Exploring Emotion Embedding and the Dimensional Model of Emotion's Predictive Power in Sentiment

by

Sean Chanheum Cho

A research paper
presented to the University of Waterloo
in partial fulfillment of the
requirement for the degree of
Master of Mathematics
in
Computational Mathematics

Supervisor: Prof. Jesse Hoey

Waterloo, Ontario, Canada, 2017

© Chanheum Cho 2017

I hereby declare that I am the sole author of this report. This is a true copy of the report, including any required final revisions, as accepted by my examiners.

I understand that my report may be made electronically available to the public.

Abstract

Artificial intelligence (AI) has been integrated to people's daily lives. However, the AI system's lack of understanding of human emotions impede an effective communication between the system and the human interactants. As an example, if a search engine understands the intent of the searcher, it should be able to return favorable results regardless of the correct queries. Emotions take a big part in communication as they allow humans to empathize and understand one another. An AI would also need to understand emotions for an effective interaction with humans. Affective computing research has surged recently in order to tackle this problem.

Sentiment analysis, which can be thought as a subset of affective computing, allows an AI system to understand a limited portion of human emotions, and has been used widely in systems that involve reviews to recommend related products, such as movies, electronics, or books. The problem with the traditional sentiment analysis is that it only uses the polarity label as a proxy to the full emotion, which makes the system difficult to make fine-grained judgments. There exists a model of emotion that represents an emotion as a point in the emotion space [26], a more detailed model than the one-dimension polarity model. The model consists of three independent axes: evaluation, potency, and activity in the EPA format [29], or equivalently, valence, dominance, and arousal in the VAD format [36]. We believe that by learning the three-dimension emotion model and extracting sentiment from it, we can predict the sentiment more precisely than the use of the typical polarity based model.

In this paper, we explore validity of using the three-dimension emotion model as opposed to the naive polarity model in predicting the sentiment of given text data. We set the work of Tang et al. [34] as the baseline where they construct word embeddings with integrated sentiment information, called *sentiment embeddings*. As opposed to their approach of using polarity label to guide the word embedding and the corresponding classifier, we use the three-dimension emotion model, namely the VAD vectors [36], and train *emotion embeddings*.

In the experiment, a recently established corpus with emotion labels, EmoBank [5], is used along with a common corpus for sentiment, Stanford Sentiment Treebank (SST) [32], and a large text dataset, text8 [22]. We compare and contrast the prediction power between the *sentiment embeddings* and the *emotion embeddings* on EmoBank corpus, while checking their generality on the SST corpus. We also analyze *emotion embedding* itself by visualizing the embeddings using t-distributed Stochastic Neighbor Embedding (t-SNE) [20]. The visualization showed the lack of context generalization in *emotion embedding*,

and we discuss possible reasons: language models, and datasets. The possible causes are studied independently. We analyze the language model by observing related researches using the same language model and their results. In analyzing the datasets, the skip-gram model [24] in Word2Vec is used, so the language model factor does not cause any unexpected results in the analysis of dataset. The text8 corpus was used for comparing the context generalization.

The paper then discusses two possible future extensions to fight the lack of generalization. One method is semi-supervised learning, which uses a small set of labeled data along with a large set of unlabeled data. The small labeled data guides the emotion judgement while the large unlabeled data fine-tunes the context representation. The other method is mining labeled data using proxies in social network platforms such as emoticons on Twitter, or emotion buttons in Facebook, which doesn't require an expert annotator. These two methods may be able to improve the context representation of the *emotion embedding*.

We end the paper with possible applications which may help humans by employing an intelligent system that can understand emotions, hence empathize with interactants such as persons suffering from depression or loneliness. Especially, we look into chatbots and robotics where the interaction with humans is important.

Acknowledgements

I would like to thank my colleagues in CHIL, and Computational Mathematics for all their help and all the little people who made this possible.

Table of Contents

List of Tables **viii**

List of Figures **ix**

1 Introduction **1**

1.1 The Emotion Space 2

1.2 Neural Probabilistic Language Model 4

1.3 Sentiment Analysis and Sentiment Embeddings 5

2 Dataset **9**

2.1 EmoBank 9

2.1.1 Reader versus Writer Perspectives 10

2.2 Stanford Sentiment Treebank 11

3 Models **13**

3.1 Sentiment Model 13

3.2 Emotion Model 15

3.3 Hyperparameters 16

3.4 Methods 16

4 Results 19

4.1 Sentiment versus Emotion Experiment 19

4.2 Cross-dataset Experiment 20

4.3 Investigating *Emotion Embeddings* 21

4.3.1 t-distributed Stochastic Neighbor Embedding 22

5 Discussions and Future Work 30

5.1 Contextual Generalization 30

5.1.1 Semi-supervised Learning 30

5.1.2 Labeled Data Mining 31

5.2 Bi-perspective Modeling 32

5.3 Applications 33

5.3.1 Chatbots 33

5.3.2 Robotics 34

6 Conclusion 35

References 37

List of Tables

1.1 Top 5 closest words to *good* and *bad* from pre-trained Word2Vec [24]. 6

2.1 Domain Distribution of EmoBank [5]. 10

2.2 A sample from the EmoBank corpus [32]. The labels show the emotion felt from the reader's perspective where values scale from zero to four. Four is the most positive, active, and dominant in the VAD format, respectively. . 11

2.3 A sample from the SST corpus [32]. The labels show the sentiment from zero being the most negative to four being the most positive. 12

4.1 Effect of Emotion Information. 20

4.2 Cross-Dataset Experiment 21

5.1 An example of a smooth transition between two points in the hidden space [3]. 33

List of Figures

1.1 Example of words represented in the EPA space. Each word has a corresponding EPA point. The values are extracted from Interact [9], an ACT simulator. The plot is scaled to $[-3, 3]$ range for a better presentation than the use of the full range of EPA values. 3

1.2 Neural architecture of NPLM where w_t is the word to predict, given a context of w_i^j. The embeddings have m dimensions while the linear layer and the hyperbolic tangent layer has h dimensions where h is the length of the hidden layer. The output is the probability of the word w_t given the context. h is a hyperparameter a user can set, and we use $h = 20$ for all our experiments as described in Section 3.3. The model produces a probability value for each of the vocabulary, so the dimension is $|V|$, the number of distinct words in the dataset. 5

1.3 The sentiment model from the baseline work of Tang et al. [34]. The context to hyperbolic tangent layers, and the word prediction layer on the top right branch are the same as NPLM. Another linear layer is added at the same level as the word prediction layer, which transforms the h dimension vector to a vector with $|class|$, which represents the number of distinct sentiment labels in the dataset. As an example, $|class|$ is five for a five-class sentiment classification. The sentiment model simultaneously trains the context and the sentiment. In our experiment, $|class|$ is 5. A softmax layer is added on top of the second linear layer, so the sentiment results are represented as probabilities. In classification settings, the model selects the class with the highest probability as its output. 7

3.1 The emotion model. Notice the three classifiers added to the basic NPLM structure. The subscripts V, A, D stand for valence, arousal, and dominance, respectively. 15

3.2 The context-only model. It has an additional classification layer on top of the basic NPLM model. The weights in the shaded area are trained with the basic NPLM as shown in Figure 1.2, and fixed. Then, the classifier (the unshaded portion) is trained. 18

4.1 t-SNE visualization of similar words and their relationships. The t-SNE algorithm is applied to the pretrained Word2Vec [24] to visualize word relations [25]. 23

4.2 t-SNE visualization of *sentiment embeddings* learned from EmoBank. Only the first 500 words in the vocabulary are plotted for presentation. Each point is the location of a word projected to the two-dimension space from a 50-dimension embedding space. All points are annotated with their word. . 24

4.3 t-SNE visualization of *emotion embeddings* learned from EmoBank. Only the first 500 words in the vocabulary are plotted for presentation similar to Figure 4.2 25

4.4 t-SNE visualization of word embeddings learned from text8 corpus. Similar to Figure 4.2 and Figure 4.3, only the first 500 words from the vocabulary is plotted for a reasonable presentation. 27

4.5 t-SNE visualization of word embeddings learned from EmoBank corpus. Only the first 500 words from the vocabulary is plotted. 28

5.1 The general autoencoder framework, where x is an input, f is an encoding function, g is a decoding function, and $\tilde{x}$ is the reconstructed input obtained by $g(f(x))$. L is a loss function for guiding $g(f(x))$ towards x such as the mean squared error [10]. 31

Chapter 1

Introduction

Artificial intelligence (AI) has made astonishing breakthroughs in the recent years, dissolving itself to human society from word predictions in messaging services to recommendations for food, movies, and events. AI systems now analyze people's life patterns and assist them in making decisions efficiently. Yet, the systems' inability to understand human emotions has provided unreasonable responses to the interactant. As an example, a productivity maximizing tool would fit as many tasks as possible within the working hours without considering the user's feelings such as mood or tiredness. An ideal system would empathize with the user and customize their routine for the maximum productivity by scheduling a meeting when they are the most productive, and booking some time off. The field of affective computing has surged to tackle the problem of better understanding the human emotion, and investigate methods to integrate the models of human emotions in computer systems for better interactions. In societal interactions, the emotion between the interactants is important for effective communication, which suggests an intelligent system that understands emotions would cooperate better with humans. Ideally, if a search engine can understand the intent of the search user, it would be able to return preferable results even when the query is not accurately referring to the intended results.

The research in emotion analysis has used proxy data such as speeches, images, or videos to infer one's emotion. Recently, Martinez at al. [21] used physiological signals and unsupervised deep learning techniques to predict a person's emotion in discrete states, which outperformed traditional statistical methods in predictions. The emotions can also be inferred from text data. The most common form of emotion analysis in text is sentiment analysis in natural language processing (NLP) research, which predicts the polarity of a given text. However, the usual information a system extracts from one dimensional polarity label, may it be multi-class discrete or sometimes continuous, is not enough to represent the perceived emotions. In this paper, we investigate extending the sentiment analysis framework to include other dimensions in emotions, and the effect of using such emotion label in analyzing sentiments in texts in comparison to only using naive polarity label commonly used in the sentiment analysis. We used Neural Probabilistic Language Model [2] as the language model to represent the texts and integrated a simple multi-layer perceptron (MLP) for classification, following the work of Tang et al. [34].

1.1 The Emotion Space

The emotion space, which consists of three independent dimensions, was suggested by Osgood [26] as shown in Figure 1.1. The three dimensions are evaluation, potency, and activity in EPA format [29, 12, 11], and valence, dominance, and arousal in VAD format [36], respectively. It differs from other emotion models where emotions are presented as discrete states as in Ekman's six basic emotions [8]. The continuous multidimensional model of emotion allowed researchers to apply mathematical operations on emotions. We use the terms the emotion space and the affective space interchangeably.

One example of the use of the affective space is affect control theory (ACT) [29] in the field of social psychology. ACT models an interaction between two agents where emotions emerge from the interaction. It approaches the interaction as a game where two agents try to minimize the difference in their emotions relative to the social norm given their situation. In this framework, an AI system can predict what it has to do or how it should react in an interaction with a human.

ACT was also extended to Bayesian affect control theory (BayesACT) [12], a probabilistic variant of ACT. ACT was a deterministic model that yields only one vector representation which means that given the situation, context and a certain action, the agent must be feeling only one emotion. This is not true in real life settings where mixed emotions can be felt simultaneously, and may cause some misunderstandings in an interaction. BayesACT captures these various possibilities and yields a list of probable emotions in an interaction. BayesACT was applied in a tutoring system [11] and proved that it can lead to an efficient management of human resources via effective communications. Students performed better with an AI system that minimized the difference in the agents' emotions. It allowed the students to focus their cognitive power solely on studying and learning the subjects rather than wasting their energy observing and reacting to the tutor's emotion. This is an example of how an AI system that understands human emotions can positively impact the society.

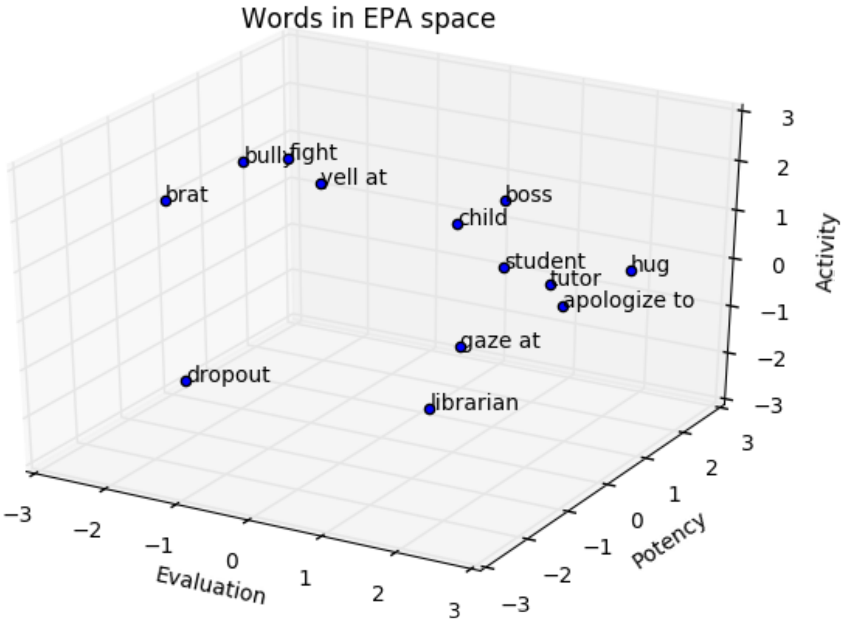

Figure 1.1: Example of words represented in the EPA space. Each word has a corresponding EPA point. The values are extracted from Interact [9], an ACT simulator. The plot is scaled to $[-3, 3]$ range for a better presentation than the use of the full range of EPA values.

In this paper, we use the three-dimension emotion space as the labels of the given text data. We hope the emotion labels accurately describe how the text is being perceived, hence improve the polarity prediction in sentiment analysis. This approach may overcome the limitation in ACT formulation where it requires the sentences to be structured as actor, behavior, object, and setting. The texts in real-life scenarios, such as reviews, comments, or tweets don't always follow the correct grammatical structure, and transforming the sentences in this format may lose some information such as nuance. Hence, we believe that preserving the structure and building new models for the textual setting to determine the emotion would be a better choice.

1.2 Neural Probabilistic Language Model

Language modeling (LM) has been the core component in NLP. The key in LM is learning numeric vector representations of words based on the distributional properties of words [2]. Learning the word distributions help NLP systems understand grammatical structure, information about the training corpora, and more. The learned vectors called *word embeddings*, lead breakthroughs in speech recognition [23], machine translation [1], as well as sentiment analysis [34]. Often, better language models trained on larger dataset yield better performance in the given tasks [14]. Many language models have been developed using neural networks such as fully connected networks [2, 24], recursive neural networks [31], or convolutional neural networks [16]. There exist language models that learn the word distributions without the use of label, via unsupervised learning, such as the skip-gram and continuous-bag-of-words models in Word2Vec framework [24]. However, the unsupervised methods require a large dataset, typically more than a million words. Since the labeled dataset we use is small with only a few tens of thousands of words, we did not consider using the unsupervised approach in our experiments.

We use the neural probabilistic language model (NPLM) developed by Bengio et al. [2] to compare the results with the baseline research by Tang et al. [34]. NPLM learns the model $f(w_t, \cdots, w_{t-n+1}) = \hat{P}(w_t|w_{t-n+1}^{t-1}), w_t \in V$ where V is the vocabulary set from data, w_t is the t-th word, $w_i^j = (w_i, w_{i+1}, \cdots, w_{j-1}, w_j)$ is a sub-sequence, and $\hat{P}(w_t|w_{t-n+1}^{t-1}) \approx \hat{P}(w_t|w_1^{t-1})$ in a statistical model of language. In other words, the model learns to predict the most probable word given a sequence of words, w_i^j, in a sentence. This sequence of words, w_i^j is also called the *context*.

The likelihood function consists of two steps in NPLM. The first step is a lookup procedure where a lookup matrix $C \in \mathbb{R}^{|V| \times m}$ translates i-th word from V to its word embedding with m dimensions. In this step, m is a hyperparameter that we can define, and the word embeddings are learned via training after being randomly initialized. A popular pre-trained word embedding, Word2Vec, uses 300-dimension vectors for word embeddings [24], but we set m as 50 due to the computing power, dataset size, and time constraints. This step is also called a *lookup layer* in the neural network setting.

The second step is the part where the model learns the probability function over the words in a neural network framework. The function learns the conditional probability given a context, defined as $(C(w_{t-n+1}, \cdots, C(w_{t-1}))$. Let this function be noted as g so that $f(i, w_{t-1}, \cdots, w_{t-n+1}) = g(i, C(w_{t-1}), \cdots, C(w_{t-n+1})) = \hat{P}(w_t = i|w_{t-n+1}^{t-1})$. The learning process adjusts the weights in the layers to maximize this probability. The resulting neural architecture is shown in Figure 1.2. The lookup matrix C can be used as a

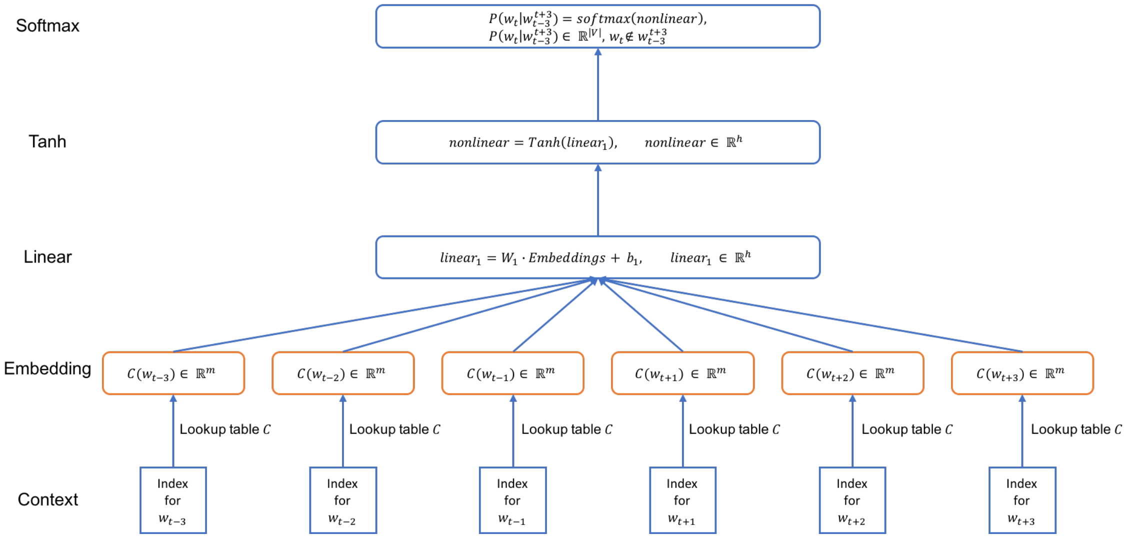
Softmax
$P(w_t|w_{t-3}^{t+3}) = softmax(nonlinear)$,
$P(w_t|w_{t-3}^{t+3}) \in \mathbb{R}^{|V|}, w_t \notin w_{t-3}^{t+3}$
Tanh
$nonlinear = Tanh(linear_1)$, $nonlinear \in \mathbb{R}^h$
Linear
$linear_1 = W_1 \cdot Embeddings + b_1$, $linear_1 \in \mathbb{R}^h$
Embedding
$C(w_{t-3}) \in \mathbb{R}^m$
$C(w_{t-2}) \in \mathbb{R}^m$
$C(w_{t-1}) \in \mathbb{R}^m$
$C(w_{t+1}) \in \mathbb{R}^m$
$C(w_{t+2}) \in \mathbb{R}^m$
$C(w_{t+3}) \in \mathbb{R}^m$
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Context
Index for w_{t-3}
Index for w_{t-2}
Index for w_{t-1}
Index for w_{t+1}
Index for w_{t+2}
Index for w_{t+3}

Figure 1.2: Neural architecture of NPLM where w_t is the word to predict, given a context of w_i^j. The embeddings have m dimensions while the linear layer and the hyperbolic tangent layer has h dimensions where h is the length of the hidden layer. The output is the probability of the word w_t given the context. h is a hyperparameter a user can set, and we use $h = 20$ for all our experiments as described in Section 3.3. The model produces a probability value for each of the vocabulary, so the dimension is $|V|$, the number of distinct words in the dataset.

generic embedding matrix for word embeddings once the model is trained, similar to the pre-trained Word2Vec embeddings.

1.3 Sentiment Analysis and Sentiment Embeddings

Sentiment analysis is a crucial component in NLP research in understanding human reactions to a given context such as products or movie reviews. It is considered as a classification task where the sentiment analysis system evaluates whether the given text is positive or negative. Sentiment analysis has a big potential when integrated in various applications. One example could be recommendation systems where the system suggests movies that the user responded with a positive comment. Most of the advancements in sentiment analysis has been made in building a better classifier using different techniques [15, 32, 13, 19] while some research tried to fine-tune the dataset to extract more information. Especially, the work of Socher et al. [32] established a common dataset researchers in sentiment analysis can use to compare their results. Not only that, they fine-tuned and relabeled the existing movie review dataset developed by Pang et al. [27] creating the dataset with five class labels, similar to five star rating systems commonly found in review platforms. Systems integrated with a five-class sentiment classifier would be able to provide finer responses and more personalized services to their users.

An important part of developing such sentiment classifier is to develop a general, yet highly representable word embeddings. The research in sentiment analysis has been focusing on the classifying methods [15, 32, 13, 19], and use pre-trained word embeddings on a large data such as Word2Vec [24], or GloVe [28]. Although these pre-trained embeddings have a great semantic representation due to the size of the data they are trained on, it lacks the information regarding sentiments. This is because the pre-trained word embeddings are trained under unsupervised learning with the distributional aspect of the language being the only focus of the learning. The result is that their representation of words with opposite sentiments are mapped similar to one another. For example, in the pre-trained Word2Vec [24], the word *good* and *bad* are within the five-nearest words from each other as shown in Table 1.1 because those two words can be used interchangeably in sentences in distributional aspect. The sentences still make sense, but they mean the opposite.

Table 1.1: Top 5 closest words to *good* and *bad* from pre-trained Word2Vec [24].

good	great	**bad**	terrific	decent	nice
bad	**good**	terrible	horrible	Bad	lousy

The work of Tang et al. [34] tried to tackle this problem by injecting sentiment information into the word embeddings, effectively creating *sentiment embeddings*. In their work, this is achieved by transforming the unsupervised learning aspect of NPLM to a supervised learning.

The language models such as NPLM are considered unsupervised because the data they use typically don't have an explicit label. The language models *infer* the label by trying to predict the target word, w_t, given its surrounding words, w_i^j. The label is not explicitly annotated, but is taken from the data itself. Note that because it is predicting a word based on the surrounding words, it is possible to have multiple labels. For example, the model can predict *good* for the target word given a context "The book is *target*.", but it can also predict *bad* which can also appear in the same context. This is one of the reason why the word embeddings based only on language models show similarities among words with opposite meanings. On the other hand, sentiment analysis heavily rely on the annotated data where the data is labeled by experts. Sentiment prediction is a classification task where the classifier learns the correct output by comparing the outcome to the ground-truth label. The model developed by Tang et al. makes use of both where the context is learned without labels, and the sentiment of the context is learned with the explicit labels in the dataset. Some underlying weights are shared as shown in Figure 1.3, so the model learns the weight values that optimizes for both the word prediction likelihood, and the sentiment prediction, given a context, w_i^j. They collected a large labeled dataset from Twitter with the emoticons as proxies to sentiment labels, automating the annotation process. However, they also reported that a portion of the data they collected have been deleted from Twitter, and they couldn't collect the same data again for the future use.

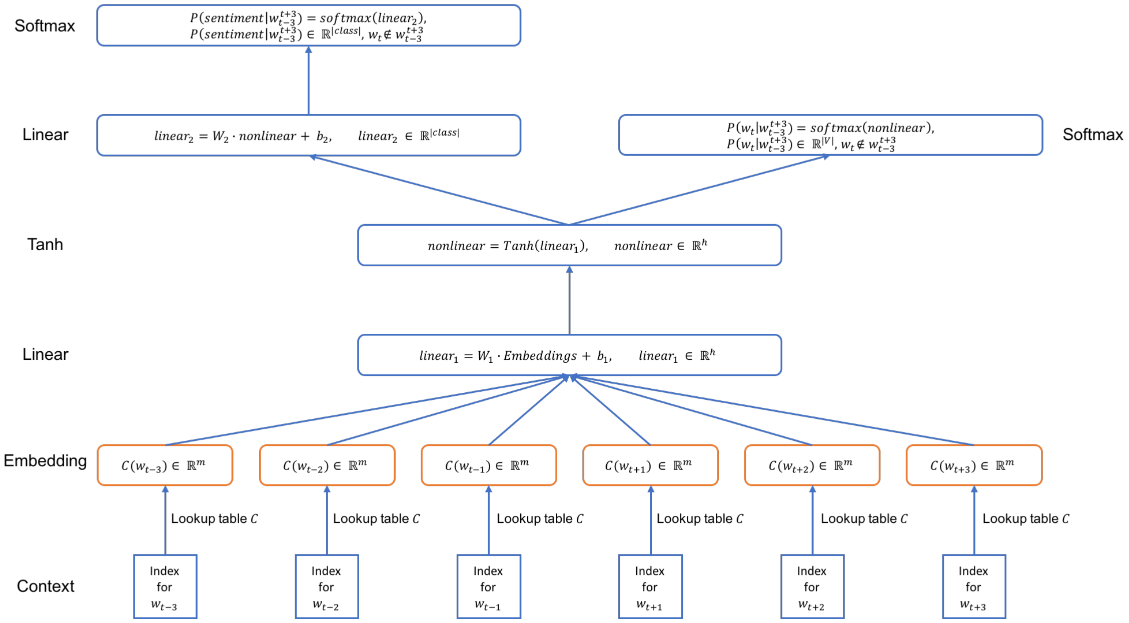
Softmax
$P(sentiment|w_{t-3}^{t+3}) = softmax(linear_2)$,
$P(sentiment|w_{t-3}^{t+3}) \in \mathbb{R}^{|class|}$, $w_t \notin w_{t-3}^{t+3}$
Linear
$linear_2 = W_2 \cdot nonlinear + b_2$, $linear_2 \in \mathbb{R}^{|class|}$
$P(w_t|w_{t-3}^{t+3}) = softmax(nonlinear)$,
$P(w_t|w_{t-3}^{t+3}) \in \mathbb{R}^{|V|}$, $w_t \notin w_{t-3}^{t+3}$
Softmax
Tanh
$nonlinear = Tanh(linear_1)$, $nonlinear \in \mathbb{R}^h$
Linear
$linear_1 = W_1 \cdot Embeddings + b_1$, $linear_1 \in \mathbb{R}^h$
Embedding
$C(w_{t-3}) \in \mathbb{R}^m$
$C(w_{t-2}) \in \mathbb{R}^m$
$C(w_{t-1}) \in \mathbb{R}^m$
$C(w_{t+1}) \in \mathbb{R}^m$
$C(w_{t+2}) \in \mathbb{R}^m$
$C(w_{t+3}) \in \mathbb{R}^m$
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Context
Index for w_{t-3}
Index for w_{t-2}
Index for w_{t-1}
Index for w_{t+1}
Index for w_{t+2}
Index for w_{t+3}

Figure 1.3: The sentiment model from the baseline work of Tang et al. [34]. The context to hyperbolic tangent layers, and the word prediction layer on the top right branch are the same as NPLM. Another linear layer is added at the same level as the word prediction layer, which transforms the h dimension vector to a vector with $|class|$, which represents the number of distinct sentiment labels in the dataset. As an example, $|class|$ is five for a five-class sentiment classification. The sentiment model simultaneously trains the context and the sentiment. In our experiment, $|class|$ is 5. A softmax layer is added on top of the second linear layer, so the sentiment results are represented as probabilities. In classification settings, the model selects the class with the highest probability as its output.

In our work, we extend this model by incorporating a few more classifiers in parallel, each for additional emotion labels. We investigate the full emotion model's predictive power in sentiment, and analyze the resulting *emotion embeddings* through our experiments.

Chapter 2

Dataset

We use two different datasets for the experiment. Firstly, we use EmoBank [5] to learn the *emotion embeddings*. Then, we use the learned *emotion embeddings* on Stanford Sentiment Treebank (SST) dataset to see its generality. In this chapter, we explain the data.

2.1 EmoBank

EmoBank is the first fully annotated corpus with the three-dimension emotion values in VAD format. It contains 10,062 English sentences from various sources, and they are annotated bi-perspectively: reader's and writer's. The details regarding the perspectives is explained in Chapter 2.1.1. The motivation for the authors was that the research trend in sentiment analysis shifted from binary classification to fine-grained classification since the fine-grain labeled sentiment data [32] were released. They pointed out that the inadequate models of emotion hinge the research in understanding emotions [33], hence the data used in sentiment analysis should be labeled with a model that better represents human emotions, the three-dimension emotion space [26].

The corpus complements abundant texts from social media, and reviews by adding several domains such as newspapers, travel guides, or fictions, to construct a balanced corpus which may be used for a variety of tasks including sentiment analysis. The annotations were collected via a reduced version of self-assessment manikin (SAM) [4], the only standardized instrument in acquiring VAD values. In the original SAM, there are nine choices for each dimension in VAD. However, the choices are reduced to five in collecting the annotations to reduce cognitive load on the survey participants [5]. Each sentence in the corpus was labeled by five different annotators per emotional dimension per perspective. They filtered fraudulent and overrepresented responses as well as divergent responses, and created EmoBank with 10,062 sentences. The domain distribution of the sentences are shown in Table 2.1.

A portion of EmoBank's corpus is labeled in two emotion models, the three-dimension model and the discrete model. The purpose of this is to find a transformation function from the discrete states to the continuous model. This allows the comparison between the research performed with the discrete emotion models and the the dimensional model of emotions. EmoBank is the largest standard for any emotional format [5], so the experiments in this paper used its corpus to learn *emotion embeddings*. A sample of the data is shown in Table 2.2

Table 2.1: Domain Distribution of EmoBank [5].

Domain	# Sentences
news headlines	1,192
blogs	1,336
essays	1,135
fiction	2,753
letters	1,413
newspapers	1,314
travel guides	919

2.1.1 Reader versus Writer Perspectives

Emotions typically emerge from an interaction according to the formulation of ACT [29]; the interactants do not share the same feeling. In textual data, this can be thought of as an interaction between the writer and the reader where they perceive the text differently. As an example, a newspaper title ”Germany defeats Brazil in the World Cup Semi-Final” may be neutral from the writer's perspective, but may evoke different emotions if the reader is German or Brazilian. Similarly, the emotions from readers' perspective, and the writers' perspective did not converge during the annotation process, hence the authors decided to include labels from both perspectives. In our experiment, we only use the labels in reader's perspective because the reader perspective was found to converge better

Table 2.2: A sample from the EmoBank corpus [32]. The labels show the emotion felt from the reader's perspective where values scale from zero to four. Four is the most positive, active, and dominant in the VAD format, respectively.

Text	Valence	Arousal	Dominance
we believe that any terrorist act is a crime against humanity and against the will of god, because it deliberately indents to injure and kill innocent people	2	2	2
why is someone so young not having fun with friends on a Friday night	1	2	2
a few weeks ago, someone broke into their shed to steal their garden tools	3	1	0

among the annotators, and emits stronger emotions according to the response analysis of EmoBank [5].

2.2 Stanford Sentiment Treebank

While EmoBank is the first gold standard for emotion labeled corpus, Stanford Sentiment Treebank (SST) [32] has been the standard for sentiment analysis since its inception in 2013. Many sentiment analysis research [15, 32, 19] compare their results on this dataset to distinguish a better algorithm for sentiment classification. SST was developed by Socher et al. [32], by decomposing the IMDB movie review dataset [27] for their recursive neural tensor network (RNTN) framework and relabeling them with fine-grain annotations. The corpus includes 11,855 sentences, similar to the size of EmoBank corpus. Since the goal of this paper is not about pushing the boundary of the state-of-the-art sentiment classification, we use this dataset as out-of-sample test data for generality only. A small sample of the dataset is shown in Table 2.3.

Table 2.3: A sample from the SST corpus [32]. The labels show the sentiment from zero being the most negative to four being the most positive.

Text	Label
the story loses its bite in a last minute happy ending that's even less plausible than the rest of the picture	0
is this progress?	1
near the end takes a whole other meaning	2
occasionally melodramatic, it's also extremely effective	3
a masterpiece four years in the making	4

Chapter 3

Models

In our experiment, we set the work of Tang et al. [34] as the baseline and train the model on EmoBank corpus. We first re-implement the model presented by them with slight modifications, and feed only the valence dimension, which represents the polarity, as the label. Then, two additional classifiers for the arousal and the dominance dimensions are added in parallel to the valence dimension and the contexts to guide the model to learn emotion representations. We can separate the affective dimensions and conduct experiments on them independently because the dimensions are orthogonal to each other, forming three independent axis [5].

Our hypothesis is that the sentiment prediction accuracy will improve as we provide closely related, but independent extra information to the system. It is analogous to how people can better understand a complex research paper when related visualizations are also provided on top of the textual information. Texts and images are independent sources of understanding, yet they refer to the same concept. Similarly, the valence, which only provides the polarity information, and the other two dimensions are independent, but refer to the same emotion. We examine the effect of using the additional dimensions by comparing the sentiment prediction accuracies on five-class fine-grained classification. This is different from the baseline work where the classification is binary because we believe the detailed information should help find precise sentiment polarity of a given text.

3.1 Sentiment Model

Typical word embeddings such as Word2Vec based on word distributions excelled on learning the semantics, but failed to capture the sentiment aspect in text data. As a result, words with opposite sentiment that appear in similar contexts are considered close to each other in the embedding space as shown in Table 1.1. Ideally, they should be far apart from one another due to their opposing meanings. In an attempt to dissolve sentiment information in word embeddings, Tang et. al [34] proposed a few different models which creates *sentiment embeddings*. Their models combine NPLM with a fully connected layer at the same level as the context layer in NPLM as shown in Figure 1.3. A softmax function is applied to the output of the classifier branch to convert it to a probability.

They suggested models with two different schemes. One was a prediction model where the contexts and sentiments were trained to predict a word, and the corresponding sentiment given the context. The model tried to yield the most probable output $\hat{P}(v, w_t|w_1^{t-1})$ given the sequence of words in a sentence in this scheme, where v is the predicted label in valence. For an example, the context portion yields a likelihood $\hat{P}(w_t|w_1^{t-1})$, the probability of obtaining w_t as the output given the context w_1^{t-1}. Similarly, the classifier portion predicts the label given the same context. The model learns to maximize the context likelihood, and minimizes the classification error simultaneously.

The other scheme has the same structure as the first scheme, but it optimizes the hinge loss function which compares the relative likelihood of one output in comparison to all other possible outputs. Because it yields the highest likelihood based on relative results, it is called the ranking model. This optimization method effectively maximizes the margin or errors in support vector machine (SVM) classification framework [30].

The experiment reported by Tang et al. [34] indicates the prediction model performs better in general, so we use the prediction model as our baseline model. In this paper, we call this the *sentiment model* because it follows the traditional sentiment analysis framework which only use the polarity label. We made some modifications to their original model including the use of a hyperbolic tangent function, $tanh(x)$, instead of the hard hyperbolic tangent, $hTanh(x)$, and performing a fine-grained five-class classification instead of a binary classification. The hard hyperbolic tangent was defined as Equation 3.1 in the work of Tang et al. [34] where the hyperbolic tangent is a smooth function defined as in Equation 3.2.

$$hTanh(x) = \begin{cases} \text{-1}, & \text{if } x < \text{-1}, \\ x, & \text{if -1} \leq x \leq 1, \\ 1, & \text{if } x > 1. \end{cases} \tag{3.1}$$

$$tanh(x) = \frac{sinh(x)}{cosh(x)} \tag{3.2}$$

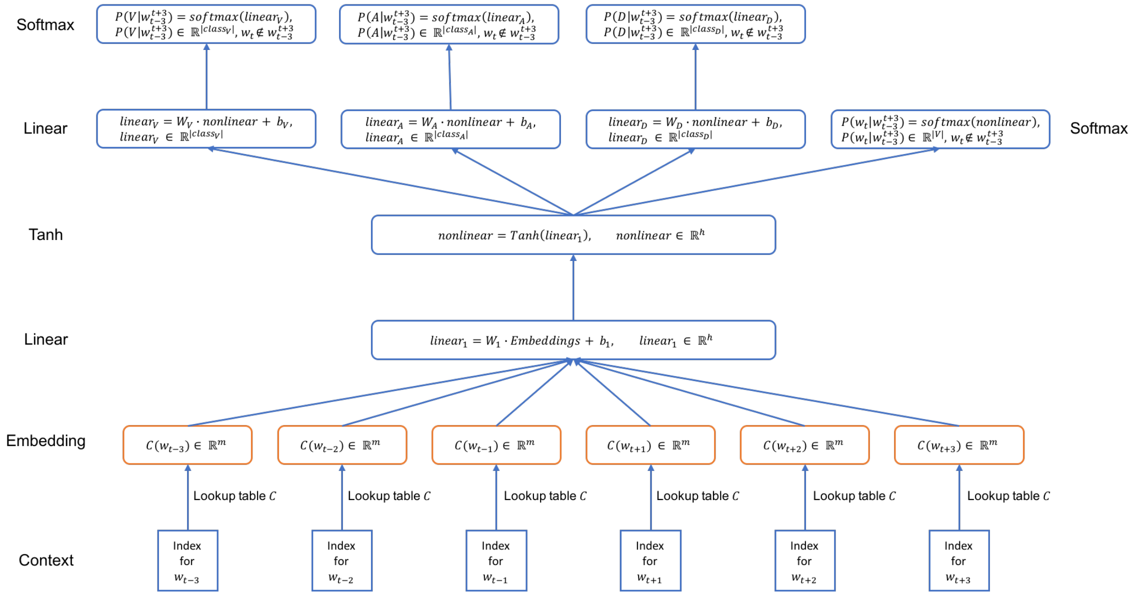
Softmax
$P(V|w_{t-3}^{t+3}) = softmax(linear_V)$,
$P(V|w_{t-3}^{t+3}) \in \mathbb{R}^{|class_V|}, w_t \notin w_{t-3}^{t+3}$
$P(A|w_{t-3}^{t+3}) = softmax(linear_A)$,
$P(A|w_{t-3}^{t+3}) \in \mathbb{R}^{|class_A|}, w_t \notin w_{t-3}^{t+3}$
$P(D|w_{t-3}^{t+3}) = softmax(linear_D)$,
$P(D|w_{t-3}^{t+3}) \in \mathbb{R}^{|class_D|}, w_t \notin w_{t-3}^{t+3}$
Linear
$linear_V = W_V \cdot nonlinear + b_V$,
$linear_V \in \mathbb{R}^{|class_V|}$
$linear_A = W_A \cdot nonlinear + b_A$,
$linear_A \in \mathbb{R}^{|class_A|}$
$linear_D = W_D \cdot nonlinear + b_D$,
$linear_D \in \mathbb{R}^{|class_D|}$
$P(w_t|w_{t-3}^{t+3}) = softmax(nonlinear)$,
$P(w_t|w_{t-3}^{t+3}) \in \mathbb{R}^{|V|}, w_t \notin w_{t-3}^{t+3}$
Softmax
Tanh
$nonlinear = Tanh(linear_1), \quad nonlinear \in \mathbb{R}^h$
Linear
$linear_1 = W_1 \cdot Embeddings + b_1, \quad linear_1 \in \mathbb{R}^h$
Embedding
$C(w_{t-3}) \in \mathbb{R}^m$
$C(w_{t-2}) \in \mathbb{R}^m$
$C(w_{t-1}) \in \mathbb{R}^m$
$C(w_{t+1}) \in \mathbb{R}^m$
$C(w_{t+2}) \in \mathbb{R}^m$
$C(w_{t+3}) \in \mathbb{R}^m$
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Context
Index for w_{t-3}
Index for w_{t-2}
Index for w_{t-1}
Index for w_{t+1}
Index for w_{t+2}
Index for w_{t+3}

Figure 3.1: The emotion model. Notice the three classifiers added to the basic NPLM structure. The subscripts V, A, D stand for valence, arousal, and dominance, respectively.

3.2 Emotion Model

Our hypothesis is that the sentiment prediction accuracy would improve given more information related to the sentiment. The extra information being the two dimensions, arousal and dominance, in the full emotion model in the emotion space. A classifier for these dimensions are added at the same stage as the classifier for valence and the likelihood output of a word given the context as shown in Figure 3.1. We expect this model to perform better because it is given more information. The model will decide whether the information actually help predicting the sentiment during the training, but the presence of the abundant data is expected to positively impact the model learning the representation. We call it the *emotion model*, and the resulting word embedding the *emotion embedding*.

Training this model can also be viewed as a *multitask learning* [6], as the network is jointly trained on multiple classification tasks with its underlying weights shared. Improved performance on one task, when the network is jointly trained on multiple tasks such as part-of-speech (POS) tags, semantic roles prediction, and/or named entity tags, was reported by Collobert et al. [6] due to better generalization. This is similar to our hypothesis that the related labels in the arousal and dominance dimension should improve the sentiment prediction when they are trained simultaneously.

3.3 Hyperparameters

We use the same hyperparameters as the *sentiment model* because we try to reproduce the same learning environment as Tang et al. [34]. The values of the word embedding are initialized as a uniform distribution, $U(-0.01, 0.01)$, and the weights and biases in the neural net layers such as the linear layer and the hyperbolic tangent layer are initialized with a different uniform distribution that follows $U(\frac{-0.01}{InputLength}, \frac{0.01}{InputLength})$. The context length or the window size is set as 7, so the models look at the three words before and after as the context for the middle or the target word. The size of the embedding dimension is set as 50, and the length of the hyperbolic tangent layer output in Figure 1.3 and Figure 3.1, also known as the length of the hidden layer, is set to 20. We used AdaGrad [7] as our optimization algorithm. These values match the original research conducted by Tang et al. [34]. We ran the all experiments for 10 epochs for a fair comparison.

3.4 Methods

In the experiment, we report the sentiment prediction accuracies from the two models. We train the word embeddings using the sentiment model, and the emotion model. The learned embeddings from the two models are called *sentiment embedding* [34], and *emotion embedding*, respectively. To compare their predictive performance in sentiment, we take a look at the inferred result in the valence classifier in each model because valence is the dimension responsible for polarity in emotion [5]. More precisely, we obtain the class predicted by $P(sentiment|w_i^j)$ in Figure 1.3, and $P(V|w_i^j)$ in Figure 3.1, and compare their accuracies. The i and j are $t-3$ and $t+3$ respectively because we set the context length as 7. The inferred result from the valence classifier is the predicted sentiment of the given context. By comparing the valence classifier result, both models can be compared in the same measure where the only difference between them is the use of the arousal and dominance dimensions during the training.

The use of two extra dimension in the emotion space will have the shared weights in the tangent layer and the first linear layer to differ between the models. This is because the weights in the emotion model have to optimize for predicting all three dimension simultaneously as well as the context where the weights in the sentiment model only optimize for valence and the context.

We also tested the context-only model where the model does not use any labels for training the word embeddings. This is the basic NPLM discussed in Chapter 1.2. In the context-only model's case, we replaced the softmax layer for word prediction with a multilayer perceptron (MLP), the same classifier used in the sentiment model and the emotion model, on top of the NPLM architecture so it can predict the sentiment. There are two training procedures for this model. The initial training was done on the basic NPLM structure as shown in Figure 1.2, so it learns the word embedding with only word distribution, and the corresponding weights in the hidden layers. The learned embeddings and weights are used in the architecture as shown in Figure 3.2 to learn sentiment predictions. The sentiment prediction test for the context-only model is performed in this model after training.

Notice that there is no word prediction in the model architecture. This is because the training architecture and the inference architecture are different in this model. The weights, and the embeddings up to the hyperbolic tangent layer is learned through the basic NPLM architecture as in Figure 1.2. In the context-only model, we only train the second linear layer and the softmax layer for sentiment prediction. This effectively learns a classifier that predicts the sentiment given a learned embedding that only includes word distribution information.

The modification in the context-model has to be done because the basic NPLM in Figure 1.2 doesn't have the sentiment classification ability in the model. As opposed to the context-only model, both the sentiment model and the emotion model includes the classifier(s) in their architectures. Therefore, the context and the sentiment are trained simultaneously, and the valence classifier included in their architectures are used in testing.

As the purpose of the experiment is to investigate the embedding's predictive power, the end-to-end training in the context-only model is not considered. This is because if we trained the model end-to-end without context learning, the embeddings would not be useful as general word embeddings; it would instead learn some representation where word embeddings with context information is required on top of the representation to solve any NLP problem.

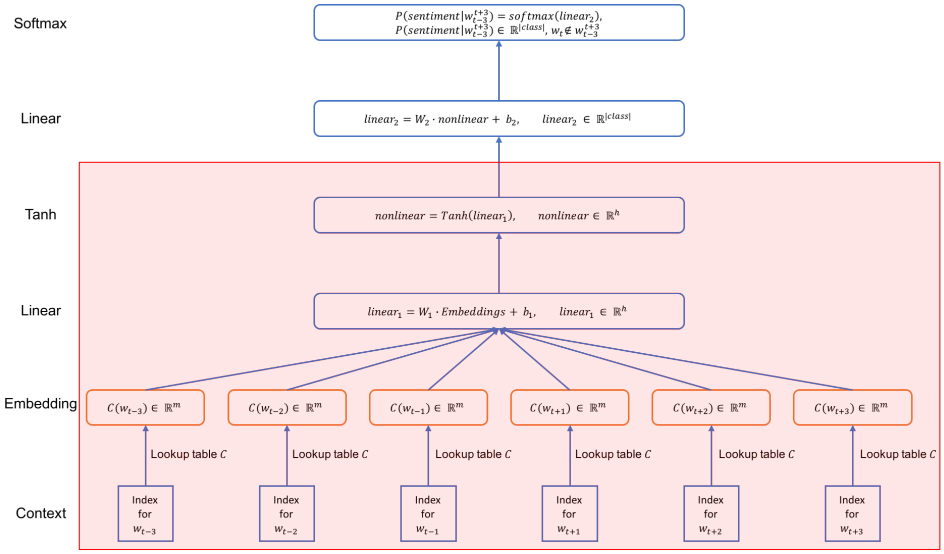
Softmax
$P(sentiment|w_{t-3}^{t+3}) = softmax(linear_2)$,
$P(sentiment|w_{t-3}^{t+3}) \in \mathbb{R}^{|class|}, w_t \notin w_{t-3}^{t+3}$
Linear
$linear_2 = W_2 \cdot nonlinear + b_2, \quad linear_2 \in \mathbb{R}^{|class|}$
Tanh
$nonlinear = Tanh(linear_1), \quad nonlinear \in \mathbb{R}^h$
Linear
$linear_1 = W_1 \cdot Embeddings + b_1, \quad linear_1 \in \mathbb{R}^h$
Embedding
$C(w_{t-3}) \in \mathbb{R}^m$
$C(w_{t-2}) \in \mathbb{R}^m$
$C(w_{t-1}) \in \mathbb{R}^m$
$C(w_{t+1}) \in \mathbb{R}^m$
$C(w_{t+2}) \in \mathbb{R}^m$
$C(w_{t+3}) \in \mathbb{R}^m$
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Lookup table C
Context
Index for w_{t-3}
Index for w_{t-2}
Index for w_{t-1}
Index for w_{t+1}
Index for w_{t+2}
Index for w_{t+3}

Figure 3.2: The context-only model. It has an additional classification layer on top of the basic NPLM model. The weights in the shaded area are trained with the basic NPLM as shown in Figure 1.2, and fixed. Then, the classifier (the unshaded portion) is trained.

Chapter 4

Results

We present the comparison between the use of emotion information and the sentiment information in Table 4.1. The context-only model is included to see the improvement over unsupervised setting. The context-only model, *sentiment model*, *emotion model* are all trained with EmoBank corpus, where the dataset was randomly split with 7:1:2 train, validation, test ratio, which is equivalent to 5430, 761, and 1532 sentences respectively. This is because the model filters sentences shorter than seven words, the defined context length. After the filtering, there were only 7723 sentences left. The split sets were labeled with the split tags; zero for train, one for validation, and two for test, so all the models can use the same data for training, validating, and testing for a fair comparison.

4.1 Sentiment versus Emotion Experiment

As shown in Table 4.1, the use of emotion information significantly improves the sentiment predictability over the context-only model. The context-only model was tested for sanity check, and it was surprising to see the 17% accuracy which is about the same as random guesses in five-class classification. It was shown to have no predictive power in EmoBank corpus. The use of additional dimension is found to be helpful in improving the sentiment prediction accuracy although not as significant from the context-only model. The accuracy with the full VAD format labels improved about 1% over the *sentiment model*, and this improvement was found to be statistically significant with 99% significance level with a p-value of 0.001356 according to t-test analysis.

From this experiment, we were able to find that the inclusion of sentiment labels boosts the predictive power in classification tasks, and the more related information the models are fed, the better representation it learns. The better representation translates to the better prediction results. The more related information in our experiment was the emotion labels, which includes the arousal and the dominance dimensions. It means that the embeddings learned from the full emotion model in the affective space better represent sentiments than the embeddings learned from only the valence dimension. Intuitively, sentiment is a subset of the full emotion, so predicting the subset from a model trained on the full information must be simple. This also aligns with the findings of Collobert et al. [6] that the network jointly trained on independent but highly related tasks yield better performance than the model that is trained to perform only one task.

Furthermore, the emotion model also predicts the other two dimensions in emotion when it predicts for the valence dimension. Extrapolating the full emotion from the sentiment model is impossible as it requires the labels in the other two dimensions. In order for the context-only model and the sentiment model to achieve the same task, the models need to learn the embeddings and the weights according to their model, and extra layers for the arousal and the dominance classifier have to be added after the initial training. This is similar to how the context-only model needs two training steps for sentiment classification. In this aspect, the emotion model is superior because it learns the full emotion simultaneously without any extra training steps, and yields predictions in all three emotion dimensions. We report the average of five experiments with different random seeds in Table 4.1.

Table 4.1: Effect of Emotion Information.

Models	Valence Accuracies	Arousal Accuracies	Dominance Accuracies
Context Only	16.93 %	N/A	N/A
Sentiment Model	55.75 %	N/A	N/A
Emotion Model	57.02 %	59.38 %	65.79 %

4.2 Cross-dataset Experiment

The classification accuracy of above 50% is astonishing considering five-class classification because a random guess would yield about 20% accuracy, which is similar to the result from the context only model in Table 4.1. Excited with the result, we tested the trained models on the SST dataset to investigate the generality of the learned embeddings. When the cross-dataset test was performed, we used the same model that is trained on the train set from the EmoBank corpus. The difference here is that the models are tested on the SST dataset.

One preprocessing we had to do for this experiment was to filter the sentences in SST corpus that has less than 100% word coverage with the vocabulary from EmoBank. In other words, only the sentences in SST corpus that can be built from a combination of words from EmoBank is used as the test data. This is because the smallest language unit in NPLM is a word instead of a character, thus it is unable to map unseen words, also known as out-of-vocabulary (OOV) words, to corresponding embeddings. The embeddings for the OOV words would be randomly initialized similar to the initialization of word embeddings in training. This means that the OOV words would not contain any information regarding the context nor the emotion. Therefore, only the sentences that has 100% word coverage with respect to the EmoBank dataset were used as the cross-dataset experiment.

Table 4.2: Cross-Dataset Experiment

Models	Accuracies
Sentiment Model	20.67 %
Emotion Model	20.03 %

The accuracy seems to drop in the emotion model by about 0.6 %. However, the results were the same for both models as the difference is statistically insignificant as shown in Table 4.2. Both the sentiment model and the emotion model had no predictive power in an unseen dataset because 20% is the same with a random guess in five-class classification. This was surprising because the word coverage for these sentences was 100%, meaning the sentiment and emotion embeddings for the words in the sentences were learned during the training. Curious with the result, we proceeded in analyzing the embeddings.

4.3 Investigating *Emotion Embeddings*

In order to find the validity of our embeddings as a general representation of emotions in text, we compared and contrasted the embeddings learned from both models in Word2Vec [24] framework. This is because the skip-gram or the continuous-bag-of-words (CBOW) models that learn word embeddings without supervision are proven to be general enough that many NLP applications simply use the pre-trained Word2Vec model instead of training their own embeddings [15]. We visualize the embeddings using t-distributed Stochastic Neighbor Embedding, and analyze the reason why the embeddings failed to generalize.

4.3.1 t-distributed Stochastic Neighbor Embedding

t-distributed Stochastic Neighbor Embedding (t-SNE) [20] is a data visualization technique that maps high-dimensional data to a two or three-dimension space. t-SNE has been commonly used in the deep learning community [18] because it visualizes incomprehensible, hidden high dimensional data. It is also easier to optimize and produce better visualizations than other methods such as principle component analysis (PCA) [20]. Data such as texts or images, which resides in two dimensional space, contains high dimensional information that can be extracted by learning the data. The absolute positions of data in t-SNE visualization have no meaning as they focus on placing similar datapoints together in a neighborhood of each other. Hence, similar words appear in one region of the visualization according to relative distances in the embedding space among the words.

For example, the absolute positions of *king* and *queen* in Figure 4.1 do not have any meaning except the difference in their position is similar in their plural forms. This example shows that the t-SNE preserves the word relations although the dimensions are reduced for visualization. In this specific example, similar singular words like *king* and *queen* take similar vectors to become plural. The word location, and the difference between *king* and *queen* are in a 50-dimension space due to the embedding length. The t-SNE algorithm projects them in a two-dimension space for visualization which inherits the relations given the embeddings include reasonable semantic information as in the pre-trained Word2Vec [25].

Techniques like t-SNE help us understand the data via dimensionality reduction and visualization. The word embeddings learned from *sentiment model* and *emotion model*, are both 50-dimension vectors, so we used t-SNE to reduce the 50-dimension to two-dimension and visualize them. By observing the visualization, we analyzed why their prediction performance were lower in the cross-dataset experiment. Figure 4.2 visualizes the learned embeddings in the *sentiment model*, and Figure 4.3 shows the learned embeddings in the *emotion model*. As shown in the figures, the embeddings have been clustered without much semantic connections amongst clustered words, indicating that the learning had been primarily based on the sentiment or emotion labels rather than the word distribution. For example, in a region near $(x, y) = (20, -20)$ in Figure 4.2, the word *force* is near *forces*. They are clustered together because they have close meanings; they appear on similar contexts with similar sentiments. However, words like *press* or *east* are also in the same

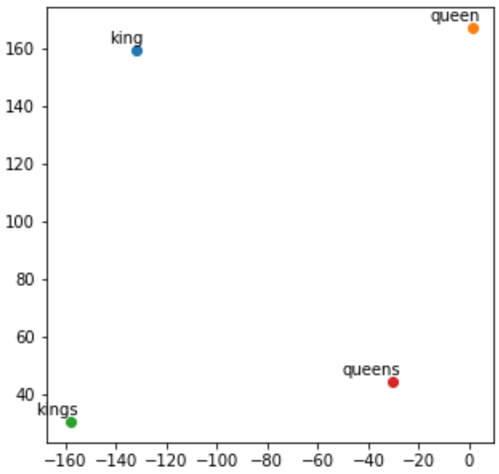

Figure 4.1: t-SNE visualization of similar words and their relationships. The t-SNE algorithm is applied to the pretrained Word2Vec [24] to visualize word relations [25].

region. The word *press* may be there because it is used as a verb and appear on similar sentiment, but the word *east* have no relation to words like *force* or *press*; it is not a verb. The only reason it is clustered in the same region could be that it appears in texts with similar sentiments as the words like *force* or *press*.

The embeddings learned a mixed representation of the sentiment or emotion, and the word contexts. The *learned* embeddings can be observed in Figure 4.2, and in Figure 4.3 as the words are clustered in different regions. If it had not learned the representation, the embeddings would be randomly scattered in the t-SNE visualization as they have no relations with one another. We trained the embeddings with two measures: the word distribution, and the sentiment or emotion labels. The visualized embeddings does not display reasonable relations in word distribution as described in the previous example, yet they clustered in some regions. This means that the embeddings learned more about the sentiment or emotion than it did about the word contexts. Therefore, the words are clustered in a region that yields similar sentiment predictions.

The lack of representations in word distributions in the learned embeddings could be the reason why the model failed to generalize in the cross-dataset experiment. At this point, we had two hypotheses that may explain why the models failed to learn the word distribution. One was that the NPLM model is not a good language model to learn the context of the

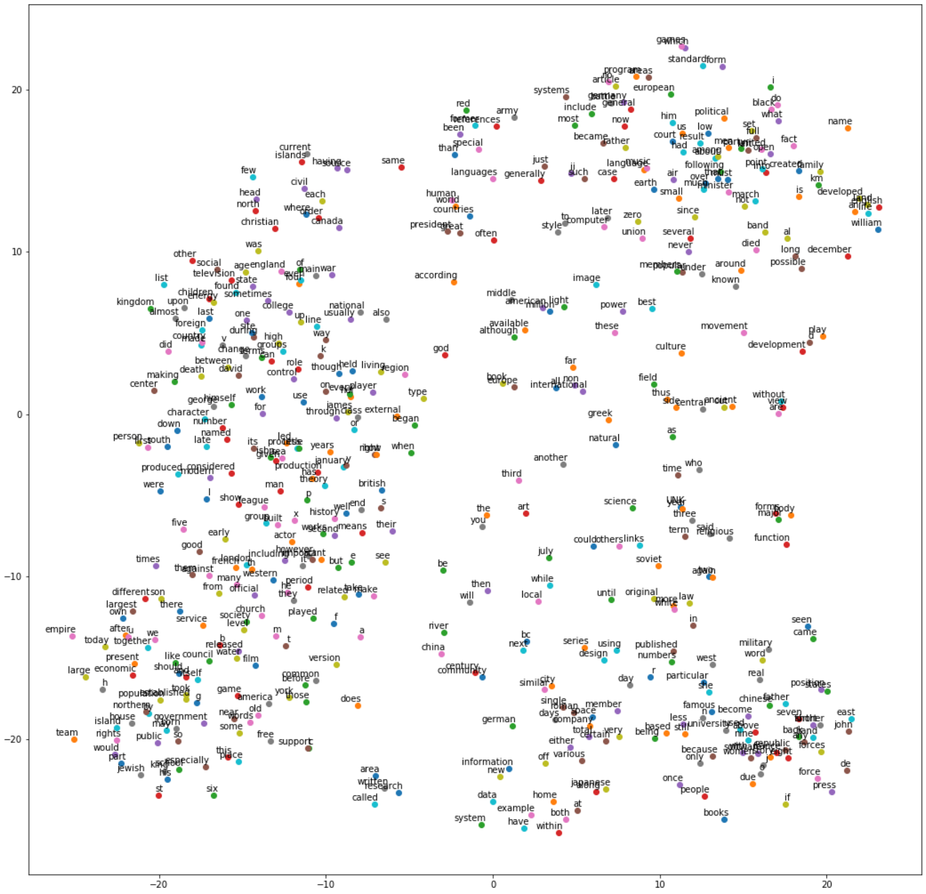

Figure 4.2: t-SNE visualization of *sentiment embeddings* learned from EmoBank. Only the first 500 words in the vocabulary are plotted for presentation. Each point is the location of a word projected to the two-dimension space from a 50-dimension embedding space. All points are annotated with their word.

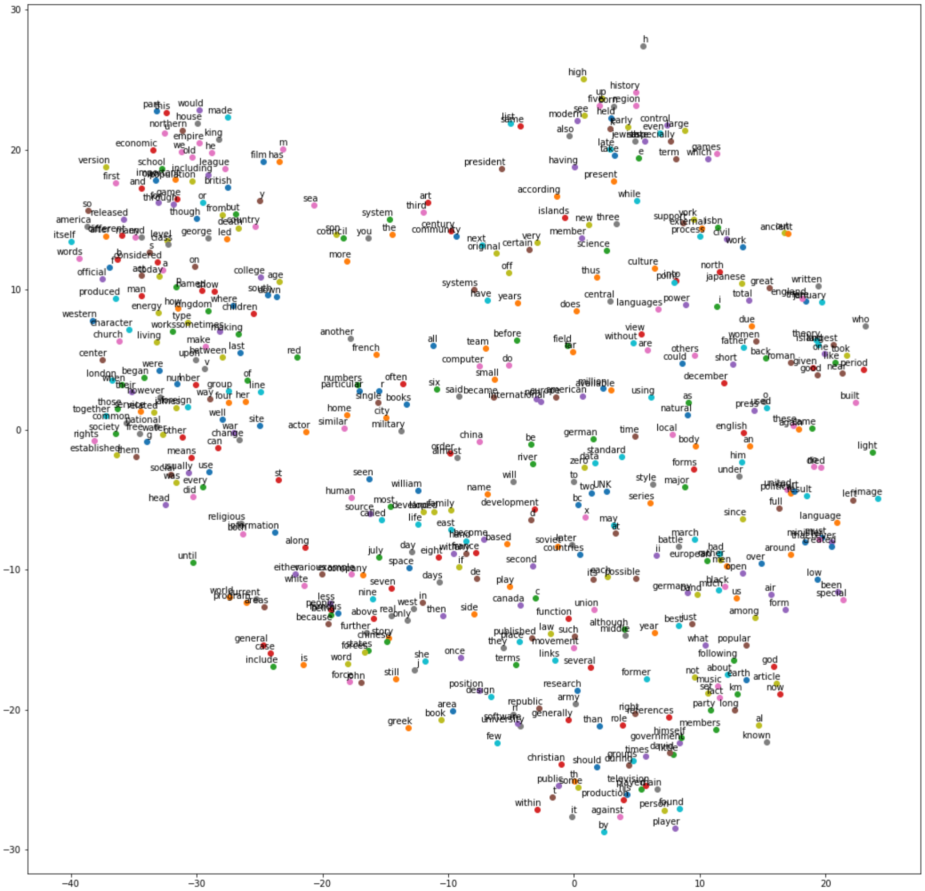

Figure 4.3: t-SNE visualization of *emotion embeddings* learned from EmoBank. Only the first 500 words in the vocabulary are plotted for presentation similar to Figure 4.2

dataset. After all, more advanced language models such as Word2Vec models are commonly used in NLP [15] due to improved ability to learn the word distribution with faster training time. This hypothesis was rejected immediately because research has shown [2, 34] that it was able to learn the word distribution. Furthermore, the baseline research also showed that it successfully dissolved the sentiment information in the word embeddings as they were able to query words which returned similar words in sentiment. As an example, when they queried the word "good", it returned words like "excellent", or "nice", which are not only similar in semantics, but also in sentiment.

The second hypothesis was that the dataset was not large enough to learn the general word distribution of the language. This was highly probable because in the baseline work, Tang et al. [34] trained their model on a large dataset they collected from Twitter. In order to test this hypothesis independent of the first one, we used the Word2Vec framework, namely the skip-gram model, to compare the contexts encoded in the embeddings via unsupervised learning. The embeddings learned from EmoBank corpus were compared against the embeddings learned from a large text corpus, text8 dataset, commonly used as a dataset to test the embedding algorithms [22].

We first present the word embeddings learned from text8 corpus with the skip-gram model on Figure 4.4. It is easily observable that word embeddings successfully dissolved the contexts in its dimensions as words with similar semantics are clustered together. For example, letters are clustered on the top left region of the two-dimension space, while modal verbs are put together on the top right region. Note that the letters are in the dataset as one-character words. The hyperparameters are set to be the same as the NPLM ones such as the embedding dimensions, and number of words to look before or after as a context.

Similarly, we learned the word embedding from EmoBank in the same setting, and visualized them in Figure 4.5. The skip-gram model only relies on the word distribution, so it only looks at the context in data. As it can be seen in the figure, the embeddings did not learn much about the context; the only reasonable similarity that can be found is at the region $(x, y) = (10, -7)$ where some numbers are clustered together. This means that the learned embedding successfully learned an indicator that can represent the emotions as a combination of the embedding dimensions as it was seen in the sentiment prediction accuracies, but the embeddings could not fully dissolve the context information with the EmoBank dataset.

Looking at the number of distinct words in the datasets, EmoBank had 16,142 words, whereas text8 corpus had 17,005,207 words. The number of words were several orders of magnitude greater with the text8 dataset where the skip-gram model was able to extract

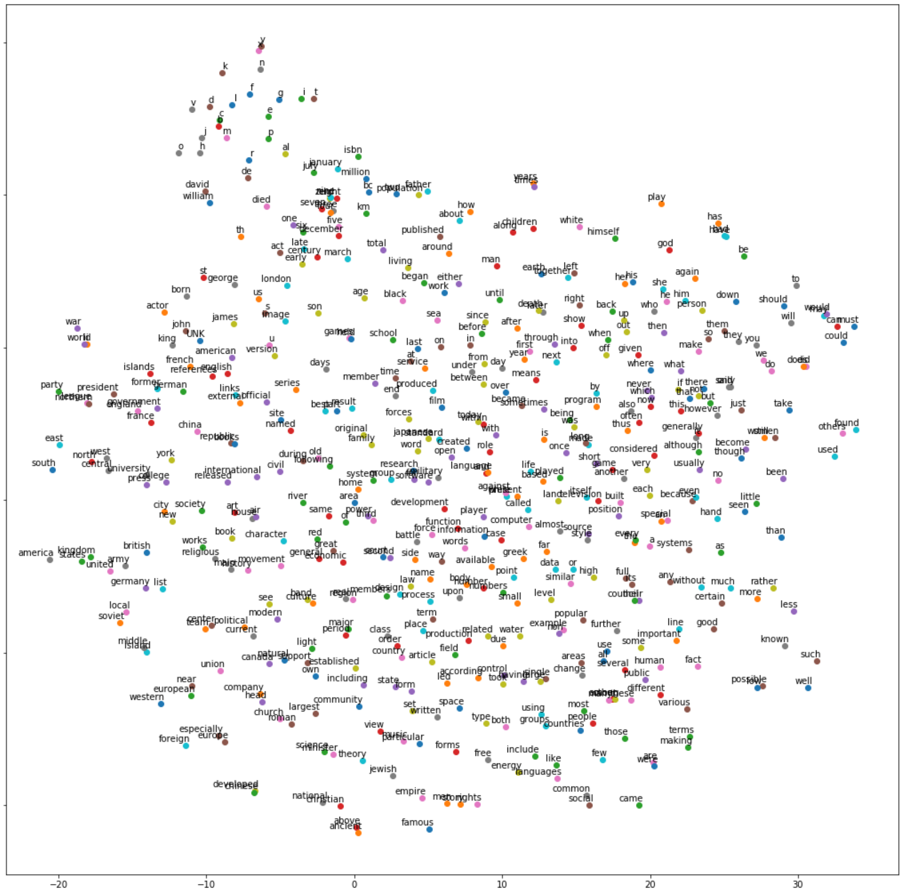

Figure 4.4: t-SNE visualization of word embeddings learned from text8 corpus. Similar to Figure 4.2 and Figure 4.3, only the first 500 words from the vocabulary is plotted for a reasonable presentation.

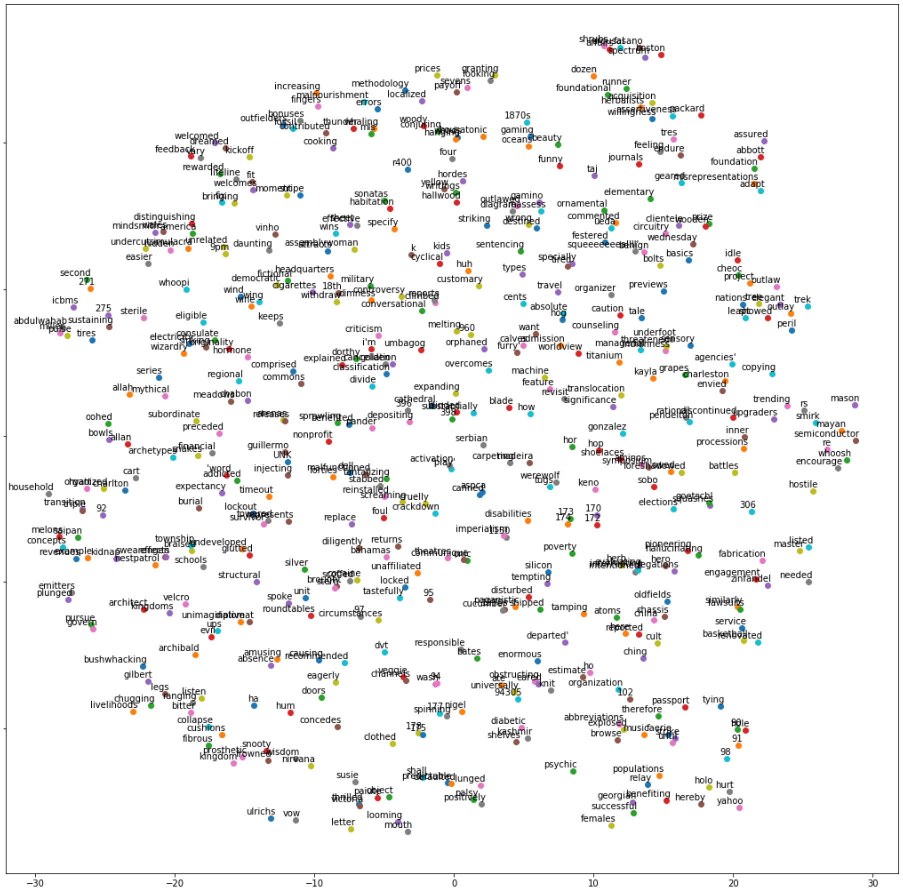

Figure 4.5: t-SNE visualization of word embeddings learned from EmoBank corpus. Only the first 500 words from the vocabulary is plotted.

context reasonably. From this difference, we were able to discover that EmoBank corpus alone was not enough for learning the general context, although its emotion labels are critical in better predicting fine-grained sentiments than the use of polarity only labels.

Chapter 5

Discussions and Future Work

In this chapter, we discuss possible methods for resolving the generalization problem, modeling both the writers and the readers perspectives simultaneously on a given text, and applications of the *emotion embeddings* whether the embeddings are generalized by the methods discussed or trained for specific tasks on the specific dataset.

5.1 Contextual Generalization

It was observed in our experiment that *emotion embedding* had only limited power in representing contexts in general. Specifically, it learned representations that only works for the EmoBank corpus. We suggest two different approaches that may improve the context generalization.

5.1.1 Semi-supervised Learning

Semi-supervised learning may help generalize the embeddings. Semi-supervised learning framework has been known to improve accuracy of a supervised NLP system by integrating unsupervised word representations as additional features [35]. The problem of learning generalized *emotion embedding* is perfectly suited for semi-supervised learning because it involves small set of labeled data and large set of unlabeled data. The EmoBank corpus can be served as the small labeled dataset while the unlabeled data can be acquired by using different large datasets such as text8, or scraping the web.

Semi-supervised approaches for sentiment classification in text has been gaining popularity recently [17, 38, 37] especially with the rise of popularity in generative models. Generative models in NLP utilizes a type of autoencoder that encodes the meaning of a text, whether it is a sentence or a document, then tries to reconstruct the input by decoding the encoded representations. Mathematically, the training of such autoencoders minimizes $L(x, g(f(x)))$, where the general framework is described as in Figure 5.1 [10]. Intuitively, this means that if the system can reproduce or generate the same sentence, it understands the concepts behind the sentence where the concepts are the encoded representation of the text.

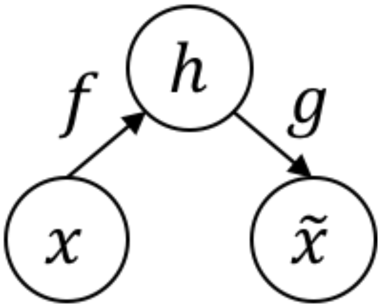

Figure 5.1: The general autoencoder framework, where x is an input, f is an encoding function, g is a decoding function, and $\tilde{x}$ is the reconstructed input obtained by $g(f(x))$. L is a loss function for guiding $g(f(x))$ towards x such as the mean squared error [10].

Semi-supervised learning methods such as autoencoders may help improve the contextual understanding, so the *emotion embedding* better represents the context as well as the emotion in the context.

5.1.2 Labeled Data Mining

Gathering more labeled data can be an easy solution to generalize the contextual representation of *emotion embedding*. The data acquisition process may be performed in two ways: by surveying a large population, or automatically scraping the data although the label may be noisy. As large amount of data is required, the first method may not be practical as it is expensive and time consuming to survey a large population and label the responses manually.

The latter method that utilizes new features in social media platforms seems more practical. In a social network platform like Twitter, the emoticons may be used as a proxy to represent the emotions of a posting where the text in the posting becomes the data and the emoticon the label for the data entry. Another popular social media platform Facebook recently added a feature where people can express their feelings about a post beyond the *like* button; "*love*", "*haha*", "*wow*", "*sad*", and "*angry*" buttons are added. These can be mapped to discrete model of emotions, which may be further translated to the dimensional model of emotion such as EPA or VAD format, using the subset of EmoBank where the text data is labeled in both a discrete model and a dimensional model. With the help of these social media platforms, obtaining a large labeled data may not be as a difficult task as it is perceived.

Note that Tang et al. [34] used this approach in gathering their sentiment labeled data. However, they did not publish the dataset, and reported that some of the data they used have been deleted. Publishing the gathered full emotion data may gain some momentum in using the emotion in sentiment analysis.

5.2 Bi-perspective Modeling

Emotions emerge from an interaction between interactants [29], but the emotions felt by the interactants may be different. In fact, the annotators for EmoBank could not obtain a convergent label on a text from two different perspectives; writer's, and reader's. We used the reader's emotion as the label for given text data as it expresses richer emotions [5].

The two perspectives may be modeled together with some modification of the language model. Instead of having one hidden space where the language model learns the context and the emotion, it may have a shared weights and biases on the context, but the emotions are learned in two independent hidden spaces; one representing the writer's emotion and the other representing the reader's emotion. For example, Figure 3.1, would have 3 more classifiers for the VAD labels of the writer. It would be interesting to see the impact of the bi-perspective modeling as it may discover how the readers and writers perceive a given corpus differently, or generalize better in the sentiment predictability due to joint training [6].

5.3 Applications

The *emotion embedding* may be used in multiple domains where an effective communication between an AI and a human agent is important. In this section, we look into possible application of *emotion embeddings* in chatbots, and robotics.

5.3.1 Chatbots

Kingma et al. [17] argues that semi-supervised learning is of great practical interest in fields such as genomics, natural language parsing, or speech analysis because the unlabeled data is abundant, but gathering the labeled data is expensive and time consuming. They proposed a semi-supervised variational autoencoder, where the encoding function is a variational approximation of the true encoding function of the distribution of the hidden variables. In variational autoencoder (VAE) framework, the variables are assumed to follow probability distributions. The advantage of the probability assumption of the hidden variable, h is that it is continuous, so one can walk through the hidden space which yields sentences that gradually transforms to another by changing nouns, verbs, or structures. These examples are shown by Bowman et al. [3], where they observed the local consistent transition of one sentence to another as shown in Table 5.1.

Table 5.1: An example of a smooth transition between two points in the hidden space [3].

“i want to talk to you . ”
“i want to be with you . ”
“I do n’t want to be with you . ”
i do n’t want to be with you .
she did n’t want to be with you .

With emotion embeddings, these sentence generating frameworks can be used in a chatbot to generate coherent sentences that may empathize with the interactants. These chatbots may be used to help diagnose depression, or accompany elderly people during the day when their family are either at school or work.

5.3.2 Robotics

Applications in robotics would be similar to that in chatbots. The major difference is that robots can give more diverse feedback to human interactants such as speech, facial expression, or physical interaction. An intelligent robotic system may be able to comfort people suffering depressions by providing a warm tea, or hugging the patients. A robot that can empathize and communicate with people reasonably may be great at accompanying elderly people as they can rely on the physical presence of the robot. Robots may learn to detect the emotion of people from additional sources such as facial expressions, or speech which may help constructing a general purpose *emotion embedding* that can be used in multiple domains beyond the textual settings, and provide more fine-tuned responses.

Chapter 6

Conclusion

Our findings suggest that the use of labels in emotion space indeed provides more useful information for an AI system to predict the sentiment of a given text. This was seen in the first experiment where we compared the prediction accuracies on two different models, one provided with polarity only labels, and the other with the full three-dimension VAD labels. This finding also means that the sentiment being perceived by people is affected by a combination of independent variables in emotion, although the valence dimension is the main contributor. Therefore, the use of VAD vectors as a precursor for understanding how people feels allows an AI agent to understand the interactant's emotion in detail.

However, the embeddings learned from the first experiment could not be used as a general word embedding as the pretrained Word2Vec [24]. This is because the dataset was too small to learn enough contexts, even though the embeddings dissolved the emotion information properly during the training. This was shown in the second experiment where we tested the trained models and the corresponding embeddings, with a different dataset. When the *emotion model* was trained on the first dataset, EmoBank, it was able to improve the predictions compared to the *sentiment model*. The test on the SST corpus, however, showed that both models were practically yielding random guesses.

Two hypotheses were constructed to discover the reason for failed generalization. The first hypothesis was the power of the language model; perhaps the model is not good at capturing the context. However, we soon rejected this hypothesis because the results reported by the baseline work of Tang et al. [34] showed that they used the model successfully across multiple datasets. The second hypothesis is the size of the dataset, EmoBank. This idea was spurred by the fact that the baseline research collected a large set of data by scraping Twitter [34]. This idea was tested on a separate context learning framework, the skip-gram model in Word2Vec, which has been proved to work well across various domains in NLP, and is a popular choice for generic word embedding [24, 15]. In comparing the learned embeddings from EmoBank and text8, we visualized the embeddings on two-dimensional space via t-SNE. The visualization showed that EmoBank corpus was not large enough to learn general context of a language. This was discovered by observing the embedding clusters; embeddings learned from text8 reasonably clustered similar words together in the projected two-dimension space while embeddings learned from EmoBank clustered without much contextual relations to one another.

The results suggested that the supervised learning framework of using the VAD emotion model as labels improve the sentiment predictability of an AI system, but in our experiment, the embeddings were not generalized enough to be used as a plug-and-play embeddings like the pretrained Word2Vec embeddings. This was discovered by testing the learned embeddings on a different dataset, SST, than the trained dataset, EmoBank. We visualized the embeddings with t-SNE to observe the lack of generality of the *emotion embeddings*, especially in terms of context representations.

We also suggested a few different approaches that may fix the problem of contextual generalization. Semi-supervised learning was one suggestion that is suitable for improving context representations as the framework uses small set of labeled data and large set of unlabeled data. The EmoBank corpus can be used as the small labeled data in this framework while the large set of unlabeled corpus can be obtained via using different text corpus or scraping them from the web. The other suggestion was to gather more labeled data, mainly by extracting discrete emotions from emoticons in Twitter, and emotion buttons in Facebook, which can be served as labels to a given text data.

Possible applications of *emotion embeddings* include chatbots and robotics where an interaction between an AI agent and a human is important. It may be used in both areas for treating depression by empathizing, and accompanying lonely elderly people.

References

[1] *WLM '12: Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[2] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March 2003.

[3] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany, August 2016. Association for Computational Linguistics.

[4] Margaret M Bradley and Peter J Lang. Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1):49–59, 1994.

[5] Sven Buechel and Udo Hahn. Emobank: Studying the impact of annotation perspective and representation format on dimensional emotion analysis. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 578–585, Valencia, Spain, April 2017. Association for Computational Linguistics.

[6] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA, 2008. ACM.

[7] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, July 2011.

[8] Paul Ekman. An argument for basic emotions. *Cognition & emotion*, 6(3-4):169–200, 1992.

[9] Clare Francis and David R. Heisse. Mean affective ratings of 1,500 concepts by indiana university undergraduates in 2002-3 [computer file].

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

[11] Jesse Hoey and Tobias Schroeder. Bayesian affect control theory of self, 2015.

[12] Jesse Hoey, Tobias Schroeder, and Areej Alhothali. Bayesian affect control theory, 2013.

[13] Rie Johnson and Tong Zhang. Semi-supervised convolutional neural networks for text categorization via region embedding. In *Advances in neural information processing systems*, pages 919–927, 2015.

[14] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*, 2016.

[15] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.

[16] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. In *AAAI*, pages 2741–2749, 2016.

[17] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

[18] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[19] Moshe Looks, Marcello Herreshoff, DeLesley Hutchins, and Peter Norvig. Deep learning with dynamic computation graphs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

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

[21] H. P. Martinez, Y. Bengio, and G. N. Yannakakis. Learning deep physiological models of affect. *IEEE Computational Intelligence Magazine*, 8(2):20–33, May 2013.

[22] Tomas Mikolov, Armand Joulin, Sumit Chopra, Michael Mathieu, and Marc'Aurelio Ranzato. Learning longer memory in recurrent neural networks. *arXiv preprint arXiv:1412.7753*, 2014.

[23] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, page 3, 2010.

[24] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[25] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[26] Charles E Osgood. The nature and measurement of meaning. *Psychological bulletin*, 49(3):197, 1952.

[27] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*, pages 115–124, 2005.

[28] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[29] Dawn T Robinson, Lynn Smith-Lovin, and Allison K Wisecup. Affect control theory. In *Handbook of the sociology of emotions*, pages 179–202. Springer, 2006.

[30] Lorenzo Rosasco, Ernesto De Vito, Andrea Caponnetto, Michele Piana, and Alessandro Verri. Are loss functions all the same? *Neural Comput.*, 16(5):1063–1076, May 2004.

[31] Richard Socher, Christopher D Manning, and Andrew Y Ng. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9, 2010.

[32] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, Christopher Potts, et al. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642, 2013.

[33] Carlo Strapparava and Trento FBK-irst. Emotions and nlp: Future directions. In *WASSA@ NAACL-HLT*, page 180, 2016.

[34] D. Tang, F. Wei, B. Qin, N. Yang, T. Liu, and M. Zhou. Sentiment embeddings with applications to sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(2):496–509, Feb 2016.

[35] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[36] Amy Beth Warriner, Victor Kuperman, and Marc Brysbaert. Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior Research Methods*, 45(4):1191–1207, 2013.

[37] Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. Variational autoencoder for semi-supervised text classification. In *AAAI*, pages 3358–3364, 2017.

[38] Shuangfei Zhai and Zhongfei (Mark) Zhang. Semisupervised autoencoder for sentiment analysis. In *AAAI*, pages 1394–1400, 2016.