-blogging and deal with the problems shown in Section 1.2, our motivation is to develop deep learning models which can capture the characteristics of Twitter social networking without considering any laborious feature engineering. To accomplish this, we propose novel deep

learning methods to recognize the characteristics of tweet words towards to consider the relationship of the words, because, with the unique properties of Twitter social networking, building a good extractor for laborious feature engineering is difficult due to noisy data. Moreover, the problems of traditional machine learning methods are challenging to adapt well to different domains or different languages.

In reality, most people often consider the sentiment overall of tweets firstly, then, regarding the specific targets of the tweets before making a decision. For example, the tweet: *"The battery is too short, but the phone is still good"*. In this tweet, the overall is positive. However, the aspect of the tweet is *"battery"* with negative sentiment. Therefore, we formulate our problems into two main categories: tweet-level sentiment analysis and aspect-level sentiment analysis where the tweet-level task is to summarize the sentiment orientation of a tweet, while aspect level task is to pay more attention to the detail of the tweet, specifically, the sentiment polarity of each aspect of the tweet. In other words, our end goal of the aspect-level task is to figure out *"What people think about X"*, where X can be a target such as a brand, product, event, company or celebrity. Tweet-level sentiment analysis is considered as a summarization process, which describes the conclusion of the aspects of the tweet.

To deal with the lack of domain data problem, we consider a knowledge transfer approach which is a powerful aspect for both humans and machines. Human beings can gain more by sharing and teaching each other, and machines can perform this idea in the same way. In layman terms, *transfer learning* is *knowledge sharing*. Transfer learning is the technique of using the knowledge gained by a model trained for a source task to solve a target task. In most cases, the target task will be in a related or a similar domain of data. Moreover, the knowledge gained refers to the weights learned. On the other hand, the weakness of deep learning models requires significant data. Thus, transfer learning approaches deal with large limits to the effectiveness of deep learning models.

In the rest of this chapter, we state research methodologies and end with the chapter organization of the thesis.

1.4 Research methodologies

The main proposal of this thesis is to develop the solutions for improving the sentiment analysis on Twitter micro-blogging. We design a general methodology that consists of three major components: *Data pre-processing*, *Feature formulation* and *Deep neural networks*.

Data Processing

The raw tweet data is processed and reduced the data sparsity. Specifically, the unique properties of Twitter social networking: *Username*, *None* and *Repeated Letters*, *retweets*, *tweet repetition*, *stop words*, *links*, *mentions*, *folksonomies* and *accentuation* are processed. Moreover, a rule-based semantic approach is applied to clean the raw tweet as well. For deep learning models, the data preprocessing step contributes a big role in increasing the classification performance, since the deep learning networks try to learn the probability distributions of tweets for prediction.

Feature Formulation

We investigate and develop a series of augmentation methods to cast flavor features for our deep learning networks. These flavor-features are built upon word embeddings or character embeddings which present the characteristics of tweet words. The purpose of these features is to incorporate real-valued hints (different views) into the deep learning networks for modeling the tweet structure in order to capture the correct contextual words in the tweet.

Deep Neural Networks

We investigate and develop deep neural networks by incorporating knowledge information. The rule-based semantic approach, multiple attention mechanisms, and an iterative attention mechanism are applied to produce reliable and robust sentiment analysis results.

Finally, we conduct a comprehensive evaluation and in-depth analysis of the inner workings of our proposed models compared to the strong state-of-the-art baselines which allow us to understand the problems of sentiment analysis in different views.

1.5 Chapter Organization

The chapters of our thesis are organized into four parts as follows:

Part 1: Introduction

- Chapter 1. We introduce our motivation and fundamentals of sentiment analysis and opinion mining. The problem statement, research objectives are described in this chapter as well.

Part 2: Background and Literature Review

- Chapter 2: In this chapter, we describe the characteristics of the Twitter social networking as well as the overview of our methodology for sentiment analysis.

Part 3: Sentiment Analysis on Social Networking

- Chapter 3. We introduce the background of deep learning networks and state the drawbacks of deep learning networks. Subsequently, we discuss the previous works in the sentiment analysis field.
- Chapter 4: We propose a novel method for Tweet-level sentiment classification which combines flavor-features with word embeddings.
- Chapter 5: We introduce novel approaches which take into consideration the aspect level of tweets.
- Chapter 6: We present a transfer approach using deep neural networks to take into consideration other source domains in order to solve the limitation of the aspect level task.

Part 4: Discussion and Conclusion

- Chapter 7. We discuss and conclude the works presented in this thesis as well as the future works.

Chapter 2

Sentiment Analysis on Twitter Social Networking

In this chapter, the unique properties of social networking and the characteristics of Twitter social networking are introduced. Understanding the characteristics of social networking supports us to interpret the unique aspects of social networking to extract effective features for learning models. The section 2.1 formulates the properties of social websites, while, the subsection 2.1.1 is diving more into representative social networking named Twitter social networking. Based on these unique attitudes, proposed models are introduced to overcome the challenges at two level tasks: Tweet-level sentiment classification and aspect-level sentiment classification tasks.

2.1 Social Networking Characteristics

Social networking is web services that allow users to broadcast their messages to other users in the services. Unlike the real world, where instances are considered as homogeneous, independent, and identically distributed leads us to a substantial loss of information and the introduction of statistical bias, the nature of social networking is heterogeneous, and the entities involved are connected. Users can read messages online and send responses or even express their sentiment/ opinion about these messages. Recently, social networking has been growing rapidly and become to be the mainstream communication media. The number of users in social networking is increasing dramatically. Unlike traditional social networks such as reviews, blogs, and forums, modern social networking has special characteristics that differentiate the traditional social networks from regular websites and difficulties to handle as shown below:

User-based

In social networks, the users are center, and social websites are based on contents that are updated by one user and read by Internet visitors. Online social networks are built and directed by users themselves. Without the users, the social networks would be an empty space filled with empty forums, applications, and chat rooms. The users populate the social networks with conversations and contents. The direction of these contents is determined by anyone who takes part in the discussion. This is what makes social networks so much more exciting and dynamic for Internet users.

Interactive

The users of social networks are very interactive. This means that a social network is not just a collection of chat rooms and forums anymore. Social websites like Facebook are filled with network-based gaming applications, where users can play games together. These social networks are more than just entertainment, they are a way to connect and have fun with friends.

Community-driven

Social networks are built and thrive based on community concepts in which individuals have private hobbies, relationships, find a new friend and even reconnect to old friends in society. Social networks have sub-communities of people who share commonalities, and the users can discover new friends within these interest-based communities.

Relationships

Relationships in social networks are an important concept which creates the development of social networks. The more relationships that you have within the network, the more established you are toward the center of that network.

Short text length

The text length of messages posted on social networks is short. For example, Facebook limits 420 characters for status and Twitter is 140 characters. Due to this limitation, messages posted by the users are often abbreviation, emphatic and emotion. Evenly, the users of Twitter social networking use hashtags to illustrate the topic of tweets. However, due to the limitation of the text length, the users can easily write or receive their messages via various platforms and devices, such as mobile phones, tablets, laptops. On the other hand, the limitation of the length

causes the problem: the users may express an implicit opinion that sentiment detection becomes more difficult.

Informal language

Languages used in social networks are arbitrary due to the character limitation constraint. Users tend to use informal languages and non-standard texts to express the main points. For example, in Twitter social networking, abbreviation (e.g., *LOL* means laughing out loud), misspelling, emphatic lengthening (e.g., *"goooooooood"*), emoticons (e.g., *":("*, *":)"*) and even, the users use wrong grammar. These problems make data to become noisier and sparse.

Topic/ domain variation

Unlike traditional social media such as forums and blogs, modern social networks are a domain-mixing environment which allows users to post any topics in any domains without restrictions. These topics are also sorted following any properties. For example, Facebook social networking has a feed stream that status is shown based on the degree of interaction between users and their friends.

Language style variation

The number of users using social networks is huge and come from many countries. Therefore, there is a mixing of users with different backgrounds and preferences as well as the variety of language styles. Additionally, because of the popular in various countries, social networks are often mixed with many languages. For example, some countries like Singapore, the Philippines can use English to present their language, however, depending on different situations, the meaning of words is different. With these problems, the sentiment analysis on social networking is difficult.

Big and real-time data

More than 2 billion users are using social networks, and the number of messages is largely generated every day. Moreover, social networks operate on the real-time data stream where data is updated immediately. This causes a big problem related to time, storage space and analysis for big data.

2.1.1 Twitter Characteristics

In this thesis, we use Twitter social networking as a representative case study for our research. However, our models can be expanded to other social networks due to

the similar properties of social networks. Additionally, a transfer learning method is utilized to approach knowledge from various resources of social networks.

Twitter social networking is one of the most popular micro-blogging services that allow users to post their messages, called *tweets*. These tweets are displayed at the *timeline* of *followers* who follow the users. The length of a tweet is limited to 140 characters. Users must deliver their message in 140 characters, but this can be seen as a very positive element of Twitter. The character limit makes Twitter fun and exciting so users are not left scrolling through pages of information if they enjoy the tweet they can click on an attached link or reply to the person who composed the original tweet. This characteristic encourages the interaction and conversation between users. Another characteristic that makes Twitter so fantastic is that it is free. Users do not have to pay to share their opinion.

Twitter is designed so that users can check up on what is happening or join a conversation during their favorite TV show. This aspect also allows for multitasking, and users can do something else and share that experience with their followers. Twitter social networking provides a public application programming interface (REST API), which allows programmers and third-party applications to interact with the data on Twitter social networking. Users in Twitter social networking are freely following other users without any permission. Users can create a profile to interact with friends and other people around the world via tweets. The profile can contain information about the user including his or her name, contact details, education, interests, and any other information he or she wants to share. The user can set the account as either public or private. Users can contact other users via private (direct) messages or by replying to other peoples mentions of his or her account name. This is done through a universal tweeting system that uses three basic symbols: *"@"* followed by a Twitter account name for a mention or tweet, *"RT"* for *"Retweeting"* a message, and *"#"* followed a word/phrase for initiating or participating in a *hashtag"*. These properties can be used as features encoded into word embeddings for deep learning models. Figure 2.1 shows the example of a tweet with *Hashtag*, *Mention*, *URLs*, *Emoticon* and *Retweet*.

Hashtag

The hashtag is a word or a phrase starting with *"#"* symbol in order to express the topic of a tweet. For example, the tweet *"I love Apple #Iphone"*. The hashtag can be used to extracting a *trending topic*, *hot topic* which many people discuss.

Retweet

The retweet is an action to re-post or forward a message posted by other users on Twitter social networking. The format of Retweet is *" RT @Username"*, where

Username is the twitter name of the user, who writes the message re-posted by others. The content of retweet can be not changed. As such, the users agree with the content of the original message.

URL

Twitter social networking allows users to refer to external contents, such as news, photos via URLs. URLs enable users to give more information about their opinion due to the limitation of the tweet length.

Mention

The tweet contains *"@Username"* in the body. It is normally used for replying comments or referring to other users. The user mentioned in the tweet will be received a notification message. This creates a relationship between users besides the follow-network.

Figure 2.1: An example of the tweet on Twitter social networking.

2.2 The Overview of Proposed System

In this section, we propose the overview of our system at two levels: tweet-level sentiment classification and aspect-level sentiment classification. Our system takes Twitter corpus as an input and performs *data processing* before feeding into deep learning models. The deep neural networks in Chapter 5 and Chapter 6 consider more aspects extracted from the Twitter corpus for the aspect-level sentiment classification task. For each model, flavor-features are initialized via *feature creator* component before putting to the deep learning models. These flavor-features contribute to shaping the initial views of the deep neural networks about Twitter data and is fine-tuned during the training process of the deep neural networks. Deep neural networks have a promising performance for sentiment analysis task without any laborious feature engineering. This solves the difficult problems of traditional machine learning in term of feature extraction.

Our system is as an integrated approach which considers the different views of a tweet. Specifically, tweet-level sentiment classification is considered first in term of the sentiment summarization of a tweet. Subsequently, the different perspectives of the tweet are exploited. Intuitively, the specific parts of the tweet are recognized and extracted based on their importance scores via multiple attention mechanisms. We propose multiple attention mechanisms to detect the sentiment polarity of each aspect in the tweet. A pipeline of our work is shown in Figure 2.2. Chapters in this thesis concerning each task are shown in this Figure 2.2. In summary, the final output of our system consists of two parts: The sentiment summarization of a tweet and the different aspects of the tweet.

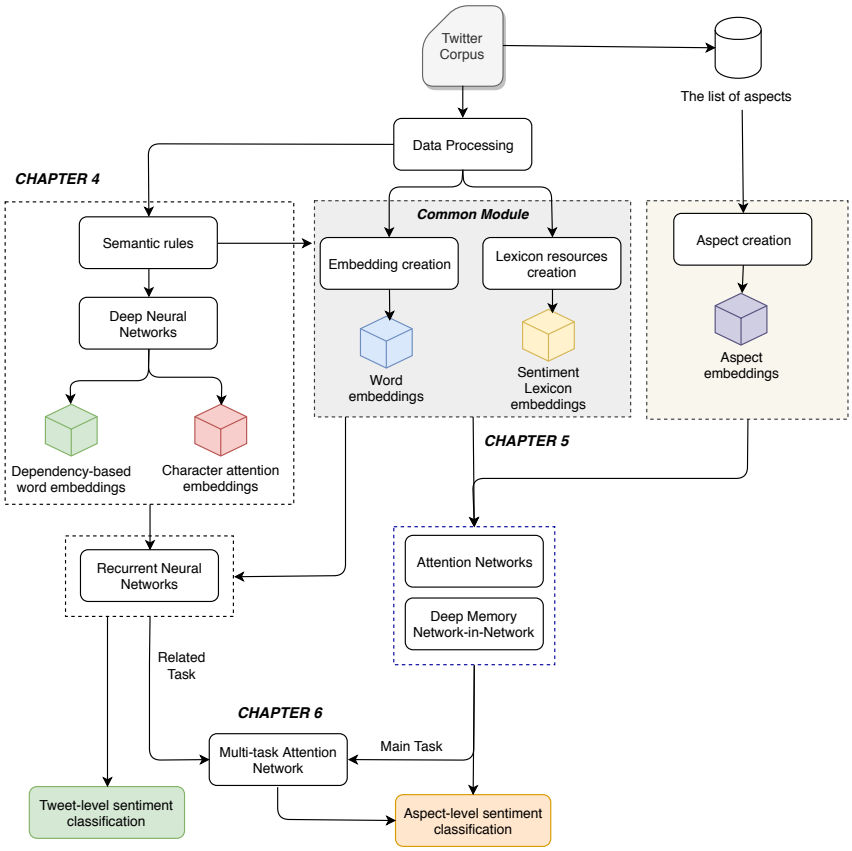


Figure 2.2: The overview of the proposed system architecture.

Chapter 3

Background and Literature Review

In this chapter, we introduce the background of deep learning techniques which are utilized and improved in our thesis. Additionally, the literature review is conducted to introduce the current knowledge including substantive findings, as well as theoretical and methodological contributions to sentiment classification tasks.

3.1 Background of Deep Learning Networks

Deep learning models are the application of *artificial neural networks* to learning tasks using networks of multiple layers. The input feature of deep learning is word embeddings in which the characteristics of words are exploited. The deep learning is inspired by the structure of the biological brain and consists of a large number of information processing units (called neurons) organized in layers. The neural networks adjust the connection weights between neurons which resemble the learning process of a biological brain. Neural networks can be categorized into *Feedforward Neural Networks (FNN)* and *Recurrent Neural Networks (RNN)* in which FNN is the first type of artificial neural networks invented and are simpler than RNN. A simple version of Feedforward Neural Networks is described in Figure 3.1 which consist of four layers. Layer 0 is input layer which forms the input vector (x_1, x_2, x_3). Layer 3 is output layer corresponding to the output vector (y_1, y_2). The middle of the neural network is hidden states corresponding to Layer 1 and Layer 2. The output of Layer 1 and 2 are not visible as the output of Layer 3 called *activation function*. The lines between the neurons represent the connections of the flow of information. Each connection is associated with a weight which controls the signal value between two neurons. The weights are adjusted

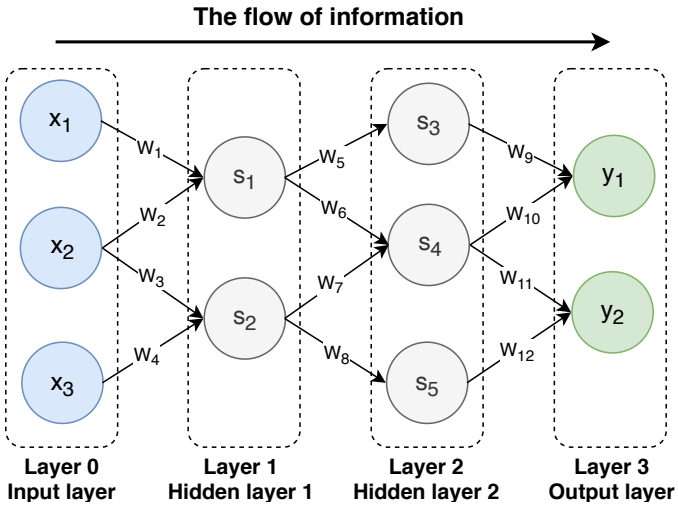


Figure 3.1: A feedforward neural network with information flowing left to right.

via the learning process of the neural network in which information flows through them and is processed in order to generate output to the next layers. After the learning process, the neural network will achieve a complex form of a hypothesis and fits the data. The computational operation in hidden layers can be described as follows: each neuron in Layer 1 takes input (x_1, x_2, x_3) and output a value $f(W_{tx}) = f(\sum_{i=1}^{3} W_i x_i + b)$ by activation function f, where W_i are weights of the connections and b is bias. f function is commonly non-linear function. The common function of f is *Sigmoid* function, hyperbolic tangent function (*tanh*), or rectified linear function (*ReLU*). The formulas of these functions as follows:

$$f(W_{tx}) = sigmoid(W_{tx}) = \frac{1}{1 + exp(-W_{tx})} \tag{3.1}$$

$$f(W_{tx}) = tanh(W_{tx}) = \frac{e^{W_{tx}} - e^{-W_{tx}}}{e^{W_{tx}} + e^{-W_{tx}}} \tag{3.2}$$

$$f(W_{tx}) = ReLU(W_{tx}) = max(0, W_{tx}) \tag{3.3}$$

The *Sigmoid* function takes a real-valued number and squashes it to a value in the range between 0 and 1. The function has been used frequently in the previous time due to its nice interpretation as the firing rate of a neuron: 0 for not firing or 1 for firing. However, the problem of the non-linearity of the *sigmoid* function is that its activation can easily saturate at either the tail of 0 or 1, where gradients are almost zero, and the information flow would be cut. Another thing is that its output is not zero-centered, which could cause undesirable zig-zagging dynamics in the gradient updates for the connection weights in training. The *tanh* function has been more preferred recently in practice because its output range is zero-centered

[-1, 1]. The *ReLU* function has become popular lately because its activation is thresholded at zero when the input is less than 0. The *ReLU* function is easy to compute, fast to converge in training and produce better performance in neural networks against to *sigmoid* and *tanh* functions.

In the last layer of neural networks, a *softmax* function is utilized to normalize the logits of networks to produce a final prediction which squashes a K-dimensional vector X of arbitrary real values to a K-dimensional vector $\sigma(X)$ of real values in the range (0, 1). The functional equation is as follows:

$$\sigma(X)_j = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}} \tag{3.4}$$

Where $j = 1, ..., k$. *Stochastic gradient descent* is used to training a neural network via *Back propagation*. The purpose is to minimize *Cross-entropy* loss. Gradients of the loss corresponding to weights from the last hidden state to the last layer are computed firstly. Subsequently, gradients of the expressions with respect to weights between upper network layers are calculated recursively in a backward manner. The weights between layers are adjusted accordingly through those gradients. It is an iterative refinement process until certain stopping criteria are met.

In a nutshell, deep learning utilizes multiple layers of non-linear for extracting and transforming features. The lower layers try to learn the simple features, while, the higher layers learn more complex features derived from lower layers. Recently, end-to-end neural networks with sophisticated structures have achieved promising performance and showed great potentials. In the next sections, we review some popular Deep learning neural networks which are utilized in our thesis.

3.1.1 Convolutional Neural Networks

Convolutional neural network (CNN), a class of artificial neural networks that have become dominant in computer vision, natural language processing tasks, is attracting interests across a variety of domains. CNN is designed to automatically and adaptively learn spatial hierarchies of features through backpropagation by using multiple building blocks, such as convolution layers, pooling layers, and fully connected layers.

Figure 3.2 shows a Deep convolutional neural network (DeepCNN) from [Nguyen and Nguyen, 2018] to capture the morphology of a word by recognizing the characteristics of characters. DeepCNN has two wide convolution layers. The first layer extracts local features around each character windows of a word and using a max pooling over character windows to produce a global fixed-sized feature vector for the word. The second layer retrieves important context characters and transforms the representation at the previous level into representation at a higher abstract

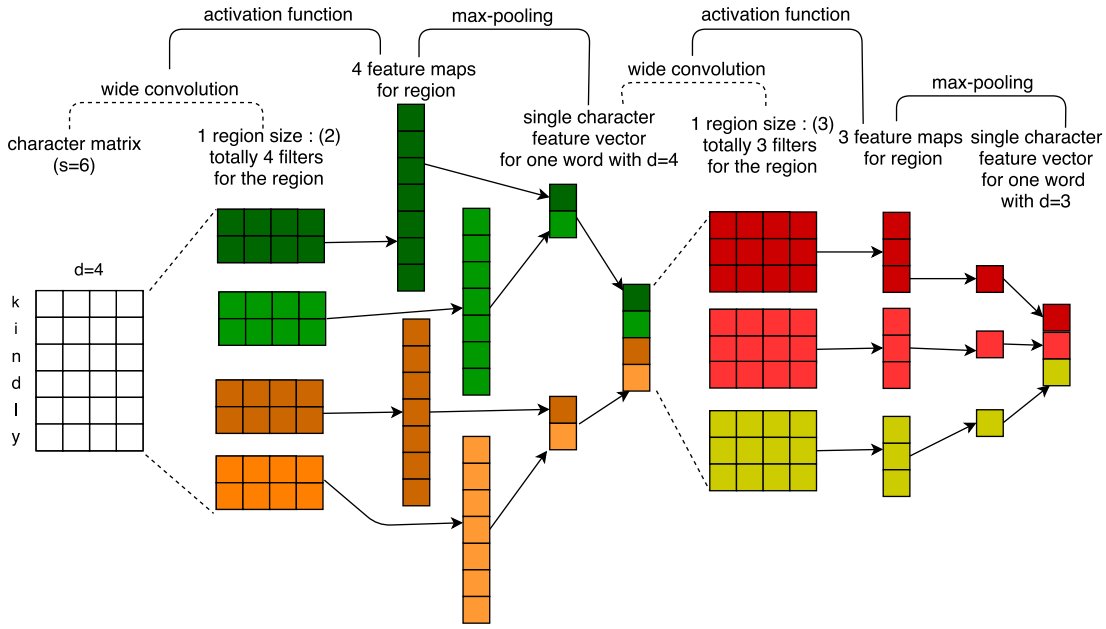


Figure 3.2: A structure of Convolutional Neural Network capturing the local path on the characters of a word [Nguyen and Nguyen, 2018].

level. To extract such local features, DeepCNN utilizes a filter to scan the characters of a word. The filter is an array of numbers (called weights or parameters) which projects on each region of the input matrix. The filter convolves by multiplying its weight values with the original values of the character matrix (*element-wise multiplications*). The multiplications are all summed up to a single value which is a representative of the receptive field. Each representative produces a number. The output of scanning is called *activation map* or *feature map*. Following the convolutional layer is a *subsampling* (or *pooling*) layer which progressively reduces the spatial size of the representation. As such, the *pooling* layer is to reduce the number of features and the computational complexity of the neural network. For example, DeepCNN in Figure 3.2 utilizes the filter of size (2×4) with four filters in the first layer to produce 4 *feature maps*. Afterward, *max pooling* reduces the *feature maps* to a single vector of size (1×4). The last layer of Convolutional Neural Network is a *softmax* layer to normalize logits for prediction.

Convolutional Neural Network often plays a role of feature extractor, which extracts local features. CNN has a different spatially local correlation by enforcing a local connectivity pattern between neurons. Such a characteristic is useful for natural language processing classification, in which we expect to find reliable local clues that these clues can appear in the different places of input. For example, CNN can capture *N-gram* of a sequential data to determine a topic of a document/

sentence. Due to the unique properties of Social networking (e.g., the abbreviation case of Twitter social networking), CNN can be utilized to capture the morphology of each word in a tweet to cast scalar indicators in order to recognize the word.

3.1.2 Recurrent Neural Networks

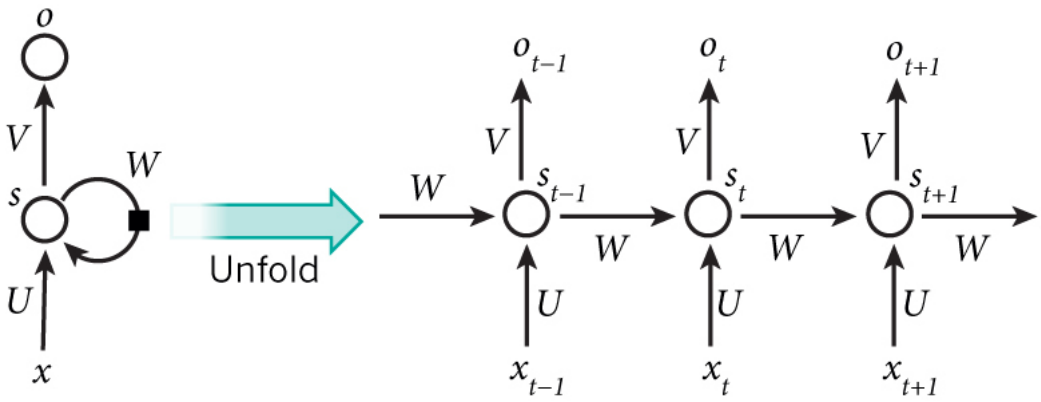


Figure 3.3: A recurrent neural network and the unfolding in time of the computation involved in its forward computation from *Nature*.

Recurrent Neural Networks (RNNs) are a class of neural networks in which the connection weights between neurons form a directed cycle. The RNNs are popular models that have shown great promising performance in many NLP tasks. The idea behind Recurrent Neural Networks is to process sequential information that assumes that all inputs (and outputs) are dependent on each other. Whereas, traditional neural networks accept that all inputs and outputs are independent of each other. In many cases, this is a terrible idea, because if we want to predict the next word in a sentence, we better know which words came before it. RNNs are called *Recurrent* because they perform the same task for each element of sequential data with each output being dependent on all previous computations. This mechanism is considered as a *Memory* as well which remembers all necessary information about what has been processed so far.

Figure 3.4 shows an RNN being unrolled (or unfolded) into a full network. It means that we write out the network for the complete sequence by unrolling. For example, if the sequence is a sentence of 5 words, the network would be unrolled into a 5-layer neural network, one layer corresponds to a word. The formulas that govern the computation happening in RNN are as follows:

- x_i is an input vector at each time step t. The input vector here can be a word embedding or one-hot vector.

- s_t is a hidden state at each time step t and is calculated based on the previously hidden state and the input at the current time step:

$$s_t = f(W^{hs} s_{t-1} + w^{hx} x_t) \tag{3.5}$$

  Where f function is usually a non-linearity $tanh$, or $ReLU$ function. The first hidden state, is typically initialized to all zeroes. W^{hx} is the weight matrix to condition the input x_t.

- o_t is the output at step t which has the equation as follows:

$$o_t = softmax(W^o s_t) \tag{3.6}$$

The hidden state s_t is as a memory of the neural network which captures information about what happened in all the previous time steps. The output at step o_t is calculated solely based on the memory at the time t. RNN shares the same parameter (W^{hs}, W^{hx}, W^o) which is different from Feedforward Neural Networks with different parameters at each layer. Additionally, RNN performs the same task at each step with different inputs. These greatly reduce the total number of parameters needed to learn. However, there are some drawbacks for RNN is that the memory s_t of the neural network can not capture information from too many time steps ago due to *vanishing gradient* problem. Researchers improve this by developing more sophisticated types of RNN to deal with the shortcomings of the standard RNN: bidirectional RNN, deep bidirectional RNN, Long-Short-Term-Memory networks. The basic idea of bidirectional RNN is that the output at each time step not only depends on the previous elements but also depends on the next elements in a sequential data. In other words, the model may look at both the left and right context to predict a missing word in a sequence. The bidirectional RNN consists of two RNNs, in which the first one processes the input from the left to the right context, while the second one processes the reversed input. The output is computed based on the hidden states of both RNNs. The deep bidirectional RNN is similar to bidirectional RNN. However, it is stacked many layers per time step and requires many training data for higher learning capacity.

3.1.3 Long-Short-Term-Memory Networks

Inspired by the drawbacks of RNN model, Long Short-Term Memory networks usually called LSTMs are an improved version of RNN. RNN has a simple structure having a single neural layer. Instead, LSTM is more complicated with four layers interacting in a special way and two states: hidden state and cell states. The core idea behind LSTMs is the cell state which can maintain its state over time, and

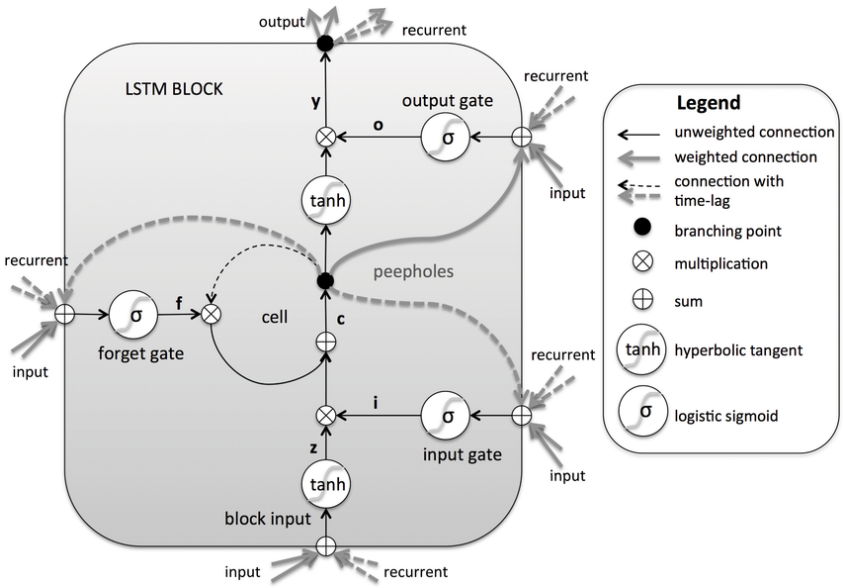


Figure 3.4: The architecture of Long-Short-Term-Memory unit from [Zazo et al., 2016].

non-linear gating units which regulate the information flow into and out of the cell. The following composite function implements a single LSTM memory cell:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{3.7}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{3.8}$$

$$c_t = f_t c_{t-1} + i_t tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{3.9}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{3.10}$$

$$h_t = o_t tanh(c_t) \tag{3.11}$$

where σ is the logistic sigmoid function, i, f, o and c are the *input gate, forget gate, output gate, cell* and *cell input* activation vectors, respectively. All of them have a same size as the hidden vector h. W_{hi} is the hidden-input gate matrix, W_{xo} is the input-output gate matrix. The bias terms which are added to i, f, c and o have been omitted for clarity. The advantage of LSTM compared to standard RNN is capable of learning long-term dependencies. A slight variation of LSTM is Gated Recurrent Unit proposed by [Cho et al., 2014]. It combines the *forget* and *input* gates into a single *update* gate. Additionally, the cell state and hidden state are merged and made some other changes. The GRU is simpler than LSTM and has been growing in popularity.

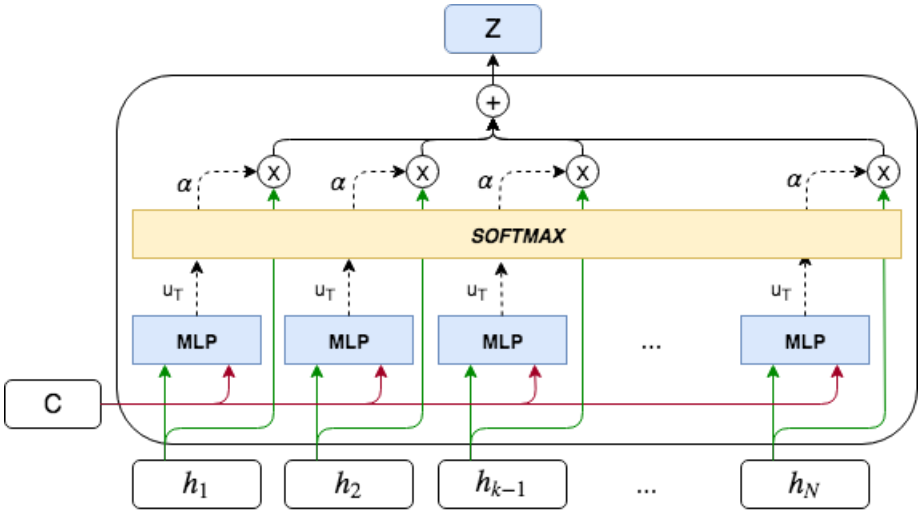


Figure 3.5: The structure of an interactive attention mechanism.

3.1.4 Attention Mechanism

Long-Short-Term-Memory networks are capable of learning long-term dependencies in sequential data. However, in practice, the long-range dependencies are still problematic handle due to mathematical nature: it suffers from *Gradient Vanishing/ Exploding* which means it is hard to train when sentences are long enough. A potential issue with the neural networks needs to be able to compress all the necessary information of a source sentence into a fixed-length vector. A critical and apparent disadvantage of this fixed-length context vector design is incapability of remembering long sentences. Often it has forgotten the first part once it completes processing the full input. Instead of encoding the input sequence into a single fixed context vector, we let the model learn how to generate a context vector for each output time step. That is we let the model learn what to attend based on the input sentence and what it has produced so far. As such, attention mechanisms are proposed to allows the neural networks to attend to different parts of the source sentence at each step of the output generation. The first version of the attention mechanisms is proposed by [Bahdanau et al., 2015] for an encoder-decoder framework where the attention mechanism is used to selecting the reference words of a source sequence for the words of a target sequence before translation. This attention mechanism is a global configuration, where all the encoder states are considered while calculating attention weights. Recently, the attention mechanism has been utilized in classification tasks in order to capture the importance of text representations.

Figure 3.5 describes the structure of an interactive attention mechanism which

is a local configuration proposed in our model. It is improved from the idea of [Bahdanau et al., 2015] in which at each time step t, each encoder state is considered while calculating attention weights instead of considering all the encoder states. The local configuration allows the context vector C to peak at a small segment of the source sequence, in order to the context vector C could selectively focus on the tokens in that small segment. It can be broken down into a few key steps:

- Feed-forward network (MLP network): A one layer MLP acting on the hidden state of the word and generate a Word-level context via dot product.
- Softmax: The resulting vector is passed through a softmax layer.
- Dynamic context: The attention vector from the softmax is combined with the hidden state that was passed into the MLP.

These key steps can be illustrated in Equations in which $h_1, h_2, ..., h_N$ is hidden states encoded by RNNs/ LSTMs and C is a context vector (e.g., In an aspect-level sentiment classification task, the context vector can be a fixed-length aspect vector):

- The single layer MLP is an aggregator which aggregates the values of C and h_i. It is to take the words, rotate/ scale them and then translate them. In other words, it rearranges words into its current vector space. The *tanh* activation then twists and bends the vector space into a manifold. There is no information lost in this step. In this step, the dot product is utilized to compute the correlation between the context vector C and each hidden state h_i. The parameter W learns information of this new vector space as a combination of the context vector C and the hidden state h_i according to their relevance to the problem at hand. In order word, the alignment model assigns a score α_i to the pair of the context vector C and the hidden state h_i at each position i based on how well they match. The set of α_i are weights defining how much of each hidden state should be considered for each output and shows the correlation between source and context vector.

$$e_i = align(C, h_i) = tanh(h_i.W.C^T) \tag{3.12}$$

- Finally, the alignment scores is normalized by *softmax* layers and multiply with the hidden states to extract the importance of the input sequence:

$$\alpha_i = softmax(e_i) \tag{3.13}$$

$$c_i = \sum_{j=1}^{N} \alpha_i h_j \tag{3.14}$$

3.1.5 Word Embeddings

Word representations are central to deep learning and an essential feature extractor which encode the different features of words in their dimensions. Word embeddings are a technique for Language modeling and Feature learning, which transforms words in a vocabulary into vectors of consecutive real numbers. This technique transforms words from high-dimensional sparse vector space (e.g., one-hot encoding vector space, in which each word takes a dimension) to a lower-dimensional dense vector space. Each dimension of the embedding vector represents a latent feature of a word which may encode linguistic regularities and patterns.

Word embeddings can be built by using neural networks or matrix factorization, such as [Collobert and Weston, 2008a], Neural network language model [Bengio et al., 2003] and CBOW (Continuous Bag-of-Word) [Mikolov et al., 2013a], Word2Vec [Mikolov et al., 2013b] and Doc2Vec [Le and Mikolov, 2014]. These methods learn word embeddings from context because words with similar context but opposite sentiment polarities may be mapped to nearby vectors in the embedding space. Inspire by the above methods, [Maas et al., 2011] improved word embeddings that can capture both semantic and sentiment information. [Bespalov et al., 2011] improved a suitable embedding for sentiment classification from *n-gram* model. Later on, [Labutov and Lipson, 2013] re-embedded word embeddings with logistic regression as a regularization term. [Tang et al., 2014b] proposed many kinds of Sentiment-specific word embeddings (SSWEs) which learn both semantic and sentiment information. [Levy and Goldberg, 2014] tackled the disadvantages of word embedding learning models by using a dependency tree to capture only relevant words for a target word. Recently, feature enrichment and multi-sense word embeddings have been investigated for sentiment classification. For example, [Qian et al., 2015] introduced two advanced models, namely Tag-guided recursive neural network (TG-RNN) and Tag-embedded recursive neural network/ Recursive neural tensor network (TE-RNN/ RNTN) to learn tag embeddings. Moreover, [Vo and Zhang, 2015] obtained additional automatic features using unsupervised learning techniques to integrate into word embeddings for Twitter sentiment analysis. Additionally, [Ren et al., 2016] proposed methods to learn topic-enriched multi-prototype word embeddings for Twitter-level sentiment classification. Most of these ideas utilize additional features to increase the information of words.

3.2 Sentiment Analysis on Social Networking

3.2.1 Tweet-level Sentiment Analysis

Tweet-level sentiment analysis is to determine the sentiment expressed in a given tweet. As discussed earlier, the sentiment of a tweet can be inferred with sub-

jectivity classification or polarity classification. Same as sentence-level sentiment classification, the tweet representation produced by neural networks is important for representing the information of a tweet. The ubiquitous nature of Twitter social networking is the use of short text messages in term of noise, relevance, emoticons, folksonomy, and slang. As such, some syntactic and semantic information (e.g., parse trees, opinion lexicons, and part-of-speech tags) may be used to helping. Additional information such as review ratings, social relationship, and cross-domain information can be considered too. For example, a social relationship has been exploited by discovering sentiments in social media data. In previous studies, some researchers dealt with these problems by using a tree structure providing some semantic and syntactic information and combining with words as an input so that the sentiment can be inferred better. However, deep learning models have become popular and are the application of artificial neural networks to extract automatic features by using multiple layers. Such impressive models are the Convolutional neural networks (CNN) and Long-Short-Term-Memory networks (LSTM). The CNN and LSTM models can learn the intrinsic relationships between words in a sentence. We introduce some improved models based on the idea of LSTM and CNN models as the detail below.

[Socher et al., 2011] proposed a semi-supervised recursive auto-encoders network (RAE) for sentence-level sentiment classification, which obtains a reduced dimensional vector representation for a sentence. Later on, [Socher et al., 2012] proposed a Matrix-vector Recursive Neural Network (MV-RNN), in which each word is additionally associated with a tree structure matrix. Additionally, [Socher et al., 2013] introduced a recursive neural tensor network (RNTN), in which tensor-based compositional functions are used to capturing the interactions between elements.

Regarding to the CNN idea, [Kalchbrenner et al., 2014] introduced a Dynamic CNN (DCNN) for modeling the semantics of sentences by improving a CNN model. The difference is that the DCNN uses a dynamic max-pooling as a non-linear subsampling function. The feature map produced by the model can capture word relations. Inspire by DCNN, [Collobert et al., 2001] proposed MAX-TDNN by generalizing the DCNN model. In next time, [Kim, 2014] proposed the many types of Convolutional Neural Networks (CNN) with the multi-channel and single-channel of convolutional layers and [Santos and Gatti, 2014] proposed Character CNN by using two convolutional layers for the characters of a words to perform sentiment analysis of short texts. Besides, [Wang et al., 2015] utilized an LSTM for Twitter-level sentiment classification by simulating the interactions of words during the compositional process.

In another aspect, [Wang et al., 2016a] and [Wang et al., 2016b] used the benefits of LSTM and CNN models to construct a combination of them for sentiment

classification of short texts. [Guggilla et al., 2016] introduced an LSTM-based or CNN-based deep neural networks which utilizes *word2vec* and *linguistic embeddings* for claim classification. [Guan et al., 2017] developed a weakly-supervised CNN model for a sentence/ aspect-level sentiment classification. It is composed of two steps: it first uses overall view ratings for learning a supervised weakly sentence representation and then uses the representation for fine-tuning.

In recent studies, researchers tended to combine lexicon-based approaches with deep learning models in order to enhance the important information of words. Specifically, [Tang et al., 2016a] proposed a context-sensitive lexicon-based method for sentiment classification based on a simple weighted-sum model. The model from [Mishra et al., 2017] utilizes a CNN model to automatically extract cognitive features from the eye-movement data of human readers and uses them as enhanced features alongside textual features for sentiment classification. On the other hand, [Huang et al., 2017] introduced a tree structure LSTM model to encode the syntactic knowledge (e.g., part-of-speech tags) in order to enhance a phrase representations and a sentence representation. Futhermore, the model of [Qian et al., 2017] combines linguistic resources such as sentiment lexicons, negation words and tensity words into a LSTM model (a linguistically regularized LSTM model) to capture the sentiment effects of sentences. Inspired by the advantages of the above models, we consider learning the advanced word embeddings for deep neural networks.

Figure 3.6 illustrates the connection between our tweet-level deep learning model and previous models by using the advantages of previous deep learning models and effective traditional features to tackle the unique characteristics of Twitter social networking.

3.2.2 Aspect-level Sentiment Analysis

Different from tweet-level sentiment classification, aspect-level sentiment analysis considers both the sentiment and the target information in a tweet/ sentence. The target is usually an entity, or an entity aspect generally called aspect. Given a tweet/ sentence and an aspect, aspect-level sentiment analysis aims to infer the sentiment polarity of the aspect in its context. For example, the sentence: *"the iPhone screen is very good, but the battery life is too short"*. The sentiment polarity of the aspect *"iPhone screen"* is positive, whereas, the sentiment polarity of *"battery life"* is negative. There are three important tasks in aspect-level sentiment classification using neural networks. The first one is to identify the context of aspect in a sentence or document. The context means the contextual words concerning a given aspect. The second task is to generate an aspect representation, which can interact with its context. The last task is to determine the important sentiment context words in a sentence or document. Our task is the third task,

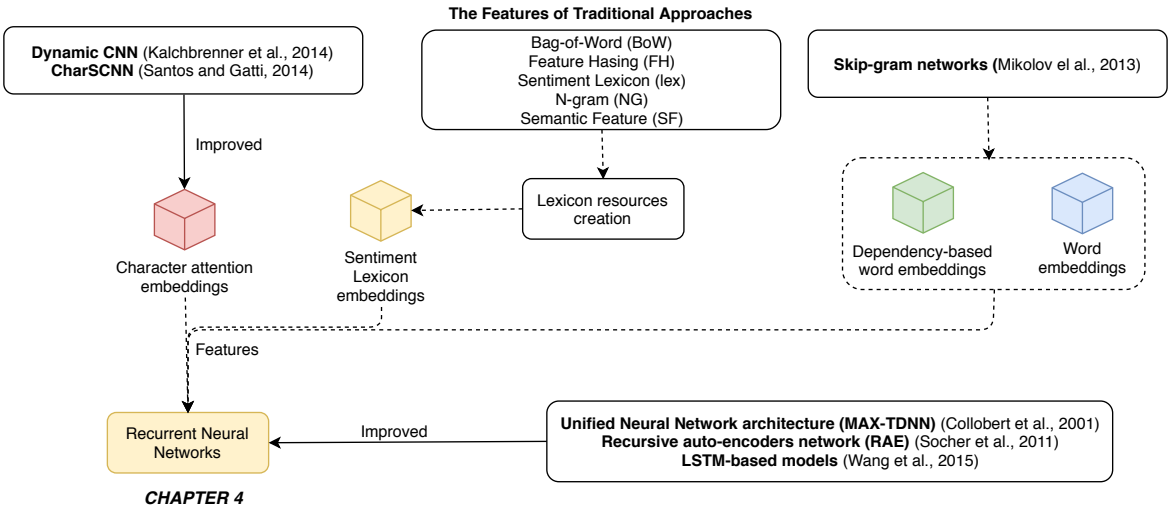


Figure 3.6: The connection between our model and previous models.

in which we develop a neural network to classify the contextual sentiment words. For example, the word *"good"* is extracted for the aspect *"screen"* and classify to positive.

Aspect-level sentiment analysis has a challenge because modeling the relation between an aspect and its context is difficult. The main reason is that different context words have different influences on the sentiment polarity of the aspect and the words of the aspect do not have equal importance. Recently, neural networks have achieved promising performance in recognizing the relativeness between sentence words as well as computing the relativeness between an aspect and its context words.

Ideally, [Dong et al., 2014] proposed an Adaptive recursive neural network (AdaRNN) for target-dependent twitter sentiment classification, which learns to propagate the sentiment information of words towards the target. [Vo and Zhang, 2015] developed aspect-based twitter sentiment classification by making use of rich automatic features, which are additional features obtained by unsupervised learning methods. LSTM models are still popular which can capture semantic relations flexibly. For example, [Tang et al., 2016a] developed models based on LSTM models such as Target-dependent LSTM (TD-LSTM) and Target-connection LSTM (TC-LSTM) to consider a specific aspect as a feature and concatenate it with its context for aspect-level sentiment analysis. These representative models achieves remarkable results for aspect-level sentiment classification.

Later on, [Wang et al., 2016c] proposed an attention-based LSTM neural network by using an attention mechanism to enforce the neural model to attend to the specific parts of context words. Additionally, [Yang et al., 2017] proposed two

attention-based bidirectional LSTM models to improve classification performance. Furthermore, [Tang et al., 2016b] and [Chen et al., 2017] introduced end-to-end memory networks for aspect-level sentiment classification. These models utilize the benefit of attention mechanisms and external memory to capture the importance of context words towards a specific aspect. The memory networks have multiple computational layers which calculate the importance degrees of an aspect-specific representation and transforms the representation to a higher abstract-level representation. The common point of these models is the use of attention mechanisms to look into the specific parts of a context sentence towards a specific aspect.

Deep learning models assisted by attention mechanisms have played a significant role and showed promising performance for aspect-level sentiment analysis, however, to our knowledge, there are still no dominating techniques in the literature that can tackle the problems of aspect-level task thoroughly. Specifically, our models are improved from the attention-based models and the deep memory networks to tackle the challenges of aspect-level sentiment classification. Figure 3.7 illustrates the connection between our aspect-level models and the strong state-of-the-art models in which the attention networks are improved from TD-LSTM+ATT and ATAE-LSTM, while Deep Memory Network-in-Network is improved from attention networks and end-to-end memory networks. Furthermore, the advantages of tweet-level sentiment classification model is utilized as a related task for multi-task learning model in order to enhance the performance of aspect-level sentiment classification task due to the lack of data in social networking. In the next section, we describe the advantages of multi-task learning which contributes to increasing the performance of aspect-level sentiment classification.

3.2.3 Multitask-based Aspect-level Sentiment Analysis

We enhance the aspect-level sentiment analysis problem to multitask-based aspect-level sentiment analysis in order to deal with the limitations of aspect-level data on Twitter social networking by using transfer learning. Transfer Learning is a powerful aspect for both humans and machines. Human beings can gain more by sharing and teaching each other and machines are in the same way. Transfer learning is the technique of using the knowledge gained by a model trained for a source task to solve a target task. In most cases, the target task will be in a related or a similar domain of data. Moreover, the knowledge gained refers to the weights learned. In other words, if additional data is available, transfer learning can often be used to improving performance on the target task. The kind of transfer learning used for our task is multi-task learning which comes in many guises: joint learning, learning to learn, and learning with auxiliary tasks. In Machine Learning (ML), we typically care about optimizing for a particular metric, whether this is a score on a certain benchmark. In order to do this, we generally train a single model or

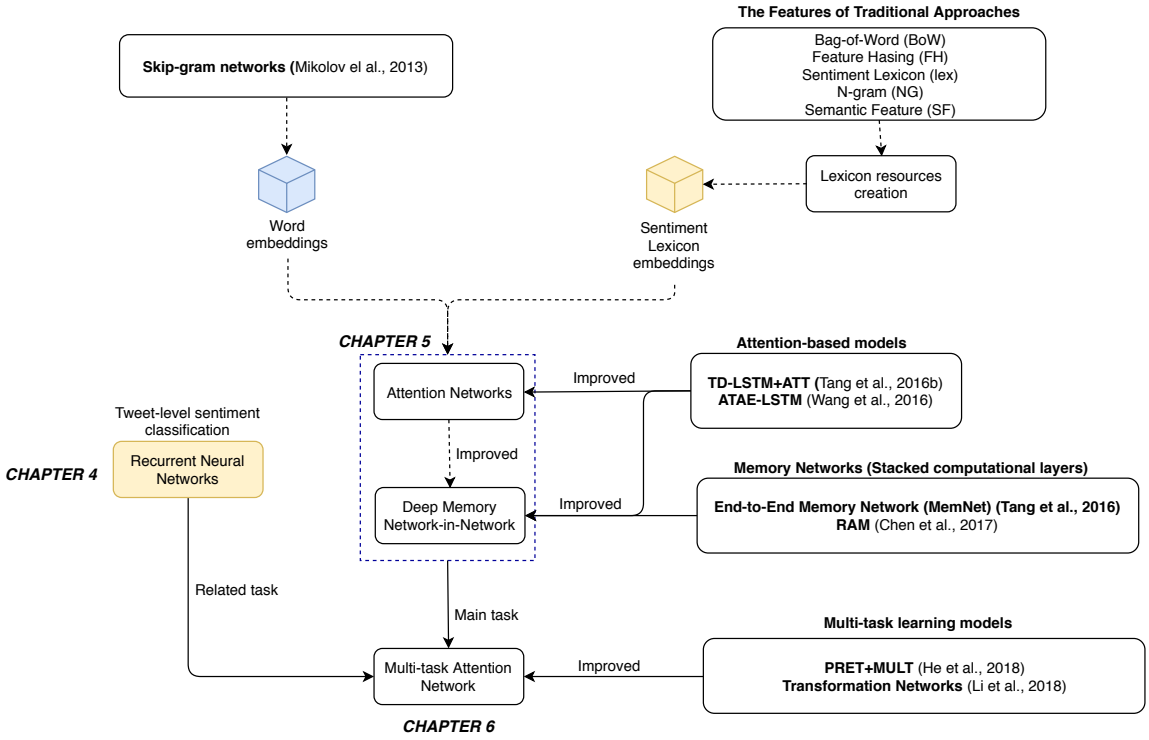


Figure 3.7: The connection between our aspect-level models and the strong state-of-the-art models.

an ensemble of models to perform our desired task. We then fine-tune and tweak these models until their performance no longer increases. While we can generally achieve acceptable performance this way, by being laser-focused on our single task, we ignore information that might help us do even better on the metric we care about. Specifically, this information comes from the training signals of related tasks. By sharing representations between related tasks, we can enable our model to generalize better on our original task.

Multi-task learning has been used successfully across all applications of machine learning, from natural language processing [Collobert and Weston, 2008b] and speech recognition [Deng et al., 2013] to computer vision [Ren et al., 2015]. Multi-task learning can be motivated in different ways: 1) Biologically, we can see multi-task learning as being inspired by human learning. For learning new tasks, we often apply the knowledge we have acquired by learning-related tasks. 2) Pedagogical perspective: we often learn tasks first that provide us with necessary skills to master more complex techniques. 3) A machine learning point of view: We can view multi-task learning as a form of inductive transfer. Inductive transfer can help improve a model by introducing an inductive bias, which causes a model to

prefer some hypotheses over others. There are two kinds of multi-task learning: *Hard parameter sharing* and *Soft parameter sharing*.

- *Soft parameter sharing.* Figure 3.8 illustrates soft parameter sharing in which each task has its own model with its own parameters. The distance between the parameters of the model is then regularized in order to encourage the parameters to be similar.

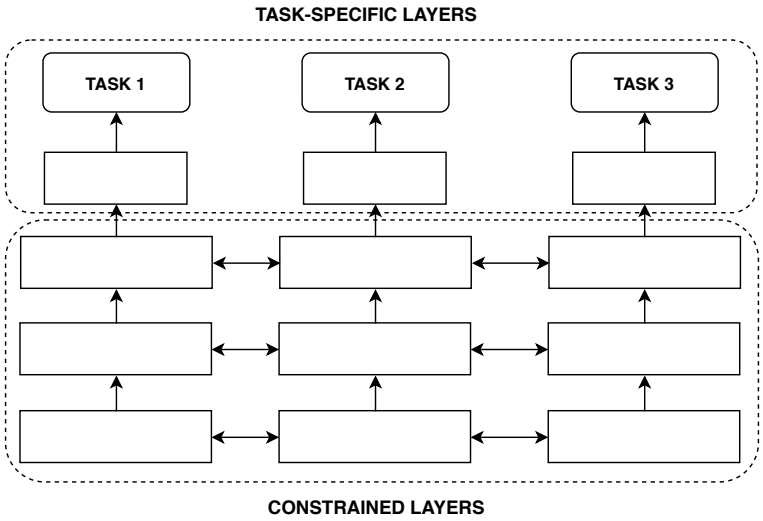


Figure 3.8: The structure of an soft parameter sharing of multi-task learning.

- *Hard parameter sharing.* Figure 3.9 describes hard parameter sharing which is the most commonly used approach to multi-task learning in neural networks. It is generally applied by sharing the hidden layers between all tasks, while keeping several task-specific output layers. Hard parameter sharing greatly reduces the risk of overfitting [Baxter, 1997].

There are some several advantages of multi-task learning:

- *Implicit data augmentation.* Multi-task learning effectively increases the sample size that we are using for training our model.

- *Attention focusing.* If a task is very noisy or data is limited and high-dimensional, it can be difficult for a model to differentiate between relevant and irrelevant features. Multi-task learning can help the model focus its attention on those features which provide additional evidence for the relevance or irrelevance of those features.

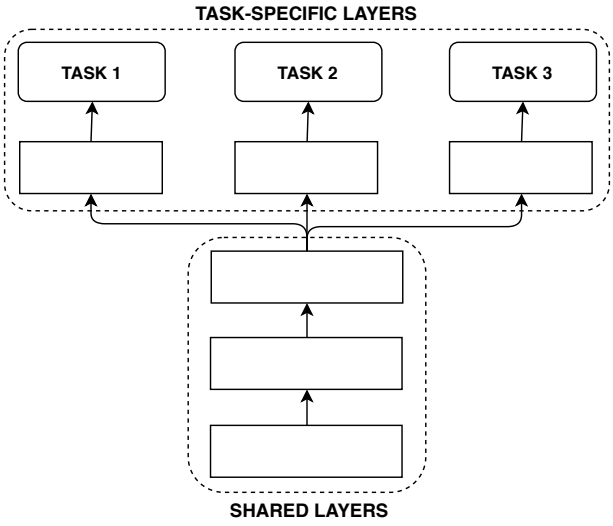


Figure 3.9: The structure of an hard parameter sharing of multi-task learning.

- *Eavesdropping.* Some features G are easy to learn for some task B while being difficult to learn for another task A. Multi-task learning allows the model to eavesdrop in order to learn G through task B.
- *Representation bias.* Multi-task learning biases the model to prefer representations that other tasks also prefer.
- *Regularization.* Multi-task learning acts as a regularizer by introducing an inductive bias. As such, it reduces the risk of overfitting as well as the Rademacher complexity of the model towards its ability to fit random noise.

Thanks to the benefits of multi-task learning, [Hashimoto et al., 2017] utilized multi-task learning to define joint hierarchical learning solving many natural language processing tasks: Chunking, POS Tagging, a Dependency tree, and Entailment. The work which is most relevant to our model is [He et al., 2018]. [He et al., 2018] proposed a kind of multi-task learning model for aspect-level sentiment analysis task in which a pre-trained document-level model is utilized to share interactive knowledge between two tasks. Multi-task learning contributes a big role in tackling the drawbacks of aspect-level sentiment analysis task on Twitter social networking.

Chapter 4

Tweet-level Sentiment Analysis

We introduce a novel *Multiple Features-Aware Contextual Neural Network* for Twitter-level sentiment classification which tries to learn *Multi-characteristics* of each word in a tweet. More specifically, our approach performs a series of feature augmentation methods to cast scalar features upon word embeddings and provide real-valued hints (high-level features) to enrich word embeddings for improving the representation learning process. Such feature-enriched words are incorporated into a neural network by modeling tweet structures which allow the model learns to adaptively focus on the correct sentiment words and contextualize the sentiment words in a tweet. This improves the drawbacks of state-of-the-art models which utilize simple word structures to capture the semantics of words in a tweet. Our model is an end-to-end differentiate neural network and achieves state-of-the-art performance on three Twitter datasets: STS, Sanders, and HCR.

4.1 Introduction

Recent years have witnessed the development of information technology and machine learning technology applied to social media. Social networking such as Facebook, Twitter, and Flickr, where people share their opinions or emotions by uploading texts, pictures or expressing their sentiments of events, products, and phenomenon. Such user data are extensive for investigating useful knowledge. For example, companies want to know the opinion of customers about their products or a person can notify an important event to people and listens to people about this event.

These works are involved in text mining and NLP (Natural Language Processing) tasks in which tweet-level sentiment classification is one of the critical tasks of sentiment analysis. Recently, neural networks (or deep learning) have garnered considerable attention for sentence-level sentiment classification [Socher

et al., 2011], [Collobert et al., 2001], [Kalchbrenner et al., 2014], [Santos and Gatti, 2014], [Wang et al., 2015]. Notably, the dominant state-of-the-art systems for many benchmarks are now neural models, almost completely dispensing with traditional feature engineering techniques altogether. With utilizing the distributed representations of words to represent phrases or sentences, the laborious feature engineering can be extracted automatically by deep learning models using multiple layers and capture the semantics of words. Such distributed representations or word embeddings encode the different features of words in their dimension. Specifically, words with similar context but opposite sentiment polarities may be mapped to nearby vectors in the embedding space such as C&W [Collobert and Weston, 2008a], Neural network language model [Bengio et al., 2003] and CBOW (Continuous Bag-of-Word) [Mikolov et al., 2013a]. Later on, the works of [Bespalov et al., 2011], [Labutov and Lipson, 2013], [Levy and Goldberg, 2014], [Qian et al., 2015], [Vo and Zhang, 2015] and [Ren et al., 2016] tried to incorporate additional features into word embeddings such as N-gram, logistic regression, dependency tree and topic-enriched multi-prototype to learn both semantic and sentiment information. The additional features are a form of feature augmentation and learned from neural networks or lexicon resources in order to enrich the information of each word for word embeddings. We can realize that building good word embeddings can assist deep learning models to learn word relationship for improving classification performance. However, these works were performed for sentence-level sentiment classification in which a sentence is formal and different from a tweet. The ubiquitous nature of Twitter social networking is the use of short text messages in term of noise, relevance, emoticons, folksonomies, and slangs (unique properties). In other words, it is difficult to exploit features and capture the unique characteristics of words. For example, the tweet *"@kirstiealley my dentistis great but shes expensive...=("*. We can observe that the tweet is very short, even wrong vocabulary at the word *"dentistis"* and contains the emoticon. Additionally, the main sentiment part of the tweet is *"she's expensive... =("*, instead of focusing on the whole tweet. To tackle these problems, our approach is targeted at serving two important purposes: 1) Tweets are normalized assisted by a Semantic rule-based approach. 2) Modeling multiple views of each word in a tweet by integrating multiple flavor-features via a Bidirectional Contextual Gated Recurrent Neural Network (Bi-CGRNNet). The Bi-CGRNNet is improved from Gated Recurrent Neural Network (GRNNet) which applies a Gating mechanism to control how much interactive information between the features would pass through to the final result.

We introduce three flavor-features: Character Attention Vectors (CharAVs), Lexicon Embeddings (LexW2Vs) and Dependency-based Word Vectors (DependencyW2Vs). Such multiple features provide different views of words and learn

high-quality representations that could be used for prediction. More specifically, CharAVs is developed from a DeepCNN network to extract local features around each character window of a word in order to capture the morphology of the word. The purpose of LexW2Vs is to highlight the essential sentiment words in a tweet. Finally, DependencyW2Vs is to derive the syntactic contexts of a word via syntactic relations of the word. These features are constructed as an Input Tensor for Bi-CGRNNet.

In summary, our novel model is a differentiate end-to-end neural network that a *Tweet processor* firstly deals with the problems of Twitter social networking in which *Semantic rules* are utilized to assist model in attending important parts of tweets. Subsequently, the *multiple features* are developed and incorporated into word embeddings, then, scoring word relationship via *Bi-CGRNNet*. There are several advantages to this design, e.g., it allows to reduce noise and deal with negation and specific PoS particles in tweets, multi-features provide multiple flavors to each word and learn to attend the correct contextual words in a tweet.

Our Contributions:

The principal contributions of this chapter are as follows:

- We propose a *Tweet processor* which normalizes tweets using *Semantic rules*.
- An Input tensor constructing *flavor-features* is developed to capture multiple perspectives of each word and learn to attend the contextual tweet words.
- A standard Bi-GRNNet is improved into Bi-CGRNNet in order to connect the information of words in a sequence and maintain the order of words for a Tweet-specific representation.
- We conduct a comprehensive and in-depth analysis of the inner workings of our proposed model.

We introduce related works in Section 4.2. Next, the model architecture is described in Section 4.3. This section illustrates our idea and the structure of our model. We show experimental results along with discussion and analysis in Section 4.4. We finish by drawing important conclusions.

4.2 Related Works

Sentiment analysis task can be often interpreted as a multi-class (or binary) classification problem in which many decades of research have been dedicated to building features and running them through Traditional machine learning models such as

Support Vector Machine (SVM), Naive Bayes (NB) and Maximum Entropy (MaxEnt) classifiers [da Silva et al., 2014], [Saif et al., 2012]. These models utilize feature engineerings such as lexicons, n-gram features or parse-tree features.

Recently, end-to-end neural networks such as the Long Short-Term Memory networks have demonstrated promising performance on tweet-level sentiment analysis. The work of [Socher et al., 2011] proposed a Semi-supervised Recursive Autoencoders Network (RAE) for Tweet-level sentiment classification, which obtains a reduced dimensional vector representation for a tweet. Later on, the authors [Collobert et al., 2001] proposed a Unified Neural Network architecture (MAX-TDNN) and learning algorithm that can be applied to various natural language processing tasks. The models mainly learn internal representations by vast amounts of mostly unlabeled training data instead of exploiting human-made input features optimized for each task.

From the idea of [Collobert et al., 2001], a Generalization of MAX-TDNN named Dynamic CNN (DCNN) by [Kalchbrenner et al., 2014] is for semantic modeling of sentences by improving from Convolutional Neural Network (CNN). DCNN uses a Dynamic max-pooling as a Non-linear subsampling function. Our work is most relevant to the work of [Santos and Gatti, 2014] using Character Convolutional Neural Network (CharSCNN). CharSCNN uses two Convolutional layers to extract relevant features from words and sentences of any size to perform sentiment analysis of short texts. Inspired by CharSCNN, we propose a novel Convolutional mechanism which tries to cast a single real-valued character hint of each word.

It is worthy to mention that Long Short-Term Memory (LSTM) can be used for Tweet-level Sentiment Classification. For example, [Wang et al., 2015] proposed the variations of Long Short-Term Memory Recurrent Network (RNN-FLT, RNN-TLT, LSTM-TLT) for Twitter-level sentiment prediction. With the help of Gates and Constant error carousels in a Memory block structure, the model could handle interactions between words flexibly. The LSTM with Gating operation is efficient for exploiting word relationship and controlling information in a tweet. In our model, the interactive information amongst multi-features is computed and controlled via the Gating mechanism. Additionally, the most above models focus on improving the computational architecture of a model in order to capture word relationship. However, the characteristic of a word improved via adopting different flavors may assist a deep learning model to adaptively learns to attend the correct contextual words in a tweet.

4.3 Proposed Model

The overall model architecture illustrated in Figure 4.1 has three major components: *Tweet processor*, *Input tensor* and *Deep neural network*. In this model,

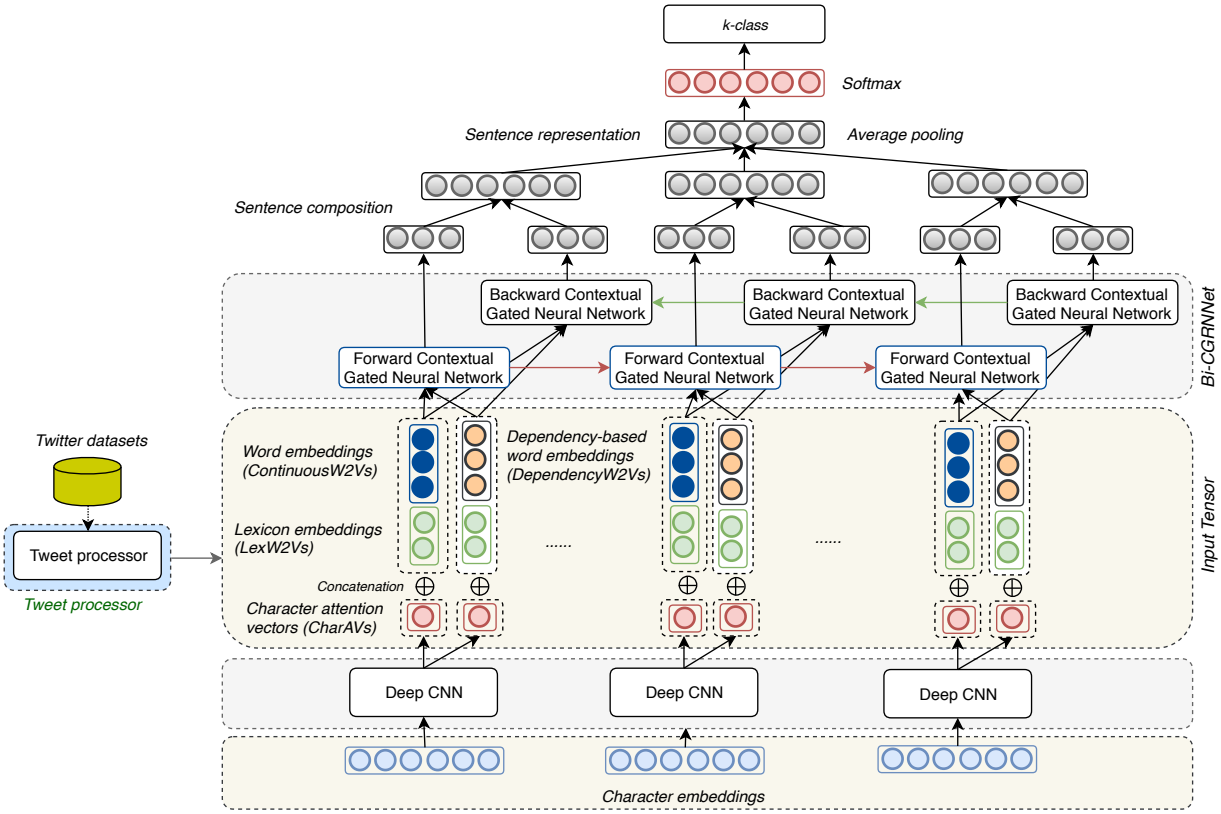


Figure 4.1: The Multiple Features-Aware Contextual Neural Network.

Tweet processor normalizes tweets before feeding into Input tensor for building multiple features. Bi-CGRNNet computes the interactive knowledge between these features and learns to score word relationship for a Tweet-specific representation. In this section, we describe our model architecture module-by-module.

4.3.1 Task definition

A tweet consists of a sequence of n words, denoted as $S = \{w_1, w_2, ..., w_i, ..., w_n\}$, where w_i referring to the position of i-th word in the tweet. The purpose of this task is to classify the sentiment overall of a tweet into positive or negative. For example, the tweet *"I love iPhone #Iphone"*. The sentiment overall of this sentence is positive. Our model like an ensemble model by combining multiple modules to increase the important information of words.

4.3.2 Tweet Processor

In this section, the Tweet processor has two main phases: 1) Pre-processing tweets to remove the unique properties of tweets. 2) Eliminating unnecessary parts of a tweet by Semantic rules (SR). In our view, the pre-processing steps are important for the deep learning models and contribute to affect the classification accuracy.

Pre-processing

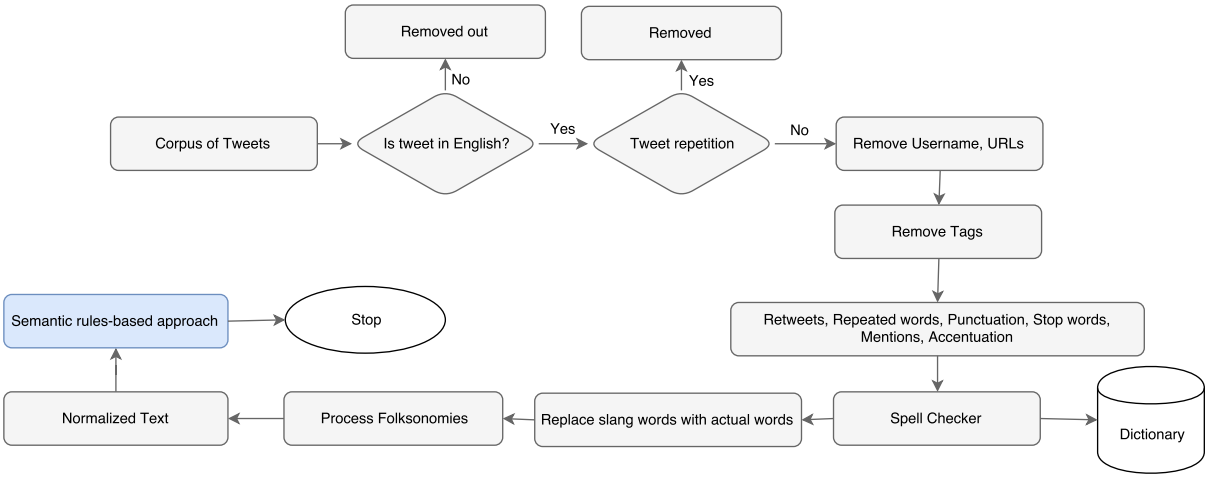


Figure 4.2: The work-flow of the Pre-processing step.

Twitter is a famous micro-blogging and social networking service which users share, deliver their status as a tweet. In the micro-blogging services, users make spelling mistakes and use emoticons for expressing their views and emotions. In most of the social media, languages used by the users is very informal. Users create their own words: *spelling shortcuts*, *punctuation*, *emoticons*, *misspellings*, *slangs*, *new words*, *URLs*, *genre-specific terminology* and *abbreviations*. Therefore, the Pre-processing steps are playing a big role and can be used to reducing the feature space and such kinds of text need to be corrected. Figure 4.2 illustrates the Pre-processing work-flow in which unique properties such as *Username*, *None* and *Repeated Letters* are processed firstly. Then, *retweets*, *tweet repetition*, *stop words*, *links*, *mentions*, *folksonomies* and *accentuation* are processed. We keep emoticons instead of removing them as the study of [Go et al., 2009]. The authors [Go et al., 2009] indicated that emoticons make noise for traditional machine learning models. However, we consider emoticons as a kind of flavor-features captured by Lexicon embeddings. We believe that emoticons are significant for deep learning models since deep neural networks are capable of capturing the semantic of words.

Semantic Rules (SR)

Rule	Semantic rules	Example - STS Corpus	Output
R11	If a sentence contains "but", disregard all previous sentiment and only take the sentiment of the part after "but".	@kirstiealley my dentist is great *but* she's expensive...=(	she's expensive...=(
R12	If a sentence contains "despite", only take sentiment of the part before "despite".	I'm not dead *despite* rumours to the contrary.	I'm not dead
R13	If a sentence contains "unless", and "unless" is followed by a negative clause, disregard the "unless" clause.	laptop charger is broken - *unless* a little cricket set up home inside it overnight. typical at the worst possible time.	laptop charger is broken
R14	If a sentence contains "while", disregard the sentence following the "while" and take the sentiment only of the sentence that follows the one after the "while".	My throat is killing me, and *While* I got a decent night's sleep last night, I still feel like I'm about to fall over.	I still feel like I'm about to fall over
R15	If the sentence contains "however", disregard the sentence preceding the "however" and take the sentiment only of the sentence that follows the "however".	@lonedog bwahahah...you are amazing! *However*, it was quite the letdown.	it was quite the letdown.

Table 4.1: Semantic rules

In a tweet, some sub-sentences of a tweet can be un-important and need to be removed because the sentiment overall of a tweet may focus on a sub-sentence only instead of the whole tweet. For example:

- *@lonedog bwahahah...you are amazing! However, it was quite the letdown.*
- *@kirstiealley my dentist is great but she's expensive...=(*

In two above examples, the sentiment overall polarities are negative. However, the sub-sentences expressing the overall sentiment of tweets follow *but* and *however*. This inspires an approach to remove unessential parts in the tweets. We realize that tweets with sub-sentences connected together by Conjunction and Conjunctive adverbs such as *"but, while, however, despite, however"* always have opposite

polarity and the sentiment overall only focus on a sub-sentence instead of the whole tweet. We propose the Semantic rules to handle such cases in tweets. Table 4.1 describes the summarization of the five rules including the examples which are used for the Tweet Processor.

4.3.3 Input Tensor

Input tensor tries to learn the multiple features and converts each flavor-feature to a dense word representation. Dense word representations are concatenated together in order to construct into two Advanced word embeddings:

- Advanced continuous word embedding $v_i = [r_i; e_i; l_i]$ is constructed by three sub-vectors: the continuous word-level embedding $r_i \in R^{d^{word}}$, the character attention embedding $e_i \in R^l$, where l is the length of the filter of wide convolutions, the lexicon embedding $l_i \in R^{d^{score}}$ where d^{score} is list of sentiment scores for that word in lexicon datasets. Advanced continuous word embedding contains semantic and is improved information from LexW2Vs and CharAVs.

- Advanced dependency-based word embedding $d_i = [de_i, e_i; l_i]$ is also built by three sub-vectors: the dependency-based word embedding $de_i \in R^{d^{word}}$, the character attention embedding e_i and the lexicon embedding l_i. The advanced dependency-based word embedding contains syntactic contexts and is enhanced information from LexW2Vs and CharAVs.

These advanced embeddings deal with three main problems:

- Sentences have any different size.

- Important information of characters that can appear at any position in a word is extracted.

- The interactive knowledge between flavor-features are captured via Bi-CGRNNet in order to produce a Tweet-specific representation.

Different from LexW2Vs and DependencyW2Vs, to learn CharAVs, we use a DeepCNN to capture the morphology of each word via character embeddings as Sub-section 4.3.3. The output of DeepCNN is a single real-valued feature. In the next Sub-sections, we introduce methods to build LexW2Vs, CharAVs and DependencyW2Vs features.

Word/Character Embeddings

The different kinds of embeddings are constructed by using a fixed-sized word vocabulary $\mathbb{V}^{word}$ and a fixed-sized character vocabulary $\mathbb{V}^{char}$. Given a word w_i is composed from characters $\{c_1, c_2, ..., c_M\}$, the character-level embeddings are encoded by column vectors u_i in the embedding matrix $W^{char} \in \mathbb{R}^{d^{char} \times |\mathbb{V}^{char}|}$, where $\mathbb{V}^{char}$ is the size of the character vocabulary. For continuous word-level embedding r_{word}, each word $w_i \in \mathbb{V}^{word}$, where $\mathbb{V}^{word}$ is a fixed-sized word vocabulary. The word embedding matrix is simply $W_E \in \mathbb{R}^{d \times \mathbb{V}^{word}}$, where d is the dimension of the word embeddings. Each word embedding w_i is mapped by using a pre-trained word embeddings. The character embeddings are constructed by initialization randomly.

Lexicon Embeddings (LexW2Vs)

Semantic embeddings ignore the sentiment polarity of words in the sentence and map words with similar semantic context but opposite sentiment polarity. To integrate the sentiment polarity for words in the sentence, LexW2Vs are constructed by taking scores from various lexicon datasets. Sentiment lexicons are valuable resources that can be considered much for building embeddings for deep learning models. The sentiment lexicons present the different states of each word in different contexts which can be trained by using neural networks.

In lexicon datasets, each word contains key-value pairs in which the key is a word, and the value is a list of sentiment scores for that word.

For each word $w_i \in V^{word}$, where V^{word} is a fixed-sized word vocabulary, a lexicon embedding is constructed by concatenating all of the scores among lexicon datasets with respect to w_i. If w_i does not exist in a certain dataset, 0 value is substituted. The lexicon embedding is a form of a vector $l_i \in R^{d^{score}}$, where d^{score} is the total number of scores across all lexicon datasets. We use seven lexicon datasets for building LexW2Vs:

- Bing Liu Opinion Lexicon [Hu and Liu, 2004].
- NRC Hashtag Sentiment Lexicon [Mohammad et al., 2013].
- Sentiment140 Lexicon [Go et al., 2009].
- NRC Sentiment140 Lexicon [Go et al., 2009].
- MaxDiff Twitter Sentiment Lexicon [Kiritchenko et al., 2014b].
- National Research Council Canada (NRC) Hash-tag Affirmative and Negated Context Sentiment Lexicon [Kiritchenko et al., 2014b].

- Large-Scale Twitter-specific Sentiment Lexicon [Tang et al., 2014a].

The LexW2Vs capture the different sentiments of words in tweets and is worth incorporating to improve the coverage. Table 4.2 illustrates the type of words for each dataset.

Lexicon dataset	**The type of words**
Bing Liu Opinion Lexicon	Sentiment adjective words
NRC Hashtag Sentiment Lexicon	Hashtag emotion words & Hashtag topic words
Sentiment140 Lexicon	Emoticons & Sentiment words
NRC Sentiment140 Lexicon	Affirmative context words & Sentiment140 Negated Context words
MaxDiff Twitter Sentiment Lexicon	Twitter sentiment words
Hashtag Affirmative and Negated Context Sentiment Lexicon	Hashtag affirmative words & Negated contextual words
Large-Scale Twitter-Specific Sentiment Lexicon	Colloquial words & Emoticons

Table 4.2: The types of words in lexicon dataset.

Lexicon-based features are considered as lexicon embeddings/ sentiment embeddings because they are a feature input of deep learning model and describe the properties of words in tweets. Each word in each lexicon datasets has many values that can be built by training a neural network. The deep learning model uses this input for calculating a computational graph (weight matrix) that describe relatedness among words (n-gram order).

Character Attention Vectors (CharAVs)

In this sub-section, we introduce a neural network to develop a Character attention vector (CharAVs feature) which represents the morphology of each word. Figure 4.3 describes DeepCNN with two *wide convolutions* to form a Character attention vector. The first convolution produces a *fixed-size character feature vector* named *n-gram* features by extracting local features around each character window of the given word and using a max pooling over vertical character windows. The second convolution retrieves the *fixed-size character feature vector* (Character feature matrix) and transforms the representation to yield a Character attention vector. This method is most relevant to our work [Nguyen and Nguyen, 2018]. However, we distinguish their work by improving the second convolution. The second convolution transforms the representation by performing *max pooling* on each row of the

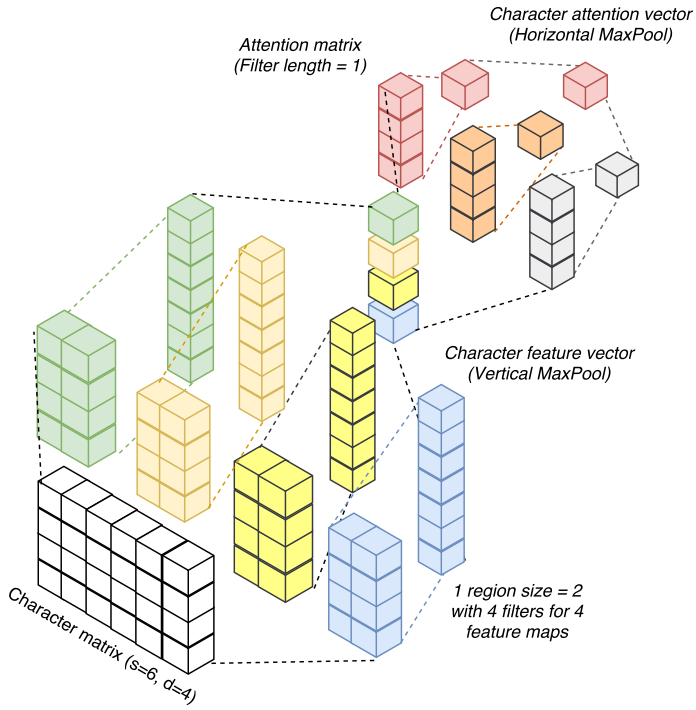


Figure 4.3: DeepCNN for the sequence of character embeddings of a word. For example with one region size is 2 and Four feature maps in the first convolution and one region size is 3 with three feature maps in the second convolution. The CharAVs is then created by performing max pooling on each row of the attention matrix.

Character feature matrix instead of each column. The purpose of this method is to attend on the highest *n-gram* feature in order to transform this *n-gram* features at previous level into representation at a focused abstract level and to produce attention over the best feature vector.

Additionally, DeepCNN is constructed from two *wide convolutions* which can learn to recognize specific *n-grams* at every position in a word and allow features to be extracted independently of these positions in the word. These features maintain the order and relative positions of characters and are formed at a higher abstract level. Character attention vector has two advantages: 1) This model could adaptively assign an importance score to each piece of word embeddings concerning its semantic relatedness. 2) Another advantage is that this attention model is differentiated so that it could be easily trained together with other components in an end-to-end fashion. In the next sub-section, we introduce the background structure of Convoluational Neural Network (CNN) with Wide convolution.

The Effect of Convolutional Neural Network: The convolution is an operation between a vector of weights $m \in R^m$ and a vector of inputs viewed as a sequence $w \in R^w$. The vector m is the filter of the convolution. Concretely, s as an input word and $s_i \in R$ is a single feature value associated with the $i-th$ character in the word. The convolution has a filter vector m and take the dot product of filter m with each *m-grams* in the sequence of characters $s_i \in R$ of a word in order to obtain a sequence c:

$$c_j = m^T s_{j-m+1:j} \tag{4.1}$$

Based on Equation 1, we have two types of convolutions that depend on the range of the index j. The narrow type requires that $s \geq m$ and produce a sequence $c \in R^{s-m+1}$. The wide type does not require on s or m and produce a sequence $c \in R^{s+m-1}$. Out-of-range input values s_i where $i < 1$ or $i > s$ are taken to be zero. The result of the narrow convolution is a subsequence of the result of wide convolution.

Wide Convolution: Given a word w_i composed of M characters $\{c_1, c_2, ..., c_M\}$, we take a character embedding $u_i \in R^{d^{char}}$ for each character c_i and construct a character matrix $W^{char} \in R^{d^{char}} \times |V^{char}|$ as following Equation 4.2:

$$W^{char} = \begin{bmatrix} | & | & | \\ u_1 & ... & u_M \\ | & | & | \end{bmatrix} \tag{4.2}$$

.

The values of the embeddings u_i are parameters that are optimized during training. The trained weights in the filter m correspond to a feature detector which learns to recognize a specific class of *n-grams*, where $n \leq m$, and m is the width of the filter. The use of a wide convolution has more advantages than a narrow convolution because a wide convolution ensures that all weights of the filter m reach the whole characters of a word at the margins. Besides, the wide convolution guarantees that the filter m always produces a valid non-empty result c, independently of the width of m and the sequence length s. Therefore, this is particularly significant when the width of the filter m is set from 7 to 14 to represent a word. The resulting matrix has dimension $d \times (s + m - 1)$.

Dependency-based Word Vectors (DependencyW2Vs)

To construct context embeddings, we use the idea of [Levy and Goldberg, 2014] to derive syntactic contexts based on the syntactic relations of a word. Most previous works on neural word embeddings take the contexts of a word by computing *linear-context* words that precede and follow the target word. However, these contexts can be exploited similar by generalizing the *SKIP-GRAM* model. The model for

learning Dependency-based Word Vectors is improved from *SKIP-GRAM* model in which the linear bag of words contexts are replaced with arbitrary word contexts from a dependency tree. Syntactic contexts are derived from produced dependency parse-trees. Specifically, the bag-of-words in the *SKIP-GRAM* model yield *broad topical similarities*, while the dependency-based contexts yield more *functional similarities* of a *cohyponym nature*. In the *SKIP-GRAM* model, the contexts of a word are the words surrounding it in the text. However, there is a limitation of *SKIP-GRAM* word embeddings: Contexts no need to correspond to all of the words and the number of context-types maybe larger than the number of word-types. Therefore, dependency-based contexts capture more information than bag-of-words contexts. In Figure 4.4, the contexts are extracted for each word in the sentence, and the contexts of a word are derived from syntactic relations of a word in the sentence. For parsing syntactic dependencies, we use a parser from [Goldberg and Nivre, 2013] for Stanford dependencies and the corpus are tagged with parts-of-speech using Stanford parser [1].

After parsing each sentence, we consider word context as Figure 4.4: For a target word w with modifiers $m_1, m_2, ..., m_n$ and a head h, we form the contexts as $(m_1, lbl_1), ..., (m_n, lbl_n), (h, lbl_h^{-1})$, where lbl is the type of the dependency relation between the head and the modifier, lbl^{-1} is used to marking the inverse-relation. The advantages of syntactic dependencies are inclusive and more focused than bag-of-words. Besides, they can capture relations that out-of-reach with small windows and filter out contexts that are not directly related to the target word. For example, *Australian* is not used as the context for *discovers*. We have more focused embeddings that capture more functional and less topical similarity. As such, DependencyW2Vs contains syntactic information captured via dependency trees.

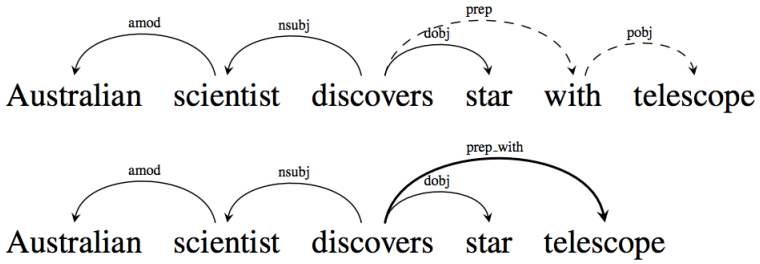
4.3.4 Contextual Gated Recurrent Neural Network (CGRN-Net)

In this section, we describe the formulation of CGRNNet architecture from scratch based on a Gated Recurrent Neural Network.

Gated Recurrent Neural Network

The Bi-GRNN is a version of [Cho et al., 2014] in which a Gated Recurrent Unit (GRU) has two gates, a reset gate r_t and an update gate z_t. Intuitively, the reset gate determines how to combine the new input with the previous memory, and the update gate defines how much of the previous memory to keep around. The

[1] https://nlp.stanford.edu/software/lex-parser.shtml



WORD	CONTEXTS
australian	scientist/amod^{-1}
scientist	australian/amod, discovers/nsubj^{-1}
discovers	scientist/nsubj, star/dobj, telescope/prep_with
star	discovers/dobj^{-1}
telescope	discovers/prep_with^{-1}

Figure 4.4: Dependency-based context extraction example [Levy and Goldberg, 2014]

idea behind a GRU layer is quite similar to that of an LSTM layer. The basic idea of using a gating mechanism to learn long-term dependencies is the same as in an LSTM, but there are a few key differences: a) A GRU has two gates, an LSTM has three gates. b) GRUs do not possess an internal memory (c_t) that is different from the exposed hidden state. They do not have the output gate that is present in LSTMs. c) The input and forget gates are coupled by an update gate z, and the reset gate r is applied directly to the previously hidden state. Thus, the responsibility of the reset gate in an LSTM is split up into both r and z. d) A second nonlinearity is not applied when computing the output.

We use GRUs for our model instead of LSTMs because GRUs are quite new and their tradeoffs have not been fully explored yet. On the other hand, GRUs have fewer parameters (U and W are smaller) and thus may train a bit faster or need fewer data to generalize. The equation 4.3 illustrates the construction of a GRU cell:

$$\begin{aligned} r_t &= \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r) \\ z_t &= \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z) \\ \hat{h}_t &= g(x_tW_{xh} + (r_t \odot h_{t-1})W_{hh} + b_h) \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t. \end{aligned} \tag{4.3}$$

Contextual Gated Recurrent Neural Network

Based on the idea of GRNN model, we build a power of syntactic contexts into a standard Bi-GRNN model which adapts GRNN cell to take both words and syntactic contexts by modifying the equations representing the operations of the GRNN cell. A GRNN cell is added a Dependency-based word vector T to reset gate, update gate and hidden state. In the Equation 4.4, the term in bold is the modification made to the original GRNN equation. Based on these equations, adding the Dependency-based word vector T is corresponding to considering a composite input $[x_i, T]$ to the GRNN cell that concatenates the Advanced continuous word embeddings and Advanced dependency-based word embeddings.

$$\begin{aligned} r_t &= \sigma(W_{xr}x_t + W_{hr}h_{t-1} + \mathbf{W_{Ti}T} + b_r) \\ z_t &= \sigma(W_{xz}x_t + W_{hz}h_{t-1} + \mathbf{W_{Ti}T} + b_z) \\ \hat{h}_t &= g(x_tW_{xh} + (r_t \odot h_{t-1})W_{hh} + \mathbf{W_{Ti}T} + b_h) \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t. \end{aligned} \tag{4.4}$$

This approach of concatenating syntactic contexts and word embeddings works better in practice and deal with the context challenge in sentiment analysis. The CGRNNet is constructed into Bi-CGRNNet and take two Advanced word embeddings from Input Tensor as input. Bi-CGRNNet computes the interactive knowledge between these Advanced word embeddings and controls the information via the Gating mechanism. Word relationship is calculated in order to produce a final Tweet-specific representation.

4.3.5 Model Training

The deep learning model is trained in a supervised manner by minimizing the cross-entropy error of sentiment classification. The deep learning model can be trained in an end-to-end way by back-propagation to calculate the gradients of all the parameters, and update them with stochastic gradient descent. The goal of training is to minimize the cross-entropy error between y and $\hat{y}$ for all sentences, where y be the target distribution for sentence, $\hat{y}$ be the predicted sentiment distribution.

$$loss = -\sum_i \sum_j y_i^j log\hat{y}_i^j + \lambda||\theta||^2 \tag{4.5}$$

4.4 Evaluation

This section shows models are used to comparing to our model and presents an evaluation metric for comparison. We perform *Binary classification* for evaluation.

To evaluate the significant of our model, a series of evaluations is conducted in which Accuracy and F1-measures is applied. We set the sizes of word embeddings amongst {50, 100, 200, 300} and apply multiple Word2Vec such as *GoogleW2V*, *Twitter GloveW2V* and *SSWE* **?**.

4.4.1 Datasets and Experimental Setting

Three datasets are used for evaluating binary classification accuracy of our model. These datasets are popular and applied in many early studies:

- *Stanford - Twitter Sentiment Corpus (STS Corpus):* STS Corpus contains 1,600K training tweets collected by a crawler from [Go et al., 2009]. [Go et al., 2009] constructed a test set manually with 177 negative and 182 positive tweets. The Stanford test set is small. However, it has been widely used in different evaluation tasks [Go et al., 2009], [Santos and Gatti, 2014], [Bravo-Marquez et al., 2013].

- *Sanders - Twitter Sentiment Corpus:* This dataset consists of hand-classified tweets collected by using search terms: *#apple*, *#google, #microsoft and #twitter*. We construct the dataset as [da Silva et al., 2014] for binary classification.

- *Health Care Reform (HCR):* This dataset was constructed by crawling tweets containing the hashtag *#hcr* [Speriosu et al., 2011]. The task is to predict positive/negative tweets [da Silva et al., 2014].

The summary statistics of datasets is described in Table 4.3.

Hyperparameters

Table 4.4 shows the summary of Hyperparameters which is applied to our model. Training is done through stochastic gradient descent over shuffled mini-batches with Adadelta update rule.

Pre-trained Word Vectors

The publicly available *Word2Vec*[2] is trained from 100 billion words from Google and *Twitter Glove*[3] of Stanford is performed on aggregated global word-word co-occurrence statistics from a corpus. Additionally, three kinds of Sentiment-Specific Word Embedding for Twitter sentiment classification *(SSWE)* from [Tang et al.,

[2]code.google.com/p/word2vec/

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

Data	Set	N	c	l_w	l_c	$\|V_w\|$	$\|V_c\|$	*Test*
STS	Train	80K	2	33	110	67083	134	-
	Dev	16K		28	48			
	Test	359		21	16			
Sanders	Train	991	2	31	33	3379	84	CV
	Dev	110		27	47			
	Test	122		28	21			
HCR	Train	621	2	25	70	3100	60	-
	Dev	636		26	16			
	Test	665		20	16			

Table 4.3: Summary statistics for the datasets after using semantic rules. c: the number of classes. N: The number of tweets. l_w: Maximum sentence length. l_c: Maximum character length. $|V_w|$: Word alphabet size. $|V_c|$: Character alphabet size.

Hyperparameters	**STS**	**HCR**	**Sanders**
Mini-batch Size	100	4	4
Filter Window Size for Layer 1	7, 6		
Filter Window Size for Layer 2	1, 14		
Dropout Rate	0.5		
l_2 Constraint	True		
Learning Rate	0.1		
Momentum	0.9		

Table 4.4: The summary of hyperparameters

2014b] are applied. These pre-train word embeddings are used for cross comparison. *Word2Vec* has the dimensionality of 300, *Twitter Glove* has the dimensionality of 200 and *SSWEs* have the dimensionality of 50. Words that do not present in the set of pre-train words are initialized randomly.

Evaluation Metric

In the Stanford Twitter Sentiment Corpus (STS Corpus), the training data is selected 80K tweets randomly, and the development set is selected 16K tweets randomly from the training data. The STS dataset is a corpus from [Go et al., 2009].

For Sander dataset, standard 10-fold cross validation is conducted for Binary classification. A development set is selected about 10% randomly from 9-fold training data.

In Health Care Reform Corpus (HCR Corpus), the development set is selected 10% randomly in a training set.

For evaluation measures, we apply Accuracy and F1 measures [Balikas and Amini, 2016] to evaluate the effectiveness of our proposed model. The evaluation F1 - measure F_1^{PN} is as follows:

$$F_1^{PN} = \frac{F_1^P + F_1^N}{2} \tag{4.6}$$

Where F_1^P is the F1 score for the $POSITIVE$ class:

$$F_1^P = \frac{2\pi^P \rho^P}{\pi^P + \rho^P} \tag{4.7}$$

Here, π^P and ρ^P denote precision and recall for the $POSITIVE$ class, respectively:

$$\pi^P = \frac{PP}{PP + PU + PN} \tag{4.8}$$

$$\rho^P = \frac{PP}{PP + UP + NP} \tag{4.9}$$

where PP, UP, NP, PU, PN are the cells of the confusion matrix shown in Table 4.5. Additionally, the evaluation accuracy measure is as follows:

$$Accuracy = \frac{\sum_{i=1...N}(1 - (target_i - threshold(f(\overrightarrow{x_i}))))}{N} \tag{4.10}$$

Where $threshold(f(\overrightarrow{x_i}))$ equal 0/1.

		Gold Standard		
		POSITIVE	*NEUTRAL*	*NEGATIVE*
Predicted	*POSITIVE*	PP	PU	PN
	NEUTRAL	UP	UU	UN
	NEGATIVE	NP	NU	NN

Table 4.5: The confusion matrix.

4.4.2 Baselines

We compare our proposed model to strong state-of-the-art models on the Twitter datasets. The methods are separated into two classes: Traditional methods and Deep learning methods as follows:

- (SVM, Maximum Entropy (MaxEnt), Random Forest (RF), Logistic Regression (LR), Multinominal Naive Bayes (MNB))-BoW + lex [da Silva et al., 2014]: Traditional separate methods are combined with Bag-of-Word (BoW), feature hashing (FH), and lexicon (lex).

- ENS(SVM + RF + LR + MNB)-BoW + lex [da Silva et al., 2014]: an ensemble model incorporating SVM, Random Forest, Logistic Regression, and Multinominal Naive Bayes together is combined with Bag-of-Word (BoW), feature hashing (FH), and lexicon (lex).

- Naive Bayes (NB) - NG + POS + SF [Saif et al., 2012]: the model of [Saif et al., 2012] using a Naive Bayes is combined with N-Gram (NG), Part-of-Speech (POS) and Semantic feature (SF) as useful features.

- RAE [Socher et al., 2011]: is a semi-supervised Recursive auto-encoders network (RAE) for sentence-level sentiment classification, which obtains a reduced dimensional vector representation for a sentence.

- MAX-TDNN [Collobert et al., 2001]: is a Unified Neural Network architecture and learning algorithm that can be applied to various natural language processing tasks. The model learns internal representations from vast amounts of mostly unlabeled training data instead of exploiting human-made input features optimized for each task.

- DCNN [Kalchbrenner et al., 2014]: is a generalization of MAX-TDNN. Dynamic CNN (DCNN) is for semantic modeling of sentences by improving from standard CNN. The DCNN uses Dynamic max-pooling as Non-linear subsampling function.

- RNN-FLT, RNN-TLT, LSTM-TLT [Wang et al., 2015]: Long Short-Term Memory Recurrent Networks for Twitter sentiment prediction. With the help of Gating mechanisms and Constant error carousels in a Memory block structure, the model could handle interactions between words through a flexible compositional function.

- CharSCNN [Santos and Gatti, 2014]: a Deep learning model uses Convolution Neural Network (CNN). The model of Santos and Gatti [2014] applies two convolutions to capture the information of characters instead of words in a sentence.

Model	**STS**	**HCR**	**Sander**
MaxEnt	83.00	-	-
NB	82.70	-	-
SVM	82.20	-	-
SVM-BoW	-	73.99	82.43
SVM-BoW + lex	-	75.94	83.98
RF-BoW	-	70.83	79.24
RF-BoW + lex	-	72.93	82.35
LR-BoW	-	73.83	77.45
LR-BoW + lex	-	74.73	79.49
MNB-BoW	-	72.48	79.82
MNB-BoW + lex	-	75.33	83.41
ENS (RF + MNB + LR) - BoW	-	75.19	-
ENS (SVM + RF + MNB + LR) - BoW + lex	-	**76.99**	-
ENS (SVM + RF + MNB + LR) - BoW	-	-	82.76
ENS (SVM + RF + MNB) - BoW + lex	-	-	**84.89**
CharSCNN/Pre-trained	86.40	-	-
CharSCNN/Random	81.90	-	-
SCNN/Pre-trained	85.20	-	-
SCNN/Random	82.20	-	-
MAX-TDNN	78.80	-	-
DCNN	**87.4**	-	-
RAE	77.60	-	-
RNN-FLT	80.20	-	-
RNN-TLT	86.40	-	-
LSTM-TLT	87.20	-	-
Bi-CGRNN + CharAVs + LexW2Vs + GoogleW2Vs + SR	**88.57**	78.47	84.96
Bi-CGRNN + CharAVs + LexW2Vs + GloveW2Vs + SR	87.74	81.00	**85.79**
Bi-CGRNN + CharAVs + LexW2Vs + SSWE-h + SR	85.23	77.74	84.69
Bi-CGRNN + CharAVs + LexW2Vs + SSWE-r + SR	84.67	74.58	85.42
Bi-CGRNN + CharAVs + LexW2Vs + SSWE-u + SR	84.67	79.24	85.42
Bi-CGRNN + CharAVs + LexW2Vs + GoogleW2Vs	84.67	78.79	83.40
Bi-CGRNN + CharAVs + LexW2Vs + GloveW2Vs	83.84	**80.00**	85.32
Bi-CGRNN + CharAVs + LexW2Vs + SSWE-h	83.56	78.34	83.77
Bi-CGRNN + CharAVs + LexW2Vs + SSWE-r	82.72	78.79	85.60
Bi-CGRNN + CharAVs + LexW2Vs + SSWE-u	81.33	77.74	85.42

Table 4.6: Accuracy of different models for binary classification.

Model	Pos. F1	Neg. F1	Avg. F1
STS Corpus			
SVM-BoW	68.50	66.30	67.40
SVM-BoW + lex	74.90	72.70	73.80
RF-BoW	69.10	63.60	66.35
RF-BoW + lex	75.70	72.90	74.30
LR-BoW	72.70	50.80	61.75
LR-BoW + lex	78.30	74.00	76.15
MNB-BoW	71.50	71.10	71.30
MNB-BoW + lex	79.30	79.40	79.35
ENS (SVM + RF + LR + MNB)-BoW	73.80	70.20	72.00
ENS (SVM + RF + LR + MNB)-BoW + lex	81.80	80.20	81.00
NB-NG + POS + SF	**82.50**	**85.30**	**83.90**
Bi-CGRNN + CharAVs + LexW2Vs + SR + GoogleW2Vs	91.01	**89.01**	**90.01**
Bi-CGRNN + CharAVs + LexW2Vs + SR + GloveW2Vs	92.31	86.91	89.60
Bi-CGRNN + CharAVs + LexW2Vs + SR + SSWE-h	92.30	84.98	88.64
Bi-CGRNN + CharAVs + LexW2Vs + SR + SSWE-r	93.25	84.05	88.65
Bi-CGRNN + CharAVs + LexW2Vs + SR + SSWE-u	**92.62**	84.24	88.43
HCR Corpus			
SVM-BoW	36.60	83.60	60.10
SVM-BoW + lex	41.60	84.80	63.20
RF-BoW	31.70	81.50	56.60
RF-BoW + lex	32.30	83.10	57.70
LR-BoW	31.50	83.80	57.65
LR-BoW + lex	33.90	84.40	59.15
MNB-BoW	48.50	81.20	64.85
MNB-BoW + lex	**53.10**	83.30	**68.20**
ENS (SVM + RF + LR + MNB)-BoW	35.80	84.60	60.20
ENS (SVM + RF + LR + MNB)-BoW + lex	41.80	85.70	63.75
NB-NG + POS + SF	50.30	**86.00**	68.15
Bi-CGRNN + CharAVs + LexW2Vs + SR + GoogleW2Vs	**60.63**	76.31	68.47
Bi-CGRNN + CharAVs + LexW2Vs + SR + GloveW2Vs	55.39	86.72	**71.06**
Bi-CGRNN + CharAVs + LexW2Vs + SR + SSWE-h	18.82	85.94	52.38
Bi-CGRNN + CharAVs + LexW2Vs + SR + SSWE-r	25.98	84.84	55.41
Bi-CGRNN + CharAVs + LexW2Vs + SR + SSWE-u	43.65	**87.31**	65.48

Table 4.7: Cross comparison results for different traditional methods. LR, RF, SVM, MNB and NB refer to Logistic Regression, Random Forest, Support Vector Machine, Multinominal Naive Bayes and Naive Bayes, respectively. BoW refers to Bag-of-Words, lex refers to lexicon, NG refers to N-gram, POS refers to Part-of-Speech and SF refers to Semantic.

Model	Pos. F1	Neg. F1	Avg. F1
Sanders Corpus			
SVM-BoW	81.50	83.30	82.40
SVM-BoW + lex	83.20	84.70	83.95
RF-BoW	78.60	79.80	79.20
RF-BoW + lex	81.80	82.90	82.35
LR-BoW	75.50	79.10	77.30
LR-BoW + lex	78.30	80.60	79.45
MNB-BoW	77.70	81.60	79.65
MNB-BoW + lex	82.00	84.60	83.30
ENS (SVM + RF + LR + MNB)-BoW	81.70	83.70	82.70
ENS (SVM + RF + LR + MNB)-BoW + lex	**84.20**	**85.50**	**84.85**
Bi-CGRNN + CharAVs + LexW2Vs + SR + GoogleW2Vs	90.46	85.51	87.98
Bi-CGRNN + CharAVs + LexW2Vs + SR + GloveW2Vs	88.63	79.31	83.97
Bi-CGRNN + CharAVs + LexW2Vs + SR + SSWE-h	91.18	84.78	87.98
Bi-CGRNN + CharAVs + LexW2Vs + SR + SSWE-r	**92.49**	85.65	**89.07**
Bi-CGRNN + CharAVs + LexW2Vs + SR + SSWE-u	91.33	**85.87**	88.60

Table 4.8: Cross comparison results for different traditional methods. LR, RF, SVM, MNB and NB refer to Logistic Regression, Random Forest, Support Vector Machine, Multinominal Naive Bayes and Naive Bayes, respectively. BoW refers to Bag-of-Words, lex refers to lexicon, NG refers to N-gram, POS refers to Part-of-Speech and SF refers to Semantic.

4.4.3 Experimental results

Table 4.6 shows the accuracy results of our model compared to other models. The variations of our model are constructed to evaluate the effectiveness of our model on classification accuracy. We compare our model performance with the approaches of [Go et al., 2009], [Socher et al., 2011], [Collobert et al., 2001], [Saif et al., 2012], [Kalchbrenner et al., 2014], [Santos and Gatti, 2014], [Wang et al., 2015] for STS corpus.

The models of [Go et al., 2009], [Socher et al., 2011], [Collobert et al., 2001], [Saif et al., 2012], [Kalchbrenner et al., 2014], [Santos and Gatti, 2014] display the good results in previous time and the model of [Wang et al., 2015] reports the state-of-the-art model so far. Our model shows the best prediction accuracy for STS Corpus.

For Sanders and HCR Corpus, we compare the performance with the models of [Saif et al., 2012], [da Silva et al., 2014] using the ensemble of multiple base classifiers (ENS) such as Naive Bayes (NB), Random Forest (RF), Support Vector Machine (SVM) and Logistic Regression (LR). The ENS model of [da Silva et al., 2014] is combined with Bag-of-Words (BoW), Feature hashing (FH) and Lexicons (lex). The model of [Saif et al., 2012] utilizes N-gram, Part-of-Speech and Semantic

Word embedding	Size 50	Size 100	Size 200	Size 300
STS Corpus				
SSWE-h	85.23	-	-	-
SSWE-r	84.67	-	-	-
SSWE-u	84.67	-	-	-
Twitter Glove	87.74	87.74	86.9	-
GoogleW2V	-	-	-	**88.57**
HCR Corpus				
SSWE-h	77.74	-	-	-
SSWE-r	74.58	-	-	-
SSWE-u	79.24	-	-	-
Twitter Glove	77.89	78.34	**80**	-
GoogleW2V	-	-	-	78.47
Sanders Corpus				
SSWE-h	84.69	-	-	-
SSWE-r	85.42	-	-	-
SSWE-u	85.42	-	-	-
Twitter Glove	83.41	85.60	**85.79**	-
GoogleW2V	-	-	-	84.96

Table 4.9: Accuracy of models using the different sizes of word embeddings for binary classification.

Model	STS	HCR	Sander
Bi-CGRNN + GoogleW2Vs	85.70	77.74	83.68
Bi-GRNN + CharAVs + GoogleW2Vs	86.00	**79.09**	83.41
Bi-GRNN + LexW2Vs + GoogleW2Vs	**88.00**	78.79	**84.42**

Table 4.10: Accuracy of models using GoogleW2Vs for binary classification.

Model	STS	HCR	Sander
Bi-CGRNN + GloveW2Vs	84.95	76.99	83.96
Bi-GRNN + CharAVs + GloveW2Vs	86.35	**79.25**	82.77
Bi-GRNN + LexW2Vs + GloveW2Vs	**88.02**	78.34	**84.6**

Table 4.11: Accuracy of models using GloveW2Vs for binary classification.

feature as effective features for Naive Bayes. Our model outperforms the model of [da Silva et al., 2014] and [Saif et al., 2012] so far. To sum up, Figure 4.5 illustrates the summary of the accurate comparison of the models for each dataset.

On the other hand, the cross-comparison between the different traditional methods and our model are conducted for F1-measure in Table 4.7 and Table 4.8. Our model outperforms other models on three datasets. In order to evaluate the effectiveness of the separate flavor-features, the experiments of them are controlled one-by-one and showed in Table 4.10 and Table 4.11. Additionally, the experiments of the different size of Word embeddings are handled to evaluate the effectiveness of the word embeddings in Table 4.9 and Figure 4.6.

4.4.4 Analysis

Table 4.6 indicates that our model built on top of *GoogleW2Vs* is more effective than *GloveW2Vs* for STS corpus meanwhile *GloveW2Vs* is effective on HCR and Sanders corpus. We believe *GoogleW2Vs* captures many words more than *GloveW2Vs* and *SSWEs* for STS dataset. However, the experiment results of *SSWEs* and *GloveW2Vs* still achieve the good performance compared to early models. Generally, the models assisted by Semantic rules are more effective than the models without Semantic rules.

In Table 4.7 and Table 4.8, *SSWEs* performs well for negative tweets compared to other Word embeddings in HCR and Sanders corpus. *GoogleW2Vs* still works well for STS Corpus and *GloveW2Vs* is for HCR Corpus.

Additionally, we can observe that the experimental results of the separate flavor-features in Table 4.10 and 4.11 show that CharAVs and LexW2Vs achieve good performance and contribute to enhancing information of words. LexW2Vs is more effective than CharAVs on STS and Sanders datasets, whereas, Char-

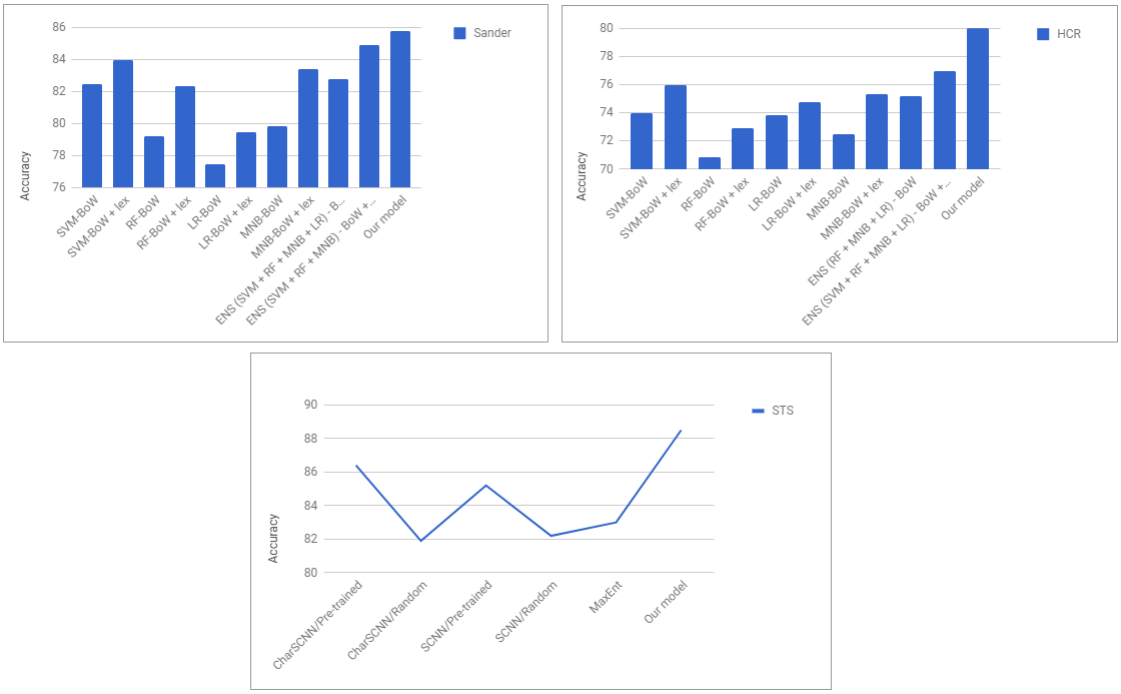


Figure 4.5: The chart of accuracy comparison for each corpus

AVs is better than LexW2Vs on HCR dataset. On the other hand, when the flavor-features are constructed separately, *GloveW2Vs* is more impressive than *GoogleW2V*. Because *GloveW2Vs* is Twitter embeddings and sufficient for Twitter-level classification since the datasets are Twitter datasets.

Besides, the size of Word embeddings affects the classification accuracy as well. Table 4.9 indicates the model achieves good performance with the larger word embedding size. However, for *Twitter Glove*, the embedding size of *100* is better than other sizes for STS corpus.

For error analysis, we show the experimental results in Table 4.12 to show the effectiveness of our model in predicting some cases. We can observe that the Bi-GRNNet using LexW2Vs captures the positive words (green words) and negative words (red words) for computing scores and predicts wrong labels due to the contexts of tweets meanwhile the Bi-CGRNNet enhanced with CharAVs and LexW2Vs can recognize contexts of the tweet. For example, the model using LexW2Vs predicts the third tweet to be positive because the *'strong'* word has sentiment stronger than *'no'* word, however, the context of this tweet is negative. The syntactic feature supports in dealing context problems in tweets, and the lexicon feature supports in dealing the sentiment of tweets. The differences between our model and other approaches are the ability of our model to capture multiple

Figure 4.6: The chart of accuracy comparison on the different sizes of word embeddings for each corpus

features, combine these features at the high level and increase the information of words in tweets. Besides, the usage of DeepCNN for characters can learn a structure of words at a higher abstract level. LexW2Vs and syntactic contexts contribute to support information for word embeddings. This helps the model not only learns to recognize single n-grams of a word, negation, but also patterns in n-grams towards to form a structure significance of a tweet.

4.5 Conclusions

This chapter proposes a novel approach which tries to learn the different perspective of each word via multiple features. There are three main contributions of our proposed method. First, we propose a Tweet processor combined with Semantic rules to deal with the unique properties of Twitter social networking. Second, flavor-features which represent the characteristics of each word in a tweet are developed in order to attend the contextual sentiment words of the tweet. Third, Bi-GRNNet is proposed to capture the semantics of words and tries to learn a tweet-specific representation via the multiple perspectives of words. We can observe that the multiple useful features are important ingredients in increasing classification accuracy for Twitter sentiment classification. Additionally, this works is an effort to see the effectiveness of pre-processing on twitter data for the

Model		Input from HCR Corpus	Gold Label	Prediction
Bi-CGRNN CharAVs LexW2Vs	+ +	seanbaran74 well that's what's next. after #hcr they'll save the environment, give us CFLs and take away our TVs.	Negative	True
Bi-GRNN LexW2Vs	+			False
Bi-CGRNN CharAVs LexW2Vs	+ +	All of us fighting for #HCR ask ourselves who #imherefor. Who are you fighting for? http://bit.ly/9-st	Positive	True
Bi-GRNN LexW2Vs	+			False
Bi-CGRNN CharAVs LexW2Vs	+ +	Stephen Lynch strong 'no' on health bill despite talk with President obama http://bit.ly/cQIujP #hcr #tcot #tlot	Negative	True
Bi-GRNN LexW2Vs	+			False
Bi-CGRNN CharAVs LexW2Vs	+ +	Another reason we need #HCR now. RT @GordBarnes 15 Executives Who Get Paid Millions To Deny You Health Care Coverage	Negative	True
Bi-GRNN LexW2Vs	+			False
Bi-CGRNN CharAVs LexW2Vs	+ +	@MPOTheHill: Rep. Allen Boyd (D-Fla.) will vote for #hcr. A flip from no to yes. http://bit.ly/9nTKHH	Positive	True
Bi-GRNN LexW2Vs	+			False

Table 4.12: The label prediction between the Bi-GRNN model using LexW2Vs and the Bi-CGRNN model using CharAVs and LexW2Vs (The red words are negative, and the green words are positive).

fortification of sentiment classification, especially regarding semantic rules.

Our results indicate that the flavor-features are useful for the deep learning model to improve classification performance. Our model outperforms other models which mainly utilized the simple word embeddings and improved the architecture of deep neural networks. The main advantage of deep neural networks is trying to learn the probability distribution of a sequence in which words are recognized and differentiated from others. This means that the semantics of a word and the relationship between words are captured better if the sequential data has specific perspectives which represent the nature of that sequence.

Chapter 5

Aspect-level Sentiment Analysis

In this chapter, we propose novel methods to tackle the challenges of the aspect-level task. A **L**exicon-Aware **W**ord-**A**spect **A**ttention **N**etwork (LWAAN) and a **D**eep **M**emory **N**etwork-in-**N**etwork (DMNN) are proposed by using effective multiple attention mechanisms and lexicon information to form an aspect-specific representation at two levels: Phrase level and Context level. In order to deal with this, the aspect and its context in a sentence are treated separately and learn their representations by attention mechanisms. Additionally, the important information of aspect and its context are highlighted by the sentiment lexicons and encoded by Long Short-Term Memory (LSTM) to produce a lexicon pooling aspect vector. Such the pooling aspect vector is to preserve the information of full context aspect and increase the information of the aspect-specific representation. To evaluate the performance, we construct many kinds of models and evaluate our models in three domains: Twitter, Laptop, and Restaurant. The experimental results indicate that our models improve the performance for aspect-level sentiment classification.

5.1 Introduction

With the advent of social networking websites such as Facebook, Twitter, and Flickr as well as the development of machine learning technology, we have observed an increase in the number of opinions shared by people on social networking. The people often share their opinions about the aspects of an event and a product through social websites. Therefore, a large of the number of such useful data can be widely applied in public opinions analysis and product recommendation. These works are involved in the problem of aspect-level sentiment classification in which aspects can be identified as the aspects of the product or the event. For instance, a company would like to know the quality of the *"screen"* of a phone or the *people situation* after an earthquake (an event).

Recent years, the ASA task has grown to be one of the most active research areas in natural language processing (NLP) and become important to business and society. This work is involved in the problem of modeling the relationship of a specific aspect term and its context. In order to tackle this, traditional machine learning approaches and lexicon-based approaches were utilized in the first time by [Go et al., 2009] [Liu, 2010] [Saif et al., 2012] and [Kiritchenko et al., 2014a]. While most of the traditional machine learning is supervised machine learning (e.g., Support Vector Machine, Maximum Entropy, Naive Bayes) which requires significant laborious feature engineering, lexicon-based approaches were applied as additional features for the traditional machine learning models. Specifically, lexicon-based approaches mainly use knowledge-based or lexicon-based methods, which utilize public available lexicon resources (e.g., WordNet, SentiWordNet) and classify the sentiment of texts based on the overall sentiment polarity of lexicons [Taboada et al., 2011]. However, the drawbacks of these methods are not capable of modeling the semantic relationship between an aspect and its context sufficiently. Additionally, another problem with these methods is difficult to adapt well to different domains or different languages. As such, the task of ASA introduces a challenging problem of incorporating aspect information into learning models for making predictions.

With the success of deep neural networks using the distributed representations of words (Word embeddings) to merge word representations to represent phrases or sentences, such models are capable of capturing the semantic relation between an aspect and its context without the particular features engineering. More specifically, end-to-end neural networks ([Dong et al., 2014], [Wang et al., 2016c], [Sukhbaatar et al., 2015], [Tang et al., 2016b], [Ma et al., 2017], [Liu and Zhang, 2017] and [Chen et al., 2017]) have demonstrated promising performance on aspect-level sentiment analysis tasks without requiring any laborious feature engineering. Such models can incorporate aspect information into neural architectures by learning to attend the different parts of a context sentence towards a given aspect term via an attention mechanism. Furthermore, such models are the attention-based LSTM models which try to fuse aspect information by adopting a naive concatenation of an aspect and its context words to extract important parts towards the given aspect. Consequently, these models meet the following drawbacks: First, the simple concatenation causes an extra burden for attention layers of modeling sequential information dominated by aspect information. Second, most of the models are heavily rooted in LSTM networks as well and do not treat an aspect and its context separately. As such, it incurs additional parameter costs to LSTM layers towards to hardly model the relationship between the aspect and its context words. Third, the attention-based models assume that the words of an aspect have the equal contribution, while the aspect information should be

an important factor for judging the aspect sentiment polarity. Finally, these models capture the correct context words based on the semantics of pre-trained word embeddings (e.g., *Glove*) that ignore the sentiments of the words.

To overcome the above challenges, we propose a deep neural network which treats an aspect and its context separately and utilizes multiple attention mechanisms to focus on the crucial parts of the aspect and its context. The attention mechanisms are (*Intra attention* and *Interactive attention*)in which the intra-attention mechanism is to extract the critical parts of an aspect (informative phrase-level information) first and then, learn the word-aspect relationship between the informative words of the aspect and its sentiment context words via the interactive-attention mechanism. More specifically, an aspect and its context words are augmented by sentiment lexicon information to form lexicon-augmented word embeddings first. The purpose of the lexicon information is to enforce the model to pay more attention to the sentiment of words instead of only the semantics of the words. Then, the lexicon-augmented word embeddings are encoded into their representations by LSTM encoders. Subsequently, the vital information words of the aspect are captured via an intra-attention mechanism and are utilized to learn to attend correct sentiment context words. Furthermore, to capture the different perspectives of each sentiment context word, we try to develop two kinds of the aspect representations (the phrase-level representation and the aggregation-level representation to compute two interactive-attention vectors and interact knowledge between them.

Our model performs well and tackles the challenges of the aspect-level task via multiple attention mechanisms. However, there is a remaining challenge that the attention cannot allow the model to consider the entire history explicitly so far and look back previous examples which are relevant. We propose a Deep Memory Network-in-Network (DMNN) to tackle this by using an iterative attention mechanism. The iterative attention mechanism is developed by constructing an interactive attention mechanism into many computational layers. This mechanism is a *searching* mechanism which confirms the internal representation of context words based on the relevant information from an aspect many times and tries to extract the correct sentiment words.

Our contributions:

The principal contributions of this chapter are as follows:

- A novel model is developed to try to learn the associative word-aspect relationship via the multiple attention mechanisms.
- Lexicon information is proposed to highlight the important information of an aspect and its context via lexicon-augmented word embeddings. These embeddings enforce the model to pay more attention to the sentiment context words in a sentence via multiple attention mechanisms.

- We conduct a comprehensive and in-depth analysis of the inner workings of our proposed model.

In the remaining sections, the task definition of aspect-level sentiment classification is formed in Section 5.2.1. The proposed models are described in Section 5.2. We express the experiments, and analysis in Section 5.3 and finish by drawing important conclusions.

5.2 Proposed Models

In this section, an ILWAAN model is introduced to learn to attend correct sentiment context words given an aspect term. Subsequently, the variants of the ILWAAN model is developed to conduct the partial evaluation of our model to assess the significance of our model for aspect-level sentiment analysis. To the best understanding, we illustrate the architecture of the ILWAAN model layer-by-layer with the functionality of each component.

5.2.1 Task definition

This section describes the ultimate goal of our model in identifying the sentiment polarity of an aspect in the context of a tweet/ sentence. the tweet consists of a sequence of n words, denoted as $S = \{w_1^C, w_2^C, ..., w_i^C, ..., w_N^C\}$ with with w_i referring to the position of i-th word in the tweet. An aspect term is constructed from a sequence of M words, denoted as $T = \{w_1^T, w_2^T, ..., w_i^T, ..., w_M^T\}$. The purpose of this task is to classify the sentiment polarity of the aspect in the context of the tweet into positive, negative or neutral. For example, given the tweet: *"the great screen, but the battery life is not good"*, the sentiment of the aspect *"screen"* is positive while the sentiment of the aspect *"battery life"* is negative.

5.2.2 Basic Idea

Our proposed model is comprised of three major components: Lexicon-aware Input tensor, LSTM layers, and Attention layers. In this model, a specific aspect and its context sentence are treated separately to model the associative relationship effectively. Specifically, given aspect embeddings and its context word embeddings augmented sentiment lexicon embeddings, LSTM models encode the aspect and its context into their representations. Then, an *average-pooling* and an intra-attention mechanism are applied to the aspect embeddings to form an aggregation-level aspect vector and a phrase-level aspect vector. Next, interactive attention mechanisms compute the associative relationship between the informative words

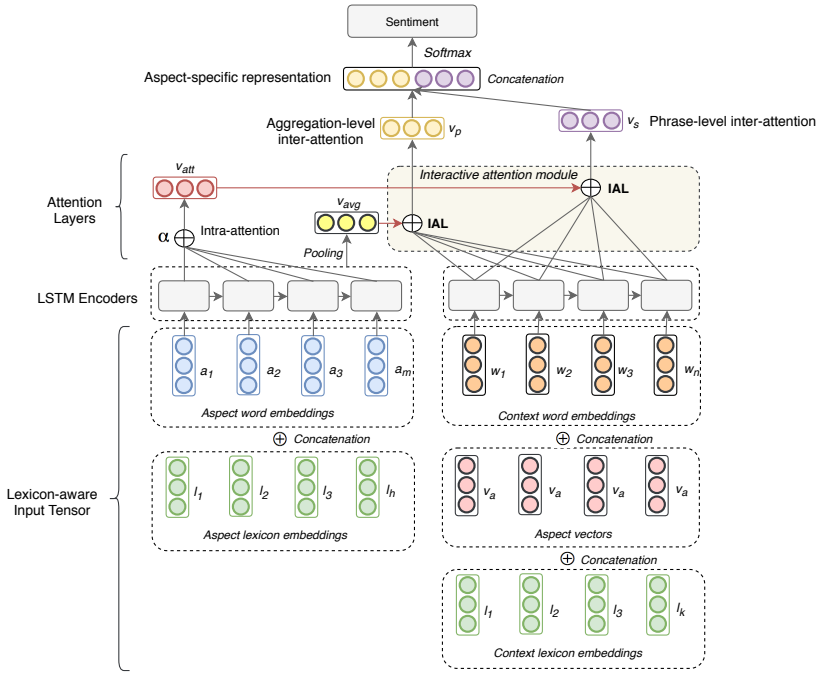


Figure 5.1: The architecture of ILWAAN model.

of the aspect and each sentiment context word to form an aggregation-level inter-attention vector and a phrase-level inter-attention vector. These inter-attention vectors are consolidated by exploiting the knowledge between them. Finally, the final representation is formed by concatenating two inter-attention vectors for making predictions. We believe that these inter-attention vectors provide different perspectives for our model to learn to attend the relationship between an aspect and its sentiment context words.

5.2.3 Interactive Lexicon-Aware Word-Aspect Attention Network (ILWAAN)

As shown in Figure 5.1, our model consists of three major components: Lexicon-aware Input Tensor, LSTM layers and Attention Layers. We describe the detail of each component in the next sections.

Lexicon-aware Input Tensor

As the name of this component, the lexicon-aware input tensor is a feature input for a deep neural network which is augmented the crucial feature: sentiment lexicon information. Specifically, three features are utilized to construct the lexicon-aware

input tensor for capturing the different views of an aspect and its context words: sentiment lexicon embeddings, an aspect vector, and semantic word embeddings.

Lexicon Embeddings (LexW2Vs): The Lexicon embeddings in the aspect-level task are same as in Subsection 4.3.3. The Lexicon embeddings are built by utilizing various useful lexicon resources. The purpose of the Lexicon embeddings is to highlight the critical sentiment words to assist multiple attention mechanisms in order to capture the informative parts of an aspect and its context.

Aspect Vector: A given aspect term is composed of M words. Each word w_i^T is associated with pre-trained word embeddings as the idea of [Mikolov et al., 2013b] to form an aspect embedding. An aspect vector v_a is formed by applying *average-pooling* over the aspect embeddings as:

$$v_a = \sum_{i=1}^{M} w_i^T / M \tag{5.1}$$

Where $v_a \in \mathbb{R}^{d'}$, with d' is the dimension of the aspect embedding. The aspect vector is an internal feature to compute inter-dependence between the aspect and its context words.

Lexicon-augmented Word Embeddings: The inputs to our model are a context sentence along with an aspect term which are indexed into semantic word embedding matrices. The context word embedding matrix and aspect embedding matrix are $W_C, W_A \in \mathbb{R}^{d \times |V|}$, where d is the dimension of the word embedding and $|V|$ is the size of a vocabulary. To develop the lexicon-aware input tensor, each context word embedding is built up by adopting a lexicon context embedding and an aspect vector into $[w_i^C \oplus v_a \oplus l_i^C] \in \mathbb{R}^{d+d'+ld}$ to form a lexicon-augmented word embedding, where, ld is the dimension of the lexicon context embedding, w_i^C is the context word embedding mapped by using pre-trained word embeddings, v_a and l_i^C are the aspect vector and the lexicon context embedding, respectively.

Similarly, each aspect embedding is constructed into $[w_i^T \oplus l_i^T] \in \mathbb{R}^{d+ld}$ to form a lexicon-augmented aspect embedding, where, w_i^T is the aspect embedding mapped by using pre-trained word embeddings and l_i^T are the lexicon aspect embedding. The lexicon embeddings and the aspect vector is commonly imagined as internal flavor-features dispensing with word embeddings altogether.

Long Short Term Memory Networks (LSTMs):

To encode the lexicon-augmented word embeddings and the lexicon-augmented aspect embeddings, we utilize Long Short Term Memory networks (LSTMs) from [Hochreiter and Schmidhuber, 1997]. The model calculates the hidden states $[\overrightarrow{h_i}]$ from these embeddings (feature vectors). The LSTMs have the number of LSTM units corresponding to the number of the feature vectors. The LSTM unit has an

input gate i_t, a memory cell c_t, a forget gate f_t and an output gate o_t . that h_t is computed as follows:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1} + b_i) \\ f_t &= 1.0 - i_t \\ g_t &= tanh(W_g x_t + U_g h_{t-1} + b_g) \\ c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + V_o c_t + b_o) \\ h_t &= o_t \odot tanh(c_t) \end{aligned} \tag{5.2}$$

Where x_t can be a word embedding or an aspect embedding (the feature vector) of the context word w_i^C or the aspect word w_i^T, σ is sigmod function, $\odot$ is element-wise multiplication. W_i, U_i, V_i, b_i, W_g, U_g, b_g, W_o, U_o, V_o and b_o are trainable LSTM parameters. Therefore, we have the hidden states $[\overrightarrow{h_1^T}, \overrightarrow{h_2^T}, ..., \overrightarrow{h_M^T}]$, $[\overrightarrow{h_1^C}, \overrightarrow{h_2^C}, ..., \overrightarrow{h_N^C}]$ corresponding to an aspect and a context sentence, respectively.

Attention Layers

Traditional LSTM model cannot capture the information about which words are important to the meaning of a specific aspect. In order to address this problem, we design multiple attention mechanisms which drive the model to concentrate on such words.

Intra-attention Layer: The intra-attention called self-attention is used for mapping a variable-length sequence of symbolic representation to another sequence of an equal length where the output is computed as a weighted sum of the values. The intra-attention is applied on the hidden states of a specific aspect to construct a phrase-level vector $v_{att} \in \mathbb{R}^{d_{att}}$. Let the set $[\overrightarrow{h_1^T}, \overrightarrow{h_2^T}, ..., \overrightarrow{h_M^T}]$ is the hidden states of the aspect. The intra-attention function is define as follows:

$$v_{att} = H\alpha = \sum_{i=1}^{M} \alpha_i \cdot h_i^T;\ v_{att} \in \mathbb{R}^{d_{att}} \tag{5.3}$$

Where the attention vector $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_M\}$ is a self-attention vector and computed by feeding the hidden states into a bi-layer perceptron, as:

$$\begin{aligned} \alpha_i &= \frac{exp(\gamma(h_i^T))}{\sum_{j=1}^{M} exp(\gamma(h_j^T))} \\ \gamma(h_i^T) &= tanh(W_a \cdot h_i^T + b_a) \end{aligned} \tag{5.4}$$

The purpose of the intra-attention mechanism is to capture the informative words of an aspect term. For example, given a aspect term *"Iphone screen"*, the word *"screen"* is going to has a higher attention weight compared to the word *"Iphone"*.

Interactive-attention Layers (IALs): After obtaining the informative words of a specific aspect via the phrase-level vector v_{att}, the interactive-attention layers are utilized to compute the word-aspect correlation between the informative aspect words and each sentiment context word by using multiple *Feedforward networks* (MLPs). As shown in Figure 5.1, two interactive-attention mechanisms perform parallel in which the model learns to attend the sentiment context words conditioned on the aspect words. To capture the different perspectives of each sentiment context word, we try to develop two kinds of the aspect representations (the phrase-level representation v_{att} and the aggregation-level representation v_{avg}) to compute two inter-attention vectors and interact knowledge between them.

Let $[\overrightarrow{h_1^T}, \overrightarrow{h_2^T}, ..., \overrightarrow{h_M^T}]$ and $[\overrightarrow{h_1^C}, \overrightarrow{h_2^C}, ..., \overrightarrow{h_N^C}]$ are the aspect hidden states and its context hidden states, respectively. An average pooling layer is applied for the aspect hidden states to form the aggregation-level vector v_{avg} as:

$$v_{avg} = \sum_{i=1}^{M} h_i^T / M;\ v_{avg} \in \mathbb{R}^{d_{att}} \tag{5.5}$$

Noted that the aggregation-level vector and the phrase-level vector have the same dimension. Subsequently, the inter-attention vectors are generated as follows:

$$v_s = \sum_{i}^{N} \beta_i \cdot h_i^C;\ v_s \in \mathbb{R}^{d'_{att}} \tag{5.6}$$

$$v_p = \sum_{i}^{N} \beta'_i \cdot h_i^C;\ v_p \in \mathbb{R}^{d'_{att}} \tag{5.7}$$

Where β and β' are interactive attention weights as:

$$\begin{aligned} \beta_i / \beta'_i &= \frac{exp(\gamma(h_i^C, v_{rep}))}{\sum_{j=1}^{n} exp(\gamma(h_j^C, v_{rep})} \\ \gamma(h_i^C, v_{rep}) &= tanh(h_i^C \cdot W_a \cdot v_{rep}^T + b_a) \end{aligned} \tag{5.8}$$

Where $v_{rep} \in \mathbb{R}^{d_{att}}$ is the representative notation of v_{att} or v_{avg}, d_{att} is the length of v_{att} and v_{avg}, v_{rep}^T is the transpose matrix of v_{rep} and W_a is the *shared weight matrix* between two interactive-attention mechanisms. Finally, the aspect-specific representation is built by concatenating v_s and v_p for final prediction $v_{final} = [v_s \oplus v_p];\ v_{final} \in \mathbb{R}^{d'_{att}+d'_{att}}$.

5.2.4 The variants of ILWAAN model

In this section, we construct the variants of ILWAAN model in order to evaluate the crucial components such as the sentiment lexicon embeddings and the aggregation-level aspect vector in our proposed model.

Lexicon-Aware Word-Aspect Attention Network (LWAAN)

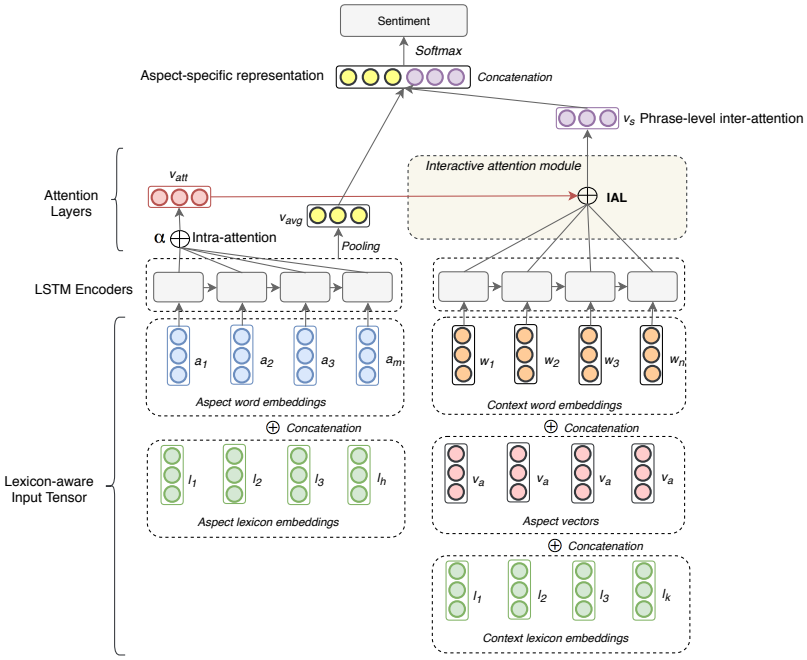


Figure 5.2: The architecture of LWAAN.

The model is different from the ILWAAN model in computing an aggregation-level inter-attention vector. As shown in Figure 5.2, the aggregation-level aspect vector v_{avg} encapsulated the sentiment information of a specific aspect is concatenated into the aspect-specific representation v_{final} for final prediction, i.e.,

$$v_{final} = [v_{avg} \oplus v_s] \in \mathbb{R}^{d_{att}+d'_{att}} \tag{5.9}$$

Where $\oplus$ denotes the concatenate operator, d_{att} is the dimension of the aggregation-level vector and d'_{att} is the dimension of the phrase-level inter-attention vector. In this case, we try to enrich the sentiment information of an aspect into the aspect-specific representation so that the model recognizes the aspect information concerning its sentiment context words in the final representation.

Word-Aspect Attention Network (WAAN)

In this model, sentiment lexicon information is eliminated to evaluate the affectation of the sentiment lexicon information. In this case, The model computes the word-aspect relationship based on the semantics of words. To better take advantage of aspect information, an aspect vector appended into each context word

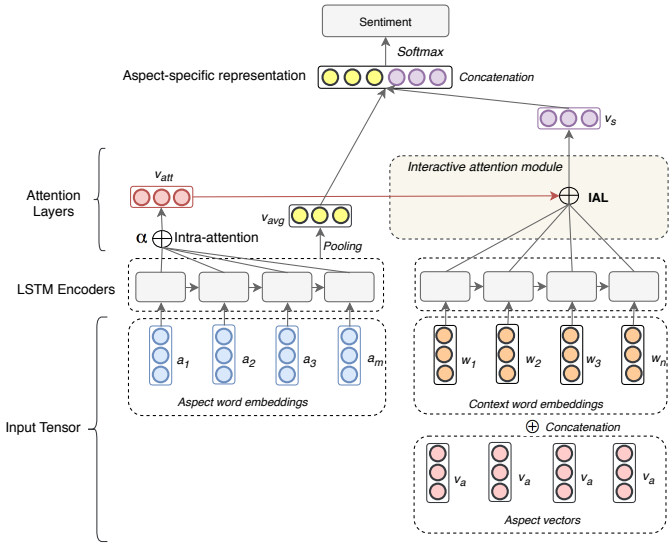


Figure 5.3: The architecture of WAAN.

vector form an aspect-augmented word embedding $\hat{w}_i^C$, i.e.,

$$\hat{w}_i^C = [w_i^C \oplus v_a] \in \mathbb{R}^{d+d'};\ i \in [1, M] \tag{5.10}$$

where $\oplus$ denotes the concatenate operator, d' is the dimension of the aspect vector, w_i^C is the context word embedding mapped by using pre-trained word embeddings, v_a is the aspect vector. The output hidden states $[\overrightarrow{h_1^C}, \overrightarrow{h_2^C}, ..., \overrightarrow{h_N^C}]$ of a context sentence have the information of the aspect vector v_a in order to allow the model to compute the inter-dependence between an aspect and its context.

Similar to the LWAAN model, the aggregation-level vector v_{avg} is adopted into the final aspect representation to increase the sentiment information of an aspect concerning its sentiment context words.

Attention Network (AN)

This is our basic baseline which only constructs an aspect and its context separately. As shown in Figure 5.4, we utilize the LSTM models and the multiple attention mechanisms to form an aspect-specific representation. Specifically, the aspect vector and the lexicon embeddings are eliminated from the model. The model tries to model the word-aspect relationship between an aspect and its context without the inter-dependence between the aspect and its context and sentiment lexicon information. In this case, the word-aspect relationship is captured by computing a phrase-level inter-attention vector between the informative aspect words and each context words via the semantics of words. The aggregation-level vector is concate-

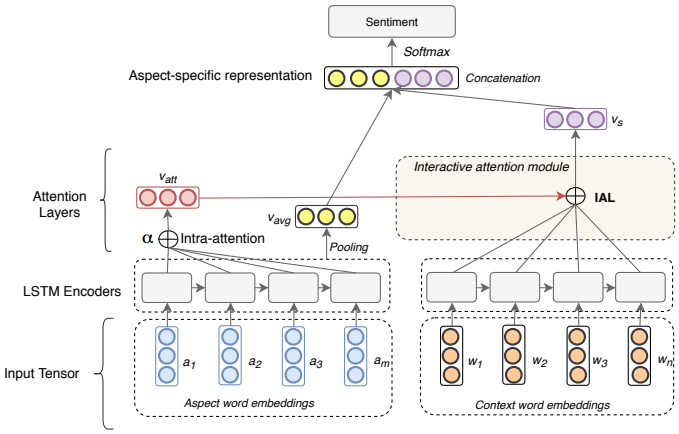


Figure 5.4: The architecture of AN.

nated into the final aspect representation for increasing the sentiment information of an aspect concerning its semantic context words.

5.2.5 Deep Memory Network-in-Network (DMNN)

In previous models, the multiple attention mechanisms contribute a big role to overcome the challenges of aspect-level task: 1) Reducing the extra burden for the attention layer of trying to score the relationship between aspect and its context words as well as minimize parameter costs to LSTM Layer. 2) The informative phrases of an aspect are captured. Indeed, the multiple attention mechanisms solve the drawbacks of encoders (e.g., RNN, LSTM) which are the limitation of compressing all the necessary information of sequential data into a single fixed-length vector. Instead, the attention mechanism let the model learn to generate a context vector for each output time step. Such context vector extracts the important part of aspect and its context. However, there is a remaining challenge that the attention cannot allow the model to consider the entire history so far explicitly and look back previous examples which are relevant. Additionally, the attention mechanism cannot refine the internal representation based on the relevant information in order to make sure that either the necessary information is enough or not. To tackle this, we propose a novel Deep Memory Network-in-Network (DMNN) improved from ILWAAN models which utilize the benefit of the multiple attention mechanisms to solve the remaining challenge for aspect-level sentiment classification. Our model is inspired by End-to-End Memory Network of [Sukhbaatar et al., 2015] in which our model iteratively performs multiple attention mechanisms to score the importance of context words and reduce them into a final joint representation of aspect and its context.

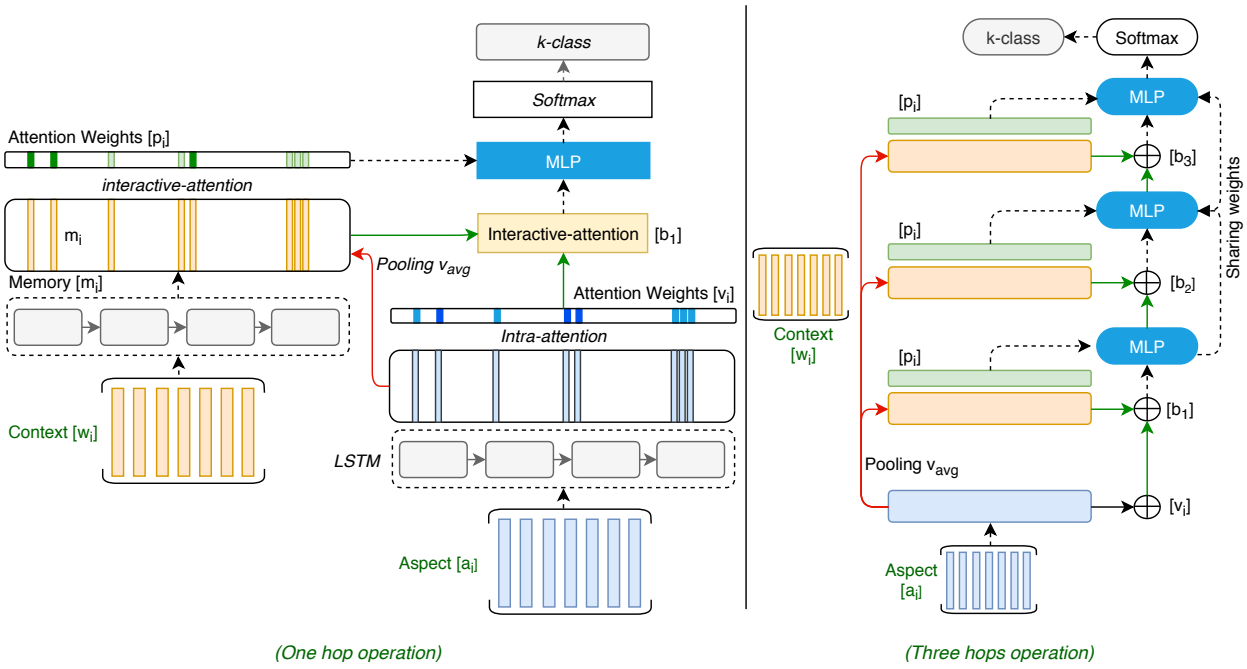


Figure 5.5: The architecture of Deep Memory Network-in-Network.

In Figure 5.5, the left side is the structure of our model with one hop operation, while we construct a model with three hops in the right side. Our Deep Memory Network-in-Network is described in a single layer case, which implements a single memory hop operation. It subsequently can be stacked to give multiple hops in memory.

Single Layer

- **Input memory representation:** Suppose that a given input set $\{w_1^C, w_2^C, ..., w_N^C\}$ is stored in memory. The memory slot m_i of dimension d is computed by first embedding each word w_i^C. Thus, the entire set of $[w_i^C]$ are converted into internal memories $[\overrightarrow{m_i}]$ through a Long-Short-Term Memory network. The aspect embeddings $[a_i]$ is also embedded by a Long-Short-Term Memory network and then used for building an internal state $[v_i]$ via an intra-attention. An interactive attention mechanism is applied on the memory vectors $[m_i]$, conditioned on v_{avg}, to produce an inter-attention vector $[p_i]$, where v_{avg} is a aggregation-level vector encoded by *average pooling* over the aspect's hidden states $(\overrightarrow{h_1^T}, \overrightarrow{h_2^T}, ..., \overrightarrow{h_M^T})$.

- **Output memory representation:** Another interactive attention mechanism is proposed to filter away un-important context and produce an inter-attention vector $[b_i]$. The function from input to output is smooth. We can

easily compute gradients and back-propagate through it. Two inter-attention vectors represent the different perspectives of each context word: informative phrase-level and aggregation-level information.

- **Generating the final prediction:** In the single case, the output vector o_i of the inter-attention vector $[b_i]$ and the inter-attention vector $[p_i]$ is passed through a *feed-forward network (MLP)* and a *softmax* function to achieve a final sentiment prediction. The role of *MLP* is an aggregator which aggregates the information of two inter-attention vectors and computes the correlation between them. The parameter matrix W_a learns the information of this new vector space at two levels: informative phrase-level and aggregation-level information according to their relevance to the problem at hand. The overview of our model is shown in Figure 5.5.

$$o_i = tanh(p.W_a.b^T) \tag{5.11}$$

$$\hat{y} = softmax(o_i) \tag{5.12}$$

Multiple Layers We extend our model to handle K hop operations. The output of hop K is parametrized by the *feed-forward network* with a *softmax* layer and this network is jointly trained with other parts of the model in order to produce an aspect-specific representation. The weights of the *feedforward network* is shared between hops:

$$o_k = tanh(p.W_a.b_k^T) \tag{5.13}$$

$$aspect_k = softmax(o_k) \tag{5.14}$$

From second hop onwards, the output of the first hop is the input of the next hop. The vector o_k is fed into another interactive-attention layer in order to form another interactive-attention weights b_k.

By iteratively applying attention mechanism on the context words, we refine the internal representation of the context words based on the relevant information from the aspect. By aggregating relevant information into the final representation, we provide necessary information to the answer selection layer, to predict the sentiment polarity to the aspect. This form of multi-hop *search* on context words allows the model to learn to perform some sophisticated reasoning for solving specific challenging tasks.

The deep memory network which contains many computational layers can learn representations of data with multiple levels of abstraction [LeCun et al., 2015]. Each layer receives important context and transforms the representation in previous layers into higher abstract level representation. When the transformation constructs enough composition, the complex functions of sentence representation towards an aspect can be learned.

5.2.6 The Effect of Multiple Attention Mechanisms

Attention Mechanism has become an integral part of neural network-based models designed for Natural language processing. Both sequence-to-sequence tasks, like machine translation, document summarization, and classification tasks, like sentiment analysis, PoS tagging, document classification benefit by including this ability to focus on segments of the sequence selectively.

In this chapter, we try to throw light on the fragments of attention mechanisms by building the variant of deep learning models using the benefit of multiple attention mechanisms. The works of [Vaswani et al., 2017] show the effectiveness of attention mechanism which plays a crucial role in selective reduction and is an essential component of state-of-the-art models so far.

5.2.7 Model Training

The deep learning model is trained in a supervised manner by minimizing the cross-entropy error of sentiment classification. The deep learning model can be trained in an end-to-end way by back-propagation to calculate the gradients of all the parameters, and update them with stochastic gradient descent. The goal of training is to minimize the cross-entropy error between y and $\hat{y}$ for all sentences, where y be the target distribution for sentence, $\hat{y}$ be the predicted sentiment distribution.

$$loss = -\sum_{i}\sum_{j} y_i^j log\hat{y}_i^j + \lambda||\theta||^2 \tag{5.15}$$

5.3 Evaluation

This section shows early studies are used to comparing to our models and presents an evaluation metric, datasets and the configuration of our models utilized for the comparison.

5.3.1 Datasets and Experimental Setting

We conduct experiments on three datasets to evaluate the performance of our model as early studies by using accuracy and F1-score measures. The Twitter dataset is built by [Dong et al., 2014] containing twitter posts and the aspects with respect to the twitter posts. To evaluate the significance of our models, we use more the SemEval 2014 datasets containing two categories: Restaurant, Laptop from [Pontiki et al., 2014]. Laptop and Restaurant contain user reviews in laptop domain and restaurant domain, respectively. We also remove a few examples having the "conflict" label as the compared models. All tokens are lowercased

without removal of stop words, symbols or digits, and sentences are zero-padded to the length of the longest sentence in the dataset.

Each dataset contains three classes *(Negative, Neutral, Positive)*. Table 5.1 and 5.2 display the statistics of datasets for evaluation.

Dataset	**Set**	**#Sents.**	**#Positive.**	**#Neutral.**	**#Negative.**
Laptop	Train	2291	994	870	464
	Test	639	341	128	169
Restaurant	Train	3589	2164	807	637
	Test	639	728	196	196
Twitter	Train	6248	1567	1563	3127
	Test	692	174	174	346

Table 5.1: The statistic of datasets

Data	**Set**	N	c	l_w	$\|V_w\|$	$\|V_m\|$	$\|V_l\|$
Laptop	Train	2291	3	83	3641	3328	3642
	Test	639					
Restaurant	Train	3589	3	79	4559	4378	4560
	Test	1118					
Twitter	Train	6248	3	45	16362	10248	16363
	Test	692					

Table 5.2: Summary statistics for the datasets. c: the number of classes. N: The number of sentences. l_w: Maximum sentence length. $|V_w|$: Word alphabet size. $|V_m|$: The number of words mapped into an embedding space (*Glove*). $|V_l|$: The number of words mapped into a lexicon embedding space.

Hyper-parameters

Table 5.3 shows the summary of the hyper-parameters which is applied to our deep learning models. We utilize *Random search* for choosing the best hyper-parameters. Training is done through stochastic gradient descent over shuffled mini-batches with *Adam* optimizer. In our experiments, the uniform distribution $U(-0.1, 0.1)$ is used for initializing all out-of-vocabulary words. All weight matrices are given their initial values by sampling from the uniform distribution $U(-0.1, 0.1)$, and all of biases are set to zeros.

We utilize *Glove2Vec*[1] which is performed on aggregated global word-word co-occurrence statistics from a corpus.

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

Hyper-parameters	# Laptop	# Restaurant	# Twitter
Mini-batch size		100	
Embedding dim		300	
Lexicon dim		16	
Epochs		300	
RNN dim		100	
Learning rate		2e-3	
Dropout Rate		0.5	
l_2 Constraint		0.00001	

Table 5.3: The summary of hyperparameters

Evaluation Metric

For evaluation measures, we apply the accuracy and F1-score to evaluate the effectiveness of our proposed model. The evaluation accuracy measure is as follows:

$$Accuracy = \frac{\sum_{i=1...N}(1 - (target_i - threshold(f(\overrightarrow{x_i}))))}{N} \tag{5.16}$$

Where $threshold(f(\overrightarrow{x_i}))$ equal 0/1.

The F1 score can be interpreted as a weighted average of the precision and recall. The formula for the F1 score is:

$$F1 = 2 * (precision * recall)/(precision + recall) \tag{5.17}$$

5.3.2 Baselines

We compare our proposed models to the early state-of-the-art models. The methods are separated into two categories: traditional methods and deep learning methods. The methods are listed as follows:

- **Majority** is a basic baseline approach. This baseline model assigns the majority sentiment polarity in training dataset to each sample in the test dataset.
- **Feature-SVM** is a state-of-the-art model using *N-gram* features, parsing features and lexicon features [Kiritchenko et al., 2014a].
- **AdaRNN** is proposed by [Dong et al., 2014] which learns the sentence representation toward target for sentiment.
- **LSTM** uses one LSTM network only in order to form a context representation of words. The last hidden vector is used as a sentence representation

and fed into a softmax function to estimate the probability of each sentiment label [Tang et al., 2016a].

- **TD-LSTM** is extended from LSTM using two LSTM networks in order to model left context and right context towards the target. The left and right target dependent representation are concatenated for predicting the sentiment polarity of the target [Tang et al., 2016a].

- **TD-LSTM + ATT** is also the work of [Tang et al., 2016a] and an extended model from TD-LSTM combined with an attention mechanism over hidden vectors.

- **ContextAVG** is implemented by [Tang et al., 2016a]. Context word vectors are averaged and added to an aspect vector. The output vector is fed into a softmax function for predicting the sentiment label of the aspect.

- **AE-LSTM** models context words by using LSTM and combines hidden vectors with an aspect vector in order to generate attention vectors to produce the final representation of the aspect [Wang et al., 2016c].

- **ATAE-LSTM** is designed based on AE-LSTM. However, ATAE-LSTM appends aspect embeddings with each word embedding to present the context [Wang et al., 2016c].

- **MemNet** is a model based on the idea of Memory Network of [Sukhbaatar et al., 2015] with many hops improved by [Tang et al., 2016b].

- **IAN** is an idea of [Ma et al., 2017] in which word context embeddings and aspect embeddings are formed by one LSTM network separately. Hidden states are fed into attention mechanism and pooling to produce representations. The representations are concatenated into final representation and fed into a softmax function for predicting the sentiment polarity of the aspect.

- **BILSTM-ATT-G** is proposed by [Liu and Zhang, 2017]. It models left and right contexts using two attention-based LSTMs and introduces gates to measure the importance of left context, right context, and the entire sentence for the prediction.

- **RAM** [Chen et al., 2017] is a multilayer architecture where each layer consists of an attention-based aggregation of word features and a GRU cell to learn the sentence representation.

Model	**Laptop** *(Acc.)*	**Restaurant** *(Acc.)*	**Twitter** *(Acc.)*
Majority	53.45	65.00	-
LSTM	66.45	74.28	-
TD-LSTM+ATT	66.24	74.31	-
ContextAVG	61.22	71.33	-
AE-LSTM	68.90	76.20	-
ATAE-LSTM	68.70	77.20	-
IAN	72.10	78.60	-
Feature-SVM	70.49	80.16	63.40
TD-LSTM	68.13	75.63	66.62
BILSTM-ATT-G	74.37	80.38	72.70
MemNet	70.33	78.16	68.50
RAM	75.01	79.79	71.88
AN	74.00	79.58	75.71
WAAN	74.14	80.00	75.85
LWAAN	73.42	80.44	76.28
ILWAAN	75.85	81.25	76.71
DMNN	**76.00**	**82.00**	**77.42**

Table 5.4: The experimental results compared to other models on three benchmark datasets.

5.3.3 Experimental results

As shown in Table 5.4, the majority model is the worst, only occupies *53.45%* and *65%*, respectively. Additionally, the SVM method is still alive and achieves remarkable performance on Restaurant dataset. We can observe the LSTM model is effective compared to Majority model. Generally, the LSTM-based model can identify the sentiment polarity of an specific aspect. However, the drawback of the LSTM model is heavily rooted in LSTMs to treat aspects and their contexts equally. Therefore, the LSTM model can not attend the informative words of the context words conditioned on the aspect.

TD-LSTM and TD-LSTM+ATT are attention-based models improved from the LSTM model which capture the left context and right context of a specific aspect and then, utilize an attention mechanism to extract correct context words towards the given aspect. Therefore, they outperform the LSTM model about *1%* and *2%*. However, these models are mainly rooted in LSTMs as well and do not treat an aspect and its context separately. As such, this causes a difficulty to model the relationship between the aspect and its context.

Inspire by the problems of TD-LSTM and TD-LSTM+ATT models, AE-LSTM, ATAE-LSTM and IAN models utilize an attention mechanism and construct an

aspect and its context separately. These models can incorporate aspect information into the deep neural networks by adopting a naive concatenation of an aspect and its context words to extract the critical parts of the context words. However, this causes an extra burden for the attention layer of modeling the sequential information conditioned on the aspect information and incurs parameter costs to LSTM layers. On the other hand, to tackle the limitation of above models heavily rooted in LSTMs, a gating mechanism is utilized by BILSTM-ATT-G model to control the information of the attention mechanism and capture the left and right contexts towards a given aspect. Therefore, the BILSTM-ATT-G model achieves the best performance on Laptop and Twitter datasets.

MemNet and RAM called End-to-End Memory networks [Sukhbaatar et al., 2015] utilize the benefit of an attention mechanism and external memory to capture the importance of context words with respect to a given aspect. The critical point of these models is to construct multiple computational layers to transform the aspect representation into more abstract-level representation. However, MemNNs still consider that the words of an aspect are the equal contribution. On the other hand, MemNNs do not combine the results of multiple attentions, and the vector fed to softmax is the result of the last attention, which is essentially the linear combination of word embeddings. Therefore, MemNNs form the final aspect representation into more abstract-level representation, instead of improving the relationship between the aspect and its context words sufficiently.

In our view, the previous models make use of the contexts without consideration of the critical degrees of the different words in a specific aspect. Additionally, the feature input used for the attention mechanism is semantic word embeddings. As such, the models mainly capture the semantics of words via the attention mechanism that ignores the sentiment of the words. On the other hand, the performance of those above methods is mostly unstable. For example, for the tweet in the Twitter dataset, BILSTM-ATT-G and RAM cannot perform as efficiently as they do for the reviews in Laptop and Restaurant datasets, due to the fact that they are heavily rooted in LSTMs, and the ungrammatical sentences hinder their capability in capturing the context features. Another difficulty caused by the ungrammatical sentences is that the dependency parsing might be error-prone, which will affect those methods such as AdaRNN using dependency information. Our observation and analysis are that the LSTM-based models (e.g., TD-LSTM, BILSTM-ATT-G, RAM) relying on sequential information can perform well for formal sentences by capturing more useful context features. However, these LSTM-based models are sensitive to informal texts which are tackled by the sentiment lexicon information in our model.

We can observe that our model achieves resonable improvement in accuracy against the dominant state-of-the-art models so far. Compared to IAN, AE-LSTM,

ATAE-LSTM and BILSTM-ATT-G, our model improves about *1% - 6%* on three benchmark datasets. To evaluate the significant improvement of our model, the Macro-F1 is conducted and showed in Table 5.5. Our model achieves the best performance against other models about *1 - 4%* on Restaurant and Twitter datasets. We believe that our model help to better capture opinion words due to additional knowledge from the sentiment lexicon information via the multiple attention mechanisms.

	# Laptop	**# Restaurant**	**# Twitter**
Macro-F1			
Feature-SVM	-	-	63.30
AdaRNN	-	-	65.90
TD-LSTM	68.43	66.73	64.01
MemNet	64.09	65.83	66.91
BILSTM-ATT-G	69.90	**70.78**	**70.84**
RAM	**70.51**	68.86	70.33
ILWAAN	68.89	71.22	75.24
DMNN	69.85	**72.76**	**75.50**

Table 5.5: The Macro-F1 scores of IALAN models compared to other models.

5.3.4 Analysis

We can observe the sentiment lexicon information contributes a significant role to the deep neural network. Specifically, ILWAAN model is the best model compared to other models. Additionally, LWAAN model with the sentiment information is more effective than AN and WAAN models without the sentiment information. This provides more evidence about the effectiveness of the sentiment lexicon information. Indeed, DMNN model improved from ILWAAN model is the best model compared to others.

We show case studies by the heat-map of interactive attention weights as Figure 5.6 to observe the effectiveness of our model. β (beta) and γ (gamma) are the attention weights of context words extracted by the lexicon pooling aspect vector and attention aspect vector, respectively:

- their *dinner special* are fantastic. (Aspect: *dinner special*)
- *food* was decent; but not great. (Aspect: *food*)
- they make the best *izza* in new jersey. (Aspect: *izza*)
- *dessert* was also to die for !. (Aspect: *dessert*)

Attention can compute the important parts from the whole sentence dynamically. Obviously, the model can detect the important context words affecting to the sentiment polarity of the aspect terms such as the phrases *the best*, *new jersey* from *(b)* and even the negation *but not great* for *(c)*. Besides, from *(c)*, the multi-keywords can be detected if more than one keyword is existing. *decent* and *but not great* are both detected.

In the previous works, most of the errors can be summarized as follow: the first factor is non-compositional sentiment expression. For example, the sentence *"dessert was also to die for !"* is the example described by [Tang et al., 2016b] where the aspect is *dessert*. The sentiment expression is *"die for"*, whose meaning could not be composed from its constituents *"die"* and *"for"*. The second factor is complex aspect expression consisting of many words, for example, *"dinner special"*, where many words construct the aspect term. Clearly, for sentence *(a)*, the model recognizes the important words of the aspect *dinner special* in which the word *dinner* is more important than the word *special* at the attention vector *beta*. We can observe that the lexicon pooling aspect vector and the attention aspect vector contribute much adequate information for the aspect-specific representation in order to extract the importance of its context.

5.4 Conclusions

In this chapter, we propose novel models for aspect-level sentiment classification by using LSTM, Intra-attention, Interactive-attention and Sentiment lexicons in which the model learns to attend word-aspect association at two levels: Phrase-level and Context-level information. Our models indicate that lexicon-based approaches is still alive and contribute effectively to deep learning models.

Our method no requires laborious feature engineering as well as flavor-features as Chapter 4. Our model tries to learn an aspect-specific representation via multiple attention mechanisms. The multiple attention mechanisms contribute to a significant role and enable the model to look at the crucial parts of an aspect and its context. Additionally, an iterative attention mechanism allows the model to refine context information which is relevant to the aspect. On the other hand, the sentiment lexicons assist the multiple attention mechanisms to focus on sentiment words and capture the special cases of Twitter social networking. Therefore, our model performs well, although we do not need to provide the multiple views of each word in a sequence.

Our experiments on *SemEval2014* and *Twitter* show that our model can learn useful features and provide enough information for predicting the aspect sentiment polarity. Our model shows a significant improvement in performance compared to

multiple strong neural baselines. Our source code is available at Github[2].

[2]https://github.com/huynt-plus/LWAANet

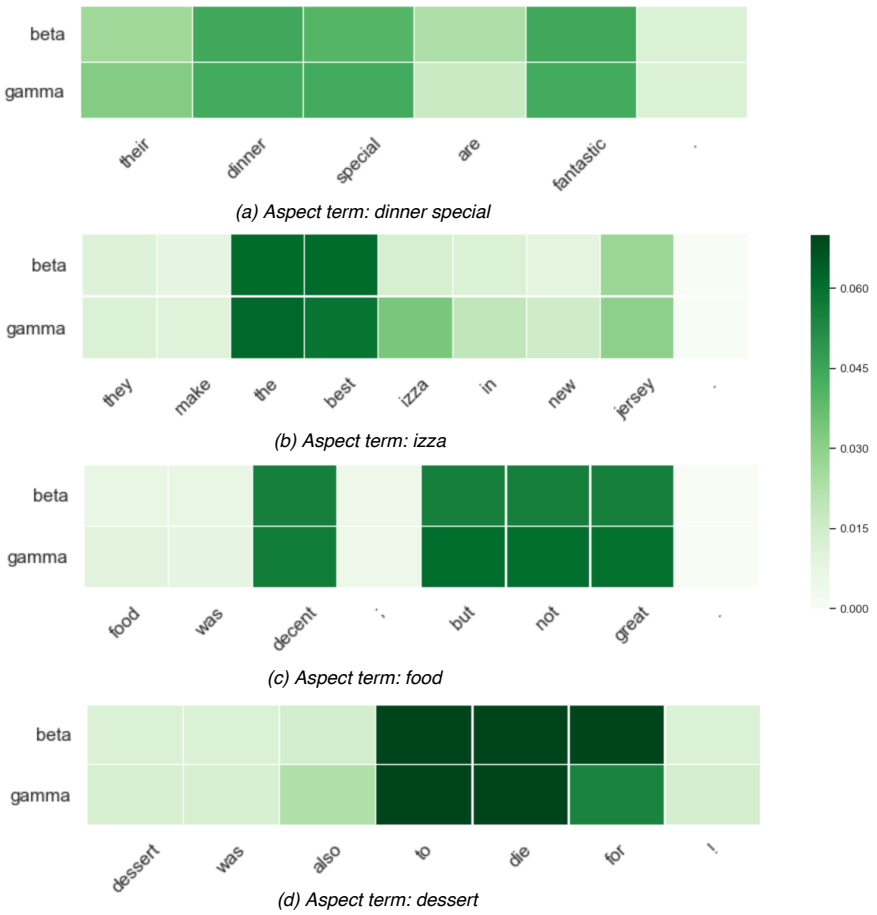


Figure 5.6: The attention visualization. The aspect terms are *dinner special*, *izza*, *food* and *dessert*, respectively. The color depth illustrates the importance of the context words affecting by the aspect terms. As can be seen, the model can detect the word *fantastic* for *(a)*, the phrases *the best*, *die for* for *(b)* and *(c)*, respectively and even negation *but not great* for *(c)*

Chapter 6

Multitask-based Aspect-level Sentiment Analysis

In this chapter, we develop an End-to-End **M**ulti-task **L**exicon-**A**ware **A**ttention **N**etwork (MLAANet) to address the drawbacks of aspect-level data and enhance the classification performance of aspect-level sentiment classification. The model learns to attend associative relationships between sentence words and an aspect term via Lexicon-aware attention operations and Interactive knowledge. More specifically, we incorporate the aspect information assisted by *lexicon information* into a neural model by modeling word-aspect relationships by an *Interactive Word-Aspect Attention Fusion (IWAA-F)*. This allows the model to simultaneously focus on exacting context words given the aspect term and integrate interactive knowledge from annotated and un-annotated corpora being much less expensive for improving the performance of aspect-level sentiment classification. This also deals with the difficulty in aspect-level data is that existing public data for this task are small which largely limits to the effectiveness of deep learning models. The experimental results show that our model outperforms the state-of-the-art models on the data: Laptop and Restaurant domains.

6.1 Introduction

Aspect-level sentiment analysis (ASA) aims to identify the sentiment polarity of an aspect term in its context. For example, the sentence *"The Iphone screen is good, but the battery life is short."*. In this sentence, there are two aspects having opposite polarities, *"Iphone screen"* is positive, whereas, *"battery life"* is negative. As such, the task of ASA introduces a challenging problem of incorporating aspect information into learning models for making predictions. Recently, end-to-end neural networks ([Tang et al., 2016a], [Wang et al., 2016c], [Ma et al., 2017]) have

garnered considerable attention and have promising performance in incorporating aspect information into neural architectures by learning to attend the different parts of a sentence towards a given aspect term. Similar to Chapter 5, we also consider the drawbacks of the state-of-the-art models so far and try to improve the previous models in Chapter 5 to tackle the drawbacks.

On the other hand, the limitation of aspect-level data is small which largely limits to the performance of deep neural networks. Therefore, we develop a multitask learning model which tries to overcome the limitations of previous models as well as the limitation of aspect-level data in order to increase the performance of our proposed model. Specifically, our multitask learning model is developed by utilizing the advantages of the aspect-level model and the sentence-level model in Chapter 4 and Chapter 5. We expand the problem of sentiment classification by using various opinion data such as document-level data.

In this chapter, we state the drawbacks of early models again. Specifically, most dominant state-of-the-art models utilize attention layers to focus on learning the relative importance of context words by simply concatenating the context words and aspect information. Consequently, this causes an extra burden for the attention layer of modeling sequential information dominated by the aspect information and incurs additional parameter costs to LSTM layer towards to hardly model the relationship between the aspect and its context words. Second, such models just make use of the contexts without consideration of the aspect information while the aspect information should be an important factor for judging the aspect sentiment polarity. In other words, the importance degrees of different words are different for a specific aspect. For example, the aspect term *"Iphone screen"*, *"screen"* plays a more important role than *"Iphone"* in its context. Finally, the attention-based models mainly utilized pre-trained word embeddings (e.g., *Glove*) which captures the semantics of words. This leads the attention mechanism via *Dot product* in extracting context words based on the semantics of the words that ignore the sentiment of the words.

In this work, we propose a novel multitask learning model that aims to tackle the weaknesses of the above challenges by considering each sentiment context word conditioned on the crucial words of an aspect. Specifically, we develop an end-to-end deep neural model constructing multiple attention mechanisms (*Intra-attention* and *Interactive-attention* mechanisms) and multi-task learning assisted by *sentiment lexicon information*. The purpose of the lexicon information is to enforce the model to pay more attention to the sentiment of words instead of only the semantics of the words. Additionally, the inter-dependence between an aspect and its sentiment context words can be captured. Besides, the goal of multi-task learning is to improve generalization on the target task by leveraging the domain-specific information contained in the training signals of related tasks. Our model called

Multi-task Lexicon-Aware Attention Network (MLAANet) treats an aspect, and its context separately and cleverly divides the responsibilities of layers to model the relationship between the aspect and its context. More specifically, aspect embeddings and its context word embeddings augmented sentiment lexicon information are firstly encoded via LSTM encoders. Subsequently, an *intra-attention* mechanism and *average pooling* are applied to obtain the information of the aspect at two levels: informative phrase-level and aggregation-level information. To the best of our knowledge, the *average pooling* summarizes the information of the aspect, while the *inra-attention* mechanism learns to weight the words and the sub-phrases within the aspect based on how important they are, and then, allowing *interactive-attention* mechanisms learn to attend the relative importance of the fused context words. Such *interactive-attention* vectors are consolidated by its tailor-made document representation via exploiting knowledge gained from document-level data. Our source is available at Bitbucket[1].

Our Contributions:

The principal contributions of this paper are as follows:

- We propose efficient multiple attention mechanisms to incorporate aspect information into a neural architecture for ASA task.
- Sentiment lexicon information is proposed to enforce the model to pay more attention to the sentiment context words via the attention mechanisms.
- A multitask learning approach is introduced to transfer knowledge from document level to aspect level via *Shared Input Tensor* and *Interactive Word-Aspect Attention Fusion (IWAA-F)* to deal with the limitation of aspect-level data.

6.2 Related Works

Today, the dominant state-of-the-art models are neural networks which incredibly are fashionable for NLP task, and ASA task is no exception. To incorporate aspect information, there are several neural architectures which are based on LSTM to model each sentence towards given the aspect term [Tang et al., 2016a], [Wang et al., 2016c], [Ma et al., 2017] and [Li et al., 2018]. These models utilize the power of LSTM layer and attention layer to model sequential information that is dominated by aspect embeddings. Multi-task learning called transfer approaches is considered as an additional flavor for ASA task and effectively increases the performance of the models [He et al., 2018].

[1]https://bitbucket.org/huynguyenplus/mlaanet/

It is worthy to mention that another class, known as MemNN or End-to-End Memory Networks, also has been used for ASA task. Such models have been applied for QA task before and recently, have formed ASA task as a question-answering problem, where the aspect is as a query and context words are as the external memory ([Tang et al., 2016b], [Chen et al., 2017]). The key idea of MemNN is an attention-based model with multiple computational layers to form aspect information into more abstract-level representation.

Our model is most relevant to [He et al., 2018] which also utilizes multi-task learning to form a tailor-made document representation for the Aspect-level task. However, the overall architecture in this paper differs significantly, which develops *Shared Input Tensor* and *Interactive Word-Aspect Attention Fusion* to integrate aspect information into the neural network.

6.3 Proposed Model

The overall model architecture illustrated in Figure 6.1 has two major tasks: *Aux* and *Main* tasks which are trained simultaneously and transfer knowledge between two tasks. In this section, we describe our multi-task deep learning architecture layer-by-layer.

6.3.1 Task definition

This section describes the ultimate goal of our model in identifying the sentiment polarity of an aspect in a tweet/ sentence. The tweet consists of a sequence of N words, denoted as $S = \{w_1^C, w_2^C, ..., w_N^C\}$ with w_i^C referring to the position of i-th word in the tweet. A document is constructed from a sequence of K words, denoted as $D = \{w_1^D, w_2^D, ..., w_K^D\}$. Similarly, an aspect term is constructed from a sequence of M words, denoted as $T = \{w_1^T, w_2^T, ..., w_M^T\}$. The purpose of this task is to classify the sentiment polarity of the aspect of the tweet into positive, negative or neutral. For example, given the tweet: *"the great screen, but the battery life is not good"*, the sentiment of the aspect *"screen"* is positive while the sentiment of the aspect *"battery life"* is negative.

6.3.2 Shared Input Tensor

The shared input tensor consists of three major components: *Lexicon Embeddings*, *Aspect Vectors* and *Shared Word Embeddings* which are concatenated together as an input for LSTM Encoders.

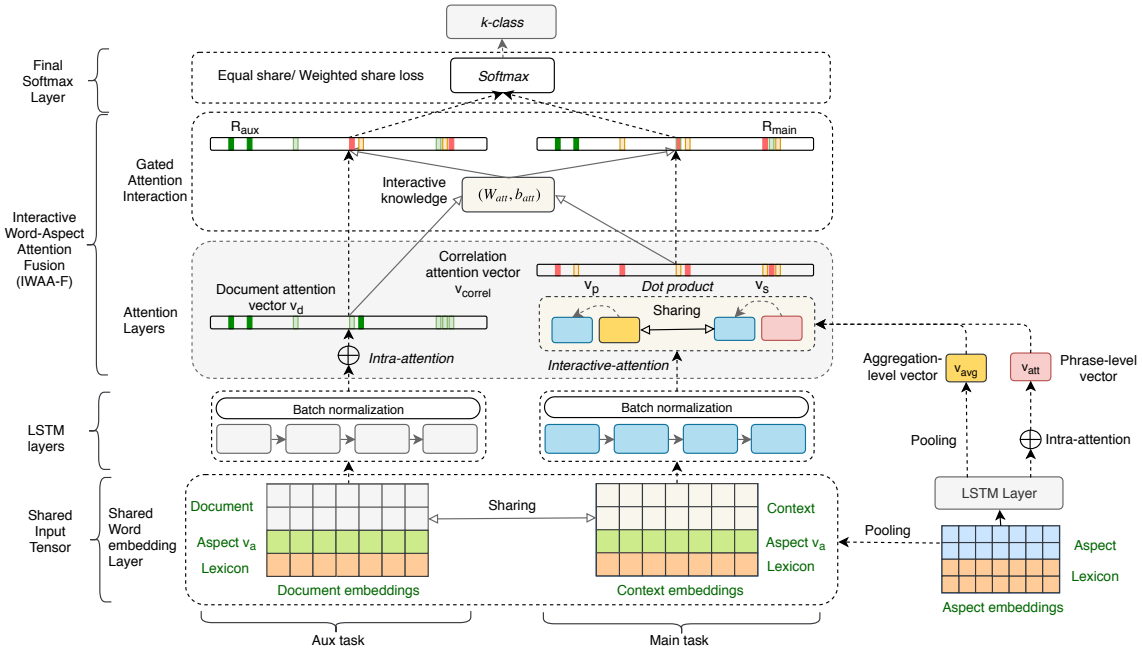


Figure 6.1: Multi-task Lexicon-Aware Attention Network Architecture (MLAANet).

Lexicon Embedding (LexW2Vs)

LexW2Vs in this Subsection is built as in Subsection 4.3.3. LexW2Vs is used to maping words into an embedding space and take the different kinds of words for capturing the different sentiment types of words. LexW2Vs support Intra-attention and Interactive-attention to capture important sentiment words in aspect and its context.

Aspect Vector

An aspect term is composed of M words. Each word w_i^T is associated with pre-trained word embeddings as the idea of [Mikolov et al., 2013b] and appended a lexicon representation to form a lexicon-augmented aspect embedding, i.e., $\hat{w}_i = [w_i^T \oplus l_i]$ where $\oplus$ denotes the concatenate operator, $\hat{w}_i \in \mathbb{R}^{d'+ld}$, $i \in [1, M]$, d' is the dimension of the aspect embedding and ld is the dimension of the lexicon representation. An aspect vector v_a is formed by applying *average-pooling* over the aspect embeddings as:

$$v_a = \sum_{i=1}^{M} \hat{w}_i / M \tag{6.1}$$

Where $v_a \in \mathbb{R}^{d_a}$, with $(d_a = d' + ld)$ is the dimension of the aspect vector. The aspect vector is an internal feature to compute inter-dependence between an aspect and its context words.

Shared Word Embedding Layer (SharedE)

The inputs of our model are a sentence and a document along with an aspect term which are indexed into word embedding matrices. For notational simplicity, we use $W_E \in \mathbb{R}^{d \times |V|}$ to present a document embedding matrix or a context embedding matrix, where d is the dimension of the word embeddings and $|V|$ is the size of a vocabulary. Each word embedding is built up by adopting a sentiment lexicon embedding and an aspect vector into $[w_i^C \oplus v_a \oplus l_i] \in \mathbb{R}^{d+d_a+ld}$, where, $\oplus$ denotes the concatenate operator, ld is the dimension of the lexicon embedding, w_i is the word embedding mapped by using pre-trained word embeddings, v_a is the aspect vector and l_i is the lexicon embedding. The lexicon embeddings and the aspect vector here are commonly imagined as internal flavor-features dispensing with the word embeddings altogether (lexicon-augmented word embeddings).

To develop the shared word embedding matrices from the lexicon-augmented word embeddings, a major shared parameter $W_{se} \in \mathbb{R}^{(d+d_a+ld)\times(d+d_a+ld)}$ is applied to interact knowledge between the document-level and aspect-level tasks as:

$$W = ReLU(W_E \cdot W_{se}) \tag{6.2}$$

Where $W \in \mathbb{R}^{d\times(d+d_a+ld)}$ is the shared word embedding matrix of a document/ a context sentence. We use $ReLU$ function as a gating mechanism to control how much the information would pass through to the final result. The shared word embeddings may allow the model to eavesdrop more features from the related task.

6.3.3 Long Short Term Memory Encoders

To encode the shared word embeddings, we utilize Long short-term memory (LSTM) which is an extension of the Recurrent neural network (RNN) by [Hochreiter and Schmidhuber, 1997]. For each shared word embedding (feature vectors), the model calculates the hidden state h_i. The LSTM has the number of LSTM units corresponding to the number of feature vectors. The LSTM unit has an input gate i_t, a memory cell c_t, a forget gate f_t and an output gate o_t that h_t is computed as

follows:

$$
\begin{aligned}
i_t &= \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1} + b_i) \\
f_t &= 1.0 - i_t \\
g_t &= tanh(W_g x_t + U_g h_{t-1} + b_g) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + V_o c_t + b_o) \\
h_t &= o_t \odot tanh(c_t)
\end{aligned}
\tag{6.3}
$$

Where x_t is shared word embedding (feature vector) of word w_i, σ is sigmod function, $\odot$ is element-wise multiplication. $W_i, U_i, V_i, b_i, W_g, U_g, b_g, W_o, U_o, V_o$ and b_o are LSTM parameters. Therefore, we have the hidden states $(h_1^T, h_2^T, ..., h_M^T)$, $(h_1^C, h_2^C, ..., h_N^C)$ and $(h_1^D, h_2^D, ..., h_K^D)$ corresponding to an aspect, a context sentence and a document, respectively.

6.3.4 Batch Normalization Layer

The Batch normalization proposed by [Ioffe and Szegedy, 2015] is utilized over the LSTM layer to increase the stability of a neural network. A Batch Normalization normalizes $|h_i|$ to ≤ 1. Batch normalization reduces the amount by what the hidden unit values shift around. A batch normalization allows each layer of a network to learn by itself a little bit more independently of other layers. The batch normalization may tackle the limitation of cross-entropy loss because neural networks could misclassify inputs that slightly different from their training data due to the poor margin of cross-entropy loss.

6.3.5 Interactive Word-Aspect Attention Fusion (IWAA-F)

An IWAA-F consists of two main components: Multiple attention mechanisms and Gated interaction which learns to attend associate relationships between individual context word and an aspect term and investigates interactive knowledge at both tasks in order to consolidate each context word with its tailor-made document representation to obtain an aspect-specific representation. In this chapter, the attention layers are reusable from the Sub-Section 5.2.3.

Intra-Attention Layer

The intra-attention is applied at both document-level and aspect-level tasks independently to construct a document attention vector $v_d \in \mathbb{R}^{d_{att}}$ and a phrase-level

vector $v_{att} \in \mathbb{R}^{d'_{att}}$. The intra-attention function is define as:

$$v_d = H\alpha = \sum_{i=1}^{K} \alpha_i \cdot h_i^D \tag{6.4}$$

$$v_{att} = H'\alpha' = \sum_{j=1}^{M} \alpha'_i \cdot h_j^T \tag{6.5}$$

Where $[\overrightarrow{h_M^T}]$ and $[\overrightarrow{h_K^D}]$ is the hidden states of LSTM encoders. Similar to the attention weights α, the attention vector $\alpha' = \{\alpha'_1, \alpha'_2, ..., \alpha'_M\}$ is a self-attention vector and computed by feeding the hidden states into a bi-layer perceptron, as:

$$\begin{aligned} \alpha'_i &= \frac{exp(\gamma(h_i^T))}{\sum_{j=1}^{M} exp(\gamma(h_j^T))} \\ \gamma(h_i^T) &= tanh(W_a \cdot h_i^T + b_a) \end{aligned} \tag{6.6}$$

The purpose of the intra-attention is to capture the informative words of an aspect term and a document. For example, given a aspect term *"Iphone screen"*, the word *"screen"* is going to has a higher attention weight compared to the word *"Iphone"*.

Interactive-Attention Layer

After obtaining the informative words of a specific aspect via the phrase-level vector v_{att}, the interactive-attention layers are utilized to compute the word-aspect correlation between the aspect and each sentiment context word by using multiple *Feedforward networks* (MLPs). As shown in Figure 6.2, two interactive-attention mechanisms perform parallel in which the model learns to attend the sentiment context words conditioned on the aspect words via the multiple MLPs. Then, the *softmax* layer is to select the correct context words adaptively. To capture the different perspectives of each context word, we try to develop two kinds of the aspect representations (the phrase-level representation v_{att} and the aggregation-level representation v_{avg}) to compute two interactive-attention vectors and interact knowledge between them.

Let $[\overrightarrow{h_M^T}]$ and $[\overrightarrow{h_N^C}]$ are the aspect hidden states and its context hidden states, respectively. An average pooling layer is applied for the aspect hidden states to form the aggregation-level vector v_{avg} as:

$$v_{avg} = \sum_{i=1}^{M} h_i^T / M;\ v_{avg} \in \mathbb{R}^{d'_{att}} \tag{6.7}$$

Subsequently, the interactive attention vectors are generated as follows:

$$v_s = \sum_{i}^{N} \beta_i \cdot h_i^C;\ v_s \in \mathbb{R}^{d_{att}} \tag{6.8}$$

$$v_p = \sum_{i}^{N} \beta'_i \cdot h_i^C;\ v_p \in \mathbb{R}^{d_{att}} \tag{6.9}$$

Where β and β' are interactive attention weights as:

$$\begin{aligned} \beta_i / \beta'_i &= \frac{exp(\gamma(h_i^C, v_{rep}))}{\sum_{j=1}^{n} exp(\gamma(h_j^C, v_{rep}))} \\ \gamma(h_i^C, v_{rep}) &= tanh(h_i^C \cdot W_a \cdot v_{rep}^T + b_a) \end{aligned} \tag{6.10}$$

Where $v_{rep} \in \mathbb{R}^{d'_{att}}$ is the representative notation of v_{att} or v_{avg}, v_{rep}^T is the transpose matrix of v_{rep} and W_a is the *shared weight matrix* between two interactive-attention mechanisms. Finally, the correlation vector $v_{correl} \in \mathbb{R}^{d_{att}}$ is built via a *dot product* layer to represent the correlation between two interactive-attention vectors: v_s and v_p. Noted that v_d and v_{correl} have the same dimension.

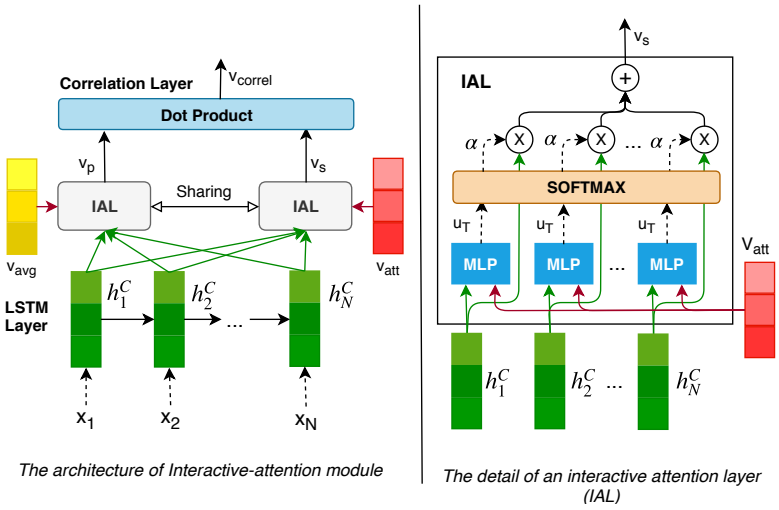


Figure 6.2: The Structure of an Interactive Attention Module.

Gated Attention Interaction (SharedG)

To share interactive attention knowledge between the main and auxiliary tasks, the Gated Attention Interaction is proposed by using a combination of an element wise

operator and a gating mechanism ($ReLU$ function). We introduce two important shared parameters: $W_{att} \in R^{d_{att} \times d_{att}}$ and $b_{att} \in R^{d_{att}}$, (d_{att} is the length of the attention vector) to fulfill the attention interaction between the main and the auxiliary tasks. The formulas of *SharedG* are described as follows:

$$\begin{aligned} R_{main} &= v_{correl} \odot ReLU(W_{att} \cdot v_d + b_{att}) \\ R_{aux} &= v_d \odot ReLU(W_{att} \cdot v_{correl} + b_{att}) \end{aligned} \tag{6.11}$$

Where v_d, v_{correl} are document attention vector and correlation attention vector, respectively, W_{att} and b_{att} are two shared parameters to be trained. W_{att} and $ReLU$ function work together to investigate interacting information between two tasks and extract useful relevant information as well. This mechanism utilizes the benefits of gating mechanism and multi-task learning as well. Therefore, the main and auxiliary tasks are capable of not only learning the shared parameters information but also learning the representations interaction from each other.

6.3.6 Final Softmax Layer

Equal Share (ES)

The straightforward method of ES is equally sharing knowledge between *aux* and *main* tasks. The total loss of ES is computed as follows:

$$L = A_{main} + D_{aux} \tag{6.12}$$

Where D and A are the loss function of document-level sentiment classification and aspect-level sentiment classification, respectively.

Weighted Share (WS)

The Weighted Share is a variation of Equal Share in which λ is utilized to control the weight of D, where $\lambda \in (0, 1)$ is a hyper-parameter. Therefore, the lower λ means the *main* task is more important than the *aux* task for training. The formula of WS as follows:

$$L = A_{main} + \lambda U_{aux} \tag{6.13}$$

6.3.7 Model Training

The deep learning model is trained in a supervised manner by minimizing the cross-entropy error of sentiment classification. The deep learning model can be trained in an end-to-end way by back-propagation to calculate the gradients of all the parameters, and update them with stochastic gradient descent. The goal of

training is to minimize the cross-entropy error between y and $\hat{y}$ for all sentences, where y be the target distribution for sentence, $\hat{y}$ be the predicted sentiment distribution.

$$loss = -\sum_{i}\sum_{j} y_i^j log\hat{y}_i^j + \lambda||\theta||^2 \tag{6.14}$$

6.4 Evaluation

6.4.1 Datasets and Experimental Setting

As shown in Table 6.1, we evaluate the proposed model on two benchmark datasets: *Laptop* and *Restaurant* are from SemEval ABSA challenge [Pontiki et al., 2014] which contains user reviews in laptop domain and restaurant domain, respectively. We also remove a few examples having the "conflict" label as compared models. All tokens are lowercased without removal of stop words, symbols or digits, and sentences are zero-padded to the length of the longest sentence in the datasets. Table 6.2 shows the document-level datasets derived from *Yelp2014* [Tang et al., 2015] and *Amazon Electronics* [McAuley et al., 2015], respectively and considered 3-class classification. We combine document-level and aspect-level datasets in same domains - *Amazon Electronics* is used by *Laptop* dataset and *Yelp2014* is utilized by *Restaurant* dataset. Evaluation metrics are Accuracy and Macro-Averaged F1 where the latter is more appropriate for datasets with unbalanced classes.

We apply *Random search* for choosing hyper-parameters. Training is done through stochastic gradient descent over shuffled mini-batches with *Adam* optimizer. In our experiments, the uniform distribution $U(-0.01, 0.01)$ is used for initializing all out-of-vocabulary words. All weight matrices are given their initial values by sampling from the uniform distribution $U(-0.01, 0.01)$, and all of the biases are set to zeros.

Dataset	Set	Sents.	Pos.	Neu.	Neg.
Laptop	Train	2328	994	870	464
	Test	638	341	128	169
Restaurant	Train	3608	2164	807	637
	Test	1120	728	196	196

Table 6.1: The statistic of aspect-level datasets

6.4.2 Baselines

In this section, we discuss the compared models which are the strong state-of-the-art deep learning models so far. Our model has three variations: *MLAANet*

Dataset	Sent.	Classes
Yelp2014	30K	3
Amazon Electronics	30K	

Table 6.2: The statistic of document-level datasets

w/o sharedE, *MLAANet w/o sharedG* and *MLAANet*. The difference is that the shared layers are combined one by one to evaluate the effectiveness of interactive knowledge at both tasks.

- **Majority** is a basic baseline approach. This baseline model assigns the majority sentiment polarity in training dataset to each sample in the test dataset.
- **Feature-SVM** is a state-of-the-art model using *N-gram* features, parsing features and lexicon features [Kiritchenko et al., 2014a].
- **LSTM** uses one LSTM network only in order to form a context representation of words. The last hidden vector is used as a sentence representation and fed into a softmax function to estimate the probability of each sentiment label [Tang et al., 2016a].
- **TD-LSTM** is extended from LSTM using two LSTM networks in order to model left context and right context towards the target. The left and right target dependent representation are concatenated for predicting the sentiment polarity of the target [Tang et al., 2016a].
- **TD-LSTM + ATT** is also the work of [Tang et al., 2016a] and an extended model from TD-LSTM combined with an attention mechanism over hidden vectors.
- **AE-LSTM** models context words by using LSTM and combines hidden vectors with an aspect vector in order to generate attention vectors to produce the final representation of the aspect [Wang et al., 2016c].
- **ATAE-LSTM** is designed based on AE-LSTM. However, ATAE-LSTM appends aspect embeddings with each word embedding to present the context [Wang et al., 2016c].
- **MemNet** is a model based on the idea of Memory Network of [Sukhbaatar et al., 2015] with many hops improved by [Tang et al., 2016b].
- **IAN** is an idea of [Ma et al., 2017] in which word context embeddings and aspect embeddings are formed by one LSTM network separately. Hidden states

are fed into attention mechanism and pooling to produce aspect-specific representations.

- **RAM** [Chen et al., 2017] is a multilayer architecture where each layer consists of an attention-based aggregation of word features and a GRU cell to learn the sentence representation.
- **PRET+MULT** [He et al., 2018] is a multi-task deep learning model in which the authors utilized the pre-trained weights of Attention-based LSTM model to update the multi-task deep learning model.
- **TNet** [Li et al., 2018] is a transformation network in which a novel Target-Specific Transformation (TST) component is proposed to generate a transformed word representation, and a CNN layer is employed to extract salient features from this transformed word representations.

6.4.3 Experimental results

Models	*LAPTOP*		*RESTAURANT*	
	Accuracy	Macro-F1	Accuracy	Macro-F1
Majority	53.45	-	65.00	-
Feature-SVM	70.49	-	80.16	-
LSTM	66.45	-	74.28	-
AE-LSTM	68.90	-	76.20	-
ATAE-LSTM	68.70	-	77.20	-
IAN	72.10	-	78.60	-
TD-LSTM	68.13	68.43	75.63	66.73
TD-LSTM+ATT	66.24	67.45	74.31	69.01
MemNet	70.33	64.09	78.16	65.83
RAM	74.49	70.51	80.59	68.86
PRET+MULT	71.15	69.73	79.11	67.46
TNET	76.54	70.63	80.79	70.84
MLAANet w/o sharedE	75.14	-	80.00	-
MLAANet w/o sharedG	74.28	-	79.58	-
MLAANet	**77.28**	**71.46**	**81.41**	**72.56**

Table 6.3: The experimental results compared to other models on Laptop and Restaurant datasets.

Table 6.3 shows the performance of our models compared to others. As shown in Table 6.3, the majority model is the worst and the SVM model achieves a remarkable result on Restaurant dataset. The traditional approaches is still alive

and can deal with the aspect-level sentiment classification task. We can observe that *TNET* model achieves the best performance against other models by *2% - 3%*. In fact, *TNET* computes the attention scores between each word of an aspect term with individual context word and utilizes multi-task approach. On the other hand, *TNET* utilizes mechanisms for preserving the information of the context words since the information can be lost after transformation steps. Therefore, *TNET* may significantly increase performance. Additionally, *PRET+MULT* approaches the multi-task learning for ASA task as well. This show that multi-task learning is an effective associative operator and improves performance for ASA task. However, *PRET+MULT* leverages *ATAE-LSTM* model and only constructs multi-task learning to transfer knowledge from document level to aspect level. As such, the model still meets the weaknesses of the Attention-based LSTM models. The work assumes that the words of an aspect have equal distribution to its aspect and all context words are considered in global configuration. Additionally, the simple concatenation between aspect and its context gives an extra burden to the attention layer and incurs a parameter cost to LSTM layer. The multi-task learning, in this case, is as an increment for overcoming the limitation of aspect datasets.

RAM and *MemNet* called End-to-end Memory Network (MemNN) utilizes the benefit of attention to construct models into many computational layers and outperform *PRET+MULT* and the LSTM-based models by *1% - 2%*. The important key of End-to-End Memory Network is an iteractive attention mechanism which is utilized to refine an internal representation and update the internal representation iteratively based on the relevant information from an aspect. However, MemNN still considers that the words of the aspect are the equal contribution and hardly computes the relationship between an aspect and each context word. We observe that our model achieves consistent improvements against the dominant state-of-the-art models so far. Additionally, the improvement of macro-F1 scores is more significant on Laptop and Restaurant datasets. We believe that our model helps to better capture domain-specific opinion words due to additional knowledge from documents via the benefit of the multiple attention mechanisms.

6.4.4 Analysis

Figure 6.3 shows the affectation of λ in multi-task learning for Restaurant dataset. As can be seen, the different share weights λ significantly influence the performance of the model. We can observe that the best performance is using a lower value of λ. A lower value of λ reduces the negative influence of the *aux* task and pay more attention to the *main* task. We can observe that the best performance is using a lower value of λ. Additionally, we can observe our model is more effective than other models in which λ ranges from *68* to *72.56* for Macro-F1.

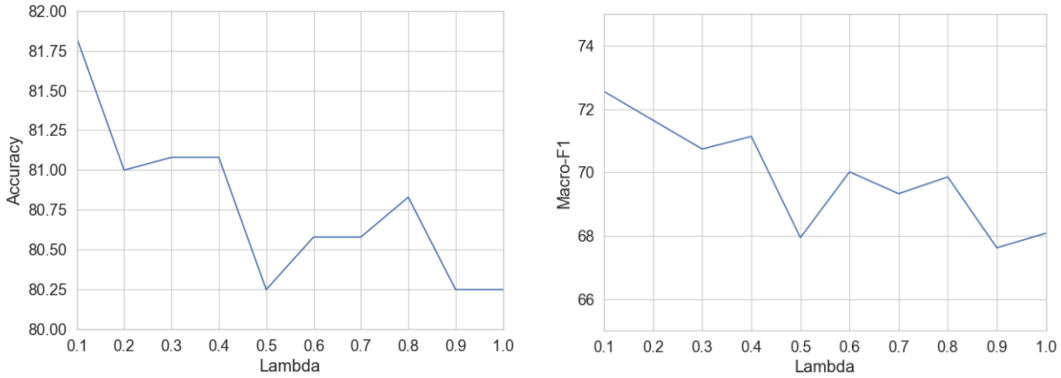


Figure 6.3: The affectation of different share weights λ on both tasks for Restaurant dataset.

We introduce case studies by the heat-map of the importance of sentence as Figure 6.4. The model can detect the important context words affecting to the sentiment polarity of the aspect terms such as the phrases *the best*, *new jersey* and the negation *but not great*. Besides, the multiple keywords can be detected if more than one keyword is existing (*decent* and *but not great*). The most of errors in previous works are non-compositional sentiment expression. For example, the model of [Tang et al., 2016b] cannot predict the sentence *"dessert was also to die for !"* for the aspect *dessert*. The main reason is the sentiment expression *"die for"*, whose meaning could not be composed from its constituents *"die"* and *"for"*. We believe that this is due to associations learned between the words, which ignores *"for"*.

6.5 Conclusion

In this chapter, we propose a novel model which fusing aspect information into a neural network by *Shared input tensor* and *Interactive word-aspect attention fusion*. Multiple attention mechanisms and lexicon information still contribute to modeling the relationship between aspects and their contexts. To tackle the limitation of aspect-level data, we propose multi-task learning to learn interactive knowledge from other related tasks in order to improve the classification performance of our model. Thanks to multi-task learning, the correlation information between two tasks are modeled and manipulated via gating mechanism which controls how much the information would pass through to the final result.

We observe that learning attentions via multi-task learning are active. The form of inductive transfer from multi-task learning can help improve a model by introducing an inductive bias, which causes a model to prefer some hypotheses

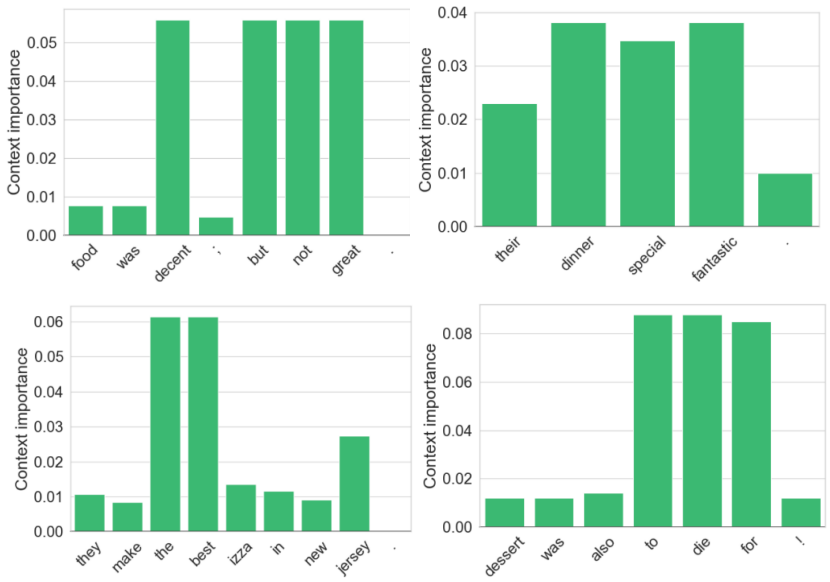


Figure 6.4: The examples showing the importance of sentences are identified by MLAANet

over others. Multiple attention mechanisms are still playing a significant role in our models by capturing the importance of aspect and its context in order to learn to attend associative relationships between exact context words and aspect term. Our model shows a significant improvement against the strong baseline models.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Sentiment analysis is highly challenging, although, the research community has attempted many subproblems and proposed a large number of solutions, none of the subproblems have been completely solved. Recently, sentiment analysis has been playing a big role in the real world, because, the large number of start-ups and established companies that offer sentiment analysis services. Indeed, there is a real need in the industry is that all businesses want to know how consumers perceive their products and services. In the past, most people asked their friends for experiences and advice related to many kinds of topics before making a decision. For example, choosing a good phone or which restaurant is the best one in the city. This is not always effective because we can not refer to many people for the best choice. Nowadays, with the rapid development of the Internet, people can go to websites and obtain the opinions and experiences of consumers before purchasing a product or service. Even, governments and private organizations are also showing strong interests in obtaining a public opinion about their policies and their public image. For example, the president elective of the United State of America with the winning of Mr. Trump thanks to social networking. However, with a huge amount of information is generated every day, filtering useful and reliable information is very difficult. Therefore, these practical needs and the technical challenges will keep the sentiment analysis field vibrant and lively for years to come. We believe that two main research directions are promising.

First, designing novel machine learning algorithms able to learn from large volumes of textual data and to extract domain-specific knowledge (Chapter 4 and Chapter 5). In particular, there is an increasing number of available datasets and an ongoing community effort that supports this approach. Indeed, deep learning networks have become popular and achieve remarkable results for sentiment

analysis task.

Second, the next generation of sentiment classification systems is to solve all subproblems at the same time, because recently, approaches have dealt with each individual subproblem. We believe that a holistic or integrated approach will likely be successful if it enables us to see the full spectrum of the problem. In our models, while tweet-level sentiment classification is sentiment summarization, aspect-level sentiment classification task is diving more into the detail of a tweet. This integration leads us to fully understand the content and relationship information for widespread practical applications. Although deep neural network has been gradually improving, we can be optimistic that the problem will be tackled satisfactorily soon for widespread deep neural networks.

Considering that there are many aspects that remain unexplored. For example, in aspect-level sentiment classification task, the available datasets are still inadequate to train robust classifiers. When bigger datasets are available, deep learning methods could be effectively developed for this task. This leads transfer learning approaches to be utilized to overcome this problem (Chapter 6). Thanks to the advantages of deep neural networks, we propose the various deep learning models to tackle the challenges of social networking, specifically, solving the challenges of Twitter social networking at two levels: Tweet-level sentiment classification and Aspect-level sentiment classification. These tasks support each other and can be an integrated approach to sentiment analysis. Additionally, our works lead to a better understanding of deep learning approaches applied to social networking and potentially make major contributions to the sentiment analysis field and to society.

In summary, this thesis not only focuses on the task of detecting the overall sentiment polarity of tweets by considering textual information but also aim to seek other characteristics of the tweets by recognizing the sentiment polarities of the aspects of the tweets. Our approach differs from existing studies in several ways and can be summarized as follows:

- We develop tweet-level sentiment classification model which classify the sentiment polarity of a tweet. To boost the performance of the sentiment deep neural network classifier, multi-characteristics of each word are considered to provide flavor real-valued hints for enriching word embeddings. Such enriched-word embeddings are modeled through the deep neural network to extract the correct sentiment contextual words in a tweet. The experimental results presented in Chapter 4 shows that our model improves the performance of tweet-level sentiment classification against the baseline models.

- For aspect-level sentiment classification task, we propose novel deep learning models which incorporate aspect information into deep neural networks by considering the advantages of multiple attention mechanisms. On the other

hand, sentiment lexicon feature is interpolated into word vectors in order to highlight the important words of aspect and its context and study the effect of this feature on Aspect-level sentiment classification. The experimental results presented in Chapter 5 indicate that our models outperform the strong state-of-the-art models.

- We propose a multi-task deep neural network to address the drawbacks of aspect-level sentiment analysis task in which transfer learning is applied to interactively study knowledge between tasks and improving the performance of aspect-level sentiment analysis task by overcoming the large limits of aspect datasets. In this sentiment model, lexicon feature weighting is still considered as an important component and contributes to the effectiveness of deep learning models. The experimental results presented in Chapter 6 show that our models outperform the strong state-of-the-art models.

Noted that the whole processes of our models do not require any laborious feature engineering. Our models are end-to-end deep neural networks and enable us to apply to the sentiment analysis of various targets.

To this end, investigating and evaluating the effectiveness of our proposed models in the different perspectives of tweet-level sentiment analysis and aspect-level sentiment analysis allow us to deeply understand the problems of sentiment analysis in different points of view. The experiment results indicate that our models are effective and the deep neural network has a promising application for sentiment analysis and address the drawbacks of traditional machine learning.

7.2 Future Work

Sentiment analysis is a trending topic because of the changes the Web from read-only to read-write in which users interact with each other and sharing various information through social networks with a huge amount of useful information every day. Future sentiment analysis systems need broader and deeper common and commonsense knowledge bases. This will lead to a better understanding of natural language opinions and will efficiently bridge the gap between (unstructured) multimodal information and (structured) machine-processable data. Multimodal information such as video and audio have been growing recently and have become useful for sentiment analysis which can be investigated by using transfer learning approaches. We can apply the multi-task learning as Chapter 6 to explore more complete knowledge and support text classification effectively.

Another direction is addressing the cross-language sentiment analysis task. Since social networking websites are the domain-mixing environment which allows users to post any topics in any domains without restrictions and language style variation. The users can post status which mixes with many languages. We can adopt a similar technique of [Pappas and Popescu-Belis, 2017] to consider various languages for sentiment classification.

Additionally, one of the important points of social networking is user relationship which has not investigated in our thesis. We can model the user relationship by a graph and train them into *Node2Vec* as the work of [Grover and Leskovec, 2016] for deep neural networks. It enables the classifiers to learn more about user similarity towards to improve the classification performance of the classifiers.

Finally, most of our methods in aspect-level sentiment classification task suppose that aspects are given as an input. However, it would be better if the system can detect both the aspects and the sentiment expressed toward given aspects automatically. For example, we can apply unsupervised deep learning or deep neural networks combined with Conditional Random Field (CRF) to extract the aspects in a tweet and subsequently, identifying the sentiment polarities of the aspects. Additionally, we would like to investigate the method for analyzing the opinion on the low resource language (e.g., Vietnamese language) in term of lack of corpus data, sentiment lexicon, and NLP tools.

Bibliography

A. Aue and M. Gamon. Customizing Sentiment Classifiers to New Domains: a Case Study. In *the International Conference on Recent Advances in Natural Language Processing, RANLP-05*, 1 2005.

D. Bahdanau, K. Cho, and Y. Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR 2015*, 2015.

G. Balikas and M.-R. Amini. TwiSE at SemEval-2016 Task 4: Twitter Sentiment Classification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 85–91, Stroudsburg, PA, USA, 2016. Association for Computational Linguistics. doi: 10.18653/v1/S16-1010.

J. Baxter. A Bayesian/Information Theoretic Model of Learning to Learn via Multiple Task Sampling. *Machine Learning*, 28(1):7–39, 1997. doi: 10.1023/A:1007327622663.

Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, 3(Feb):1137–1155, 2003.

D. Bespalov, B. Bai, Y. Qi, and A. Shokoufandeh. Sentiment classification based on supervised latent n-gram analysis. In *Proceedings of the 20th ACM international conference on Information and knowledge management - CIKM '11*, page 375, New York, New York, USA, 2011. ACM Press. ISBN 9781450307178. doi: 10.1145/2063576.2063635.

F. Bravo-Marquez, M. Mendoza, and B. Poblete. Combining strengths, emotions and polarities for boosting Twitter sentiment analysis. In *Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining - WISDOM '13*, pages 1–9, New York, New York, USA, 2013. ACM Press. ISBN 9781450323321. doi: 10.1145/2502069.2502071.

P. Chen, Z. Sun, L. Bing, and W. Yang. Recurrent Attention Network on Memory for Aspect Sentiment Analysis. In *Proceedings of the 2017 Confer-*

ence on Empirical Methods in Natural Language Processing, pages 452–461, Stroudsburg, PA, USA, 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1047.

K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning Phrase Representations using RNN EncoderDecoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Stroudsburg, PA, USA, 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179.

R. Collobert and J. Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning - ICML '08*, pages 160–167, New York, New York, USA, 2008a. ACM Press. ISBN 9781605582054. doi: 10.1145/1390156.1390177.

R. Collobert and J. Weston. A unified architecture for natural language processing. In *Proceedings of the 25th international conference on Machine learning - ICML '08*, pages 160–167, New York, New York, USA, 2008b. ACM Press. ISBN 9781605582054. doi: 10.1145/1390156.1390177.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural Language Processing (Almost) from Scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2001.

N. F. da Silva, E. R. Hruschka, and E. R. Hruschka. Tweet sentiment analysis with classifier ensembles. *Decision Support Systems*, 66:170–179, 10 2014. doi: 10.1016/J.DSS.2014.07.003.

L. Deng, G. Hinton, and B. Kingsbury. New types of deep neural network learning for speech recognition and related applications: an overview. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8599–8603. IEEE, 5 2013. ISBN 978-1-4799-0356-6. doi: 10.1109/ICASSP.2013.6639344.

L. Dong, F. Wei, C. Tan, D. Tang, M. Zhou, and K. Xu. Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 49–54, Stroudsburg, PA, USA, 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-2009.

A. Go, R. Bhayani, and L. Huang. Twitter Sentiment Classification using Distant Supervision. In *Processing*, volume 150, pages 1–6, 2009. ISBN 1012341234. doi: 10.1016/j.sedgeo.2006.07.004.

Y. Goldberg and J. Nivre. Training Deterministic Parsers with Non-Deterministic Oracles. *Transactions of the Association of Computational Linguistics*, 1:403–414, 2013.

A. Grover and J. Leskovec. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pages 855–864, New York, New York, USA, 2016. ACM Press. ISBN 9781450342322. doi: 10.1145/2939672.2939754.

Z. Guan, L. Chen, W. Zhao, Y. Zheng, S. Tan, and D. Cai. Weakly-supervised deep learning for customer review sentiment classification. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 3719–3725. AAAI, 2017. ISBN 9781577357704.

C. Guggilla, T. Miller, and I. Gurevych. CNN- and LSTM-based Claim Classification in Online User Comments. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2740–2751, 2016.

K. Hashimoto, c. xiong, Y. Tsuruoka, and R. Socher. A Joint Many-Task Model: Growing a Neural Network for Multiple NLP Tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933, Stroudsburg, PA, USA, 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1206.

R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier. Exploiting Document Knowledge for Aspect-level Sentiment Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 579–585, 2018.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997. doi: 10.1162/neco.1997.9.8.1735.

M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04*, page 168, 2004. ISBN 1581138889. doi: 10.1145/1014052.1014073.

M. Huang, Q. Qian, and X. Zhu. Encoding Syntactic Knowledge in Neural Networks for Sentiment Classification. *ACM Transactions on Information Systems*, 35(3):1–27, 6 2017. doi: 10.1145/3052770.

S. Ioffe and C. Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, pages 448–456. JMLR.org, 2015.

N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A Convolutional Neural Network for Modelling Sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 655–665, Stroudsburg, PA, USA, 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1062.

Y. Kim. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Stroudsburg, PA, USA, 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181.

S. Kiritchenko, X. Zhu, C. Cherry, and S. Mohammad. NRC-Canada-2014: Detecting Aspects and Sentiment in Customer Reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 437–442, 2014a. doi: 10.3115/v1/S14-2076.

S. Kiritchenko, X. Zhu, and S. M. Mohammad. Sentiment Analysis of Short Informal Texts. *Journal of Artificial Intelligence Research*, 50:723–762, 8 2014b. doi: 10.1613/jair.4272.

A. Kumar and T. M. Sebastian. Sentiment Analysis on Twitter. In *International Journal of Computer Science Issues, IJCSI*, pages Vol. 9, Issue 4, No 3, 2012.

I. Labutov and H. Lipson. Re-embedding words. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 489–493, 2013.

Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, pages II–1188. JMLR.org, 2014.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 5 2015. doi: 10.1038/nature14539.

O. Levy and Y. Goldberg. Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 302–308, Stroudsburg, PA, USA, 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-2050.

X. Li, L. Bing, W. Lam, and B. Shi. Transformation Networks for Target-Oriented Sentiment Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 946–956, 2018.

B. Liu. Sentiment Analysis and Subjectivity. In *Handbook of Natural Language Processing*, number 1, pages 1–38, 2010. ISBN 9781420085921. doi: 10.1145/1772690.1772756.

J. Liu and Y. Zhang. Attention Modeling for Targeted Sentiment. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 572–577, 2017.

D. Ma, S. Li, X. Zhang, and H. Wang. Interactive Attention Networks for Aspect-Level Sentiment Classification. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)*, 2017. ISBN 2164728637.

A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150. Association for Computational Linguistics, 2011. ISBN 9781932432879. doi: 978-1-932432-87-9.

J. McAuley, C. Targett, Q. Shi, and A. 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 - SIGIR '15*, pages 43–52, New York, New York, USA, 2015. ACM Press. ISBN 9781450336215. doi: 10.1145/2766462.2767755.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of International Conference on Learning Representations (ICLR 2013)*, 2013a.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119, 2013b.

A. Mishra, K. Dey, and P. Bhattacharyya. Learning Cognitive Features from Gaze Data for Sentiment and Sarcasm Classification using Convolutional Neural

Network. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 377–387, Stroudsburg, PA, USA, 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1035.

S. Mohammad, S. Kiritchenko, and X. Zhu. NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 321–327, 2013.

H. Nguyen and M.-L. Nguyen. A Deep Neural Architecture for Sentence-Level Sentiment Classification in Twitter Social Networking. In *International Conference of the Pacific Association for Computational Linguistics (PACLING 2017)*, pages 15–27. Springer, Singapore, 2018. doi: 10.1007/978-981-10-8438-6{_}2.

B. Pang and L. Lee. Opinion Mining and Sentiment Analysis. *Foundations and Trends® in Information Retrieval*, 2(12):1–135, 2008. doi: 10.1561/1500000011.

N. Pappas and A. Popescu-Belis. Multilingual Hierarchical Attention Networks for Document Classification. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1015–1025, 2017.

M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar. SemEval-2014 Task 4: Aspect Based Sentiment Analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, 2014. ISBN 9781941643242. doi: 10.3115/v1/S14-2004.

Q. Qian, B. Tian, M. Huang, Y. Liu, X. Zhu, and X. Zhu. Learning Tag Embeddings and Tag-specific Composition Functions in Recursive Neural Network. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1365–1374, Stroudsburg, PA, USA, 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1132.

Q. Qian, M. Huang, J. Lei, and X. Zhu. Linguistically Regularized LSTM for Sentiment Classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1679–1689, Stroudsburg, PA, USA, 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1154.

S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, pages 91–99. MIT Press, 2015.

Y. Ren, Y. Zhang, M. Zhang, and D. Ji. Improving Twitter sentiment classification using topic-enriched multi-prototype word embeddings. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 3038–3044. AAAI Press, 2016.

H. Saif, Y. He, and H. Alani. Semantic Sentiment Analysis of Twitter. In *Proceedings of the 11th international conference on The Semantic Web - Volume Part I*, pages 508–524. Springer-Verlag, 2012. ISBN 978-3-642-35175-4. doi: 10.1007/978-3-642-35176-1{_}32.

C. d. Santos and M. Gatti. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 69–78, 2014.

R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics, 2011. ISBN 978-1-937284-11-4.

R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211. Association for Computational Linguistics, 2012.

R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.

M. Speriosu, N. Sudan, S. Upadhyay, and J. Baldridge. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the First Workshop on Unsupervised Learning in NLP*, pages 53–63. Association for Computational Linguistics, 2011. ISBN 978-1-937284-13-8.

S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus. End-To-End Memory Networks. In *Proceedings of Advances in Neural Information Processing Systems 28 (NIPS 2015)*, pages 2440–2448, 2015. ISBN 1551-6709. doi: v5.

M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede. Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*, 37(2):267–307, 6 2011. doi: 10.1162/COLI{_}a{_}00049.

D. Tang, F. Wei, B. Qin, M. Zhou, and T. Liu. Building Large-Scale Twitter-Specific Sentiment Lexicon : A Representation Learning Approach. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 172–182, 2014a.

D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin. Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1555–1565, Stroudsburg, PA, USA, 2014b. Association for Computational Linguistics. doi: 10.3115/v1/P14-1146.

D. Tang, B. Qin, and T. Liu. Learning Semantic Representations of Users and Products for Document Level Sentiment Classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1014–1023, Stroudsburg, PA, USA, 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1098.

D. Tang, B. Qin, X. Feng, and T. Liu. Effective LSTMs for Target-Dependent Sentiment Classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307, 2016a.

D. Tang, B. Qin, and T. Liu. Aspect Level Sentiment Classification with Deep Memory Network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 214–224, Stroudsburg, PA, USA, 2016b. Association for Computational Linguistics. ISBN 978-1-945626-25-8. doi: 10.18653/v1/D16-1021.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention Is All You Need. 2017. ISBN 9781577357384. doi: 10.1017/S0140525X16001837.

D. T. Vo and Y. Zhang. Target-dependent twitter sentiment classification with rich automatic features. Technical report, 2015.

J. Wang, L.-C. Yu, K. R. Lai, and X. Zhang. Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*,

volume 2, pages 225–230, Stroudsburg, PA, USA, 2016a. Association for Computational Linguistics. doi: 10.18653/v1/P16-2037.

X. Wang, Y. Liu, C. SUN, B. Wang, and X. Wang. Predicting Polarities of Tweets by Composing Word Embeddings with Long Short-Term Memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1343–1353, Stroudsburg, PA, USA, 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1130.

X. Wang, W. Jiang, and Z. Luo. Combination of Convolutional and Recurrent Neural Network for Sentiment Analysis of Short Texts. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016b.

Y. Wang, M. Huang, X. Zhu, and L. Zhao. Attention-based LSTM for Aspect-level Sentiment Classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615, Stroudsburg, PA, USA, 2016c. Association for Computational Linguistics. doi: 10.18653/v1/D16-1058.

M. Yang, W. Tu, J. Wang, F. Xu, and X. Chen. Attention Based LSTM for Target Dependent Sentiment Classification. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI 2017)*, 2017.

R. Zazo, A. Lozano-Diez, J. Gonzalez-Dominguez, D. T. Toledano, and J. Gonzalez-Rodriguez. Language Identification in Short Utterances Using Long Short-Term Memory (LSTM) Recurrent Neural Networks. *PLOS ONE*, 11(1): e0146917, 1 2016. doi: 10.1371/journal.pone.0146917.

Publications

Journals

[1] Huy Nguyen, Le-Minh Nguyen: "A Multiple Features-Aware Contextual Neural Network for Twitter-level Sentiment Classification," Submitted in: Neurocomputing (2018).

[2] Huy Nguyen, Le-Minh Nguyen: "LWAANet: A Lexicon-Aware Word-Aspect Attention Network for Aspect-level Sentiment Classification on Social Networking," Being revised in: Expert Systems and Applications (2018).

Conference papers

[3] Huy Nguyen, Le-Minh Nguyen: "A deep neural architecture for sentence-level sentiment classification in Twitter social networking," in Proceedings: International Conference of the Pacific Association for Computational Linguistics, (PACLING 2017), 2017.

[4] Huy Nguyen, Le-Minh Nguyen: "Sentence Modeling with Deep Neural Architecture using Lexicon and Character Attention Mechanism for Sentiment Classification," in Proceedings: The 8th International Joint Conference on Natural Language Processing, (IJCNLP 2017), 2017.

[5] Huy Nguyen, Le-Minh Nguyen: "Effective Attention Networks for Aspect-level Sentiment Classification," in Proceedings: International Conference on Knowledge and Systems Engineering (KSE 2018), 2018.