A Domain-based Feature Generation and Convolution Neural Network Approach for Extracting Adverse Drug Reactions from Social Media Posts

Faculty of Engineering and Technology
Master Program of Computing

Author:
Feras Odeh

Supervisor:
Dr. Adel Taweel

This Thesis was submitted in partial fulfillment of the requirements for the Master's Degree in Computing from the Faculty of Graduate Studies at Birzeit University, Palestine

February 22, 2018

BIRZEIT UNIVERSITY

A Domain-based Feature Generation and Convolution Neural Network Approach for Extracting Adverse Drug Reactions from Social Media Posts

By Feras Odeh

Approved by the thesis committee

Dr. Adel Taweel, Birzeit University

Dr. Adnan Yahya, Birzeit University

Dr. Majdi Mafarja, Birzeit University

Dr. Radi Jarrar, Birzeit University

Date Approved:

February 27, 2018

Contents

Abstract

The recent popularity of the social media networks including forums, blogs, and micro-blogging networks changed the way patients share their health experiences and treatment options. Such forums offer valuable, unsolicited, uncensored information on drug safety and side effects directly from patients. However, it is very challenging to extract useful information from such forums due to several factors such as grammatical and spelling errors, colloquial language, and post length limitation. Furthermore, due to the sensitivity of the domain for adverse drug reactions (ADR) detection, it is more critical to identify correct ADRs (i.e., achieve higher classification precision) than identifying non-precise ones.

The aims of this thesis are: (i) to develop a new approach for ADR classification in twitter posts called Semantic Vector(SemVec); (ii) to explore natural language processing (NLP) approaches for generating domain features from text, and utilizing them for ADRs detection; and (iii) to improve convolution neural network (CNN) ADR classification precision by incorporating domain features.

This thesis proposes a dynamic and pluggable model, named SemVec, for representing words as a vector of both domain and morphological features. Based on the problem domain, domain features can be added or removed to generate an enriched word representation with domain knowledge. SemVec represents each post as a matrix of word vectors, which is fed into CNN. SemVec is scalable, can be applied to other domains by employing relevant natural language processing methods and domain lexicons. The proposed method was evaluated on Twitter (ADR) dataset. Results show that SemVec improves the precision of ADR detection by 13.43% over other state-of-the-art deep learning methods with a comparable recall score.

List of Figures

List of Tables

Chapter 1

Introduction

This chapter introduces the thesis. It describes the problem statement, motivations, goal, objectives, and organization of the thesis.

1.1 Introduction

In the United States, the fourth leading cause of death is adverse drug reactions(ADRs) ahead of diabetes, HIV, and car accidents. Each year, many injuries caused by ADRs[37]. Although new drugs should pass clinical tests in order to be marketed, many ADRs are discovered in post-approval phase as they happen to certain groups of people in certain conditions and they may take a long time to expose. To record ADRs, the US food and drug administration (FDA) has developed a database for collecting ADRs which monitors drugs safety during drug post-approval phase. Although Spontaneous ADR reporting systems can capture a set of drug ADRs, it has a set of recognized limitation, such as difficulty to determine ADR-drug relationship and it only captures a small fraction of ADRs.

The recent development of the social media networks including forums, blogs, and microblogging networks change the way we gather diseases and treatment options. It is now much easier for a patient to share his health experiences with others. In a recent survey, The Pew Research center found that 59% of all adult Internet users have used the Internet for seeking information about any of 15 health topics such as disease or treatment. Furthermore, 23% of social network users have followed their friends' personal health experiences[2]. Such social media posts offer valuable, unsolicited, uncensored information on drug safety and side effects directly from patients. So, several researchers have proposed different methods for extracting ADRs from social media [66, 41, 28, 8, 81, 33, 54, 62].

ADRs extraction research can be classified into three categories:1) text mining based methods, which utilize text mining techniques for extracting ADRs such as [77, 31, 43], 2) machine learning based methods, which use machine learning algorithms for extracting ADRs such as [49, 73, 62, 54, 33, 81], 3) deep learning based methods which utilize deep learning methods for extracting ADRs such as [28, 41, 8]. These works show that convolution neural network(CNN) can improve ADRs extraction performance. Also, it shows that domain features can be also effective in extracting ADRs, but will the combination of domain features and CNN improve ADRs classification performance?

This thesis tackles this question by proposing a new approach for ADRs detection in Twitter posts that combines domain features and CNN. In this approach, each word in a tweet is represented as a row vector of discrete features. These features include domain-specific and morphological features, employing those with domain semantic relevance and common meaningful use. This approach is dynamic, which means domain features are changeable, i.e. can be added or removed based on the problem domain. After representing each word as a row vector, each tweet is represented as a matrix of word vectors that is fed into CNN. In addition, we propose a new CNN network architecture with a single convolution layer, a single max pooling layer, two fully connected layers and finally a sigmoid activation function.

In the evaluation, our work is evaluated on Twitter ADR dataset from Arizona State University. Also, our work is compared with both: a method that uses features engineering and methods that use deep learning and CNN for ADR extraction. The results show that our approach better ADR classification precision with comparable recall compared to other methods.

1.2 Problem Statement

Adverse Event Reporting System (AERS) like systems are a major source of ADR information. These systems monitor drugs safety during the post-approval phase. Moreover, it relies on collecting ADRs from pharmaceutical companies, health-care professionals, and consumers spontaneous reports [24].

In some incidents, ADRs spontaneous reporting systems failed in protecting the public from drugs' risks as effectively as it might. The 2004 withdrawal of Rofecoxib (Vioxx) because of an increased risk of serious cardiovascular side effects was a key incident in raising such concerns[46].

Lately, a new type of social networks and forums emerged. Twitter, Ask a Patient, DailyStrength, PatientsLikeMe, and Yahoo Health and Wellness are such type of forums which provide patients the ability to report drug side-effects and share their health experiences and medications. Although the accuracy of this information may be questionable, such forums offer valuable information on drug safety and side effects directly from patients. However, it is challenging to extract ADRs from such forums due to several factors such as spelling and grammatical errors, colloquial language, distinguish real experience from news post, etc. This process requires deep statistical and linguistic methods to filter and extract useful findings [25].

This thesis aims to develop a new approach for ADR classification in twitter posts called Semantic Vector(SemVec). The approach uses domain-specific semantic features to improve convolution neural network(CNN) precision. SemVec can utilize different types of semantic features such as polarity, sentiment, subjectivity and domain-specific features. SemVec experiment results show that domain features have a clear impact on improving system classification precision and can achieve high scores even in unbalanced datasets.

1.3 Motivation

1.3.1 Why Social Media Posts ADR Detection is important

Spontaneous ADR reporting systems have a set of recognized limitations, such as it only captures a small fraction of adverse events [46]. It is also difficult to determine if an ADR is related to a particular drug, as the reports may contain other factors such as other drug exposures[60]. To overcome these limitations, social media and health forum posts can be used to extract drug side-effects from patient posts. Such forums offer valuable, unsolicited, uncensored information on drug safety and side effects directly from patients.

1.3.2 Challenges of ADR Detection in Tweets

Twitter is one of the largest social media networks that has 330 millions monthly active users and 500 millions of daily tweets [1]. Unlike other social media sites, such as Facebook, Twitter allow developers to access tweets of all users or filter them by subject-for example drug name- which can be very useful for research. However, extracting ADRs from Twitter based data sources present various NLP challenges. Leaman et al. [38] have shown that systems exposed to social media posts frequently under perform others that do not expose to social media posts due to misspellings, use of idiomatic, ambiguous and sarcastic expressions and presence of novel/creative phrases. Twitter posts, on the other hand, presents additional challenges related to tweet length constraints. These challenges are as follows [66]:

- Users express their own views about specific medications in their posts. It is hard to identify such posts unless the user explicitly states that this is his own view or someone else experience.
- Challenges related to the presence of normal misspellings particularly in drug names (e.g., 'Seroquil', 'Numbb', 'Effexer', 'Bfore'). Sometimes these errors cannot be detected which may lead to incorrect posts classification.
- Twitter post length constraint and disconnected sentences. These limitations may lead to posts missing some critical information which may cause inaccurate classification results.
- Data imbalance and noise. In this thesis, the used datasets contain far more negative ADR instances than positive ADR instances. In certain cases, classifier performance could be significantly affected by Data imbalance [29].

[1]https://www.omnicoreagency.com/twitter-statistics/ accessed 21/2/2018

- Users use ambiguous/non-standard terms to express ADRs (e.g.: 'look like a zombie', 'ton of weight'). These non-standard terms are hard to identify and requires special dictionaries.
- ADR detection is more critical than other classification tasks. In such critical tasks, precision is more important than recall.
- Some tweets are news or re-tweet from someone else post. In these cases, the classifier should verify that this post describes a user experience and not a re-tweet or news.

1.3.3 Why Domain Specific Features

Feature engineering is the process of creating or extracting features using domain knowledge to generate a representation that enables classification [19]. For example, in the text classification domain, all distinct terms present in a text corpus can be considered as features. Feature engineering process is often guided by domain knowledge [19]. In the medical domain, for example, terms related to diseases such as ADR, symptoms, and medications are manually defined by experts. The use of domain-specific features will generate a vector representation that includes the domain semantic which will guide the classifier learning process by focusing on only important domain-related features.

1.3.4 Why Convolution Neural Networks

Convolution Neural Networks (CNN)s are widely used in image classification and computer vision. CNN is composed of a set of layers with filters that convolve over to local features [40]. CNNs are directly responsible for the major breakthroughs in image classification. Today CNNs are the core of several computer vision systems such as self-driving cars and automated photo tagging.

Recently, Researchers have shown that CNN can be applied effectively to different NLP tasks and can achieve excellent results. For example Yih et al. have applied CNN in semantic parsing [78], Shen et al. have applied CNN in search query retrieval [68] and other traditional NLP tasks [17].Moreover, CNN is applied to text classification, relation extraction, and sentiment analysis. Kim[35] have done a series of experiments with CNNs for sentence-level classification tasks. The model was trained using pre-trained word vectors.

1.4 Research Questions

CNN has achieved promising results in various NLP task. Therefore, using CNN in the task of ADRs extraction or identification from tweets or free social text may improve ADRs extraction performance compared to using other classification algorithms like support vector machine(SVM). This research is trying to answer the following questions:

- What is the effect of using domain-specific and morphological features in CNN on model performance(including precision, recall, accuracy)?
- Does domain-specific features require a special CNN network model?
- What is the importance of each domain-specific feature?
- How to choose domain-specific features?

1.5 Research Methodology

In this study, we follow a scientific approach by conducting a research to observe the influence of domain specific features on ADR detection precision in Twitter posts. This thesis aims to construct a concise vector representation of words and use it as input to a convolution neural network. **SemVec** uses a set of domain specific and non domain specific features to represent each word in the post. In contrast to word2vec [47] which represents each word with a long vector from 50 to 300 embedding dimension, SemVec can represent a word with a short vector of 12 dimensions and also achieves high performance scores. Our research objectives are:

- To conduct a literature review to identify research gaps and identify research limitations.
- To collect relevant data sources that can be used in our experiment.
- To collect tweets that are medically related to use them in building word embedding model using pre-training.
- Developing new word representation layer for CNN using semantic features. This new word representation consists of a set of domain specific features. The used features are from different categories including polarity features, domain-specific features and syntactic features.
- Developing new CNN model for text classification.
- In order to examine the performance of the proposed approach, we compare classification metrics of the positive class (instances contain ADRs) with Sarker multi-corpus training method [66], Huynh et al. CNN method [28], Huynh et al. RCNN method [28],Huynh et al. CRNN method [28],Huynh et al. CNNA method [28],Term-matching based on an ADR lexicon(TM) [58] implemented by [28],Maximum-Entropy classifier with n-grams and TFIDF weightings(ME-TFIDF) [81], Maximum-Entropy classifier with mean word embeddings(ME-WE) [81] implemented by [28].
- To analyze experiment results in order to compare it with other approaches and find strength and weakness in our approach.

1.6 Hypothesis

We hypothesized that using domain-specific features and CNN in classifying Twitter posts will improve classification precision based on our evaluation methodology.

1.7 Thesis Objective

This thesis aims are:

- To develop a new approach for ADR classification in twitter posts.
- To explore natural language processing (NLP) approaches for generating domain features from text, and utilizing them for ADRs detection.
- To improve CNN ADR classification performance by incorporating domain and morphological features.

1.8 Thesis Organization

This thesis is structured as follows:

- **Chapter 2: Background.** Provides a general background of the concepts needed to understand the rest of the thesis.
- **Chapter 3: Literature review.** Reviews related works in tweets ADR classification.
- **Chapter 4: Proposed method.** Proposes a new method for tweets classification based on CNN and feature generation which improves tweets ADR classification accuracy.
- **Chapter 5: Evaluation and results.** Examines the performance of our approach, and compares it with the state of the art methods.
- **Chapter 6: Conclusion and Future Work.** Represents conclusions of thesis and future work.

Chapter 2

Background

This chapter provides a general background of the concepts needed to understand the rest of the thesis. It covers basic concepts of deep learning and more specifically CNN.

2.1 Machine Learning

Recently, machine learning has been widely used in multiple fields, including computer science, medicine, sports, etc.. So many applications and services have been using machine learning technology to solve problems. For example, email services use machine learning to filter spam messages, classify emails into important or not and recommend ads. Another machine learning technology that is widely used in social media sites is face recognition. Face recognition technology is capable of identifying persons in a given digital photograph. Today, Facebook uses face recognition to automatically suggest tags for friends in images [20].

Machine learning can be defined as ”a mechanism for pattern search and building intelligence into a machine to be able to learn, implying that it will be able to do better in the future from its own experience” [20]. So machine learning programs utilize example data or past experience to optimize model performance. In machine learning, the model is defined based on some parameters, then this computer program is executed to optimize model parameters using the training data or past experience(the learning process). Machine learning models can be classified into predictive, descriptive or both. Predictive models make future predictions while descriptive ones gain knowledge from data [9]. As shown in figure 2.1, machine learning algorithms can be classified into five subfields. The following subsections describe each subfield.

2.1.1 Supervised learning

Supervised learning is the most common form of machine learning. In supervised learning, labeled training data is used. The algorithm produces a model from training data that can be used to predict unseen data labels [39]. During training, the goal of machine learning algorithms is to minimize the error between output scores and actual scores. To

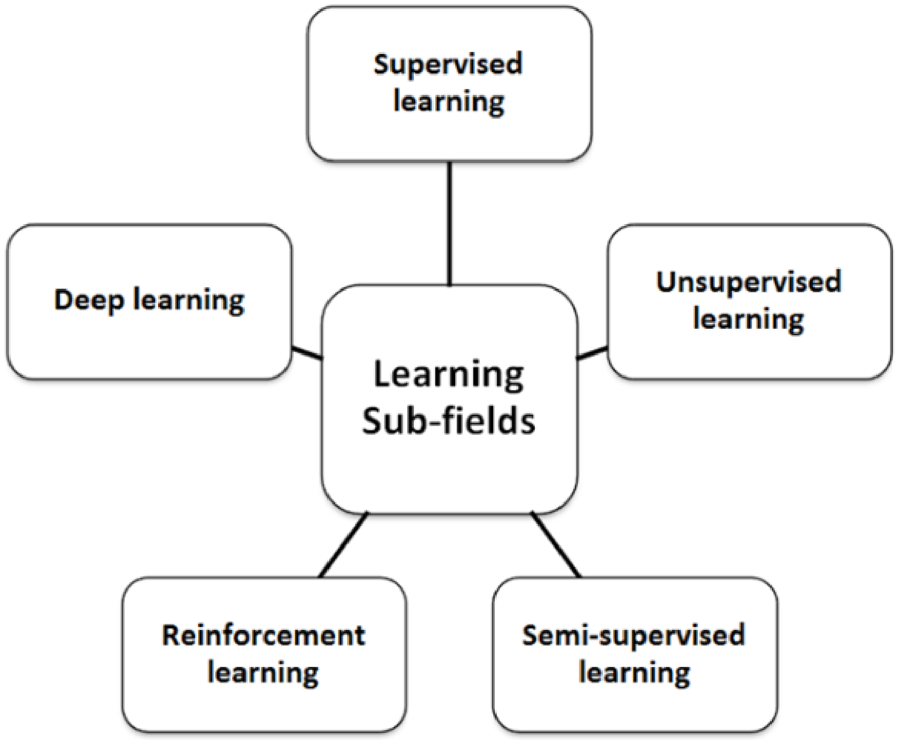


Figure 2.1: Machine learning subfields [20]

calculate error, an objective function is used to measure the distance between predicted scores and actual scores. In order to minimize error, neural nets adjust its internal parameters (also known as weights). Weights are real numbers that define the function which maps inputs to outputs. Typically, deep learning systems contain millions of weights along with millions of labeled instances [39].

To effectively optimize the weight vector, a gradient vector is computed. Using a gradient vector, the learning algorithm can detect decreases or increases in error amount when changing weights which helps in optimizing weight vector values. The most used gradient optimization algorithm is called stochastic gradient descent (SGD) [39].

2.1.1.1 Classification

Classification is the process of classifying unseen data to a set of predefined classes. A classification algorithm uses a set of labeled training data to build a classification model. Then this classification model is used to predict unseen instances classes [44]. Several applications use classification to solve diverse problems including mail classification into spam or non-spam and cells classification into malignant or benign [70]. Table 2.1 shows an example dataset used for binary classifying customers who will buy computer and who will not. The attribute set includes properties of each customer such as his name, age, income and student or not. These attribute set contains both discrete and continuous features. However, in classification problems the class label must be a discrete attribute [70].

Name	Age	Income	Student	Buys computer
Adam	30	high	no	no
Ahmad	35	high	no	yes
Ali	42	medium	no	yes
Khaled	38	low	yes	yes
Mohamad	36	low	yes	yes
Radi	30	medium	no	no
Yousef	22	low	yes	yes
John	42	medium	yes	yes
Khalil	25	medium	yes	yes
Ahmad	33	medium	no	yes
Feras	33	high	yes	yes
Fadi	42	medium	no	no

Table 2.1: Example of datasets

2.1.2 Unsupervised learning

In some machine learning problems, we have input data but we do not have specific output variables(examples are unlabeled). The main goal for unsupervised learning is to find hidden patterns and modeling underlying structure in the data. In such problems, there are no correct answers and there is no teacher. Hence, the accuracy of the resulting structure cannot be evaluated[20, 14].

2.1.3 Deep learning

Deep learning is a subfield of machine learning that unifies machine learning with AI. Deep learning works on large amounts of data which can be considered as an advancement to artificial neural networks [20].In deep learning, The core building block of a standard neural network is called neurons. Neurons can be seen as processors producing a sequence of activation. Neurons can be activated in different ways such as weighted connections from previously active neurons or through sensing the environment. Typically, connected neurons have weights that neural network adjusts during training to optimize learning using Backpropagation algorithm as described in section 2.2. Based on problem complexity, the desired behavior may require a series of deep layers and long stages of training [67].

Models with few layers(Shallow models) have been around since 1960. Due to its computation costs, Deep neural networks with multiple layers were difficult to implement in practice until early 2000 [67]. The recent drop in hardware prices, increase in processing power, and the advancement of machine learning algorithms are the main reasons for the booming in deep learning [22].

Deep networks add more layers and more neurons to each layer to represent complex functionality. In classification problems, deep learning algorithms amplify input object discriminative features in higher layers and dismiss irrelevant features. For example, in image classification tasks, each layer in a deep neural network detects a specific feature. The first layer detects image edges, motifs are detected in the second layer. Furthermore, motifs are assembled into larger parts in the third layer which matches familiar object parts. Finally, subsequent layers combine parts from previous layers which allow the network to detect objects [39].

The key feature of deep learning is the ability of each layer to detect features automatically without human intervention. Deep learning has achieved state of the art results in image recognition, speech recognition as well as other machine learning subfields [39]. Interestingly, deep learning has achieved promising results in various NLP tasks such as sentiment analysis and machine translation [39].

2.1.4 Semi-supervised learning

Semi-supervised learning is a subfield of machine learning that uses both large amount of unlabeled data and a small amount of labeled data to create a better model. Semi-supervised learning can reduce the cost associated with labeling a full training set, as labeled data often requires a skilled human agent(e.g. to label tweet for ADR and no-ADR). Instead, it uses unlabeled data which is relatively inexpensive to acquire [20].

2.1.5 Reinforcement learning

Reinforcement learning is a subfield of machine learning where a software agent tries to solve a problem by maximizing rewards for its actions and minimizing penalties. After a set of runs, the agent should learn the best sequence of actions that maximize the reward [20].

2.2 Backpropagation

Backpropagation, short for "backward propagation of errors", is an algorithm to adjust neural network weights to minimize error by calculating each neuron error contribution after processing a batch of data networks using gradient descent. The calculation of the gradient proceeds backward through the network, as the final layer gradient of weights is being calculated first then proceed through layers until the first layer gradient is calculated at last. The backward flow of the error allows more efficient computation of the gradient compared to calculating each layer gradient separately [39]. Because backpropagation is considered an efficient algorithm, it is widely used to train deep neural networks for image recognition, speech recognition, and text classification [39].

2.3 Word Embedding

The first building block for any NLP task is the word or sentence representation. Recently, a set of algorithms were introduced that provide representations of words in a vector space model by grouping similar words. One of those popular algorithms is word2vec introduced by Mikolov et al.[47]. Word2vec is a shallow, two-layer neural network pre-trained on large amounts of unstructured text data to produce a high-quality vector, typically of several hundred dimensions, representations of words. One interesting feature of word2vec it that the produced representation of word encodes many linguistic regularities and patterns. Surprisingly, using word2vec we can represent many of these relations as equations. For example, the result of this equation:

$$vec(\backslash Madrid") - vec(\backslash Spain") + vec(\backslash France")$$

is closer to vec("Paris") than to any other word vector [47]. Word2vec is widely used in deep learning text classification and relation extraction tasks as shown in chapter 3.

2.4 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNN) is a type of neural networks that is able to process complex forms of data such as multiple arrays. Images and audio; signals and sequences; and video are all samples of data represented in the form of multiple arrays [39].

Figure 2.2 shows the general CNN architecture for text classification. CNN typically is composed of series of stages. The first stages are usually composed of convolution and pooling layers which numbers may vary based on the method. The later stages consist of a set of fully connected layers [39, 22].

Inside each convolutional layer, units are organized in feature maps which are connected to previous layers through a set of filters(weights). The result of this convolution is then passed through a rectified linear unit(ReLU) layer(non-linearity) -other activation functions may be used such as softmax-. As shown in Figure 2.2, each sentence is represented as a matrix with word representations-such as word2vec- as rows. Each sentence matrix is then fed into a convolution layer which applies a filter to it and generates a feature map. Feature maps then fed into max pooling layer which extracts the most useful features and then fed it into fully connected layers which classify this sentence.

Recently, several researchers applied CNNs to different NLP tasks including text classification, relation extraction, and sentiment analysis. Kim[35] has done a series of experiments with CNNs for sentence-level classification tasks. The model was trained on top of pre-trained word vectors.

2.4.1 Types of layers

CNN is composed of a set of different layers in a hierarchy. Generally, it contains a convolution layer followed by pooling layers. The following subsections describe those

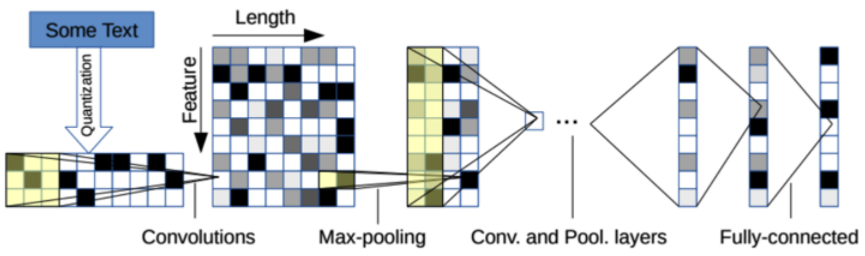


Figure 2.2: CNN architecture for text classification [79]

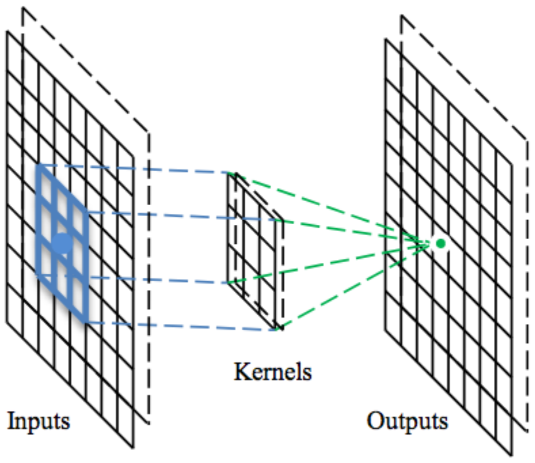


Figure 2.3: convolution layer(Kernels=filters) [22]

layer and how they are used in text classification.

2.4.2 Convolution Layer

The convolution layer is the core building block of a CNN. As shown in figure 2.3, it applies a convolution operation to the input and passing the output of this operation to the next layer. In Kim model [35] the convolution layer uses three region sizes: 3,5 and 7 words each one of them has 2 filters. Each filter has a width of the word vector dimension. So this will produce 3-gram if the filter size is 3 or 5-gram if the filter size is 5, etc... Filters apply convolution operation on the sentence matrix in order to generate (variable-length) feature maps. A feature c_i is generated from a window of words $X_{i:i+h-1}$ by

$$c_i = f(w \cdot x_{i:i+h-1} + b)$$

In this equation, f is rectify linear unit (ReLu) function and $b \in \mathbb{R}$ is bias term.

2.4.3 Pooling Layer

This layer is usually used to reduce both feature maps and network parameters dimensions. The most commonly used strategies are max pooling and average pooling. After using a max pooling layer of 2*2 size and stride of 2, feature maps of size 16*16 is reduced to 8*8 [22]. Figure 2.4 shows the operation of a max pooling layer.

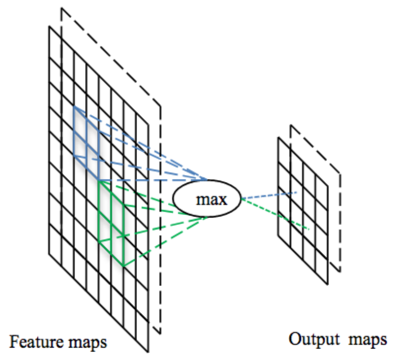


Figure 2.4: max pooling layer [22]

2.4.4 Fully-connected layers

Fully-connected layers perform the same functionality of tradition neural network. It follows the last convolution and pooling layer and allows the network to feed forward a vector into another vector of different length or into different categories [39].

2.4.5 Softmax Layer

This is usually the final layer in a CNN architecture, a softmax receives the feature vector from the previous layer and uses it to perform binary classification. The softmax equation is:

$$L_i = -log\left(\frac{e^{y_i}}{\sum_{k=1}^{k} e^{y_k}}\right)$$

In this equation, y_i represents the actual correct class.

Chapter 3

Literature Review

This chapter reviews the most commonly used text mining techniques in social media ADR extraction. The work done on ADRs can be classified into three main categories as follows:

3.1 Text Mining Based Methods

Due to the public nature of Twitter posts, multiple researchers have conducted studies on how to extract ADEs from tweets. Jiang and Zheng[31] have developed a computational approach for extracting ADRs from Twitter data. In order to automatically extract ADRs from twitter posts, the authors have developed a data processing pipeline to extract ADRs from Twitter posts. The data pipeline consists of three main steps: collecting and pre-processing tweets, using machine learning to classify personal experience tweets, Detecting ADRs from tweet text using the MetaMap software. In order to classify tweets which contains personal experience, the authors use a machine learning classifier. Natural language Toolkit(NLTK)[11] was used to extract sentiment and pronouns features which were used later in machine learning methods. The authors tested three different ML methods: Naive Bayes, Support Vector Machine, and Maximum Entropy which demonstrated the best performance. Although Twitter can be considered as a valuable ADRs source, The 140 characters limitation on user tweets limits the amount of information that can be posted and afterward extracted from each tweet.

Leaman et al.[38] have proposed a more sophisticated approach for mining the relationships between ADRs reported by patients and drugs. The proposed method used Java sentence breaker to split user comments into sentences and then tagged for part-of-speech using Hepple tagger[26]. After handling spelling errors, the authors measure string similarity to compare the individual tokens using Jaro-Winkler [76]. Finally, a normalized score was calculated by summing the similarities of the individual tokens for both original tokens and stemmed ones, and the higher of the two scores is the final score. To evaluate their system the authors manually annotate a set of user comments posted on DailyStrength [38].

A different approach for extracting ADRs from online patient forums was proposed

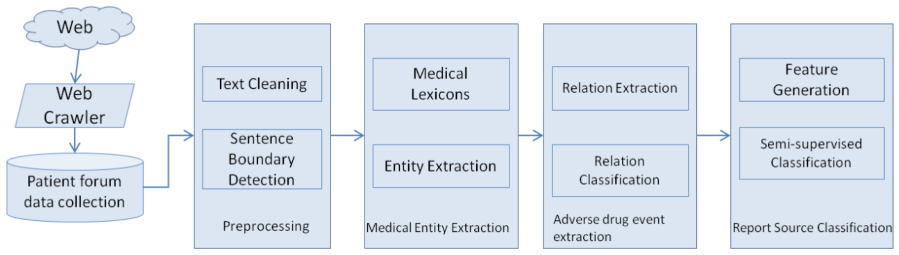

Web
Web Crawler
Patient forum data collection
Text Cleaning
Sentence Boundary Detection
Preprocessing
Medical Lexicons
Entity Extraction
Medical Entity Extraction
Relation Extraction
Relation Classification
Adverse drug event extraction
Feature Generation
Semi-supervised Classification
Report Source Classification

Figure 3.1: AZDrugMiner Framework [43]

by Liu and Chen[43]. The analytical framework (AZDrugMiner) is composed of a collection of machine learning methods to extract ADRs. As shown in Figure 3.1, AZDrugMiner contains different components. To collect the data a crawler was developed to download medical posts from patient forums. In data pre-processing, step AZDrugMiner cleans text and detect sentence boundary. To split each post into sentences, the authors used an open source OpenNLP tool[4]. After pre-processing, medical entity extraction step starts. In this step, Metamap[3] Java API maps post's text to UMLS[5].

In order to detect whether an ADR is associated with a drug in a sentence, an entity relation extraction method is needed. To help increase the precision of the extracted relation, a kernel-based machine learning method was used to extract ADRs from patient forums[42]. To reduce noise and duplication, the authors used a post source classification to filter whether an ADR reports is based on reporter experiences or not. The authors claim that no previous patient social media research addresses this issue and to solve it they develop a machine learning classifier to classify reports based on patient experience[42].

To evaluate AZDrugMiner, the authors used a manually annotated dataset which shows promising performance. To further improve AZDrugMiner system other kernel functions such as SubTree Kernel and Standard tree kernel could be used. Moreover, to improve relation detection performance multiple kernel function can be combined[42].

Sarker et al.[66] have proposed an approach for classifying Twitter posts that utilizes multi-corpus to improve classification performance. The authors have used three data sets two of them are annotated posts from social media while the third one contains annotated clinical report sentences. The proposed method generates a feature vector from a large set of semantic features (i.e: sentiment, polarity, and topic) from short text nuggets.

3.2 Machine Learning Based Methods

Nikfarjam et al.[49] have proposed a new machine learning based method for extracting ADRs from user's social media named ADRMine. The authors collected a set of drug related user posts from DailyStrength and Twitter sites. After data collection, user posts were annotated by two expert annotators. After data collection, an exhaustive list of

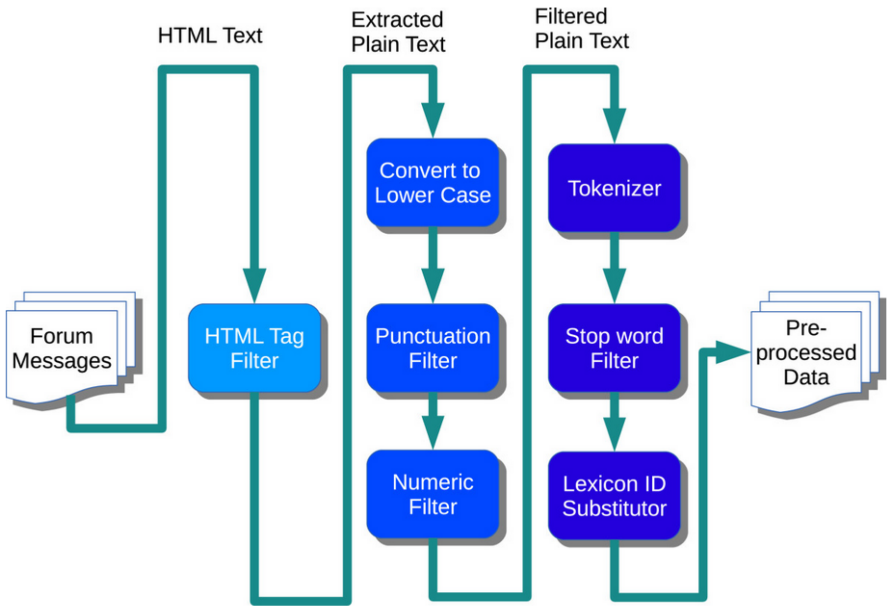


Figure 3.2: Text Processing Module [62]

ADRs and corresponding UMLS IDs were compiled. In order to extract ADRs from social media posts, CRFsuite[51] (a CRF[72] classifier implementation by Okazaki) was used. The CRF classifier was trained on a set of annotated ADR posts. By achieving an F-score of 0.82, the authors claim that ADRMine results outperform several baseline systems in extracting ADRs[49].

A most recent study conducted by Wang[73] extends Nikfarjam et al.[49] work by using different word representations(K-means clusters and Brown clusters) [16][71]. Word clusters were generated using a set of unlabeled tweets (some contains ADRs). Word clusters were constructed using Brown clustering tools [13] and Stanford GloVe tool [1] as described in[71]. Study results show that the proposed system significantly outperforms the other participating system in this study.

A different approach for extracting ADRs from posts in online healthcare forums was proposed by Sampathkumar et al.[62]. The proposed method consists of the following three modules:

- Information Retrieval Module[62]: in order to collect data, the authors built a crawler to extract data Medications and Steadyhealth websites.
- Text Processing Module[62]: in this module, a data processing pipeline was built from a collection of Natural Language Processing (NLP) tools as shown in Figure 3.2.
- Information Extraction Module[62]: used to identify entities and extract possible relationships between them. It consists of two sub-modules: Named Entity Recog-

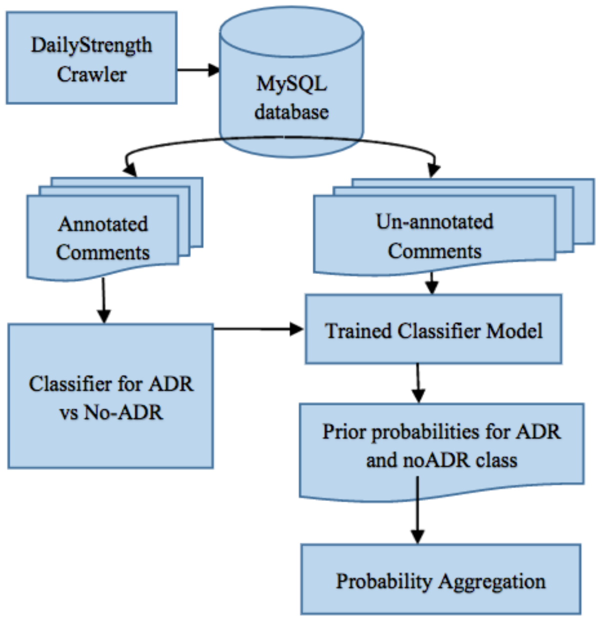


Figure 3.3: Two steps drug comments classification process[54]

nition (NER) module and Relationship Extraction (RE) sub-module. To extract a drug and ADR association the authors used Hidden Markov Model(HMM) which is a supervised machine learning method.

Patki et al.[54] proposed a novel probabilistic model for drug categorization based on user comments. As shown in figure 3.3, the proposed approach is composed of two steps: first, classify whether a comment is an ADR or not, and then check whether all drug comments shows a red flag or not. In the binary classification step, the authors used a manually annotated set of user comments to train classifiers. Two machine learning algorithms were used: Multinomial Naive Bayes (MNB) and Support Vector Machines (SVM). Finally, in order to classify drugs into a normal or black box, the authors developed a probabilistic model based on the hypothesis that a black box drug should have more ADRs than a normal one. Although the results of classifying single comments into ADRs and non-ADRs are promising, the overall classification of drugs is marginal.

Jonnagaddal et al.[33] have proposed a binary classifier using linear SVM to automatically classify ADR-assertive Twitter posts. The proposed method pre-processes each tweet to remove unnecessary characters such as hashtags, username, etc.. After pre-processing, the system extracts various features including syntactic, lexicon, polarity and topic modeling based features. Moreover, the system extracts a set of lexicon features generated using pattern matching.

Zhang and Nie[81] have proposed a new approach for classifying tweets which pro-

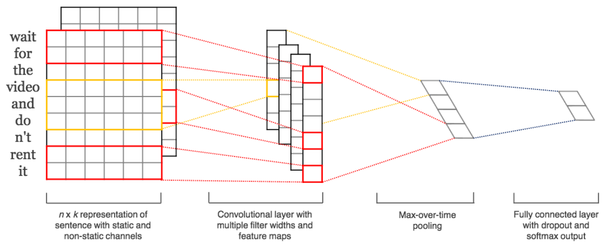

wait
for
the
video
and
do
n't
rent
it
n x k representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure 3.4: Kim's model architecture with two channels for an example sentence. [35]

duces a weighted average of four classifiers: concept-matching classier; a maximum entropy (ME) classier with TFIDF weighting scheme; an ME classier based with naive Bayes (NB) log-count ratios as feature values; and an ME classier with word embedding features. The final classifier is an ensemble of the previously mentioned classifiers and the overall score is defined as the weighted mean of output from these classifiers.

3.3 Deep Learning-Based Methods

Deep learning has recently shown interesting results in various NLP tasks. In this section, we show a review of work done in this area related to our problem.

3.3.1 Convolution Neural Networks (CNN)

Convolution Neural Networks (CNN)s are extensively used in computer vision and image classification. CNNs are directly responsible for the major breakthroughs in image classification.

Recently, several researchers applied CNNs to different NLP tasks including text classification, relation extraction, and sentiment analysis. Kim[35] has done a series of experiments with CNNs for sentence-level classification tasks. The model was trained on top of pre-trained word vectors.

Kim suggested model is shown in figure 3.4. In this model, each word is represented by k-dimensional word vector obtained from word2vec tool [1]. A sentence of length n words in this model is represented as a matrix $\mathbb{R}^{n*k}$-k stands for word representation dimension- of word vectors. In order to produce a new feature from sentence matrices, a convolution operation with multiple filter widths is applied to a window of h (h represents filter size) words to produce a new feature. For example, a feature c_i is generated from a window of words $X_{i:i+h-1}$ by

$$c_i = f(w \cdot x_{i:i+h-1} + b)$$

In this equation, f is a nonlinear such as the hyperbolic tangent and $b \in \mathbb{R}$ is a bias term. After applying this filter to each possible set of word window a feature map is

[1]https://github.com/dav/word2vec accessed 21/2/2018

produced[35]

$$c_i = [c_1, c_2, ..., c_{n-h+1}]$$

After getting the feature maps a max pooling over time is applied in order to capture the feature with the highest value - the most important feature - for each feature map. The final results of the authors work show that a simple CNN with one layer convolution performs very well. Moreover, the author found that unsupervised pre-trained word vectors are a corner block component in deep learning for NLP[35].

Johnson and Zhang[32] have proposed a CNN model without the need for pre-trained word vectors. The authors were able to directly apply CNN to high dimensional text data instead of low dimensional word vectors like word2vec or GloVe. In this paper, the authors have applied convolutions to one-hot vectors. Moreover, in order to reduce the parameters which the network has to learn, the authors have proposed a space-efficient input data representation (bag-of-words-like).

Dos Santos and Gatti[63] have proposed a new approach for sentiment analysis of Twitter messages using CNNs. In order to perform sentiment analysis, the proposed method exploits character to sentence-level information. The neural network takes a sentence as input. By passing this sentence word representation matrix through a sequence of layers the system is able to extract more complex features. Moreover, this system is able to extract features from both character-level and sentence level. Moreover, it consists of two convolutional layers to handle variable size words and sentences.

Sahu et al.[61] proposed a new approach for relation extraction using CNNs and feature generation. The proposed model takes a sentence as an input and outputs all possible relation types as probability vector. It is based on Kim[35] model architecture. The authors used generated features in the first layer in addition to word embedding and used multiple filters in all possible continuous n-grams in the sentence.

Huynh et al.[28] have proposed two new neural network models, Convolutional Recurrent Neural Network (CRNN) by concatenating recurrent neural network to convolution neural network and Convolutional Neural Network with Attention (CNNA) by adding attention weights into CNN.

Lee et al. [41] have proposed a semi-supervised CNN model for tweets classification into ADR and non-ADR. The proposed model uses several semi-supervised CNN models built from different types of unlabeled data. In the second phase, CNN is trained with annotated ADE data. The output layer uses a linear classifier that can classify if a tweet contains an ADR or not.

Sulieman et al. [69] have examined if using semantic features and word context enhances patient portal messages classification performance. The authors have constructed a different set of features such as a bag of phrases, a bag of words, word embeddings, and graph representation. The authors have trained a random forest and logistic regression classifiers, and CNN with softmax output. Researchers have found that using semantic features such as word2vec and graph representation improve classification accuracy of portal messages.

Akhtyamova and Alexandrov [8] have conducted a study which uses CNN with word2vec to classify ADRs in tweets. The authors have done unsupervised word embeddings learning from different datasets including GoogleNews, Wikipedia, and Diego lab. Results show that using CNN+GoolgeNews corpus improves ADR F-score.

To reduce training CNN computational costs, Prusa and Khoshgoftaar [57] have proposed a new character level representations of sentences. In this method, each character is given an integer value. Each character in the sentence is then replaced by the corresponding integer value. This sequence of integer values is then replaced by its equivalent binary representation which creates a vector of 0s and 1s.

3.4 Comparison

ADR detection research can be classified based on vector representation into sentence level representation and word level representation. Sarker et al. [66] proposed method generates a feature vector for each sentence from a large set of semantic features (i.e: sentiment, polarity, and topic) from short text nuggets, whereas our approach generates a vector of domain features for each word. On the other hand, word level approaches generate a representation for each word. Wang [73] proposed a method that uses a word level representation, Brown clusters, and k-means clusters, that are generated from unlabeled generic and drug-related tweets, while our approach uses word clusters as a feature in word representation. Other word level methods use CNN for ADR and text classification. For instance, Johnson and Zhang [32] have proposed a method which represents words as one-hot vectors. While Johnson and Zhang [32] used one-hot vectors on the word level, Prusa [57] represented each character in the sentence as a binary vector of 0s and 1s. However, our approach represents each word as a vector of domain features. Figure 3.5 shows a diagram of previously described methods.

One of the popular algorithms that are used with CNN for text classification tasks is word2vec introduced by Mikolov et al.[47]. Word2vec is a shallow, two-layer neural network pre-trained on large amounts of unstructured text data that produces high-quality vectors, typically of several hundred dimensions, representations of words. While word2vec utilizes a word's local context window, Pennington et al. [55] proposed a different model that combines the advantages of local context window and global matrix factorization methods. Both of these models fail to provide a good representation for rare words, which can be a domain word that can clearly affect performance. While our model utilizes lexicons and dictionaries to handle rare and domain words.

Many researchers have used word2vec and GloVe to extract ADRs from social media posts [28, 8]. While Kim [35] was the first to introduce a CNN model that uses word2vec and GloVe for short text classification. Huynh et al [28] have proposed three new neural network models CRNN, RCNN and CNNA with word2vec as input layer. Experiment results show that CNN architecture achieves the best results in ADR classification. Akhtyamova and Alexandrov [8] trained word2vec on GoogleNews, Wikipedia, and Diego lab. CNN achieved better results when using word2vec trained on GoogleNews. Compared to our approach, word2vec is unsupervised machine learning algorithm that requires a huge amount of unlabeled data to learn word representation. While our approach is supervised and so no unlabeled data is required. Further, our approach is

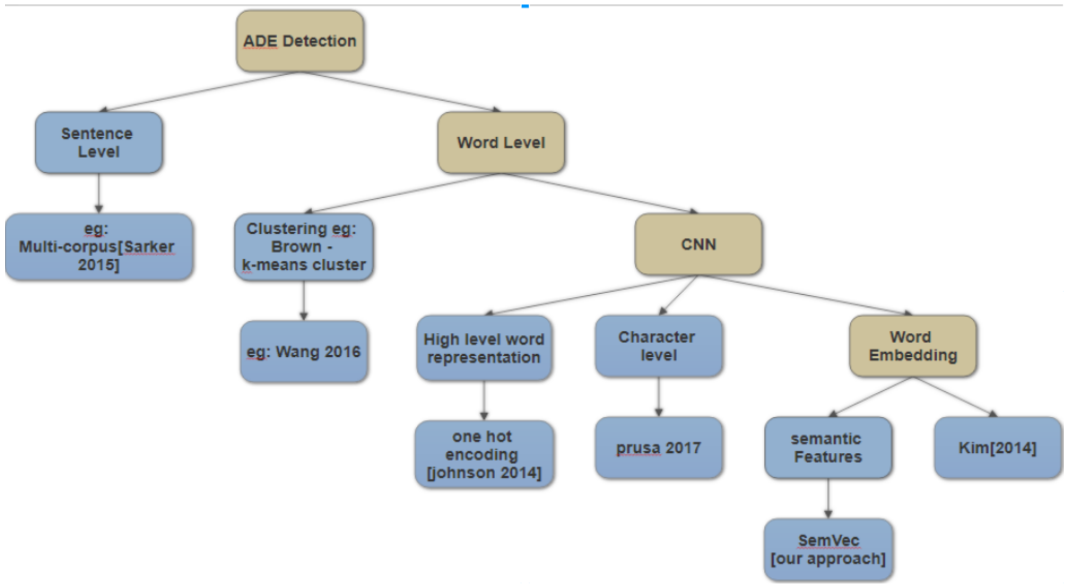


Figure 3.5: Literature review comparison

dynamic as some features can be added or removed based on the given task and word representation contains semantic and domain features that word2vec alike systems do not have.

3.5 Conclusion

The findings from this review reveal a lack of methods that explore the effect of combining domain, semantic and morphological features with CNN. Indeed, some studies used domain and semantic features but with SVM and other machine learning algorithms. Added to this, another study tried to improve CNN performance by using domain features and word2vec but for relation extraction task. On the other hand, multiple studies used CNN with different word representations-i.e. GloVe and word2vec- for ADR classification but none of them explored the effect of domain features on CNN. Hence, direction for the research discussed in this thesis is extracted from Kim, or those who have improved Kim's method, in the domain of ADR classification and Sarker and Sahu their collaborators in the domain of ADRs features engineering. However, none of these methods explored the effect of domain, semantic, and morphological features on CNN. This is considered in further detail in the next chapter as the part of the discussion of the proposed method.

Chapter 4

Proposed Method

This chapter proposes a new approach for text classification that is called Semantic Vector (**SemVec**). SemVec uses domain-specific features and CNN-section 1.3.4 describes why CNN- to improve tweets ADR classification performance. For example, in order to classify tweets into two categories which are: tweets without drug side effects and tweets which may contain ADRs, SemVec uses feature engineering to extract features related to the medical domain. Furthermore, SemVec can be applied to different text classification problems in different domains as well.

4.1 Introduction

Many text classification approaches rely on feature engineering [19]. Feature engineering process is often guided by domain knowledge. In the medical domain, for example, terms related to diseases such as ADR, symptoms, and medications are manually defined by experts [19].

SemVec is a new approach that utilizes domain-specific features to improve CNN networks ADR classification performance. In this approach, SemVec proposes a new word representation using domain-specific features instead of using word2vec [47] or GloVe [55] word embedding representation. Moreover, SemVec proposes a new CNN network model for text classification that best fit the proposed word representation.

4.2 New Word Representation

As shown in figure 4.1, SemVec represents each word in each sentence as a row vector of features. For example, the word He is represented as a vector of feature values:

$$x = f_1^1 \oplus f_2^1 \oplus f_3^1 \oplus f_4^1 \oplus f_5^1 \oplus f_6^1 f_7^1 \oplus f_8^1 \oplus f_9^1 \oplus f_{10}^1 \oplus f_{11}^1 \oplus f_{12}^1 \oplus f_{13}^1$$

Here $\oplus$ is concatenation operation so $x \in \mathbb{R}^{1\times 13}$. It is worth noting that each feature can be represented by one or more dimensions in the resulting word vector. Moreover, the sentence is represented as a matrix of words X features. In order to have a unified

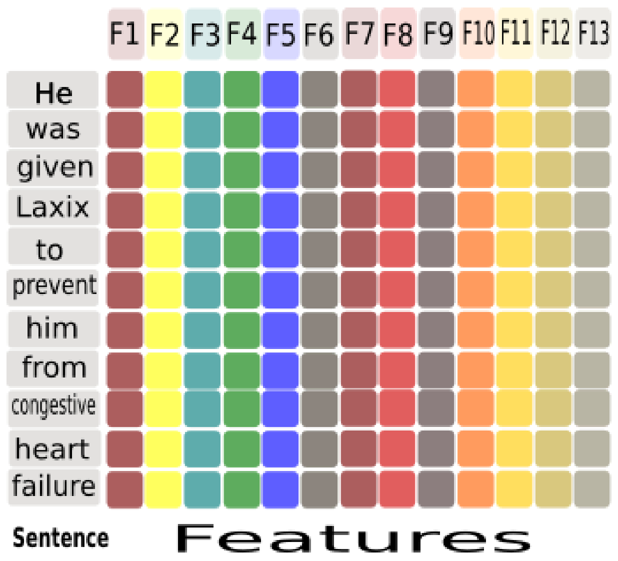


Figure 4.1: Features Layer.

number of matrix rows, we get the max number of words and zero pad any matrix that has a smaller number of words. So, each matrix has a length of 32 rows-max number of sentence words in all corpus-.

4.3 Domain Features Identification & Importance

SemVec utilizes domain specific features to improve ADR classification performance. Moreover, as SemVec is a dynamic and pluggable method, it can be applied to other domains other than ADR domain by using that domain specific features. In order to apply it to new domains, a set of the new domain features need to be used. Besides the new domain features, some of the features proposed in this thesis 4.6 can also be used.

Feature engineering process is often guided by domain knowledge. All unique terms present in a text corpus can be considered as features [19]. In the medical domain, for example, terms related to diseases such as ADR, drug names, symptoms, and medications can be considered as domain-features. In sentiment analysis tasks, language features such as good words, bad words, and emotion words can be also considered as domain-features. Moreover, emotion icons in social media text can also be considered as a domain feature in sentiment analysis as they express users' emotions and feelings.

Features importance can be measured by multiple methods. The first one is to check if this feature already exists in this corpus and how it is distributed among classes. Does this feature split the dataset into the desired classes? Or it is equally found in all classes? A feature that only exists in the desired class instances can be considered as a high-quality feature. The empirical method can be used also to measure the effect of each feature. The leave-one-out technique, where researchers remove one feature from the generated matrix, can help measure the effect of a single feature on classification performance although it can be not precise but can give an indication of feature effect.

4.4 Preprocessing

The first step before performing any feature extraction is to perform standard preprocessing. We use lowercasing, tokenization and stemming or part of speech tagging based on the feature type. Stop words are not removed from the sentences to have the whole sentence represented as a matrix. Porter stemmer is used for stemming words [56]. NLTK python library is used for part of speech tagging and tokenization [1]. In this thesis, some special social media text preprocessing were performed. In particular, Twitter user names were removed by removing any word starting with at (@) char, remove hashtags and URLs to other websites.

4.5 Features Categorization

In this thesis, SemVec was applied to a twitter ADR dataset and an additional medical reports domain dataset. A set of features related to the medical domain was extracted which can be classified into six categories. This section lists these categories with a list of included features. Features will be described in more details in the next section. Features categories are as follows:

- **Sentiment/Opinion Features:** this category indicate how positive, negative, and neutral the words are[10]. We used 4 sentiment features in this thesis. These features can be applied to other datasets-for sentiment analysis tasks-. This category includes the following features:
  - Opinion Lexicon Negative Words(F1)
  - Opinion Lexicon Positive Words(F2)
  - SentiWordNet v3.0 Lexicon Word Positive Score(F3)
  - SentiWordNet v3.0 Lexicon Word Negative Score(F4)
- **Subjectivity Features:** this category includes any word express an emotion, evaluation, opinion, stance, and speculation [75]. This feature can be applied to other datasets. This category includes the following feature:
  - Subjectivity Lexicon (F5)
- **Domain Specific Features:** represents the features that are related to a specific domain. In this thesis, we use one feature from the medical domain. This category includes the following feature:
  - ADR Lexicon (F6)

[1]http://www.nltk.org/

- **Polarity Features:** these feature set tries to capture when there is an increase/decrease in a good/bad thing. This category includes the following features:
    - More Good Lexicon (F7)
    - More Bad Lexicon (F8)
    - Less Good Lexicon (F9)
    - Less Bad Lexicon (F10)
- **Word Complexity Features:** these features are based on text statistics such as word length and word order. This category includes the following features:
    - Word Length (F11)
    - Word Order (F12)
- **Other Features:** this represents other features that may be created using unsupervised learning algorithm or any other type of algorithm. This category includes the following features:
    - Word Clusters (F13)

4.6 Features Extraction

SemVec represents each word with a vector of features. Based on the problem domain, additional features can be added or removed. The following list shows these features and how they are represented in SemVec:

- **Opinion Lexicon Negative Word (F1):** this feature represents a list of *negative English language* opinion words. This list contains also a list of ADR mention words. Hu and Liu [27] proposed a list of negative opinion words that can be used in sentiment analysis. The list contains 4817 negative words. Table 4.1 shows a sample of Opinion Lexicon Negative Words. The following steps show how to extract this feature:
    - Every word in the sentence is first stemmed using Porter stemming algorithm [56].
    - Then,this word is matched against the stemmed list of negative opinion words.
    - If the word exists in this NEGATIVE list of words we set its assigned value to **W** otherwise, it is set to **0**. **W** is a positive integer that represents this feature weight.
- **Opinion Lexicon Positive Word (F2):** this feature represents a list of *positive English language* opinion word. Hu and Liu [27] proposed a list of positive opinion words that can be used in sentiment analysis. Table 4.1 shows a sample of Opinion Lexicon Positive Words. The following steps show how to extract this feature:

Feature	Samples
Opinion Lexicon Negative Word	*abnormal** annoying aggressive sue *suicidal** zombie *ache** *addict** *allergic**
Opinion Lexicon Positive Word	abundance adequate awesome fascinating
ADR Lexicon	*infection vascular** *fecal fat increased** *luteinizing hormone decreased** *ulcer gastrojenunal** *high feeling**
More Words	enhance augment increase amplify
Less Words	drop fewer slump fall down
Good Words	beneficial improve advantage resolve
Bad Words	complication risk *adverse** *chronic** *bleeding** morbidity
Word Cluster[2]	000110 who've 1987 0001110 sshe 43 0001110 ser-ueberwacher 43 0001110 shhe 47 0001110 testasterisk 47

Table 4.1: Features Samples. (* denotes an ADR domain specific words/sentences)

- Every word in the sentence is first stemmed using Porter stemming algorithm [56].
- Then,this word is matched against the stemmed list of positive opinion words.
- If the word exists in this NEGATIVE list of words we set its assigned value to **W** otherwise, it is set to **0**. **W** is a positive integer that represents this feature weight.

- **SentiWordNet Lexicon Positive Word(F3):** this feature represents a *positive English language* sentiment word. Baccianella et al. [10] proposed the SentiWordNet v 3.0 which contains 118,000 English words associated with positive sentiment score between 0 and 1. Table 4.2 shows a sample of SentiWordNet Lexicon. The following steps show how to extract this feature:
  - Every word in the sentence is tagged by part of speech tagger.
  - Then,this word is matched against the SentiWordNet list by part of speech tag and text matching.
  - If the word exists in SentiWordNet, SemVec sets its value is calculated by the following equation:

$$x = W * SentiWordNetpositivescore$$

  W is a positive integer that represents this feature weight.
  - If the word does not exist in SentiWordNet, SemVec sets its value to **0**.

Pos Score	Neg Score	SynsetTerms Gloss
0.25	0	parturient giving birth; "a parturient heifer"
0.75	0	direct lacking compromising or mitigating elements
0.5	0.125	living (informal) absolute; "she is a living doll"
0.125	0.5	superabundant most excessively abundant
0.375	0.25	unabused not physically abused; treated properly

Table 4.2: SentiWordNet Lexicon Sample

- **SentiWordNet Lexicon Negative Word (F4):** this feature represents a *negative English language* sentiment word. Baccianella et al. [10] proposed the SentiWordNet v 3.0 which contains 118,000 English words associated with negative sentiment score between 0 and 1. Table 4.2 shows a sample of SentiWordNet Lexicon.The following steps show how to extract this feature:
  - Every word in the sentence is tagged by part of speech tagger.
  - Then,this word is matched against the SentiWordNet list by part of speech tag and text matching.

Type	Word	Prior Polarity	POS
weaksubj	abandon	negative	verb
strongsubj	aberration	negative	noun
weaksubj	abrupt	negative	adj
strongsubj	aggrieved	negative	adj

Table 4.3: Subjectivity Lexicon Sample

  - If the word exists in SentiWordNet, SemVec sets its value by the following equation:
    $$x = W * SentiWordNetnegativescore$$
    W is a positive integer that represents this feature weight.
  - If the word does not exist in SentiWordNet, SemVec sets its value to **0**.

- **Subjectivity(F5):** this feature represents the *English language* subjectivity value of words. Wilson et al.[75] have proposed a list of words with there subjectivity strength (weak and strong) and their polarity (negative, positive and neutral). Table 4.3 shows a sample of Subjectivity Lexicon. The following steps show how to extract this feature:
  - Every word in the subjectivity list is read and given a score according to strength and polarity. If it is strong (encoded to 1) or weak (encoded to 0.5) and its polarity can be positive (encoded to 1) or negative (encoded to -1) or neutral (encoded to 0). The final score for each word is calculated as follows:
    $$subjectivityscore = type * polarityscore$$
  - If the word exists in subjectivity list , SemVec sets its value by the following equation:
    $$x = W * subjectivityscore$$
    W is a positive integer that represents this feature weight.
  - If the word does not exist in subjectivity list, SemVec sets its value to **0**.

- **ADR Lexicon (F6)**: this feature represents a drug side effect(ADR). It incorporates ADR domain knowledge to the classifier by performing pattern matching with ADR lexicon. Sarker et al. [66] have built this ADR lexicon. It includes a total of 13,699 ADR mentions from different resources. Table 4.1 shows a sample of ADR Lexicon. The following steps show how to extract this feature:
  - Every word in the sentence is tagged by part of speech tagger.
  - For each sentence, we create word-level 1-gram, 2-gram, 3-gram, and 4-gram.
  - Then, resulting word-level n-gram are matched against the ADR lexicon and counter of how many matches per sentence is saved.

- If the word-level n-gram sentence exists in ADR lexicon, SemVec calculates its value by the following equation:

  $$x = W * count(ADRinsentence)$$

  W is a positive integer that represents this feature weight.

  Each word in the matching sentence will receive the value calculated by the above equation. Otherwise, it is set to **0**.

- **More Good (F7), More Bad (F8), Less Good(F9), Less Bad(F10):** Niu et al. [50] have proposed these four polarity features. More Good feature indicates more positive information in the sentence. This represents how a change happens: for example, reducing headache is considered as a positive outcome; on the other hand, increasing headache, is considered as a negative outcome. These features try to find out when there is an increase/decrease in a good/bad thing. A collection of good, bad, more and less words were used which was created by Sarker et al. [64]. In order to extract these features, a window of four words on each side of each feature word. If a Good word was found in this window, then a more-good feature is activated. Similar process were followed to activate other features. Table 4.1 shows a sample of negative, positive, less, and more words.

  - Every word in the sentence is tagged by part of speech tagger.
  - For each word we define a max boundary and min boundary of words. Min and Max boundaries are windows of words of size 4.
  - Then, if a more good, more bad, less good, less bad word occurs within the min and max boundary this word is given a score of **W1** for more good, **W2** for more bad, **W3** for less good and **W4** for less bad. Otherwise, it is set to **0**. **W1, W2, W3, and W4** are positive integers that represent each feature weight

- **Word Length (F11)**: this feature represents each word number of characters. for example: word: 4, amazing: 7, do: 2 ,etc. This feature defines how complex each word is and how complex the sentence is. The following equation shows how to calculate this feature:

  $$x = W * length(word)$$

  W is a positive integer that represents this feature weight.

- **Word Order (F12)**: this feature represents the order number of this word in the sentence. For example in the following sentence:

  I like this movie very much.

  - order(I)= 1
  - order(like)= 2
  - order(this)= 3

  - order(movie)= 4
  - order(very)= 5
  - order(much)= 6

  The following equation shows how to calculate this feature:

  $$x = W * order(word)$$

  W is a positive integer that represents this feature weight.

- **Word Clusters (F13)**: this feature is proposed by Brown et al. [13] in this feature hierarchical word clusters are generated from a huge set of unlabeled tweets via Brown clustering [13]. This produces a base set of 1,000 clusters which includes clusters for happinesses and sadness words and emotions, etc. Table 4.1 shows a sample of word clusters. The following equation shows how to calculate this feature:

  $$x = wordclusterkey(word)$$

4.7 Model Architecture

Regular Neural Nets receive a single vector input, and then a series of hidden layers transform it to generate the final class scores. Each hidden layer is composed of a set of neurons. Neurons in one layer function independently of other neurons in other layers and each neuron in any layer is connected to all neurons of the previous layer [7]. The last layer is the output layer which represents the class scores.

CNN uses convolving filters that are applied to local features [40]. The first CNN for text classification approach was proposed by Kim [35].

SemVec proposes a new CNN model for text classification. The new model improves classification results in terms of accuracy and precision. In this thesis, we compared the following network models:

4.7.1 Kim Model

Kim [35] has proposed a CNN network model for text classification. Figure 4.2 shows Kim CNN model architecture. The proposed model takes complete tweet as an input and classifies it into either positive or negative class. More information about this model can be found at [35].

4.7.2 SemVec Model

In this thesis, we propose a new CNN model for text classification. Figure 4.5 outlines the main layers in the proposed model. It only contains a single convolution layer with one filter of size 2(by experiment). All the properties of SemVec model was found through grid search experiments 5.3.6. The following describe the differences between our proposed model and Kim model:

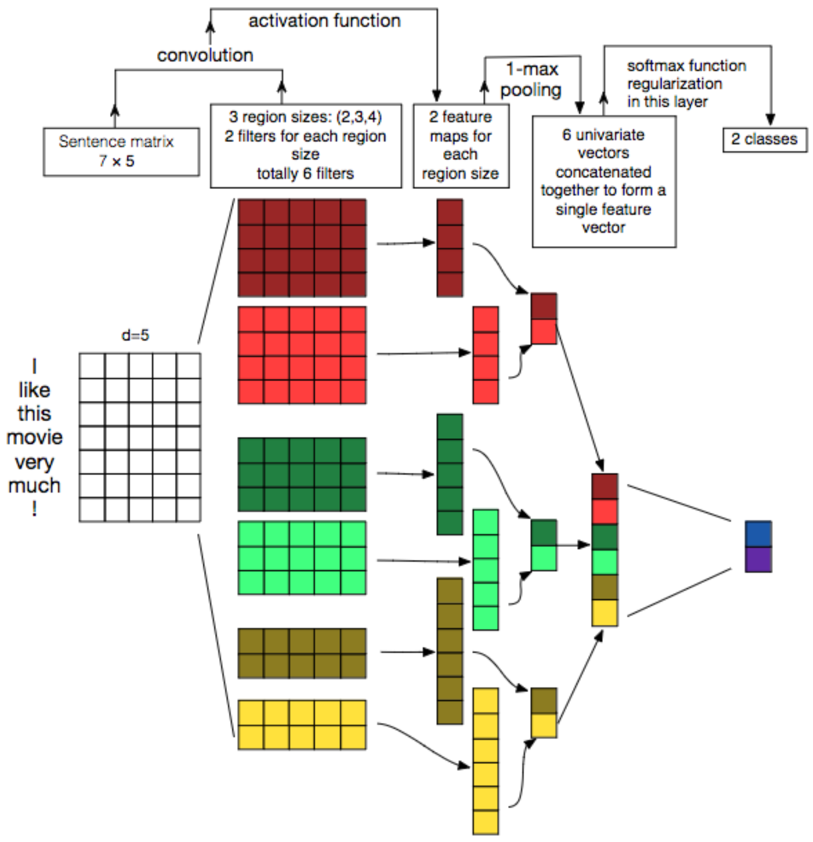
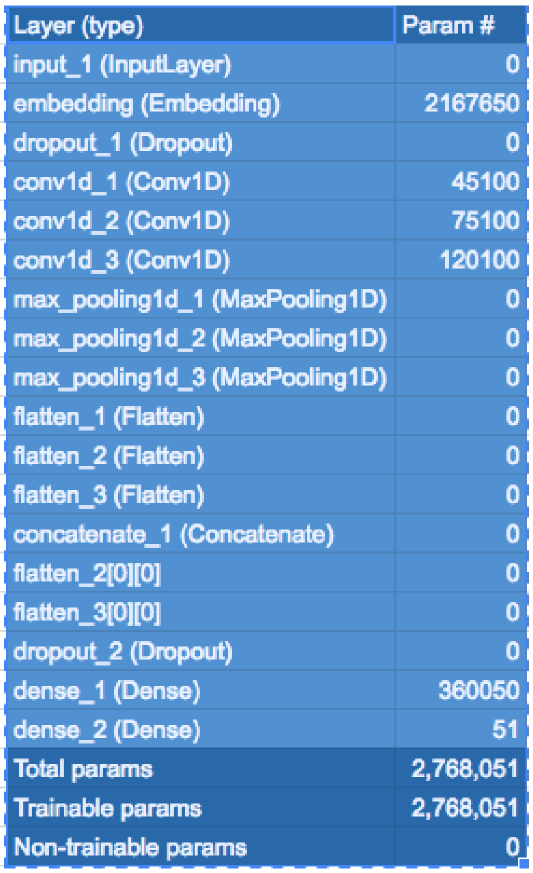

activation function
convolution
1-max pooling
softmax function regularization in this layer
Sentence matrix 7 x 5
3 region sizes: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
6 univariate vectors concatenated together to form a single feature vector
2 classes
d=5
I
like
this
movie
very
much
!

Figure 4.2: Kim [35] CNN model architecture. [80]

- In this model, each word is represented with 13 discrete features instead of using word embeddings.
- SemVec model uses a single convolution layer instead of three convolution layers used in Kim's model, one max pooling layer instead of three max pooling layers.
- SemVec model uses a filter size of 2(by experiment) instead of whole word embedding size in Kim model.
- SemVec model uses two fully connected layer of 128 and 1 respectively while Kim model does not use them.
- SemVec model uses Sigmoid activation function in the last layer instead of Softmax layer used by Kim model.

Subsequent sections describe each layer in SemVec model in details:



Layer (type)	Param #
input_1 (InputLayer)	0
embedding (Embedding)	2167650
dropout_1 (Dropout)	0
conv1d_1 (Conv1D)	45100
conv1d_2 (Conv1D)	75100
conv1d_3 (Conv1D)	120100
max_pooling1d_1 (MaxPooling1D)	0
max_pooling1d_2 (MaxPooling1D)	0
max_pooling1d_3 (MaxPooling1D)	0
flatten_1 (Flatten)	0
flatten_2 (Flatten)	0
flatten_3 (Flatten)	0
concatenate_1 (Concatenate)	0
flatten_2[0][0]	0
flatten_3[0][0]	0
dropout_2 (Dropout)	0
dense_1 (Dense)	360050
dense_2 (Dense)	51
Total params	2,768,051
Trainable params	2,768,051
Non-trainable params	0

Figure 4.3: Kim [35] CNN architecture weights.

4.7.2.1 Feature Input Layer

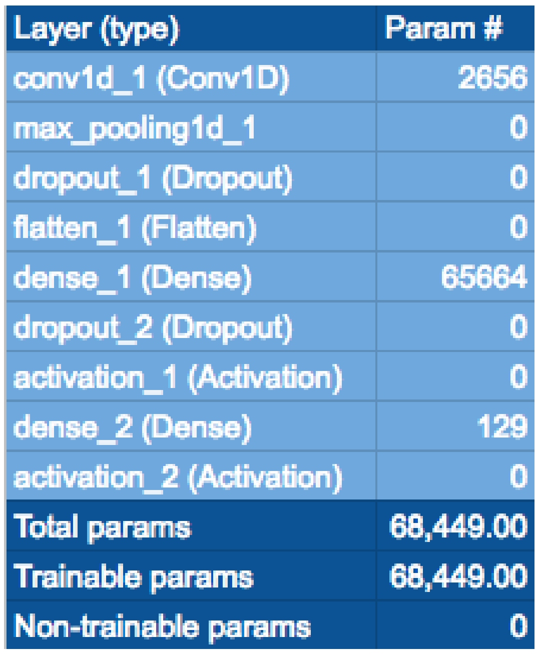
In this model, each word is represented with 13 discrete features as described in section 4.2.

4.7.2.2 Convolution Layer

In Kim [35] model, the convolution layer uses three filter sizes: 3,5 and 7 each one of them has 2 filters as described in section 2.4.2. On the other hand, SemVec model uses one filter size of 2 found by grid search as in section 5.3.6.

4.7.2.3 Max Pooling Layer

In this layer, we apply a max operation to find the most useful feature in the generated feature map from the previous layer. SemVec max pooling window size is equal to 2*2 found by grid search as in section 5.3.6.



Layer (type)	Param #
conv1d_1 (Conv1D)	2656
max_pooling1d_1	0
dropout_1 (Dropout)	0
flatten_1 (Flatten)	0
dense_1 (Dense)	65664
dropout_2 (Dropout)	0
activation_1 (Activation)	0
dense_2 (Dense)	129
activation_2 (Activation)	0
Total params	68,449.00
Trainable params	68,449.00
Non-trainable params	0

Figure 4.4: SemVec CNN architecture weights.

4.7.2.4 Dense Layers

SemVec model has 2 fully connected layers. The first layer includes 128 neurons and the second one has one neuron. A non linearity Relu activation function connects the layers together found by grid search as in section 5.3.6.

4.7.2.5 Sigmoid Layer

In the final layer, a sigmoid receives the feature vector from the previous layer and uses it to perform binary classification. The sigmoid equation is:

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

found by grid search as in section 5.3.6.

4.8 Model Weights Comparison

As shown in figure 4.4 and 4.3, SemVec uses only 68,449 weights compared to 2,768,051 for Kim [35] model. So, Kim model uses 40x more weights than SemVec to train CNN network for text classification task. This clearly impact network training time.

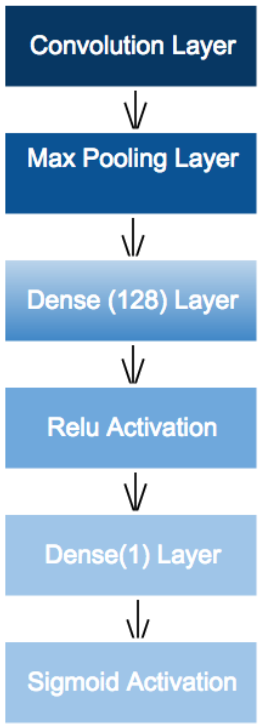


Figure 4.5: SemVec model architecture.

4.9 SemVec Use Case

SemVec can be used in post-market drug monitoring. In this phase, patients may share ADRs on social media sites which they do not report for governmental drug monitoring systems such as AERS. It is vital to capture these ADRs as some of them can be critical and may cause a drug withdrawal. This thesis proposes a system that can capture ADRs from a live Twitter stream as shown in figure 4.6.

The system listens for live twitter steam filtered by a list of predefined drugs. The second step is to use trained SemVec model to classify filtered posts into two categories: contains ADR and do not contain ADR. After this step, a human annotator can verify classification results of SemVec. The new set of annotated tweets can be used to retrain SemVec model and improve classification performance.

Based on previously classified tweets we can classify drugs into black box and normal drugs. In order to classify drugs into black box and normal, we followed Patki et al. [54] inference step that is based on the assumption that drugs in blackbox category have more ADRs posts than normal drugs. Patki et al. [54] have proposed a probalistic model that can be used to classify a drug belonging to one of the two categories. The following equations [54] shows the probability estimates for a drug to be in blackbox or normal category.

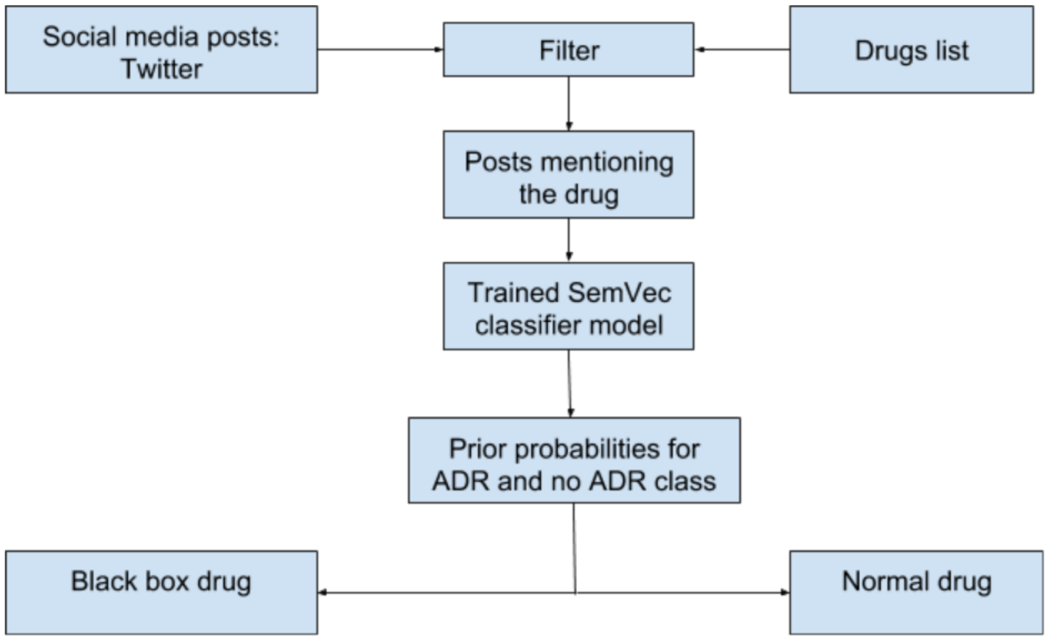

Social media posts: Twitter
Filter
Drugs list
Posts mentioning the drug
Trained SemVec classifier model
Prior probabilities for ADR and no ADR class
Black box drug
Normal drug

Figure 4.6: SemVec use case

$$P(y = N|x_1x_2...x_n) = \frac{\sum_i^n P(y = noADR|x_i)}{n}$$

$$P(y = B|x_1x_2...x_n) = \frac{\sum_i^n P(y = ADR|x_i)}{n}$$

Where N stands for normal class, B stands for black box class, x1 to xn are the comments related to that drug, and n is the total number of comments belonging to that drug.

As the number of comments without ADRs are usually much higher than comments with ADRs, this results in higher sums for noADRs compared to ADRs. As a result, the system will always be biased towards normal class. To solve, we can scale the ADR probability by a scaling factor which is the number of tweets with no ADRs to the number of tweets with ADRs [54] as shown in the following equations:

$$\alpha = \frac{\text{number of no ADR tweets}}{\text{number of ADR tweets}}$$

$$P(y = B|x_1x_2...x_n) = \alpha * \frac{\sum_i^n P(y = ADR|x_i)}{n}$$

Using this approach, if for a drug the probability of being black box drug * α ¿ probability of being normal,we categorize the drug as a black box; otherwise, we categorize it as normal. Finally, drugs classified as a black box can be then reported to government health department for further investigation.

Chapter 5

Evaluation and Results

This thesis takes an empirical approach to evaluate the performance of our proposed method by comparing it with the state of the art methods on different datasets. Due to ADRs criticality, precision is selected as the most significant metric. Also, other classification metrics are reported. This chapter represents datasets and evaluation methodology which was used in experiments. Moreover, it discusses the results of our proposed approach compared to other methods in ADRs classification.

5.1 Evaluation Methodology

This section represents thesis methodology that was followed to evaluate proposed method.

5.1.1 Datasets Selection

In order to evaluate the performance of SemVec, we run a group of experiments on three different datasets. The three datasets are publicly available and consist of manually annotated text segments. To evaluate SemVec ADR classification performance we used Twitter ADR corpus and in addition ADE corpus. Both datasets are annotated for ADRs presence or absence. Moreover, each dataset has different post types: the Twitter dataset contains a collection of tweets whereas the ADE dataset contains a collection of medical case reports which helps in evaluating SemVec performance on different post types. Furthermore, CADEC [34] dataset was excluded because it is mainly designed and annotated for ADR entity extraction and not for classification.

SemVec aims to allow classification of texts in different domains. To evaluate SemVec performance on different domains we apply it to SemEval 2015 dataset. SemEval 2015 dataset contains tweets expressing a sentiment about popular topics without ADRs. The following is a brief description of the datasets.

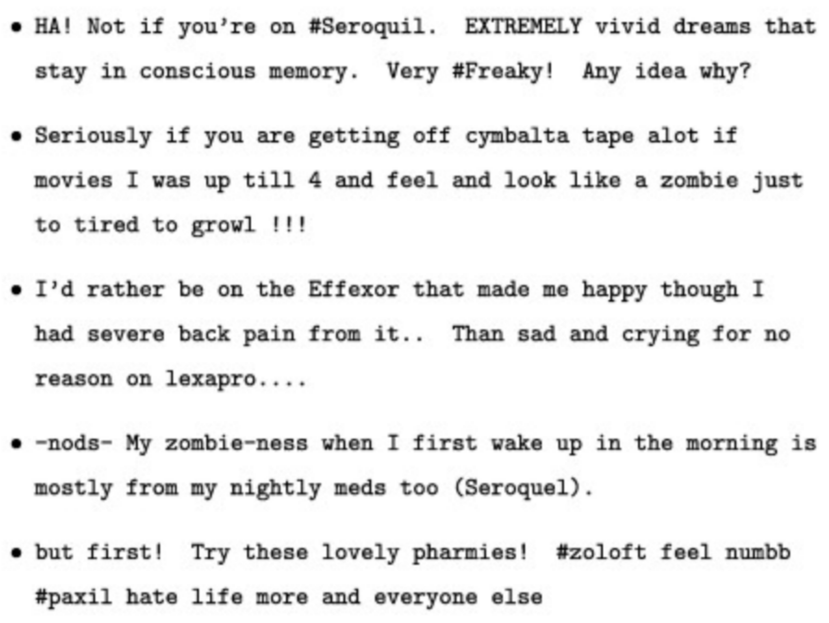

- HA! Not if you're on #Seroquil. EXTREMELY vivid dreams that stay in conscious memory. Very #Freaky! Any idea why?
- Seriously if you are getting off cymbalta tape alot if movies I was up till 4 and feel and look like a zombie just to tired to growl !!!
- I'd rather be on the Effexor that made me happy though I had severe back pain from it.. Than sad and crying for no reason on lexapro....
- -nods- My zombie-ness when I first wake up in the morning is mostly from my nightly meds too (Seroquel).
- but first! Try these lovely pharmies! #zoloft feel numbb #paxil hate life more and everyone else

Figure 5.1: Sample tweets from Twitter dataset. [66]

5.1.1.1 Twitter ADR Corpus

The first dataset has been sourced from Twitter[66]. A total of 74 drugs from IMS Health's Top 100 drugs by volume for 2013 were used. The tweets collected using the generic and brand names of drugs, including phonetic misspellings. The dataset is composed of a total of 7,574 instances of which 6,672 do not contain ADRs, and only 902 include ADR mentions. So this dataset is highly imbalanced which reflects the real world, where the number of tweets with ADRs are so small compared to non-ADR tweets,- only 11.9% of the tweets have new ADRs. Figure 5.1, shows sample tweets from twitter dataset.

5.1.1.2 ADE Corpus

ADE corpus[23] consists of a set of annotated sentences from medical case reports. The dataset is composed of a total of 23516 instances of which 6,821 (29.0%) includes ADR mentions, while the rest do not. Although this data set is of different origin we can use it to evaluate our method performance. Figure 5.2, shows sample posts from ADE dataset.

5.1.1.3 SemEval-2015 Task 10 Dataset

SemEval is the most popular Twitter sentiment analysis shared task. The dataset for this task was gathered from tweets expressing a sentiment about popular topics(general

English language domain). The 2015 dataset is a collection of previous 2 years shared task dataset and 2015 dataset. The dataset was collected for popular topics from the same time period. Testing messages have different topics than training. Tweets were skewed towards the neutral class. Moreover, the tweets that do not contain sentiment-bearing words using SentiWordNet 3 [10] were removed, to reduce class imbalance. Sentiment-bearing word is any word in SentiWordNet [10] that has a positive or negative score [59].

Corpus	Pos.	Neg.	Obj./ Neu	Total
Twitter2013-train	3,662	1,466	4,600	9,728
Twitter2013-dev	575	340	739	1,654
Twitter2013-test	1,572	601	1,640	3,813
SMS2013-test	492	394	1,207	2,093
Twitter2014-test	982	202	669	1,853
Twitter2014-sarcasm	33	40	13	86
LiveJournal2014-test	427	304	411	1,142
Twitter2015-test	1040	365	987	2392

Table 5.1: Data set statistics for SemEval 2015[59]

5.2 Evaluation Methodology

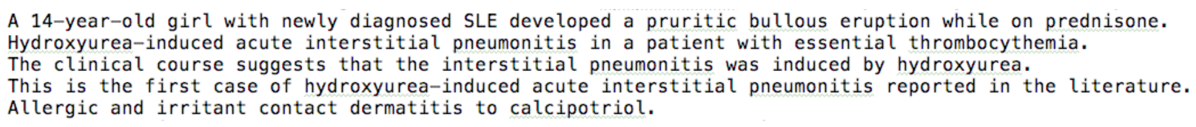
To evaluate the performance of our approach, we used evaluation metrics to evaluate the quality of our model predictions namely: accuracy, recall, precision, and f-score.

Our approach aims to achieve better precision results in comparison to the state of the art approaches. We evaluated our approach on Twitter ADR dataset as well as other datasets.

In order to correctly evaluate our approach and find out more about its limitations and strengths, a set of experiments were conducted as follows:

1. SemVec proposes a new approach using domain-specific features along with a new CNN network model to improve model precision. To validate this, experiments were conducted to measure SemVec model precision and compare it to other approaches. The results of the these experiments are presented in section 5.3.1.

2. To validate that domain specific features improve model precision and not any random numbers, experiments were conducted to compare SemVec with untrained



```
A 14-year-old girl with newly diagnosed SLE developed a pruritic bullous eruption while on prednisone.
Hydroxyurea-induced acute interstitial pneumonitis in a patient with essential thrombocythemia.
The clinical course suggests that the interstitial pneumonitis was induced by hydroxyurea.
This is the first case of hydroxyurea-induced acute interstitial pneumonitis reported in the literature.
Allergic and irritant contact dermatitis to calcipotriol.
```

Figure 5.2: Sample posts from ADE dataset.

embeddings on SemVec network model. Untrained embeddings are randomly initialized word vectors. The results of the these experiments are presented in section 5.3.1 and 5.3.2.

3. We propose a new CNN network model for text classification which uses a single convolution layer and a set of optimized parameters -found by experiments as shown in 5.3.6- to SemVec word representation. To validate that this model improves classification performance, experiments were conducted to compare its classification results with Kim Model. The results of the these experiments are presented in section 5.3.3.

4. Adding more domain-specific features can improve classification precision. To validate this, experiments were conducted to measure the changes in classification performance when adding more features. The results of the these experiments are presented in section 5.3.7.

5. Each feature has a different contribution than other features. To evaluate features contribution, a set of leave-one-out experiments were conducted to measure the contribution of each feature(one feature is removed from SemVec every time). The results of the these experiments are presented in section 5.3.8.

6. In order to determine the best hyper-parameters, several experiments were conducted to search through a manually specified subset of the hyper-parameter space measured by cross-validation on the training set. The results of the these experiments are presented in section 5.3.6.

Furthermore, the following additional experiments (outside the scope of this thesis) were conducted to evaluate SemVec:

1. To validate that SemVec can be applied to different domains, an experiment were conducted on a general English language domain (SemEval 2015 dataset in this case). The results of this experiment are presented in section 5.3.5.

2. SemVec should work on different post types. To validate this, an experiment were conducted to validate that SemVec works on different post types (ADE medical reports dataset). The results of this experiment are presented in section 5.3.4.

5.2.1 Metrics

To correctly evaluate our model we use multiple classification metrics. This section defines the metrics that are used in the thesis.

- **True Positives(TP):** total number of correctly classified posts as positive class[52](i.e: correctly classified as containing ADRs in Twitter and ADE datasets).
- **True Negatives(TN):** the number of posts correctly classified as belonging to the negative class[52](i.e: correctly classified as not containing ADRs in Twitter and ADE datasets).

- **False Positives(FP):** the number of wrongly classified posts as positive class[52](i.e: wrongly classified as containing ADRs in Twitter and ADE datasets).

- **False Negatives(FN):** the number of wrongly classified posts as negative class[52](i.e: wrongly classified as not containing ADRs in Twitter and ADE datasets).

- **Accuracy:** represents total correctly classified (positives and negatives) divided by total number of samples samples[52]. Accuracy measures the percentage of correctly classified classes(Both ADRs and non-ADRs), for which high accuracy indicates that either positive(ADR) or negative(non-ADR)(in imbalanced datasets) or both classes were correctly classified.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

- **Precision:** also known as confidence in data mining. It represents the proportion of predicted positive instances that are really positives[52]. Precision measures the percentage of correctly identified ADRs, for which high precision indicates that from the identified/extracted ADRs, high percentage correctly identified as ADRs.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall:** or sensitivity, it represents the proportion of true predicted positive objects from the total number of positive objects in the set [52]. Recall measures the proportion of correctly identified ADRs from the total number of ADR instances in the whole dataset.

$$Recall = \frac{TP}{TP + FN}$$

- **F-measure:** represents a weighted average of both precision and recall [52].

$$F - measure = 2 * \frac{1}{\frac{1}{recall} + \frac{1}{precision}}$$

- **Epoch:** represents one cycle of training on the entire training set samples [15].

- **Mean:** represents sum of all values divided by the number of those values [18].

$$mean = \frac{\sum x_i}{n}$$

- **Standard Deviation:** represents the amount of variation between set data values [12].

$$s = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N - 1}}$$

5.2.2 Experiment Design

All the experiments were conducted on Google cloud virtual machines. Table 5.2 lists virtual machine specifications. Experiments implementation were done using Python programming language version 2.7 along with Keras deep learning library with Tensorflow[6] deep as backend.

Environment Variable	**Specs**
Operating System	Debian GNU/Linux 9 (stretch)
CPU	4 vCPUs
RAM	3.6 GB

Table 5.2: Virtual machine specifications

5.2.3 Training, Learning and Test Sets

A common way to experiment the model classification performance is to randomly split the dataset into a training set, and test set. The dataset was divided into two parts: 80% training and 20% testing set [66]. We used Stratified k-fold which is a variation of k-fold which returns stratified folds. This ensures that each set contains approximately the same proportions of instances for each ADR and non-ADR class as the complete set. In this experiment, 10 stratified Fold (splits) were used. With SemEval dataset 5.3.5 we used three different datasets: training, validation and testing sets. In both cases, the classification model is built based on training set only. The classes of validation and test set instances are hidden from the model in training step.

5.2.4 Experiment Settings

The list of all the major model parameters are shown in table 5.3. A set of grid search experiments were conducted to find the optimal values for each parameter as shown in section 5.3.6. The following explains each setting:

- **BATCH_SIZE:** represents the number of instances that are used in each step of training.
- **DROPOUT_KEEP_PROB:** dropout is a method used in CNN to regularize the network. This setting controls the fraction of neurons that are enabled during training time.
- **EMBEDDING_DIM:** represents the dimensions of the vector space model.
- **FILTER_SIZES:** represents how many words in the sentence matrix to slide over in each filter.
- **NUM_FILTERS:** represents the number of filters per each filter size.

Parameter Name	Parameter Value
BATCH_SIZE	128
DROPOUT_KEEP_PROB	0.7
EMBEDDING_DIM	32
FILTER_SIZES	2
NUM_FILTERS	32

Table 5.3: Experiment settings.

5.2.5 Baselines

To compare SemVec results with other methods, 10-stratified-fold cross-validation was used. SemVec results are compared to the state of the art methods in PSB 2016 social media mining shared task on binary classification of adverse drug reactions (ADRs) and other methods. Below is a list of these methods:

- **Convolutional Neural Network (CNN):** by Huynh et al. [28] .
- **Recurrent Convolutional Neural Network (RCNN):** new architecture proposed by Huynh et al. [28] .
- **Convolutional Recurrent Neural Network (CRNN):** new architecture proposed by Huynh et al. [28].
- **Convolutional Neural Network with Attention (CNNA):** new architecture proposed by Huynh et al. [28].
- **Term-matching based on an ADR lexicon(TM):** proposed by Zhang et al. [81] with Huynh et al. [28] implementation.
- **Maximum-Entropy classifier with n-grams and TFIDF weightings(ME-TFIDF):** proposed by Zhang et al. [81] with Huynh et al. [28] implementation.
- **Maximum-Entropy classifier with n-grams and NB log-count ratio(ME-NBLCR):** proposed by Zhang et al. [81] with Huynh et al. [28] implementation.
- **Maximum-Entropy classifier with mean word embeddings(ME-WE):** proposed by Zhang et al. [81] with Huynh et al. [28] implementation.
- **Multi Corpus method:** proposed by Sarker et. al. [66]. We report the best Multi Corpus method results on single ADE data set.

These methods were described in more details in chapter 3. In addition to the above-mentioned methods, we compare the results with:

- **Word2vec:** In this experiment, CNN was used with 150 embedding dimension -$x \in \mathbb{R}^{1\times150}$-. Word2vec embeddings are real-valued vectors of configurable dimension. Nikfarjam et al. [49] generated word2vec word representation was used. The similarity between words was modeled by utilizing an unlabeled twitter dataset about drugs with more than one million tweets. Word representation is learned by training a neural network language model.

- **Untrained Embeddings:** In this experiment, CNN without pre-trained word embedding, instead CNN model learned word embedding from scratch.
- **SemVec:** In this experiment, we used CNN with our generated features as input. SemVec used 13 generated features.
- **Word2vec + generated features :** In this experiment, CNN with pre-trained word2vec embeddings plus 12 generated features as input.

5.3 Results and Analysis

In this section, SemVec is compared to other classification approaches on Twitter ADR dataset (the key evaluation for SemVec) as described in section 5.3.1 and 5.3.2. Additionally, this thesis added extra evaluation datasets including ADE dataset (section 5.3.4) and SemEval 2015 dataset (section 5.1.1.3). Table 5.4 (key evaluation results) shows the results of the comparison between SemVec and other methods on Twitter ADR data set. Table 5.5 shows the results of the comparison between SemVec and other methods on ADE data set. Table 5.6 shows the comparison between SemVec and other methods on SemEval 2015 dataset. Table 5.8 shows leave one out classification results showing how metrics are affected as one feature is removed from the set.

5.3.1 Twitter ADR Dataset Results

This section shows the experimental results of SemVec on Twitter ADR dataset compared to other state of the art methods. Full results are shown in appendix B.

As described in chapter 3, Zhang et al. [81] have achieved the second best approach in PSB task and their approach uses classifiers with engineered features. We compare our results with Huynh et al. [28] implementation of Zhang et al. [81] methods. Moreover, we compare SemVec with Huynh et al [28] and Sarker et al. [66] methods. Section 5.2.5 contains a list of all methods.

Table 5.4 shows comparison between SemVec and a set of state of the art methods(described in section 5.2.5). These results are for instances containing ADR which this thesis is interested in. In this experiment, SemVec used 13 features including preprocessing step where we clean tweet from URLs and hashtags.

Compared to other methods, SemVec achieves the best accuracy- only three methods reported accuracy two of them use CNN as the Twitter dataset is highly imbalanced and the max accuracy score if all tweets are set to no-ADR is 88.1% - and precision scores due to the use of domain specific features, which are tailored to this domain and post type. Features such as ADR lexicon, word clusters, and opinion words have clear effect on improving precision as shown in section 5.3.8 and 5.3.7. Apart from that, the

Method	Accuracy	ADR Precision	ADR Recall	ADR F-score
TM [28]	N/A	13	**89**	23
ME-TFIDF [28]	N/A	33	70	45
ME-NBLCR [28]	N/A	**79**	14	23
ME-WE [28]	N/A	27	73	40
CNN [28]	N/A	47	57	51
CRNN [28]	N/A	49	55	51
RCNN [28]	N/A	43	59	49
CNNA [28]	N/A	40	66	49
Multi-corpus [66]	86.2	N/A	N/A	53.8
SemVec	**92.5**	**79.4**	49.8	60.93
Untrained Embeddings	90.85	60.91	63.12	61.89
Majority Vote [41]	N/A	70.21	59.64	**64.50**
CNN+GoogleNews [8]	90.4	N/A	N/A	54.20

Table 5.4: ADR classification results on the Twitter datasets.

new CNN network architecture that is proposed by this thesis is clearly affecting the model classification performance.

ME-NBLCR achieves second highest ADR precision score as it assigns a high weight to one feature, which is ADR class feature. While ME-NBLCR achieves a high precision score, it has the lowest ADR Recall due to the same reason [81]. Moreover, ME-NBLCR is a fine tuned method to a set of predefined ADRs and does not take into account different features in tweets such as if this tweet expresses a user experience or a news. Due to that Zhang et al. approach uses an ensemble of ME-NBLCR, ME-TFIDF, ME-WE, and TM which results in lower precision and better recall.

Majority Vote method [41] achieves the best F-score results on Twitter ADR dataset. Compared to SemVec, it achieves lower precision and higher recall. Majority Vote achieves such high recall scores since it uses several semi-supervised CNN models built from different types of unlabeled data. In the second phase, CNN is trained with annotated ADE data. The use of several semi-supervised CNN models followed by a linear classifier which determines the final results of theses models clearly improved this method F-score.Untrained embeddings achieves the second best F-score because it achieves a higher recall compared to SemVec.

SemVec achieves the third best F-score because of low recall scores. The lower recall scores are due to the fact that this data set is highly imbalanced (see section 5.1.1.1). The second reason for low recall scores is related to a set of issues such as the use of non-standard terms, short posts, spelling errors, etc. These issues with solutions are described in more details in section 5.3.9. On the other hand, techniques like oversampling proposed by Hensman and Masko [45] can be used to fix dataset imbalance and improve classifier performance.

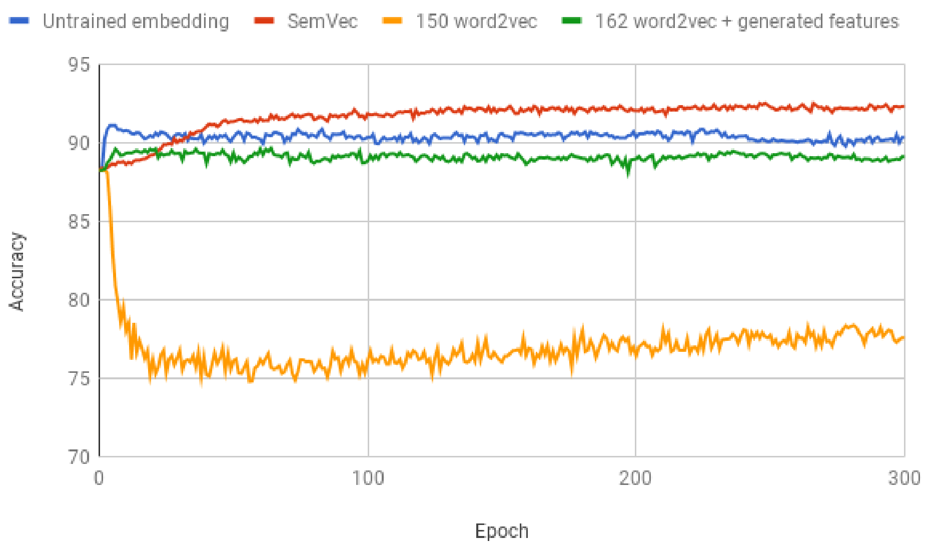


Figure 5.3: Accuracy comparison

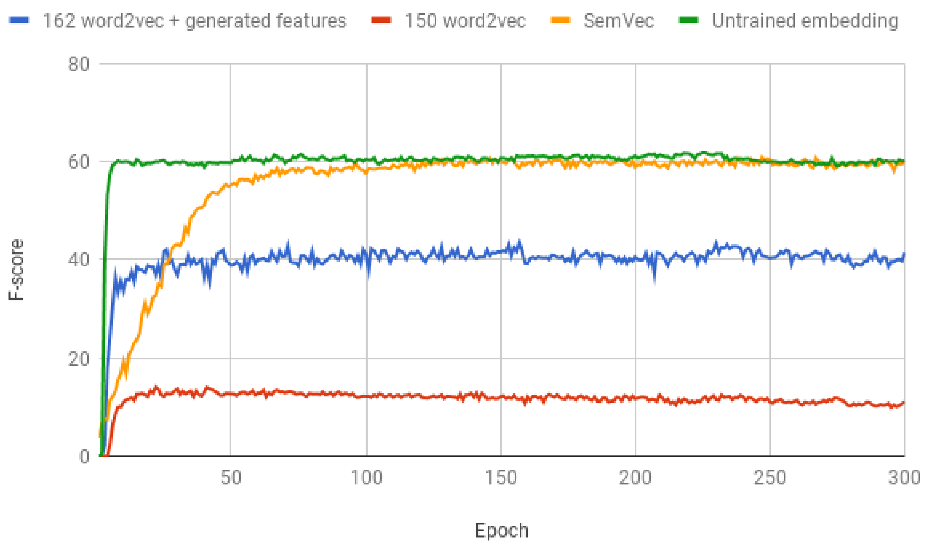


Figure 5.4: ADR F-score comparison

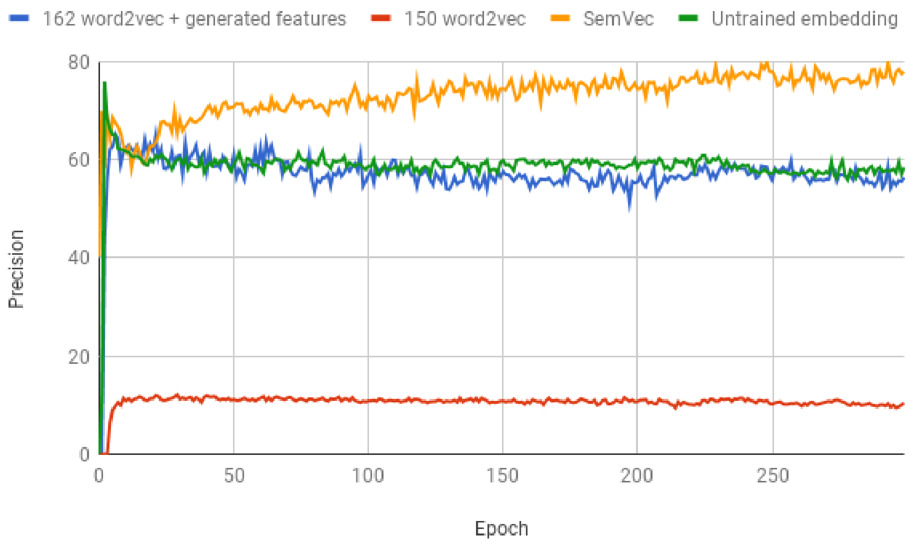


Figure 5.5: ADR Precision comparison

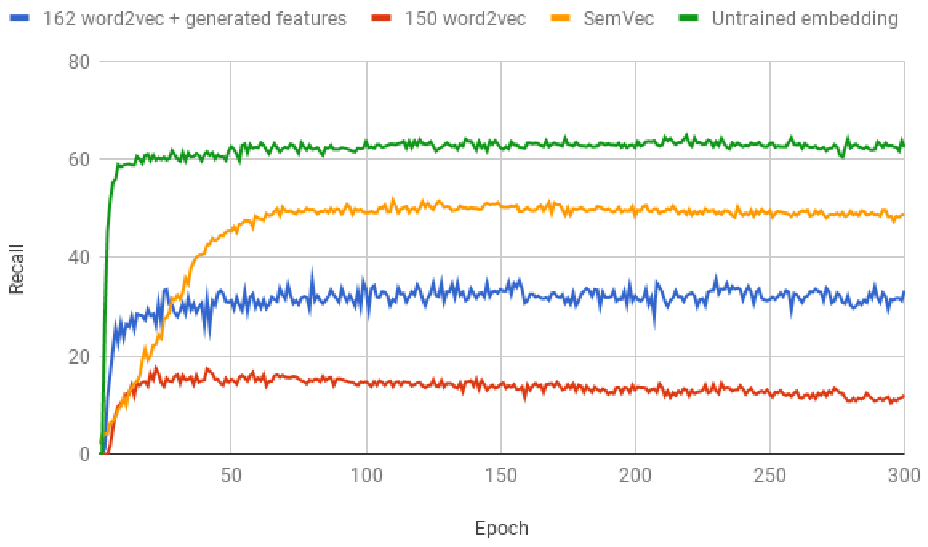


Figure 5.6: ADR Recall comparison

5.3.2 Comparison between SemVec, Word2Vec and untrained embeddings on Twitter ADR Dataset

This section compares the performance of SemVec, Word2Vec and Untrained embeddings on the Twitter ADR dataset based on several experiments conducted by this thesis using SemVec network model 4.7.2. Figure 5.3 shows accuracy scores of the above-mentioned methods on the Twitter dataset. In the first epoch, SemVec achieves a lower accuracy score compared to untrained embedding and word2vec + generated features methods. After that, SemVec accuracy score keeps improving until it achieves its maximum accuracy score. After each epoch, CNN optimizes model weights to achieve a better score. Interestingly, SemVec was able within 38 epochs to achieve the best accuracy results.

Compared to other methods, SemVec achieves the best Accuracy and Precision scores due to the use of domain specific features. Features and the new CNN network architecture that is proposed by this thesis as discussed in section 5.3.1. In addition, domain specific features are clearly improving convergence time and adding more quality features can decrease convergence time and increase model performance as discussed in section 5.3.7.

Untrained embeddings input was able to achieve better accuracy results than both word2vec and word2vec + generated features methods. This is maybe due to the use of SemVec network model. Also, CNN backpropagation algorithm is clearly has improved untrained embedding accuracy.

It is also clear that domain specific features improve word2vec accuracy and precision scores compared to word2vec without domain features.

Figure 5.4 shows f-score scores of the above-mentioned input methods on the Twitter dataset. Untrained embeddings achieve the best accuracy scores and it is clear that it converges faster than any other method. SemVec was able to achieve better f-score results than both word2vec and word2vec + generated features. Moreover, after 100 epoch SemVec achieves a very close scores to Untrained embeddings scores. F-score is clearly related to both model precision and recall. Untrained embeddings achieves higher recall values from the first epochs with lower precision scores. In other words, it classifies more posts as ADR while they are not which improves recall but affects precision. SemVec on the other hand, classify less posts as ADRs in the first epochs which affect recall and f-score while maintaining higher precision scores.

The use of domain specific features clearly improves generated features + word2vec approach f-score results as it improves both precision and recall by improving the features representing each word. Moreover, word2vec scores are lower than other approaches because word2vec word vector is considered as one unit. However, the filter in SemVec network model operates on every 2 dimensions in the word vector to extract features, which results in features without any meaning.

Figure 5.5 shows precision scores of the above-mentioned input methods on the Twitter dataset. SemVec achieves the best precision scores from the 25 epoch. After 25 steps we notice a huge gap between SemVec and other methods precision scores. Both untrained embeddings and word2vec + generated features SemVec achieve nearly equal precision scores. Word2vec, on the other hand, achieved the worst precision scores in these experiments. It is clear also that domain specific features improve word2vec precision results because it improves the word representation which improves the classifier performance. Moreover, word2vec scores are lower than other approaches because word2vec word vector is considered as one unit. However, the filter in SemVec network model operates on each 2 dimensions in the word vector to extract features, which results in features without any meaning.

Figure 5.6 shows recall scores of the above-mentioned input methods on Twitter dataset. Untrained embeddings achieve better recall scores across all the epochs. SemVec achieves the second best recall scores in this experiment. Word2vec + generated features achieve better recall scores than word2vec. Word2vec on the other hand, achieved the worst precision scores in these experiments. It is clear also that generated features improve word2vec recall results because it improves the word representation which improves the classifier performance. Moreover, word2vec scores are lower than other approaches which is because word2vec word vector is considered as one unit. However, the filter in SemVec network model operates on every 2 dimensions in the word vector to extract features, which results in features without any meaning.

The lower SemVec recall scores are due to the fact that this data set is highly imbalanced (see section 5.1.1.1). The second reason for low recall scores is related to a set of issues such as the use of non-standard terms, short posts, spelling errors, etc. These issues with solutions are described in more details in section 5.3.9.

5.3.3 Comparison between SemVec and untrained embeddings on Twitter ADR Dataset on Kim Model

This section shows the experimental results of two different methods for generating input layer using one convolution layer CNN. We note that our method SemVec achieves the best precision results.

Figure 5.7 shows accuracy scores of the above-mentioned methods on the Twitter dataset. SemVec achieves the best accuracy scores after 300 epochs. Untrained embeddings input was able to achieve better accuracy results in the first 300 epochs then accuracy starts to decrease.

Compared to SemVec model results in the previous section, we notice that Kim model is clearly affecting accuracy scores for both untrained embeddings and SemVec. Also, it seems like untrained embeddings model starts to overfit after epoch 180 which does not happen with SemVec model. While SemVec continues to achieve better results but with slow convergence time compared to SemVec model in the previous section.

SemVec achieves better accuracy and precision scores as it uses domain specific features compared to randomly initialized numbers of untrained embeddings. Moreover, the drop in performance when using Kim model could be because it convolves over the whole word vector that represents each word. This results in representing the whole word vector with a limited number of features which causes a loss in some features compared to SemVec network model which has a bigger representation of features because of smaller filter sizes.

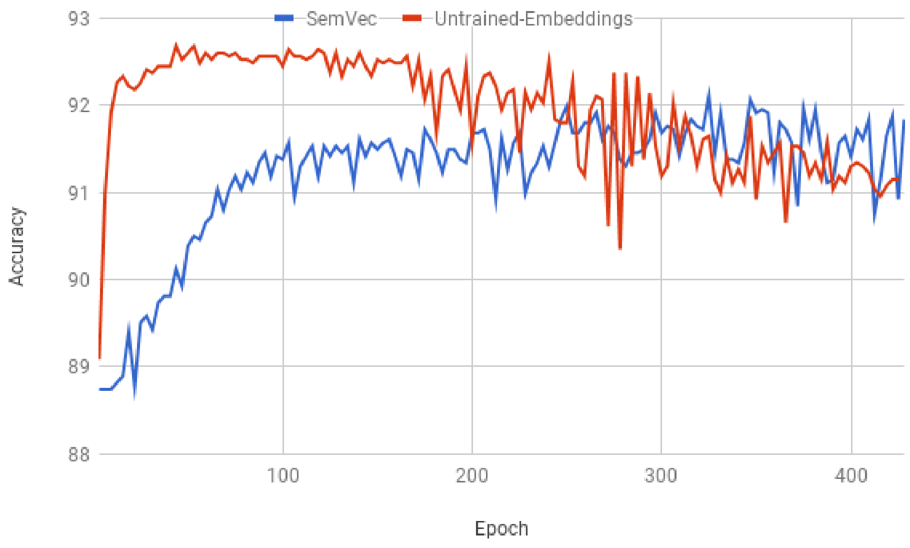


Figure 5.7: Kim Model Accuracy

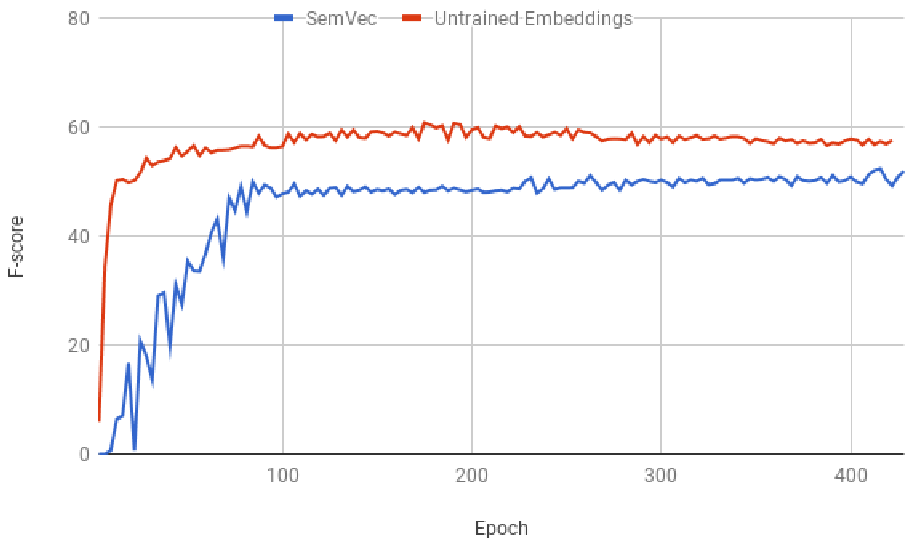


Figure 5.8: Kim Model ADR F-score

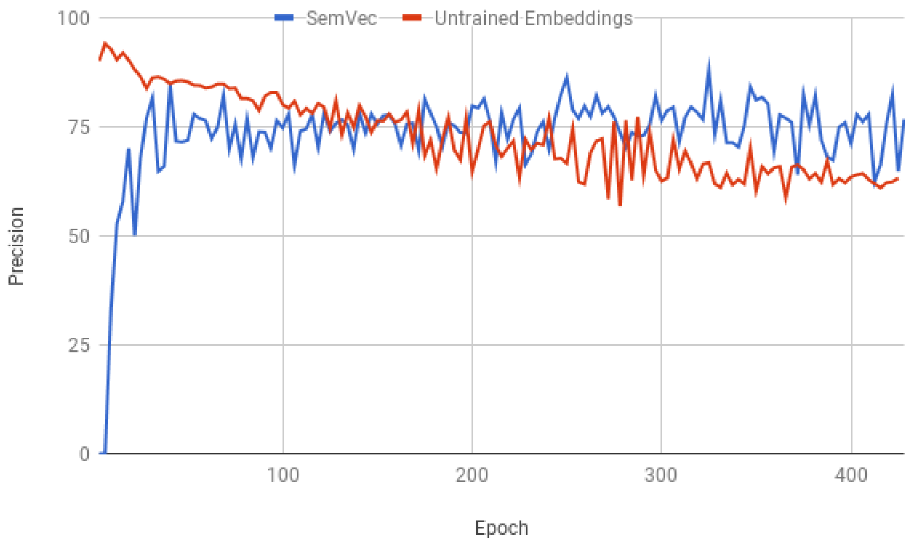


Figure 5.9: ADR Precision comparison

Figure 5.8 shows f-score scores of the above-mentioned input methods on the Twitter dataset. Untrained embeddings achieves the better accuracy scores and it is clear that it converges faster than SemVec.

SemVec low f-score scores are related to low recall scores. Low recall scores are related to dataset imbalance as described in previous sections and common errors such as spelling errors, short posts, acronyms, etc. as described in section 5.3.9.

Figure 5.9 shows precision scores of the above-mentioned input methods on the Twitter dataset. SemVec achieves the best precision scores after 200 epochs. After 200 epochs, SemVec precision is clearly better than untrained embeddings.

Figure 5.10 shows recall scores of the above-mentioned input methods on the Twitter dataset. Untrained embeddings achieve the best recall scores across all the epochs.

5.3.4 ADE Dataset Results

This section shows additional experimental results (outside the scope of this thesis) of SemVec on ADE dataset to evaluate its performance on other type datasets. ADE dataset contains sampling from MEDLINE case reports and so it is written by professionals. In addition, it does not contain data from social media. Compared to Twitter dataset the content of text segments are more likely to be different and post length is clearly bigger. ADE dataset also contains approximately twice more posts compared to the Twitter dataset [66].

Table 5.5 shows a comparison between SemVec and methods mentioned in section 5.2.5. These results, for instances containing ADR which we are interested in. In

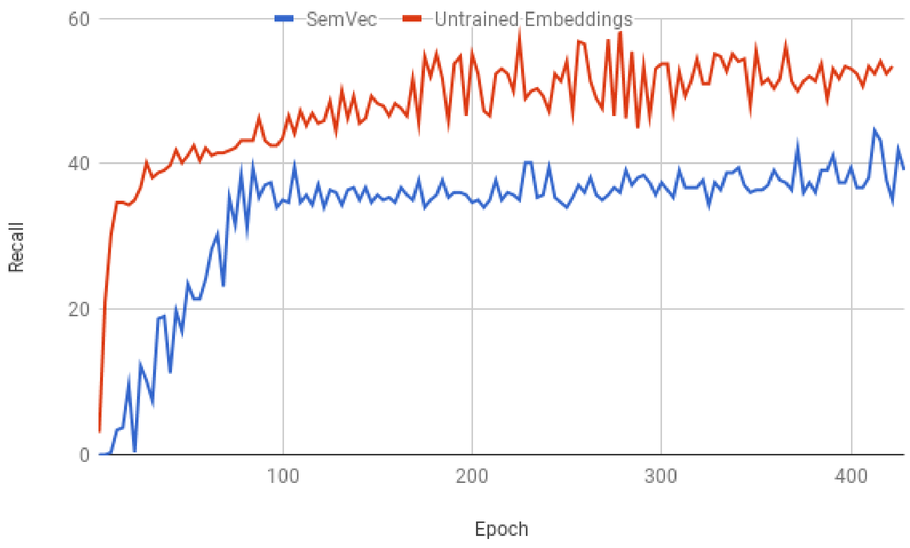


Figure 5.10: ADR Recall comparison

Method	ADR Precision	ADR Recall	ADR F-score
TM [28]	30	**99**	46
ME-TFIDF [28]	74	86	80
ME-NBLCR [28]	**91**	79	84
ME-WE [28]	48	70	57
CNN [28]	85	89	**87**
CRNN [28]	82	86	84
RCNN [28]	81	89	83
CNNA [28]	82	84	83
Multi-corpus [66]	N/A	N/A	81.2
SemVec	85.14	68.9	76.1

Table 5.5: ADE data set results comparison

	Team	Twitter 2015	Twitter 2015 sarcasm
7	IOA	62.62	65.77
4	INESC-ID	64.17	64.91
14	NLP	60.93	63.62
30	Sentibase	56.67	62.96
17	UIR-PKU	60.03	62.75
10	TwitterHawk	61.99	61.24
5	Splusplus	63.73	60.99
25	SWATAC	58.43	59.43
9	CLaC-SentiPipe	62	58.55
22	GTI	58.95	58.18
19	ECNU	59.72	57.74
41	whu-iss	24.8	57.73
21	SWASH	59.26	57.02
26	SWATCMW	57.6	56.69
27	WarwickDCS	57.32	56.58
16	Gradiant-Analytics	60.62	56.45
13	KLUEless	61.2	56.19
29	DIEGOLab	56.72	55.56
2	unitn	64.59	55.01
8	Swiss-Chocolate	62.61	54.66
20	CIS-positiv	59.57	54.3
1	Webis	64.84	53.59
37	**SemVec**	47.96	52.8
18	IIIT-H	59.83	52.67
11	SWATCS65	61.89	52.64
15	ZWJYYC	60.77	52.4
6	wxiaoac	63	52.22
24	elirf	58.58	50.66
35	RoseMerry	51.18	49.62
28	SeNTU	57.06	49.53
36	Frisbee	49.19	48.26
33	RGUSentimentMiners123	53.73	48.21
12	UNIBA	61.55	48.16
3	lsislif	64.27	46
23	iitpsemeval	58.8	43.91
32	UPF-taln	55.59	41.63
39	UDLAP2014	42.1	40.59
34	IHS-RD	52.65	36.02
38	UMDuluth-CS8761	47.77	34.4
40	SHELLFBK	32.45	25.73
31	Whu_Nlp	56.39	22.25

Table 5.6: SemEval 2015 competition results with SemVec [59]

this experiment, we used 13 features same as Twitter ADR dataset with no additional preprocessing steps. Compared to other methods, SemVec achieves the second best Precision scores the same as CNN method proposed by Huynh et al. [28].

ME-NBLCR achieves the highest ADR precision scores as it assigns a high weight to one feature, which is ADR class feature [81]. While SemVec achieves the second best precision score because we use the same preprocessing, same domain features, and same model hyper-parameters as Twitter ADR dataset although this dataset has a different nature. To improve SemVec precision and recall scores on this dataset additional features, different preprocessing steps and different model hyper-parameters should be used. CNN methods achieve high precision and recall scores as ADRs are composed of short text fragments; convolutions are enough to capture necessary ADR classification information [28]. This may suggest that removing some extra features from SemVec could potentially improve classification performance.

Moreover, SemVec F-score results lower than other methods due to lower recall scores for SemVec. The lower recall scores are due to the fact that this data set is highly imbalanced. Techniques like oversampling proposed by Hensman and Masko [45] can be used to fix dataset imbalance and improve classifier performance.

In this experiment, SemVec achieves accuracy score of 87.2% compared to 88.2% achieved by Sarker et al. [66]. Other above-mentioned methods do not report their accuracy score in their papers.

5.3.5 SemEval 2015 Dataset Results

This section shows the results of applying SemVec to SemEval 2015 Task 10 Subtask B message polarity classification dataset. In this subtask, 40 teams were competing, both newcomers and returning. All competing systems managed to beat the baseline except one system [59].

Table 5.6 show the test results of the teams competing in this subtask including SemEval results. The scores reported in this table are calculated as follows:

$$Score = \frac{FscorePositive + FscoreNegative}{2}$$

SemVec achieves the position 23 from 41 in Twitter 2015 sarcasm test and position 37 in Twitter 2015 dataset. In this experiment, SemVec used the following additional features:

- **NRC Emotion Lexicon Features:** this experiment uses eight basic emotions (surprise, sadness, trust, anger, fear, anticipation, joy, and disgust) and two sentiments (negative and positive)scores extracted from the NRC Emotion Lexicon. NRC Emotion Lexicon is a list of English words with their association with the above-mentioned emotions and sentiment. In this lexicon, each word has a score for each one of those emotions and scores [48].

- **Valence, Arousal, and Dominance Features:** SemVec use the three features from Warriner et al. [74] [74] Lexicon.

In this experiment, SemVec uses the same configuration as previous experiments except adding more features and changing epochs number and dropout. Other teams [1] used a set of different techniques -that can be used with SemVec- to improve their classifier performance such as:

- Use additional training set such as Sanders and SentiStrength twitter datasets, which are not provided by the task organizers.
- Use extra preprocessing steps to expand acronyms and emotions using special dictionaries.
- Use extra lexicons such AFFIN, Liu Bing Lexicon and others which SemVec did not use.
- Use additional features such as ngrams, trigrams, and chargrams.

Moreover, taking into consideration that some teams also rejoin this competition for the third time. It is clear that some teams optimized their methods more than SemVec for this special task. In addition to that, we believe that adding additional preprocessing with extra features can clearly improve SemVec score further.

5.3.6 Grid Search

To determine the best hyperparameters for our model, several experiments were performed. Table 5.7 shows part of these experiments. The effect of changing the optimizer, hidden dimensions, batch size, epochs, dropout, filters, kernel size and pool size on model accuracy and loss were studied. The settings that maximize model performance were used in later experiments. For a complete list of experiments can be found at appendix B.1.

5.3.7 Adding features Effect

Several classifications experiments were performed to investigate the real effect of adding features on model classification performance. However, we do not study the best combination of features as this is outside the scope of this thesis. So, we start adding features one by one to the model and track the changes in classification performance. As shown in figure 5.11, the results show that every feature added improves model accuracy. It is clear that some features have a bigger impact on the accuracy than others. It can be observed that adding the fourth feature causes a big jump in model accuracy. There is also a small jump after adding the fifth feature but adding the sixth, seventh, eighth and ninth feature has a small effect. There is also another jump when adding the tenth, eleventh, twelfth and thirteenth feature. It is also worth noting that the biggest jump in model performance when adding a new feature happens in the first 100 steps.

As shown in figure 5.12, the same as accuracy, it is clear that some features have a bigger impact on the model f-score than others. It can be observed that adding

[1] https://github.com/xiaowh7/SemEval-2015-Task-10

Accuracy	Loss	Epochs	Optimizer	Hidden Dimensions	Batch Size
89.81	0.239381	300	Adam	128	128
89.78	0.239441	300	Adam	128	64
89.78	0.238454	300	Adam	256	64
89.74	0.237184	200	Adam	256	128
89.7	0.238626	300	Adam	256	128
89.55	0.239334	300	Adam	128	256
89.53	0.236153	200	Adam	256	256
89.5	0.235099	300	Adam	256	256
89.45	0.238668	200	Adam	128	64
89.45	0.236627	200	Adam	128	128
89.44	0.231163	200	Adam	256	64
89.32	0.232866	200	Adam	64	128
89.27	0.236982	300	Adam	64	256
89.25	0.239289	300	Adam	64	64
89.25	0.239787	100	Adam	256	64
89.17	0.237499	200	Adam	128	256
89.08	0.239252	100	Adam	256	128
89.05	0.240378	100	Adam	256	256
88.95	0.23616	300	Adam	64	128
88.88	0.237877	200	Adam	64	64
88.88	0.236926	200	Adam	64	256
88.61	0.254083	100	Adam	128	256
88.53	0.252873	100	Adam	128	64
88.44	0.237259	300	RMSprop	128	256
88.13	0.298497	300	RMSprop	128	64
88	0.253993	100	Adam	128	128
87.9	0.236279	200	RMSprop	128	256
87.83	0.24536	300	RMSprop	128	256
87.83	0.261661	100	Adam	64	128
87.81	0.236815	200	RMSprop	128	256
87.72	0.26464	100	Adam	64	64
87.39	0.28228	100	Adam	64	256
87.35	0.293081	100	RMSprop	64	256
87.3	0.276044	300	RMSprop	64	256
87.22	0.293348	300	RMSprop	64	64
87.21	0.294215	300	RMSprop	128	64
87.2	0.284198	100	RMSprop	128	256
87.16	0.283954	100	RMSprop	128	256
87.06	0.293875	100	RMSprop	64	64
87.03	0.293208	200	RMSprop	128	64
87.01	0.2932	200	RMSprop	64	64

Table 5.7: Grid search results

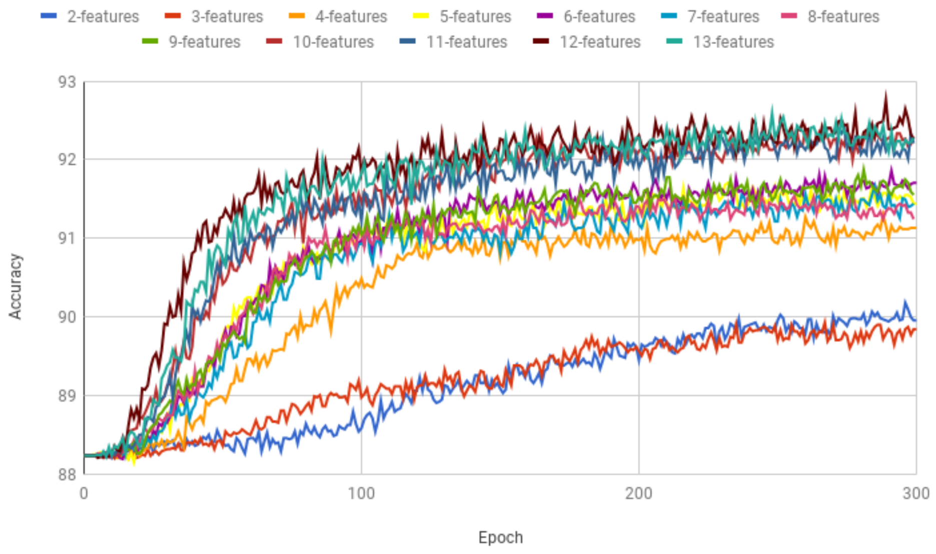


Figure 5.11: Adding features effect on model accuracy

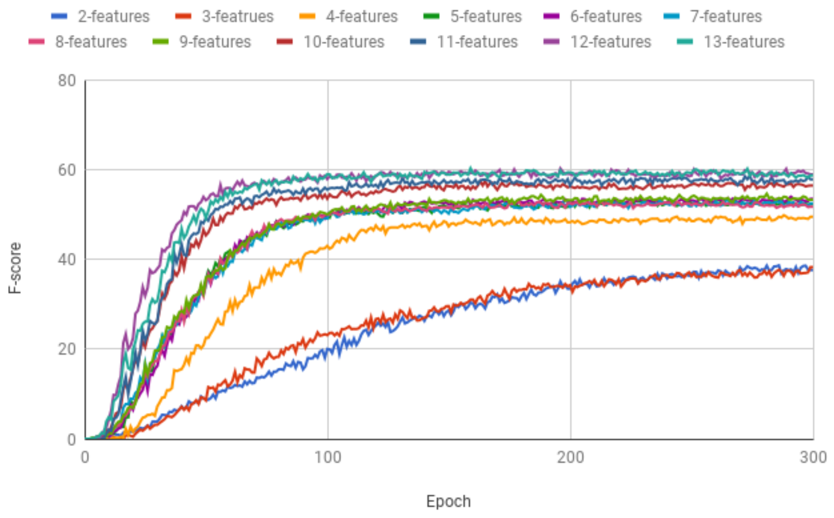


Figure 5.12: Adding features effect on model ADR F-score

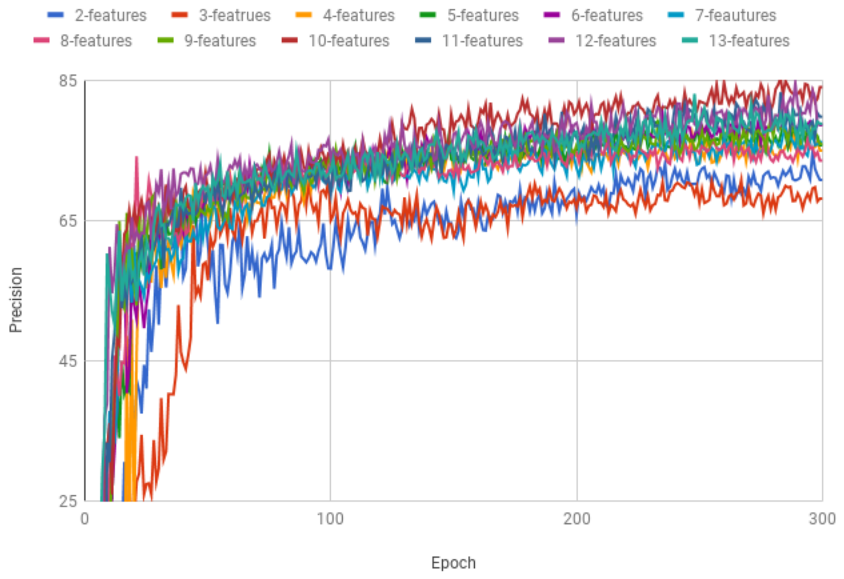


Figure 5.13: Adding features effect on model ADR precision

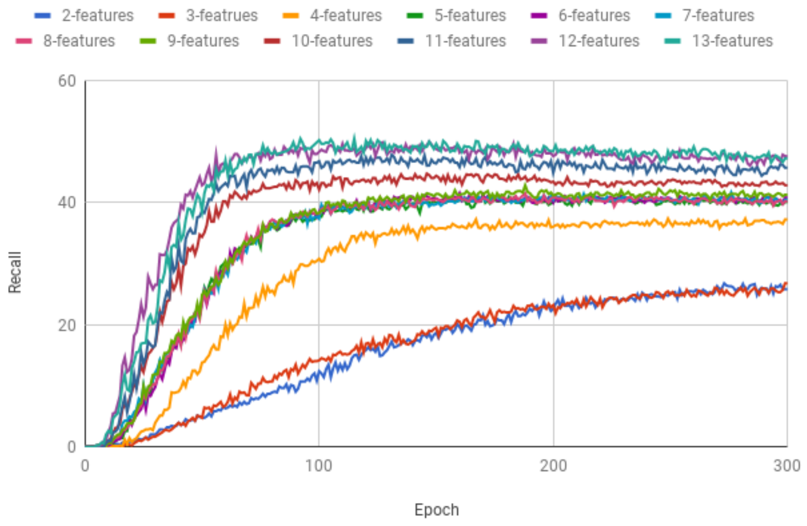


Figure 5.14: Adding features effect on model ADR recall

the fourth feature (Positive SentiWordNet feature) causes a big jump in model f-score. There is also a small jump after adding the fifth (Negative SentiWordNet) and seventh (Less Bad Word) features. The same effect of adding these features has the same effect on model recall. Positive, Negative SentiWordNet, and Less Bad Word features are language related features which as expected most of the posts contain some of these words which makes them more relevant to be returned. There is also another jump when adding the tenth(Word Length) feature. It is also worth noting that the biggest jump in model performance when adding a new feature happens in the first 100 steps.

Moreover, adding more features has the same effect on model recall as f-score and accuracy as shown in figure 5.14. It can be observed that adding the fourth feature (Positive SentiWordNet feature) causes a big jump in model Recall. There is also a small jump after adding the fifth (Negative SentiWordNet), and seventh (Less Bad Word). There is also another jump when adding the tenth feature (Word Length). It is also worth noting that the biggest jump in model recall when adding a new feature happens in the first 100 steps. Moreover, It can be observed that Word Length feature has a significant impact on ADR F-score. After investigating word length in both classes, it is clear that average word length for positive ADR is 4 while it is 5 for negative ADR instances. This may suggest that users tend to use shorter terms when discussing ADRs. The effect of adding features on model precision is different than other metrics. As shown in figure 5.13, the biggest jump happens after adding the fourth feature (Positive SentiWordNet feature). On the other hand, other jumps are small and cannot be observed and we can see that the model with 10 features is able to achieve a higher precision after the hundredth epoch.

Interestingly, adding more features make model converge faster. This means that model can achieve higher scores in less number of epochs. This is clear in accuracy, f-score, and recall performance metrics.

5.3.8 Contribution of features

Several leave-one-out classifications experiments were performed to investigate the real contribution of each feature on model classification performance. As shown in figure 5.15, the results show that every feature improves ADR F-scores and that ADR F-score drops when any feature is removed. Moreover, It can be observed that removing Word Length feature has a significant impact on ADR F-score. After investigating word length in both classes, it is clear that average word length for positive ADR is 4 while it is 5 for negative ADR instances. This may suggest that users tend to use shorter terms when discussing ADRs. Similarly, both ADR lexicon and word cluster [53] features have a clear impact on improving ADR F-score. ADR feature is important for twitter ADR dataset specifically because both positive and negative dataset classes have a set of tweets with ADR mentions. Based on that ADR feature improves recall significantly but on the other hand, it lowers precision. On the other hand, the removal of word clusters feature has clearly lowered model ADR precision and ADR recall. This is due to the fact that hierarchical word clusters were obtained via Brown clustering [13] on a huge set of unlabeled tweets which produces a base set of 1,000 clusters with in-

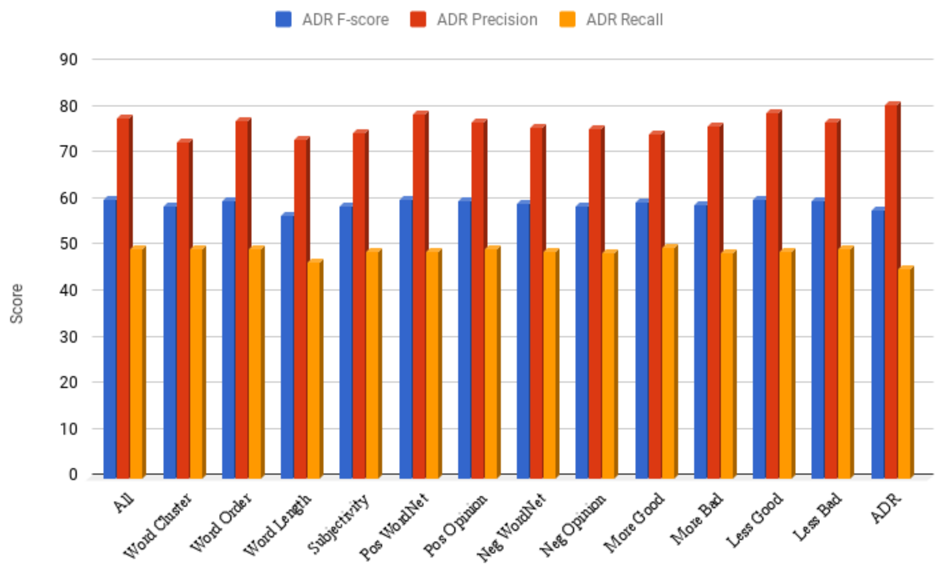


Figure 5.15: Leave-one-out classification scores

cludes for example clusters for happiness and sadness words and emotions which has a clear impact on our model performance.

Importantly, it is clear that the best score is achieved by the combination of all the features and the removal of any feature clearly lowers the model performance. This supports our initial hypothesis that adding high-quality features improves the performance of CNN. The complete results of these experiments are presented in Table 5.8.

5.3.9 Error Analysis

This section shows the leading causes of SemVec classification errors. Furthermore, it presents possible techniques that can be applied to improve SemVec classification performance. The common causes of classification errors are as follows:

- **Non-standard terms/use of English:** many ADRs are written using user's language and not in standard ADR terminology. Those instances are usually unique to specific posts, and they are not repeated[66]. A solution to this problem is to have a larger dataset with much larger ADR instances which can solve this problem.

- **Short posts:** some instances in twitter datasets contains very few terms. This makes it hard to generate a rich feature set for these instances[66]. A solution to this problem is to expand terms with their synonyms and acronym or emotion equivalent words. SemVec did not employ this solution so implementing it may improve SemVec performance.

Features	Accuracy		ADR F-score		ADR Precision		ADR Recall	
	Mean (%)	SD (±)	Mean (%)	SD (±)	Mean (%)	SD (±)	Mean (%)	SD (±)
All	92.38	0.43	**60.48**	2.41	77.93	4.99	49.68	3.65
Word Cluster	91.84	0.72	58.84	3.98	**72.72**	5.71	49.78	5.34
Word Order	92.31	0.58	60.16	3.27	77.56	6.22	49.56	4.67
Word Length	**91.74**	0.75	**56.87**	5.1	73.51	4.73	46.67	6.23
Subjectivity	92.03	0.24	59.06	1.99	74.76	3.01	49.03	3.54
Pos WordNet	92.43	0.47	60.33	3.18	78.95	4.88	49.13	4.43
Pos Opinion	92.24	0.8	60.06	3.37	77.04	7.96	49.56	3.73
Neg WordNet	92.18	0.59	59.65	3.36	75.98	4.28	49.24	3.94
Neg Opinion	92.09	0.54	59.09	3.64	75.68	4.89	48.81	5.17
More Good	92.12	0.43	59.86	2.73	74.7	3.36	**50.1**	3.65
More Bad	92.14	0.44	59.27	3.26	76.39	4.89	48.92	5.56
Less Good	**92.46**	0.53	60.4	3.08	79.22	5.04	49.03	3.91
Less Bad	92.31	0.5	60.23	3.3	77.06	4.71	49.78	4.85
ADR	92.32	0.83	58.09	4.92	**80.94**	5.8	**45.39**	4.57

Table 5.8: Leave-one-out classification scores showing how accuracies, ADR F-scores, ADR precisions and ADR recalls are affected after removing one feature.

- **Spelling errors:** often social media posts contains misspellings[66]. We can reduce this problem by using a set of medical and nonmedical dictionaries to fix spelling errors.
- **Sarcastic statements/ambiguous statements:** in some posts, the users are very ambiguous and may use positive words to represent an ADR [66].
- **Emotion icons:** it is common in social media posts to contain emotion icons. Emotion icons can express user feelings that are not included in his written post. SemVec strip emotion icons from posts. Parsing emotion icons may improve SemVec classification performance.
- **Acronyms:** often users in twitter posts use a set of acronyms to reduce their tweet size. SemVec ignores acronyms. However, expanding acronyms may improve the feature representation of post and improve SemVec accuracy, precision, and recall.
- **ADRs:** some of the negative instances has ADR text in it. This is maybe due to wrong annotation or the negative nature of the post.

The following describe the effects of these errors on our model:

- These errors are the main causes of SemVec classification errors.
- SemVec does not handle all these errors (i.e: spelling errors, scarcastic statements, emotion icons, and acronyms). It currently strip some of these information such as emotion icons. Handling these features correctly could clearly improve classification performance even further.

Usually, users use some or all of the above-mentioned issues in their posts. This makes classifying these posts very hard to classify even for human annotator.

5.3.10 Summary of results and discussions

This thesis proposes a new CNN network model for text classification. The proposed model is shown in section 4.7.2. Compared to Kim's [35] model for text classification this model uses a filter size of 2*2(by grid search) instead of full depth of the embedding dimension and random filter sizes. The use of smaller filter sizes increases the number of features extracted from each sentence which improves classification performance. Additionally, this model uses a fully connected layer of 128 neurons connected with the output. The proposed model clearly improves SemVec accuracy, precision, recall and as a result f-score. This is due to the fact that each feature in SemVec is independent, unlike word2vec which represents each word as one related feature.

Compared to other methods, SemVec achieves the best Accuracy and Precision scores as shown in table 5.4 due to the use of domain specific features, which are tailored to this domain and post type. Features such as ADR lexicon, word clusters, and opinion words have a clear effect on improving precision as shown in section 5.3.8 and 5.3.7. Apart from that, the new CNN network architecture that is proposed by this thesis is clearly affecting the model classification performance.

To evaluate the effect of SemVec proposed CNN network model, experiments were conducted to compare its classification results with Kim model using the same features input layer as shown in section 5.3.3. Figures 5.7, 5.9,5.8, and 5.10 show classification performance results for SemVec and Untrained embedding methods using Kim model. While SemVec achieves better accuracy and precision results, there is a drop in both methods performance when using Kim model. In addition, the convergence time for both methods become bigger and Untrained embeddings seems to overfit after 200 epochs.

The drop in performance when using Kim model could be because it convolved over the whole features that represent each word. This results in representing the whole word vector with a limited number of features which causes a loss in some features. On the other hand, SemVec network model has a bigger representation of features because of smaller filter sizes.

SemVec improves model convergence time. As shown in figure 5.3 and figure 5.5, SemVec was able to achieve the best results under 50 epochs. This is also related to the fact that SemVec uses a set of semantic features which improves network learning and model accuracy results.

SemVec does not achieve the best F-score results in Twitter and ADE datasets due to low recall scores. Recall scores are due to the fact that both data sets are highly imbalanced for example: in Twitter dataset number of ADR instances is 902 compared to 6,672 non-ADR instances. To improve SemVec recall scores we suggest using techniques like oversampling proposed by Hensman and Masko [45] which can fix dataset imbalance and improve classifier performance. Moreover, issues such as short posts, spelling errors and acronyms has an impact on recall and precision scores as described in section 5.3.9.

Every feature we experiment has an effect on SemVec performance as shown in section 5.3.8. Although some features have smaller impact than other features, the best score was achieved by the combination of all features as shown in table 5.8. We notice that language features (i.e:negative and positive words), domain specific features (i.e: ADR lexicon), and word length has more impact compared to other features on Twitter ADR dataset. We can also conclude that adding more high-quality features can improve SemVec performance further.

In the Twitter dataset, ADR lexicon feature, language features(i.e:negative and positive words), and word length has bigger impact on both recall and precision compared to other features as shown in table 5.8.

Finally, adding more domain specific features to SemVec improves convergence time and has a clear impact on all performance metrics as discussed in section 5.3.7. Figures 5.11, 5.12, 5.13 and 5.14 shows how adding more features clearly affect different model metrics and how some features such as Sentiment features, language, and domain features have more impact than other features.

Chapter 6

Conclusion

This section concludes our thesis. We represent a brief literature review, our model architecture, results, limitations and future work.

6.1 Introduction

This thesis has conducted a literature review of research done on ADR detection in tweets. The work can be categorized into three categories: text mining based methods, machine learning based methods and deep learning based methods.

Recently, several researchers have been using deep learning for various NLP tasks from relation extraction to sentiment analysis and POS tagging. Due to promising results that deep learning can achieve, this thesis investigates the use of deep learning in ADR classification.

Chapter 4 proposed a new approach for text classification which uses a set of engineered features as an input to CNN. The proposed method achieves better accuracy and precision results compared to other state of the art methods on Twitter ADR data set.

6.2 Contributions

The main contributions of this thesis, the development of new word representation layer that can be used as the input layer for CNN (Chapter 4). These contributions are summarized as follows:

- **SemVec** which is a new word representation layer that can be used as the input layer for CNN. SemVec extracts semantic and domain-specific features from posts and represents each word in this sentence as a vector of features. In SemVec, we combine a set of domain-specific features such as ADR lexicon and syntactic and semantic features. Some of these features are generic and can be used for other data sets without the need for any modifications. Domain-specific features can be added or removed based on the dataset and its domain.

- This thesis proposes a **new CNN model** for text classification. The proposed model is shown in section 4.7.2. Compared to Kim's [35] model, this model uses a filter size of 2*2 instead of full depth of the embedding dimension and random filter sizes. Additionally, this model uses a fully connected layer of 128 neurons connected with the output. The proposed model clearly improves SemVec accuracy, precision, recall, and as a result f-score. This is due to the fact that each feature in SemVec is independent, unlike word2vec which represents each word as one related feature.
- SemVec improves Twitter dataset classification accuracy and precision and achieves the best precision score in ADE data set. Moreover, it is able to improve model learning and minimize model convergence time.

6.3 Discussion

SemVec is evaluated by comparing it to the different state of the art methods on Twitter ADR data set as shown in chapter 5. Our work was implemented in python using Keras library and Tensorflow as backend. Results show that SemVec achieves better accuracy and precision score on the Twitter dataset and the best precision score on ADE dataset.

- SemVec achieves the best accuracy and precision results on Twitter data set.
- SemVec improves network learning and network convergence time. SemVec achieves the best results under 50 epochs.
- SemVec did not achieve the best F-score results in Twitter and ADE datasets due to low recall scores.
- Adding more quality features to SemVec improves convergence time and has a clear impact on all performance metrics. Although some features have a smaller impact than other features, the best score was achieved by the combination of all features.
- Domain-specific features have a bigger impact on model performance than other features. In SemVec ADR lexicon feature has a clear impact on both recall and precision.

6.4 Limitations and Assumptions

Three different data sets were selected to evaluate the performance of SemVec. Although these datasets are different in many characteristics such as the number of instances, post length and post nature, it is impossible to cover all the different data set types. This section covers some of the assumptions and limitations of our work:

- The maximum number of features that are used in the experiments is 13 in both Twitter and ADE datasets and 28 features in SemEval dataset, thus we could not ascertain how SemVec would behave after adding more features.
- SemVec was applied to two different Twitter datasets and ADE data set which is a medical reporting data set. Similarly, we could not ascertain how SemVec would behave on different type datasets such as news posts or any other type.
- This thesis assumed all the datasets do not have missing data in Twitter ADR dataset.
- This thesis applied the same 13 features to the 3 datasets. However, we could not ascertain how these features would behave on different type data sets.
- This thesis applied SemVec to one multi-class dataset (SemEval 2015), thus we could not be certain how it will behave on different multi-class data sets.

6.5 Future work

This section represents the main areas for future works:

- This thesis applied SemVec to three datasets. As a future work, we need to expand the number of data sets and investigate how SemVec behaves on different datasets.
- This thesis applied SemVec to one data set with three classes. As a future work, we need to investigate the performance of SemVec on other multi-class data sets.
- SemVec approach achieves higher precision scores on twitter dataset. However, we notice that recall is relatively lower than other methods. As a future work, we need to investigate different techniques to improve SemVec recall.

Appendix A

Twitter Dataset Results

A.1 Accuracy

Epoch	Untrained embedding	SemVec	150 word2vec	162 word2vec + generated features
10	90.78305671	88.61548969	78.01424732	89.32859479
20	90.36317102	89.21683767	76.16857388	89.4965385
30	90.54517899	90.18197276	76.7694466	89.32838356
40	90.18158551	91.03517471	75.23146168	88.6846844
50	90.34926519	91.34284563	76.12602908	89.34278226
60	90.20948519	91.56655348	76.21057301	89.60831322
70	90.33560579	91.6084118	75.86095577	89.3567585
80	90.46116313	91.4687198	75.4411933	88.68540609
90	90.54517899	91.72039774	75.46919858	89.07693445
100	90.34944121	91.84609589	76.12685638	88.83905673
110	90.34954683	91.76216804	76.01513447	89.28657806
120	90.13956878	91.90215928	76.25299459	89.14672764
130	90.30744208	92.22378883	75.91694874	88.78315177
140	90.30733647	92.02776942	76.08476924	89.07689925
150	90.58731894	92.09784424	76.04273491	89.23063789
160	90.51734972	92.1817721	76.91012433	88.76924593
170	90.50344389	92.12583193	76.11244009	88.95104267
180	90.55929605	92.13989618	78.01452895	88.99312982
190	90.40548699	92.06987416	76.82609086	89.18883239
200	90.34944121	92.05593313	76.44860915	88.96496611
210	90.32141832	92.08376239	78.05724978	89.11872235
220	90.67103556	92.19573073	76.60273505	89.18862116
230	90.81095639	92.16772545	77.74927963	89.16068628
240	90.41951604	92.22384164	78.08451577	89.30044868
250	90.08385744	92.26566474	77.74885717	89.0066836
260	90.02795248	92.15390763	77.79127876	89.11882797
270	90.12576856	92.083868	77.98661168	89.06272938
280	90.08391025	92.13972016	78.3087517	89.00693003
290	90.36368149	92.30757586	77.79106753	88.96505412
300	90.33579942	92.30761107	77.58100147	89.10497494

A.2 F-score

Epoch	162 word2vec + generated features	150 word2vec	SemVec	Untrained embedding
0	0	0	3.757063228	0
10	36.08691485	11.50731278	16.90394119	60.07636713
20	38.62057143	12.92864034	32.37864634	59.83946655
30	39.00712496	12.57923325	42.90805756	59.74720184
40	38.95654905	14.11638402	52.64106121	59.71113185
50	39.03137199	12.70971267	55.6742273	60.31011433
60	40.11155476	12.88498807	57.15466265	60.36063605
70	42.91171807	13.16662549	58.05095869	60.28143651
80	38.94264165	12.67273784	57.73531679	60.61361665
90	41.7759147	12.55092777	58.14846881	60.90412211
100	37.91313511	12.27616682	58.640437	60.39396933
110	40.42979126	12.31500679	59.07941198	60.62670414
120	40.20111606	11.94296054	59.81472234	60.29784208
130	40.75125632	12.22741255	60.29531299	60.52802856
140	40.27509065	12.92518783	59.57294506	60.13176056
150	41.8863586	12.35721666	59.87270696	60.77726906
160	40.32585582	11.45431748	59.88396333	61.13903691
170	40.54944591	12.57003829	60.21688524	60.99389965
180	41.29300491	11.71891053	59.75713421	60.99102602
190	40.9368632	11.25320974	59.46256256	60.7454668
200	39.96839276	12.08553473	59.31928832	60.992843
210	39.98573979	11.1804115	59.48188129	61.0664052
220	41.48768636	11.23332971	59.46679546	61.28312672
230	41.44480659	11.46333485	59.62202805	61.65449687
240	42.03404725	11.37867481	59.68018748	60.66445493
250	40.49174897	10.79006439	59.40571842	60.07031645
260	40.44064809	10.74159137	59.55086772	59.8591561
270	41.91549087	11.11977699	58.77961406	59.5365819
280	38.51637008	10.78534828	59.26842343	59.82901372
290	39.46672432	10.57429663	59.6793328	60.58226827
300	41.5301324	11.12727709	59.88703549	60.27787266

A.3 Precision

Epoch	162 word2vec + generated features	150 word2vec	SemVec	Untrained embedding
0	0	0	40.30864198	0
10	62.28786539	10.90296026	61.60909023	61.77311411
20	61.74620803	11.41918775	62.85070485	59.58431011
30	60.90332332	11.26064106	68.25833453	60.27755169
40	57.73267858	11.96963676	70.40033653	58.59828608
50	60.94530841	11.17777189	70.82102425	59.17865671
60	62.48453457	11.44115232	71.39786751	58.27878902
70	59.61967108	11.53236776	70.9657358	58.6454329
80	53.51371516	10.96794347	70.59401104	59.43678822
90	56.66283033	10.87582533	72.15446393	59.77282467
100	56.88216433	10.8991704	72.98347404	58.74518309
110	59.90823456	10.88036403	71.50872801	58.73342921
120	57.08736259	10.72203892	72.15081272	58.21111404
130	53.94071472	10.74527481	76.03685187	58.53205822
140	56.13232928	11.41215914	73.86984357	58.68172847
150	57.64322441	10.89148156	74.51784002	59.89194486
160	54.19320884	10.4125849	75.72167199	59.33750384
170	55.37223492	11.06399534	74.34056824	59.25762009
180	55.78605931	11.10099773	75.454648	59.45421271
190	57.16872814	10.29735173	74.63983504	58.93263684
200	55.50686974	10.77524727	74.42640885	58.36006234
210	56.65240985	10.7646473	75.01470218	58.18699744
220	57.3481256	10.14333916	76.73963016	59.98183533
230	57.55993842	10.81053104	75.92381947	60.60589472
240	58.15852345	10.82471908	77.32000148	58.73550179
250	55.85849432	10.21549924	77.99154508	57.22802211
260	56.76048631	10.20633586	76.05203898	56.99164638
270	56.01381607	10.64483954	75.959091	57.5839
280	55.89158018	10.54198535	75.92732579	57.25184899
290	57.74664587	10.08658383	78.08181101	58.42051165
299	56.39280585	10.44935526	77.92575619	58.457856

A.4 Recall

Epoch	162 word2vec + generated features	150 word2vec	SemVec	Untrained embedding
0	0	0	2.022166298	0
10	26.39111314	12.2524721	9.874424871	58.96692849
20	28.52385673	15.21899387	22.12308396	60.87216249
30	29.13775134	14.3902997	31.51834058	59.67487735
40	31.52469559	17.35682147	42.55954651	61.57884034
50	29.60420956	14.86946796	46.24164315	62.0605506
60	29.85205521	14.98767127	47.90919952	63.37095503
70	34.48104934	15.57741681	49.44965556	62.29949922
80	31.28447597	15.22407789	49.56658786	62.18765093
90	33.63964514	15.22280688	48.97938433	62.42024454
100	29.36144793	14.26955439	49.21960396	62.42151554
110	31.0290043	14.6178093	50.5325504	63.0138031
120	31.63273088	13.90604743	51.35361855	63.25529373
130	33.04608658	14.3902997	50.29105976	63.01253209
140	31.73949516	15.21899387	50.04829813	61.9461603
150	33.18462594	14.50723201	50.29105976	62.06436361
160	32.57708635	13.07481126	49.69241719	63.36460002
170	32.10554412	14.62797733	50.76133099	63.12565139
180	33.18081293	12.59818501	49.8093495	62.76977046
190	31.98861181	12.596914	49.57040087	63.00490607
200	31.51579857	13.6645568	49.68860418	63.59592262
210	31.13958159	11.88261013	49.45473957	64.55171713
220	32.81095096	12.94898193	48.7404357	62.77231246
230	32.69401866	12.35923638	49.33399426	62.88543176
240	33.04481558	12.24230407	49.09631663	62.88670276
250	31.96954676	11.65001652	48.26380945	63.36332901
260	31.62002084	11.5254582	49.21451994	63.24639671
270	33.75657744	11.88642314	48.14433514	61.81524696
280	29.72495488	11.40344188	48.73789369	62.88543176
290	30.89681995	11.40471288	48.50021607	63.12183838
300	33.27868019	12.00716846	48.86499403	62.52192481

Appendix B

Grid Search Results

B.1 Hyper Parameters Results

Accuracy	SD	Activation	Pool Size	Filters	Dropout	Filter Size
86.57	15.45	sigmoid	2	64	0.7	2
86.49	15.39	sigmoid	2	32	0.7	2
85.84	14.93	sigmoid	2	32	0.7	4
85.81	14.95	sigmoid	3	32	0.7	3
85.67	14.81	sigmoid	4	64	0.7	2
85.58	14.75	sigmoid	5	64	0.7	3
85.55	14.73	sigmoid	5	64	0.7	2
85.53	14.71	sigmoid	3	64	0.7	2
85.53	14.73	sigmoid	3	64	0.7	3
85.52	14.71	sigmoid	5	32	0.7	2
85.51	14.7	sigmoid	2	64	0.7	4
85.47	14.67	sigmoid	2	64	0.5	2
85.46	14.67	sigmoid	2	64	0.7	3
85.33	14.57	sigmoid	2	32	0.5	5
85.33	14.57	sigmoid	2	64	0.5	3
85.31	14.55	sigmoid	2	64	0.5	4
85.3	14.56	sigmoid	2	32	0.5	2
85.27	14.53	sigmoid	2	32	0.7	3
85.26	14.55	sigmoid	2	32	0.5	4
85.19	14.49	sigmoid	4	32	0.7	2
85.09	14.41	sigmoid	2	32	0.5	3
85.06	14.4	sigmoid	4	64	0.7	3
85.04	14.37	sigmoid	4	32	0.7	4
84.98	14.33	sigmoid	4	64	0.7	4
84.94	14.3	sigmoid	3	32	0.7	2
84.85	14.23	sigmoid	4	32	0.7	3
84.8	14.24	sigmoid	3	64	0.7	4
84.77	14.2	sigmoid	2	64	0.7	5
84.75	14.16	sigmoid	5	32	0.7	5

Accuracy	SD	Activation	Pool Size	Filters	Dropout	Filter Size
84.72	14.14	sigmoid	5	32	0.7	3
84.68	14.11	sigmoid	5	32	0.7	4
84.64	14.09	sigmoid	5	64	0.7	5
84.6	14.07	sigmoid	2	64	0.5	5
84.36	13.89	sigmoid	4	64	0.7	5
84.31	13.85	sigmoid	5	64	0.7	4
84.25	13.81	sigmoid	3	32	0.7	5
84.23	13.79	sigmoid	3	64	0.7	5
84.14	13.73	sigmoid	3	32	0.7	4
84.07	13.68	sigmoid	2	32	0.7	5
84	13.64	sigmoid	4	32	0.7	5
11.76	16.63	softmax	2	32	0.5	2
11.76	16.63	softmax	2	32	0.5	3
11.76	16.63	softmax	2	32	0.5	4
11.76	16.63	softmax	2	32	0.5	5
11.76	16.63	softmax	2	64	0.5	2
11.76	16.63	softmax	2	64	0.5	3
11.76	16.63	softmax	2	64	0.5	4
11.76	16.63	softmax	2	64	0.5	5
11.76	16.63	softmax	4	32	0.7	2
11.76	16.63	softmax	4	32	0.7	3
11.76	16.63	softmax	4	32	0.7	4
11.76	16.63	softmax	4	32	0.7	5
11.76	16.63	softmax	4	64	0.7	2
11.76	16.63	softmax	4	64	0.7	3
11.76	16.63	softmax	4	64	0.7	4
11.76	16.63	softmax	4	64	0.7	5
11.76	16.63	softmax	3	32	0.7	2
11.76	16.63	softmax	3	32	0.7	3
11.76	16.63	softmax	3	32	0.7	4
11.76	16.63	softmax	3	32	0.7	5
11.76	16.63	softmax	3	64	0.7	2
11.76	16.63	softmax	3	64	0.7	3
11.76	16.63	softmax	3	64	0.7	4
11.76	16.63	softmax	3	64	0.7	5
11.76	16.63	softmax	5	32	0.7	2
11.76	16.63	softmax	5	32	0.7	3
11.76	16.63	softmax	5	32	0.7	4
11.76	16.63	softmax	5	32	0.7	5
11.76	16.63	softmax	5	64	0.7	2
11.76	16.63	softmax	5	64	0.7	3
11.76	16.63	softmax	5	64	0.7	4
11.76	16.63	softmax	5	64	0.7	5
11.76	16.63	softmax	2	32	0.7	2

Accuracy	SD	Activation	Pool Size	Filters	Dropout	Filter Size
11.76	16.63	softmax	2	32	0.7	3
11.76	16.63	softmax	2	32	0.7	4
11.76	16.63	softmax	2	32	0.7	5
11.76	16.63	softmax	2	64	0.7	2
11.76	16.63	softmax	2	64	0.7	3
11.76	16.63	softmax	2	64	0.7	4
11.76	16.63	softmax	2	64	0.7	5
84.77	14.2	sigmoid	2	64	0.7	5
84.75	14.16	sigmoid	5	32	0.7	5
84.72	14.14	sigmoid	5	32	0.7	3
84.68	14.11	sigmoid	5	32	0.7	4
84.64	14.09	sigmoid	5	64	0.7	5
84.6	14.07	sigmoid	2	64	0.5	5
84.36	13.89	sigmoid	4	64	0.7	5
84.31	13.85	sigmoid	5	64	0.7	4
84.25	13.81	sigmoid	3	32	0.7	5
84.23	13.79	sigmoid	3	64	0.7	5
84.14	13.73	sigmoid	3	32	0.7	4
84.07	13.68	sigmoid	2	32	0.7	5
84	13.64	sigmoid	4	32	0.7	5
11.76	16.63	softmax	2	32	0.5	2
11.76	16.63	softmax	2	32	0.5	3
11.76	16.63	softmax	2	32	0.5	4
11.76	16.63	softmax	2	32	0.5	5
11.76	16.63	softmax	2	64	0.5	2
11.76	16.63	softmax	2	64	0.5	3
11.76	16.63	softmax	2	64	0.5	4
11.76	16.63	softmax	2	64	0.5	5
11.76	16.63	softmax	4	32	0.7	2
11.76	16.63	softmax	4	32	0.7	3
11.76	16.63	softmax	4	32	0.7	4
11.76	16.63	softmax	4	32	0.7	5
11.76	16.63	softmax	4	64	0.7	2
11.76	16.63	softmax	4	64	0.7	3
11.76	16.63	softmax	4	64	0.7	4
11.76	16.63	softmax	4	64	0.7	5
11.76	16.63	softmax	3	32	0.7	2
11.76	16.63	softmax	3	32	0.7	3
11.76	16.63	softmax	3	32	0.7	4
11.76	16.63	softmax	3	32	0.7	5
11.76	16.63	softmax	3	64	0.7	2
11.76	16.63	softmax	3	64	0.7	3
11.76	16.63	softmax	3	64	0.7	4
11.76	16.63	softmax	3	64	0.7	5

Accuracy	SD	Activation	Pool Size	Filters	Dropout	Filter Size
11.76	16.63	softmax	5	32	0.7	2
11.76	16.63	softmax	5	32	0.7	3
11.76	16.63	softmax	5	32	0.7	4
11.76	16.63	softmax	5	32	0.7	5
11.76	16.63	softmax	5	64	0.7	2
11.76	16.63	softmax	5	64	0.7	3
11.76	16.63	softmax	5	64	0.7	4
11.76	16.63	softmax	5	64	0.7	5
11.76	16.63	softmax	2	32	0.7	2
11.76	16.63	softmax	2	32	0.7	3
11.76	16.63	softmax	2	32	0.7	4
11.76	16.63	softmax	2	32	0.7	5
11.76	16.63	softmax	2	64	0.7	2
11.76	16.63	softmax	2	64	0.7	3
11.76	16.63	softmax	2	64	0.7	4
11.76	16.63	softmax	2	64	0.7	5
11.76	16.63	softmax	3	32	0.7	4
11.76	16.63	softmax	3	32	0.7	5
11.76	16.63	softmax	3	64	0.7	2
11.76	16.63	softmax	3	64	0.7	3
11.76	16.63	softmax	3	64	0.7	4
11.76	16.63	softmax	3	64	0.7	5
11.76	16.63	softmax	5	32	0.7	2
11.76	16.63	softmax	5	32	0.7	3
11.76	16.63	softmax	5	32	0.7	4
11.76	16.63	softmax	5	32	0.7	5
11.76	16.63	softmax	5	64	0.7	2
11.76	16.63	softmax	5	64	0.7	3
11.76	16.63	softmax	5	64	0.7	4
11.76	16.63	softmax	5	64	0.7	5
11.76	16.63	softmax	2	32	0.7	2
11.76	16.63	softmax	2	32	0.7	3
11.76	16.63	softmax	2	32	0.7	4
11.76	16.63	softmax	2	32	0.7	5
11.76	16.63	softmax	2	64	0.7	2
11.76	16.63	softmax	2	64	0.7	3
11.76	16.63	softmax	2	64	0.7	4
11.76	16.63	softmax	2	64	0.7	5

Bibliography

[1] Glove: Global vectors for word representation = http://nlp.stanford.edu/projects/glove/, note = Accessed: 2016-08-29

[2] May 12, 2011 the social life of health information, 2011. http://www.pewinternet.org/2011/05/12/the-social-life-of-health-information-2011/, accessed: 2016-08-21

[3] Metamap = http://metamap.nlm.nih.gov/, note = Accessed: 2016-08-24

[4] Opennlp = http://opennlp.apache.org/, note = Accessed: 2016-08-24

[5] Unified medical language system (umls) = http://www.nlm.nih.gov/research/umls/, note = Accessed: 2016-08-24

[6] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), http://tensorflow.org/, software available from tensorflow.org

[7] Agatonovic-Kustrin, S., Beresford, R.: Basic concepts of artificial neural network (ann) modeling and its application in pharmaceutical research. Journal of pharmaceutical and biomedical analysis 22(5), 717–727 (2000)

[8] Akhtyamova, L., Alexandrov, M., Cardiff, J.: Adverse drug extraction in twitter data using convolutional neural network. In: Database and Expert Systems Applications (DEXA), 2017 28th International Workshop on. pp. 88–92. IEEE (2017)

[9] Alpaydin, E.: Introduction to machine learning. MIT press (2014)

[10] Baccianella, S., Esuli, A., Sebastiani, F.: Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In: LREC. vol. 10, pp. 2200–2204 (2010)

[11] Bird, S.: Nltk: the natural language toolkit. In: Proceedings of the COLING/ACL on Interactive presentation sessions. pp. 69–72. Association for Computational Linguistics (2006)

[12] Bland, J.M., Altman, D.G.: Statistics notes: measurement error. Bmj 313(7059), 744 (1996)

[13] Brown, P.F., Desouza, P.V., Mercer, R.L., Pietra, V.J.D., Lai, J.C.: Class-based n-gram models of natural language. Computational linguistics 18(4), 467–479 (1992)

[14] Brownlee, J.: Master Machine Learning Algorithms: discover how they work and implement them from scratch. Jason Brownlee (2016)

[15] Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H., Rosen, D.B.: Fuzzy artmap: A neural network architecture for incremental supervised learning of analog multidimensional maps. IEEE Transactions on neural networks 3(5), 698–713 (1992)

[16] Cherry, C., Guo, H.: The unreasonable effectiveness of word representations for twitter named entity recognition. In: Proc. NAACL (2015)

[17] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. Journal of Machine Learning Research 12(Aug), 2493–2537 (2011)

[18] Dean, R.B., Dixon, W.: Simplified statistics for small numbers of observations. Analytical Chemistry 23(4), 636–638 (1951)

[19] Garla, V.N., Brandt, C.: Ontology-guided feature engineering for clinical text classification. Journal of biomedical informatics 45(5), 992–998 (2012)

[20] Gollapudi, S.: Practical Machine Learning. Packt Publishing Ltd (2016)

[21] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016), `http://www.deeplearningbook.org`

[22] Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., Lew, M.S.: Deep learning for visual understanding: A review. Neurocomputing 187, 27–48 (2016)

[23] Gurulingappa, H., Mateen-Rajpu, A., Toldo, L.: Extraction of potential adverse drug events from medical case reports. Journal of biomedical semantics 3(1), 1 (2012)

[24] Harpaz, R., DuMouchel, W., LePendu, P., Bauer-Mehren, A., Ryan, P., Shah, N.H.: Performance of pharmacovigilance signal-detection algorithms for the fda adverse event reporting system. Clinical Pharmacology & Therapeutics 93(6), 539–546 (2013)

[25] Harpaz, R., DuMouchel, W., Shah, N.H., Madigan, D., Ryan, P., Friedman, C.: Novel data-mining methodologies for adverse drug event discovery and analysis. Clinical Pharmacology & Therapeutics 91(6), 1010–1021 (2012)

[26] Hepple, M.: Independence and commitment: Assumptions for rapid training and execution of rule-based pos taggers. In: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics. pp. 278–277. Association for Computational Linguistics (2000)

[27] Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 168–177. ACM (2004)

[28] Huynh, T., He, Y., Willis, A., Rüger, S.: Adverse drug reaction classification with deep neural networks. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. pp. 877–887 (2016)

[29] Japkowicz, N.: The class imbalance problem: Significance and strategies. In: Proc. of the Int'l Conf. on Artificial Intelligence (2000)

[30] Ji, Y., Ying, H., Dews, P., Mansour, A., Tran, J., Miller, R.E., Massanari, R.M.: A potential causal association mining algorithm for screening adverse drug reactions in postmarketing surveillance. IEEE Transactions on Information Technology in Biomedicine 15(3), 428–437 (2011)

[31] Jiang, K., Zheng, Y.: Mining twitter data for potential drug effects. In: International Conference on Advanced Data Mining and Applications. pp. 434–443. Springer (2013)

[32] Johnson, R., Zhang, T.: Effective use of word order for text categorization with convolutional neural networks. arXiv preprint arXiv:1412.1058 (2014)

[33] Jonnagaddala, J., Jue, T.R., Dai, H.: Binary classification of twitter posts for adverse drug reactions. In: Proceedings of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing. pp. 4–8 (2016)

[34] Karimi, S., Metke-Jimenez, A., Kemp, M., Wang, C.: Cadec: A corpus of adverse drug event annotations. Journal of biomedical informatics 55, 73–81 (2015)

[35] Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)

[36] Kwak, H., Lee, C., Park, H., Moon, S.: What is twitter, a social network or a news media? In: Proceedings of the 19th international conference on World wide web. pp. 591–600. ACM (2010)

[37] Lazarou, J., Pomeranz, B.H., Corey, P.N.: Incidence of adverse drug reactions in hospitalized patients: a meta-analysis of prospective studies. Jama 279(15), 1200–1205 (1998)

[38] Leaman, R., Wojtulewicz, L., Sullivan, R., Skariah, A., Yang, J., Gonzalez, G.: Towards internet-age pharmacovigilance: extracting adverse drug reactions from user posts to health-related social networks. In: Proceedings of the 2010 workshop on biomedical natural language processing. pp. 117–125. Association for Computational Linguistics (2010)

[39] LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature 521(7553), 436–444 (2015)

[40] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)

[41] Lee, K., Qadir, A., Hasan, S.A., Datla, V., Prakash, A., Liu, J., Farri, O.: Adverse drug event detection in tweets with semi-supervised convolutional neural networks. In: Proceedings of the 26th International Conference on World Wide Web. pp. 705–714. International World Wide Web Conferences Steering Committee (2017)

[42] Li, J., Zhang, Z., Li, X., Chen, H.: Kernel-based learning for biomedical relation extraction. Journal of the American Society for Information Science and Technology 59(5), 756–769 (2008)

[43] Liu, X., Chen, H.: Azdrugminer: an information extraction system for mining patient-reported adverse drug events in online patient forums. In: International Conference on Smart Health. pp. 134–150. Springer (2013)

[44] Maimon, O., Rokach, L.: Introduction to knowledge discovery and data mining. In: Data Mining and Knowledge Discovery Handbook, pp. 1–15. Springer (2009)

[45] Masko, D., Hensman, P.: The impact of imbalanced training data for convolutional neural networks (2015)

[46] McClellan, M.: Drug safety reform at the fda—pendulum swing or systematic improvement? New England Journal of Medicine 356(17), 1700–1702 (2007)

[47] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)

[48] Mohammad, S.M., Turney, P.D.: Crowdsourcing a word-emotion association lexicon 29(3), 436–465 (2013)

[49] Nikfarjam, A., Sarker, A., O'Connor, K., Ginn, R., Gonzalez, G.: Pharmacovigilance from social media: mining adverse drug reaction mentions using sequence labeling with word embedding cluster features. Journal of the American Medical Informatics Association 22(3), 671–681 (2015)

[50] Niu, Y., Zhu, X., Li, J., Hirst, G.: Analysis of polarity information in medical text. In: AMIA annual symposium proceedings. vol. 2005, p. 570. American Medical Informatics Association (2005)

[51] Okazaki, N.: Crfsuite: a fast implementation of conditional random fields (crfs) (2007), http://www.chokkan.org/software/crfsuite/

[52] Olson, D.L., Delen, D.: Advanced data mining techniques. Springer Science & Business Media (2008)

[53] Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., Smith, N.A.: Improved part-of-speech tagging for online conversational text with word clusters. Association for Computational Linguistics (2013)

[54] Patki, A., Sarker, A., Pimpalkhute, P., Nikfarjam, A., Ginn, R., O'Connor, K., Smith, K., Gonzalez, G.: Mining adverse drug reaction signals from social media: going beyond extraction. Proceedings of BioLinkSig 2014, 1–8 (2014)

[55] Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: EMNLP. vol. 14, pp. 1532–1543 (2014)

[56] Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)

[57] Prusa, J.D., Khoshgoftaar, T.M.: Improving deep neural network design with new text data representations. Journal of Big Data 4(1), 7 (2017)

[58] Rastegar-Mojarad, M., Elayavilli, R.K., Yu, Y., Liu, H.: Detecting signals in noisy data-can ensemble classifiers help identify adverse drug reaction in tweets. In: Proceedings of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing (2016)

[59] Rosenthal, S., Nakov, P., Kiritchenko, S., Mohammad, S., Ritter, A., Stoyanov, V.: Semeval-2015 task 10: Sentiment analysis in twitter. In: SemEval@ NAACL-HLT. pp. 451–463 (2015)

[60] Safety, D.: Improvement needed in fda's postmarket decision-making and oversight process (2006)

[61] Sahu, S.K., Anand, A., Oruganty, K., Gattu, M.: Relation extraction from clinical texts using domain invariant convolutional neural network. arXiv preprint arXiv:1606.09370 (2016)

[62] Sampathkumar, H., Chen, X.w., Luo, B.: Mining adverse drug reactions from online healthcare forums using hidden markov model. BMC medical informatics and decision making 14(1), 1 (2014)

[63] dos Santos, C.N., Gatti, M.: Deep convolutional neural networks for sentiment analysis of short texts. In: COLING. pp. 69–78 (2014)

[64] Sarker, A., Aliod, D.M., Paris, C.: Automatic prediction of evidence-based recommendations via sentence-level polarity classification. In: IJCNLP. pp. 712–718 (2013)

[65] Sarker, A., Ginn, R., Nikfarjam, A., O'Connor, K., Smith, K., Jayaraman, S., Upadhaya, T., Gonzalez, G.: Utilizing social media data for pharmacovigilance: A review. Journal of biomedical informatics 54, 202–212 (2015)

[66] Sarker, A., Gonzalez, G.: Portable automatic text classification for adverse drug reaction detection via multi-corpus training. Journal of biomedical informatics 53, 196–207 (2015)

[67] Schmidhuber, J.: Deep learning in neural networks: An overview. Neural networks 61, 85–117 (2015)

[68] Shen, Y., He, X., Gao, J., Deng, L., Mesnil, G.: Learning semantic representations using convolutional neural networks for web search. In: Proceedings of the 23rd International Conference on World Wide Web. pp. 373–374. ACM (2014)

[69] Sulieman, L., Gilmore, D., French, C., Cronin, R.M., Jackson, G.P., Russell, M., Fabbri, D.: Classifying patient portal messages using convolutional neural networks. Journal of Biomedical Informatics 74, 59–70 (2017)

[70] Tan, P.N., Steinbach, M., Kumar, V.: Introduction to data mining. 1st (2005)

[71] Toh, Z., Chen, B., Su, J.: Improving twitter named entity recognition using word representations (2015)

[72] Turian, J., Ratinov, L., Bengio, Y.: Word representations: a simple and general method for semi-supervised learning. In: Proceedings of the 48th annual meeting of the association for computational linguistics. pp. 384–394. Association for Computational Linguistics (2010)

[73] Wang, W.: Mining adverse drug reaction mentions in twitter with word embeddings. In: Proceedings of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing (2016)

[74] Warriner, A.B., Kuperman, V., Brysbaert, M.: Norms of valence, arousal, and dominance for 13,915 english lemmas. Behavior research methods 45(4), 1191–1207 (2013)

[75] Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis. In: Proceedings of the conference on human language technology and empirical methods in natural language processing. pp. 347–354. Association for Computational Linguistics (2005)

[76] Winkler, W.E.: The state of record linkage and current research problems. In: Statistical Research Division, US Census Bureau. Citeseer (1999)

[77] Yang, C.C., Yang, H., Jiang, L., Zhang, M.: Social media mining for drug safety signal detection. In: Proceedings of the 2012 international workshop on Smart health and wellbeing. pp. 33–40. ACM (2012)

[78] Yih, S.W.t., He, X., Meek, C.: Semantic parsing for single-relation question answering (2014)

[79] Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in neural information processing systems. pp. 649–657 (2015)

[80] Zhang, Y., Wallace, B.: A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820 (2015)

[81] Zhang, Z., Nie, J., Zhang, X.: An ensemble method for binary classification of adverse drug reactions from social media. In: Proceedings of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing (2016)