Sentiment Analysis and Opinion Mining from Noisy Social Media Content

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering by Research

by

Aditya Joshi
201207619
aditya.joshi@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
April, 2019

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Sentiment Analysis and Opinion Mining from Noisy Social Media Content" by Aditya Joshi, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____
Date

_____
Adviser: Prof. Vasudeva Varma

_____
Co-Adviser: Prof. Manish Shrivastava

To teachers, who've taught me that learning never stops.

Acknowledgments

I would like to take this opportunity to acknowledge and appreciate the efforts of the people who have helped me during my research and documenting this thesis.

I am indebted to my advisor, Professor Vasudeva Varma, who has supported me at all levels during the course of this thesis. He introduced me to the area of Information Retrieval and Knowledge Discovery.

I'm indebted with gratitude to Manish Shrivastava, Manish Gupta and Marc Franco Salvador who have mentored me at the right times when I needed it the most. I'm thankful to collaborators in the works presented in this thesis, Ameya, Satarupa, and Santosh Kosgi, who helped me finish the projects I started.

I thanks my batchmates for having spent some great times with them during the course of my stay at IIIT. I thank all the members of SIEL lab especially, Ayushi Dalmia, Dharmesh Kakadiya, Harshit Jain, Kumar Rishabh, Kushal Dave, Nikhil Priyatam, Priya Radhakrishnan, and Romil Bansal for making the period of research joyous and satisfactory. I also thank Maaz, Nikhilesh and Vandan from LTRC Lab for their support.

I'm very thankful to my peers and friends Gaurav, Lokesh, Ishan and Ujjwal with whom I've spent my most time in IIIT Hyderabad. The work of writing this thesis was made more enjoyable with Sai Krishna's support and encouragement during challenging times. Finally I thank my parents and sister for being there in my support and during my research.

Abstract

User generated content, mainly through social media, blogs and review websites etc. offers a powerful outlet for peoples thoughts and feelings. This presents an enormous ever-growing source of texts ranging from everyday observations to involved discussions. This thesis approaches problems of sentiment analysis from recently popularized perspectives. We explore how sentiment of the user generated text can be computed using a combination of heuristics and machine learning approaches. This thesis contributes to the field of sentiment analysis, which aims to extract emotions and opinions from text.

Social Media and other mediums of user generated content provide a powerful source for aggregating peoples' thoughts and feelings towards a particular entity, event or topic. A robust sentiment analysis system is helpful for the business entities towards which the sentiment is expressed, providing them a solution to understand users' perception towards the product and services they offer. It is also helpful for other users in form of summary of reviews about a product and service the user wishes to pay for. There are hundreds of websites providing opinionated user-generated content, it is difficult to go through all and then make some decision. Automated sentiment detection helps identify pros and cons of a product with ease. Sentiment analysis system presents a reliable tool for decision making for various parties.

This thesis begins with sentiment analysis on social media using machine learning methods. Various signals were specifically extracted for social media setting. A method for fine grain sentiment analysis is presented using a sequence labelling approach combined with heuristics for improved features. To further enhance the sentiment analysis on social media, a solution is proposed for sentiment analysis of Code Mixed data, which is otherwise treated as noise and thus huge amount of information is lost. We conclude that feature engineering for standard machine learning methods and deep learning provide a reliable solution for sentiment analysis.

Contents

List of Figures

Figure Page

List of Tables

Table Page

Chapter 1

Introduction

"Information is the oil of the 21st century, and analytics is the combustion engine."

— Peter Sondergaard,Gartner

In the past years, the world has seen explosive growth in technology. According to the International Telecommunication Union about 3.4 billion people, or almost half of the world's population is connected to the Internet as of July 2016. The rise of Social Web has lead to rapidly growing user generated content. It has given users a medium of social exchange in the form of blogs, micro-blogs, wikis, posts, pictures etc. Commercial entities, organizations as well as individuals use social media as a way to interact with their target audience. Many other forms of websites provide a medium of interaction among the users in form of user reviews, ratings, feedback etc. These provide massive amount of data of commercial interest, which, when processed yields to better understanding which leads to better decision making.

Rather than conducting user surveys or polls, it is easier and more beneficial for organizations to process users' views from the social web. However finding and manually analyzing such data is a tedious task because of the volume of consistently expanding data. The field of Natural Language Processing (NLP) has provided new methods of automated understanding of such user generated text.

Initial automated solutions provided easy retrieval of data with the help of indexing techniques. Research in text clustering and summarization made it easier to comprehend the trend about a subject in the public. Though such methods reduce the amount of manual analysis, they do not eliminate the need of human intervention. Manual work is still required to understand the direction of the trend. Sentiment Analysis, a sub-field of NLP provides methods to understand the perception of direction of trends regarding a topic from the text.

This thesis deals with the problems of sentiment analysis. We present the problem, and the solution of sentiment analysis in social media, aspect based sentiment analysis, and sentiment analysis in noisy Code Mixed text from social media.

1.1 Sentiment Analysis

Social Influence is one of the most important factors that determine human behavior. Several psychological studies like (Sussman and Gifford, 2013) [68] popularized by (Robert Cialdini, 2001) [12] have shown how susceptible we are to be persuaded to a particular action due to social proof. User generated data often exhibits emotional perception of the user towards a subject. Sentiment Analysis is the methodology of applying Natural Language Processing (NLP) and Computational Linguistics based approaches to identify and extract sentiment expressed in a piece of text. Such computational approaches provide a way to compile large amount of textual data.

Sentiment Analysis(SA), or Opinion Mining is vast field that covers the problem of identifying emotions, opinions, moods and attitudes. There are also many names and slightly different tasks, e.g., sentiment analysis, opinion mining, opinion extraction, sentiment mining, subjectivity analysis, affect analysis, emotion analysis, review mining, etc. [38]

Table **??** shows examples of social media text with their polarities. Example #1 is a positive sentence where the author is appreciating some entity (show), which is evident from the words like *thanks*, *love* and joy.Example #2 is negative which is evident at surface form because of the use of word *dull*. Example #3 is negative about the *line at McDonalds*, which is understood from the phrases like *too long* and *can't get*. In many real life examples, sentiment can't be interpreted directly by analyzing the individual words and we need further processing.

These examples show that while it is easy for a human to understand sentiment in a piece of text, it is challenging for machines to accomplish this automatically. Meanwhile, presence of ambiguous words, discussion about multiple topics and several complex linguistic structures add to the problem. Processing noisy user generated text from social media further adds to the complexity. A good Sentiment Analysis system should be able to overcome such issues and assign sentiment polarities as close to human understanding as possible.

1.2 Sentiment in Linguistics and Sociology

Human emotion and sentiments have been quantitatively studied in the past few decades by Sociologists and Psychologists. The meaning of words has been central in the Symbolic Interactionism paradigm, which states that people act toward other people and things based on the meanings that they have given to them [18]. Sentiments and Emotions can be studied from semantic modelling point of view in Affect Control Theory. Affect Control theory (ACT) is a sociological theory which provides multi-dimensional view of affective emotion, as well as the means to interpret holistic point of view by combining the meanings of individual words in a sentence. Charles Osgood [49] [50] describes that three dimensions are sufficient to describe a sentiment bias: evaluation (good/bad), potency (strong/weak) and activity (active/inactive). These dimensions have been proved universally applicable through cross cul-

tural studies [24]. These studies have led to creation of sentiment lexicons for multiple languages and in several cultural contexts including internet users and religious groups.

Each word has an affective meaning (fundamental). But in context, the meaning of the word changes, making it transient. In a decision making situation, people gather the fundamental meanings, adjust these in the light of their context, and then act accordingly. Another measure, deflection models person's reaction to a particular situation. Deflection is the Euclidean distance between the fundamental cultural sentiments and the transient impressions [60](how much the meaning of words changes in a given context).

For example, a sentence "The terrorists were killed" would have a large deflection. Usually "terrorists" and "killed" are both usually considered negative. So simply averaging the evaluation dimensions would produce an overall negative score. But in most parts of the society, this sentence would be considered as possessing positive sentiment. If we re-evaluate the affective meaning of each of the words in the context first, the scores of each word will reflect the meaning of the sentence, and thus, further summation would yield a positive score.

1.3 Levels of Sentiment Analysis

Bing Liu [38] describes sentiment analysis at document, sentence, and aspect level. Based on the application, we can approach sentiment analysis at two major scales.

1. **Coarse Grain Sentiment Analysis**
   Coarse-grained sentiment analysis is associated with with extracting overall sentiment orientation of whole documents, where a document might be considered as a web page, blog, message or a tweet. The sentiment is generally considered as a sentiment polarity that indicates whether a given text is positive, negative or neutral. In some cases, it might be a polarity score.

2. **Fine Grain Sentiment Analysis**
   Fine-grained sentiment analysis is associated with extracting opinions at deeper levels involving the clauses, mentioned entities or the subject of discussion. Often analysis of sentiment expressed in individual sentences within larger documents is also considered as a fine-grained sentiment analysis task. Rather than assigning a sentiment polarity to the document, we identify polarity for a particular entity or a phrase.

1.4 Aspect Based Sentiment Analysis

Aspect Based Sentiment Analysis (ABSA) is a fine grain sentiment analysis problem where the aim is to build systems that receive input as a set of texts (e.g. reviews, social media text etc) discussing a particular entity. The systems attempt to detect the main aspects (features) of the entity and estimates the average sentiment based on the opinion in context. (Pavlopoulus, 2013) [57] decomposed the task

into three stages:

1. **Aspect Term Extraction**
   Aspect term identifies a feature of the entity under discussion. e.g. A review about a mobile phone might mention aspects like battery life, screen resolution etc. Aspect might also be present in the form of a co-reference or be present inherently within the sentence. These are called *implicit aspect expressions* contrary to *explicit expressions* where the aspect term is present in the text.

2. **Aspect Categorization**
   Each aspect is assigned as one of the pre-defined categories of attributes related to the entity. For example, the aspect terms *alfredo pasta* and *masala dosa* in a restaurant review can be categorized to *food*, while *waiters* and *staff* are categorized to *service*.

3. **Sentiment Polarity Classification**
   Given an aspect, find the polarity based on subjective context that an opinion about it. We classify the polarity as being positive, negative or neutral.

1.5 Code Mixed Text

Code Mixing (CM) refers to mixing of two or more languages in text. A study [29] defines it as the change of one language to another within the same utterance or in the same oral/ written text. It is a common phenomenon in societies in which two or more languages are used.

Data curated from Social Media often contains code mixed data authored by the users from multilingual background. Most of the work on Sentiment Analysis is concentrated towards processing mono-lingual text. SA systems usually ignore mixed code text as noise, which we believe leads to loss of large amount of useful information such data contains.

The work on CM text presented in this theses is done on utterances of Hindi/Urdu languages written in roman script (English) which we refer as Hindi-English Code Mixed text (Hi-En CM text). This present challenges of it's own like usages of non-standard spellings, incoherent grammatical structure etc. which are explained in more details in Chapter 5.

1.6 Challenges in Sentiment Analysis

Both fine grain and coarse grain sentiment analysis deal with processing natural human generated text which is vulnerable to issues like sentence structure, variation in vocabulary, mis-spellings etc. We highlight major challenges below:

1. A word may have multiple sentiment polarity based on the context.

2. User generated text is often noisy and shows irregular syntax.

3. Vocabulary is not limited. Words may be induced due to named entities as well as user errors and deliberate misspelling.

4. Sentences from social media might be sarcastic, thus causing a sentiment analysis system to produce incorrect result.

5. Sentiments are often be ambiguous due to mention of multiple entities or opinions about them

Dealing with sarcastic messages is out of the scope of this thesis.

1.7 Contributions

This thesis approaches top sentiment analysis problems from multiple perspectives. We conducted experiments on microtext datasets curated from Twitter, restaurant reviews and code mixed data crawled from Facebook. Major contributions of this thesis are as follows:

- Aspect Detection: We have explored several methods for aspect detection. We propose a Conditional Random Fields based system that labels each word of the given sentence to find aspect terms. We also leverage a subjectivity classifier to find if the sentence contains implicit aspects.

- Sentiment Analysis: We explored syntactic, semantic and heuristics based features for sentiment analysis. The sentiment analysis sub-system uses an ensemble method that uses probabilities computed by individual classifiers and uses them as features for the final classifier.

- Mix Code Dataset: We constructed a dataset by parsing user comments from social media (Facebook) pages and annotated them with sentiment labels. The dataset contains linguistic elements of Hindi language written in Roman script.

- LSTM Solution for Sentiment Analysis: We built a deep neural network based system for analysis of sentiments in Mix Code Dataset. Such datasets are more noisy compared to regular text due to lack of correspondence between phonetics of the languages. We experimented with implementations of standard sentiment analysis methods. our proposed system consists of a neural network architecture that tries to learn embeddings of opinion words.

1.8 Organization of Thesis

In Chapter 2, we discuss theoretical background required for the further chapters and present an overview of relevant related works. In Chapter 3, we present a system for sentiment analysis in noisy social network setting. In Chapter 4, we explain the problem of Aspect Based Sentiment Analysis.

We present our approaches for extraction of aspects and categorizing aspect terms among the relevant classes. categorization and determining their sentiment polarity from reviews. In Chapter 5, we continue our discussion of sentiment analysis and attempt to build a sentiment classification engine for Mix Code setting. Finally we conclude in Chapter 6, along with discussion on applications of the works presented in this thesis and future work.

Chapter 2

Background and Related Work

Over past two decades, sentiment analysis has become one of the most active research areas in the field of natural language processing. The rise of social media and other mediums of user generated content have generated a large amount of opinionated data. Along with NLP, researchers in management, political science, social sciences are also working in the same domain.

The work present in this thesis involves use of ideas from Natural Language Processing and Machine Learning to perform sentiment analysis on social media text, extract opinions from user reviews and interpret sentiment polarity in very noisy text. In this chapter, we will present the background and related work.

2.1 Related work

Research in sentiment analysis and opinion mining started back in 1966 with the development of General Inquirer system [67]. The typical task in sentiment analysis is text polarity classification, where the classes of interest are positive and negative, sometimes with a neutral class. There are two approaches to this problem: one may use a sentiment lexicon (a list of words with known sentiment polarity), or one may build a model based on the examples from training data.

Lexicon-driven approach is simple and easy to use, but it is inflexible. It is tedious to construct lexicons for a new domain or new kind of data. Style of writing my also change the way how lexicon-driven approach is applied. However machine learning techniques are relatively robust and capture the peculiarities of the language used in the data.

(Pang Lee, 2002) [53] presented sentiment classification techniques using machine learning and evaluated their system on movie reviews. Their results show that the machine learning techniques perform better than simple counting methods. They achieve an accuracy of polarity classification of roughly 83%. (Godbole et al. 2007) [26] explained the significance and methods for sentiment analysis of news articles and blogs. Their system consists of a sentiment identification phase, which associates expressed opinions with each relevant entity, and a sentiment aggregation and scoring phase, which scores each entity relative to others in the same class. These works initiated the problem of sentiment analysis as a

whole subfield at the intersection of NLP and Machine Learning. In the next few paragraphs, well look at various approaches for sentiment analysis in the recent past.

Discourse Analysis

Discourse analysis is a kind of text analysis beyond the sentence level - usually over multiple sentences, or smaller sub-sentences. [73] defined discourse structure are the patterns that one sees in multi-sentence (multi-clausal) texts. Analysing this structure is important to understand information in the text.

Rhetorical Structure Theory (RST) [41] is the one of most widely studied theory in discourse analysis. In RST, two non-overlapping spans of text are connected by a discourse relation. The primary text span under observation is called as nucleus, and the supporting span is called as satellite.

The discourse is identified described by constraints on nucleus, satellite, combiantion of both and the effect. The resulting structures are identified as one of the schemas - circumstance, contrast, joint, motivation-enablement, and, sequence.

An alternate theory based on Penn Discourse Treebank [58] uses discourse connectives, which are lexical items that help identify the discourse relations. Discourse analyses helps to compute sentiment of a sentence while considering other related sentences around it.

This thesis mainly covers machine learning based methods which are supplemented with lexicons to improve the quality of results.

2.2 Methods and Techniques

Recent advancements in this area have attracted techniques ranging from archaic rule based approaches to new approaches that use deep learning. In this section, we discuss various methods and topics that are relevant to this thesis.

2.2.1 Feature Engineering

Machine Learning based solutions work on the features. A feature vector is a representation of actual content (document, tweet etc) that the classification algorithm takes as an input. The purpose of a feature, other than being an attribute, would be much easier to understand in the context of a problem. A feature is a characteristic that might help when solving the problem.

In this section, we briefly discuss various techniques of feature engineering that are used in sentiment analysis.

2.2.1.1 n-Grams

If we consider each word as an independent lexical unit that is present as one of the features for the learning algorithm, it is referred as unigram approach. Unigram approach doesn't consider the word ordering in the sentence and often leads to less than optimal results. In n-gram approaches, we take each combination of 'n' words in sequential order in the sentence. Often 2 or 3 is sufficient to capture the requirements.

2.2.1.2 TF-IDF

TF-IDF [40, 33] is not a single method, but a class of techniques where similarity between queries and documents is measured via the sum of term frequency-like numbers (TFs) multiplied by terms' importance. The term importance is frequently expressed via the IDF (the inverse document frequency frequency tfidf, actually it is the logarithm of IDF that is used in practice) . TF-IDF measures how surprising it is to see the query keywords in a document given a known distribution of these keywords in the text collection.

2.2.1.3 Word2Vec

Word2vec [44] is a two-layer neural net that processes text. Its input is a text corpus and its output is a set of vectors: feature vectors for words in that corpus. While Word2vec is not a deep neural network, it turns text into a numerical form that deep nets can understand.

Word2vec is essentially about proportions of word occurrences in relations holding in general over large corpora of text. Word2vec is not a single monolithic algorithm. Word2vec contains two distinct models (CBOW and skip-gram), each with two different training methods (with/without negative sampling) and other variations (e.g. hierarchical softmax), which amount to small "space" of algorithms.

2.2.1.4 Heuristics

Along with standard algorithmic techniques, heuristics are often used to enhance the sentiment classification systems. The data is supplemented with extra information based on pre-decided rules by a human with sufficient knowledge or experience in the related domain.

In systems involving classifying data from social media, such heuristics can include hashtags (*#so-cool, #ftw*) etc. that impart the sentiments of the user. Another such social media specific signal is grapheme stretching (*yeaaaahhh* etc.) that can be leveraged to find sentiment polarity as well as the strength.

2.2.1.5 Parts of Speech

Part of Speech (POS) is a category of words which have similar grammatical properties in accordance with their syntactic functions. In English, their are eight parts of speech: the verb (VB), the noun (NN), the pronoun (PR+DT), the adjective (JJ), the adverb (RB), the preposition (IN), the conjunction (CC), and the interjection (UH). Penn Treebank recommends thirty-six fine-grain Part of Speech categories. e.g. Verb can further be classified into VB (Verb, base form), VBD (past tence), VBG (gerund), VBN (past participle), VBP (singular present), and VBZ (3rd person singular present).

A Part-Of-Speech Tagger (POS Tagger) is a piece of software that reads text in some language and assigns parts of speech to each word. POS Tagging is treated as a sequence labelling problem, often implemented using Hidden Markov Models (HMM) or Conditional Random Fields (CRF). Most sequence models can be seen as chaining together the scores or decisions from successive local models to form a global model for an entire sequence. A popular implementation Toutanova:2003 combines the idea of (i) explicit use of both preceding and following tag contexts via a dependency network representation, (ii) broad use of lexical features, including jointly conditioning on multiple consecutive words, (iii) effective use of priors in conditional loglinear models, and (iv) fine-grained modeling of unknown word features.

2.2.1.6 Dependency Parsing

Syntactic structure consists of lexical items linked by binary asymmetric relations called dependencies. Such dependencies between a word and it's neighbors provide a semantic understanding of the sentence. Each such connection connects a superior term, called governer, and an inferior term, called as subordinate.

Each dependency structure has:

- head-dependent relations (directed arcs),
- functional categories (arc labels), and,
- some structural categories (parts-of-speech).

There are several approaches for dependency parsing, each aim to breakdown a sentence into the elements listed above. Chen and Manning (2014) [8] proposed a transition-based dependency parsing algorithm that aims to predict a transition sequence from an initial configuration to some terminal configuration, which derives a target dependency parse tree. They use greedy parsing, which uses a classifier to predict the correct transition based on features extracted from the configuration. This parser builds a parse by performing a linear-time scan over the words of a sentence. At every step it maintains a partial parse, a stack of words which are currently being processed, and a buffer of words yet to be processed. The parser continues to apply transitions to its state until its buffer is empty and the dependency graph is completed.

In initial state, all the words are present in order on the buffer. A dummy node ROOT is placed on the stack. There are three possible transitions which can be applied, viz, LEFT-ARC, RIGHT-ARC and SHIFT. Figure **??** shows an example of transition based dependency parsing from Chen and Manning, 2014 [8]. The table on the bottom shows a transition sequence of the arc-standard system. Above right image shows an intermediate configuration which leads to the final dependency tree on the Top-left. With just these three types of transitions, a parser can generate any projective dependency parse. This approach has been shown to be more efficient than other approaches, it however slightly trades-off with accuracy.

2.3 Learning from Data

Supervised Learning is a class of Machine Learning based solutions which are used for wide variety of problems in Natural Language Processing, Computer Vision, Speech Recognition etc. An approach that utilizes a supervised learning algorithm requires data annotated specifically for the problem to be solved.

Classification, a type of supervised learning, is the task of choosing the correct class label for a given input. In basic classification tasks, each input is considered in isolation from all other inputs, and the set of labels is defined in advance. One of the classification problems we dealt with is finding whether a given review comments about (i) food, (ii) service, or (iii) ambience etc. Classifying a sentence for positive, negative or neutral sentiment polarity is also a classification problem.

The basic classification task has a number of interesting variants. For example, in multi-class classification, each instance may be assigned multiple labels; in open-class classification, the set of labels is not defined in advance; and in sequence classification, a list of inputs are jointly classified.

In this section, we give a brief summary of a few classification algorithms that are used in the works presented in this thesis.

2.3.1 Naive Bayes

Naive Bayes classifier is a popular method [62, 37] that begins by calculating the prior probability of each label, which is determined by checking frequency of each label in the training set. The contribution from each feature is then combined with this prior probability, to arrive at a likelihood estimate for each label. The label whose likelihood estimate is the highest is then assigned to the input value. The idea behind Naive Bayes Classifier is Bayes Theorem, which finds the probability of an event given the probability of another event that has already occurred. Naive Bayes classifier performs extremely well for problems which are linearly separable and for problems which are non-linearly separable it performs reasonable. We used the already implemented Naive Bayes implementation in Scikit Learn.

2.3.2 Maximum Entropy

The Maximum Entropy classifiers [48, 75, 31] use a model that is very similar to the model employed by the naive Bayes classifier. But rather than using probabilities to set the model's parameters, they use search techniques to find a set of parameters that will maximize the performance of the classifier. In particular, they looks for the set of parameters that maximizes the total likelihood of the training corpus.

2.3.3 Support Vector Machines

Support Vector Machine (SVM) [15] classifier constructs N-dimensional hyperplane which separates data into two categories. SVM models are closely related to a Neural Network. SVM takes the input data and for each input data row it predicts the class to which this input row belongs. SVM works for two class problems and is a non probabilistic binary linear classifier. We used libSVM classifier which is available as an add on to Weka toolkit with default parameters.

2.4 Neural Networks

Neural networks are a computational approach which is based on a large collection of neural units loosely modeling the way a biological brain solves problems. Each neural unit is connected with other such units. In such clusters, each individual link can be enforcing or inhibitory in their effect on the activation state of the connected neural units. Each unit has a summation function which combines the values of all the inputs, and a threshold function must be surpassed before it is propagated to other neurons.

Such systems have often be found to be highly effective in the situations where we have large amount of training data. They are self-learning and do not need to be explicitly programmed for domain specific feature engineering.

In a simple case, you could have two sets of neurons: ones that receive an input signal and ones that send an output signal. When the input layer receives an input it passes on a modified version of the input to the next layer. In a deep network, there are many layers between the input and output (and the layers are not made of neurons but it can help to think of it that way), allowing the algorithm to use multiple processing layers, composed of multiple linear and non-linear transformations. In this subsection, we will discuss a perceptron, which is the minimal unit of a neural network, and proceed to discuss such an example of complex architecture which is used in our work.

2.4.1 Perceptron

A perceptron takes several binary inputs, $x_1, x_2, ..,$ and produces a single binary output. In the example shown in Figure **??**, the perceptron has three inputs, x_1, x_2, x_3. In general it could have more or fewer inputs. It computes the output by combining the weight applied to each input and compares the

summation with a threshold value.

$$output = \begin{cases} 0 & if \sum_j w_j x_j \leq threshold \\ 1 & if \sum_j w_j x_j > threshold \end{cases}$$

Whether the sum is above or below the threshold determines the final output of the perceptron.

The neuron's output, 0 or 1, is determined by whether the weighted sum

Perceptron is the simplest element which isn't a complete model of decision making. However a complex network of perceptrons can make quite subtle decisions and yield a much reliable accuracy. Detailed study of neural network architectures is out of the scope of this thesis.

2.4.2 Recursive Neural Networks

A recursive neural network (RNN) is a kind of deep neural network created by applying the same set of weights recursively over a structure, to produce a structured prediction over variable-length input, or a scalar prediction on it, by traversing a given structure in topological order. RNNs have been successful in learning sequence and tree structures in natural language processing, mainly phrase and sentence continuous representations based on word embedding.

Recurrent neural networks are recursive artificial neural networks with a certain structure: that of a linear chain. Whereas recursive neural networks operate on any hierarchical structure, combining child representations into parent representations, recurrent neural networks operate on the linear progression of time, combining the previous time step and a hidden representation into the representation for the current time step.

2.5 Background

In this section, the concepts required to understand Sentiment Analysis and Natural Language processing of text and the tools used in analysing sentiment are explained.

2.5.1 Software Tools

The current field of sentiment analysis has the tools required to calculate sentiment analysis over a text written in a language. How ever, with the rise of social media platforms, we are noticing a paradigm shift in the way people communication using a language. It has become common that people tweet / post /comment / in English which when pronounced bring out words of foreign language. This kind of data is called code mixed data. Calculating sentiment among such sentences is still an unsolved problem and we try to use the existing tools to solve this problem. The tools we used while working on this problem are detailed below.

As specified earlier, finding sentiment among textual data is a solved problem and tools like Rapid-Miner [1] are available to use. *Pattern* [2] from Clips is used to calculate the subjectivity and sentiment among the code review comments. The approach generally followed to extract and analysing sentiment among textual data is shown in the Figure **??**.

2.5.1.1 Stanford CoreNLP

Stanford CoreNLP [14] [42]gives the base forms of words, their parts of speech, whether they are names of companies, people, etc., normalize dates, times, and numeric quantities, mark up the structure of sentences in terms of phrases and word dependencies, indicate which noun phrases refer to the same entities, indicate sentiment, extract particular or open-class relations between entity mentions, get quotes people said, etcetra.. Stanford CoreNLP is written in Java; current releases require Java 1.8+. Stanford CoreNLP is licensed under the GNU General Public License.

2.5.1.2 Numpy and Scikit Learn

Scikit-Learn [56] library contains a great number of machine learning algorithms. Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The library is built upon the SciPy (Scientific Python) [32] that must be installed before you can use scikit-learn. This stack that includes:

1. NumPy: Base n-dimensional array package
2. SciPy: Fundamental library for scientific computing
3. Matplotlib: Comprehensive 2D/3D plotting
4. IPython: Enhanced interactive console
5. Sympy: Symbolic mathematics
6. Pandas: Data structures and analysis

2.5.1.3 NLTK

NLTK [39] is the most famous Python Natural Language Processing Toolkit. The official description says : ”NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning.”

[1] https://rapidminer.com/

[2] http://http://www.clips.ua.ac.be/pattern

2.5.1.4 Keras

Keras [11] is a high-level neural networks library, written in Python. It is a minimalist Python library for deep learning that can run on top of Theano or TensorFlow. It has a simple and highly modular interface, which makes it easier to create even complex neural network models. This library abstracts low level libraries, namely Theano and TensorFlow so that, the user is free from implementation details of these libraries.

2.5.1.5 Mallet

A topic modeling tool takes a single text (or corpus) and looks for patterns in the use of words; it is an attempt to inject semantic meaning into vocabulary.

MALLET [43] is a Java-based package for statistical natural language processing, document classification, clustering, topic modeling, information extraction, and other machine learning applications to text.

2.5.2 Evaluation Metrics

Once we have defined our problem and prepared the data, we apply machine learning algorithms to the data in order to solve our problem. This is where evaluation metrics provide us a track of how close we are to our goal.

Figure **??** shows where evaluation measures fit in learning process. We divide the labelled data in two segments – training set, which is used to learn the model parameters, and test set, which is used to predict the values and compare the results. Because the test set is unseen during the learning process, it gives fairly more reliable estimate on how the model would perform on actual data.

We often further divide the training dataset into a training set and a validation set. Validation data set contains a set of examples used to tune the parameters of a classifier.

A more sophisticated approach than using a test and train dataset is to use the entire transformed dataset to train and test a given algorithm. This is called as *Cross Validation*. In cross-validation, we go through multiple rounds (determined by the value of k in k-fold cross-validation) of training and testing various subsets of the data. e.g. Say, in 5 fold cross validation, we use first 80% of the data as training phase and last 20% data in the test phase in the first round. In second round, we would pick first 60% and last 20% in the training data, and the left-over part would be used as test data. Thus all such five combinations will be tested.

In both the approaches, same evaluation metrics can be used for the task of classification.

Accuracy

Classification accuracy is the number of correct predictions made divided by the total number of predictions made, multiplied by 100 to turn it into a percentage. Though Accuracy gives an easy to

compute and comprehend summary of performance of a system, it often fails to explain the situation in case of biased data set.

Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. For a binary classification problem, four cells would contain the number of items falling per various categories. The sum of all the cells is equal to the number of total items that were tested.
For classification tasks, the terms true positives, true negatives, false positives, and false negatives compare the results of the classifier under test with trusted external judgments.

$$Precision = \frac{true\,positive}{true\,positive + false\,positive}$$

$$Recall = \frac{true\,positive}{true\,positive + false\,negative}$$

F-Measure

A measure that combines precision and recall is the harmonic mean of precision and recall.

$$FMeasure = 2 \times \frac{precision \times recall}{precision + recall}$$

This score takes both false positives and false negatives into account.

In this chapter, we summarized the relevant background. Further chapters describe our work in details.

Chapter 3

Sentiment Analysis in Social Media

Social media not only acts as a proxy for the real world society, it also offers a treasure trove of data for different types of analyses like Trend Analysis, Event Detection and Sentiment Analysis etc.

Sentiment Analysis in social media in general and Twitter in particular has a wide range of applications Companies/ services can gauge the public sentiment towards the new product or service they launched, political parties can estimate their chances of winning the upcoming elections by monitoring what people are saying on Twitter about them, and so on. In spite of the availability of huge amount of data and the huge promises they entail, working with social media data is far more challenging than regular text data. Being user-generated, the data is noisy; there are misspellings, unreliable capitalization, widespread use of creative acronyms, lack of grammar, and a style of writing that is very typical of its own which makes the problem of Sentiment Analysis on Twitter more challenging. Also, the cues for positive or negative sentiment in social media text are starkly different, thereby generating a whole new domain for exploration.

In this chapter, we will discuss our methods and their results on a public twitter dataset referred as SemEval'15 Twitter Dataset. SemEval 2015 Task 10 subtask B [61] specifically deals with the task of Sentiment Analysis in Twitter. Our approach encompasses feature extraction specific to social media text and using these features with classifier to compute the final results.

3.1 Methods for Sentiment Analysis of Social Media Text

Though user generated content has been evolving for more than a decade, Microblogging, specifically Twitter has recently been demonstrated for sentiment analysis and opinion mining purposes. (Pak, 2010)[51] advocated use of microblogging/twitter for the following reasons:

- Microblogging platforms are used by different people to express their opinion about different topics, thus it is a valuable source of peoples opinions.
- Twitter contains an enormous number of text posts and it grows every day. The collected corpus can be arbitrarily large.

- Twitters audience varies from regular users to celebrities, company representatives, politicians4, and even country presidents. Therefore, it is possible to collect text posts of users from different social and interests groups.

- Twitters audience is represented by users from many countries5. Although users from U.S. are prevailing, it is possible to collect data in different languages.

3.1.1 Lexicon Based Approaches

In such approaches, sentiment is seen as the function of some keywords present in the text. A prerequisite for this task is the construction of a sentiment dictionary - that indicate whether a word is positive or negative. A more preferred way is to obtain a probabilistic score of strength of a word towards a sentiment polarity. The scores of keywords in the sentence are combined to compute total sentiment of the given sentence.

Lexicon Construction

A method to build lexicons attempts to expand a smaller seed list into lexicons by recursively querying for synonyms using a dictionary. Wordnet [1] has been widely used for exploring semantic relatedness between words. (Godbole, 2007) [26] associated a polarity to a word and query both synonyms and antonyms from the wordnet. Synonyms inherit the polarity from the parent word and antonyms get an opposite polarity. The significance of the path decreases based on it's distance from the seed word. Final score of the word is obtained by summation of the scores received over all paths. Their system runs in two iterations where the second iteration considers flips of the polarity while calculating the final score.

Andrea Esuli [21] presented Sentiwordnet, a lexical resource in which each synset of Wordnet is associated to numerical scores that indicate its sentiment. The sentiment scores of a word describe how Objective, Positive and Negate the terms in the synset are. For constructing SentiWordNet, they trained ternary classifiers that decide whether a Synset is positive, negative or objective. These are adaptions of the authors' previous works, which are to decide positive-negative polarity [19] and subjective-objective polarity [20] of terms. Opinion related scores are determined by the normalized proportion of ternary classifiers that have assigned the corresponding label to it. Many sentiment analysis systems, including those presented in this thesis, use SentiWordNet scores as an additional input feature.

3.2 Supervised Techniques

These techniques, many of which have performed as state of the art, attempt to build a classifier based on manually labelled training data. Most used supervised algorithms are Support Vector Machines

[1] Princeton University "About WordNet." WordNet. Princeton University. 2010. http://wordnet.princeton.edu

(SVM), Naive Bayes classifier and Multinomial Naive Bayes. It has been shown that Supervised Techniques outperform unsupervised techniques in performance [53]. Supervised techniques can use one or a combination of approaches we saw above. For example, a supervised technique can use relationship-based approach, or language model approach or a combination of them. For supervised techniques, the text to be analyzed must be represented as a feature vector.

The method presented in this chapter is a supervised learning technique.

3.3 Our Approach for Twitter Sentiment Analysis

3.3.1 Pre-processing

We acquire a list of acronyms and their expanded forms. We use this list as a look-up table and replace all occurrences of acronyms in our data by their expanded forms. We normalize all numbers that find a place in our data by replacing them with the string 0. We do not remove stop words because they often contribute heavily towards expressing sentiment/emotion. We do not also stem the words because stemming leads to the loss of the parts of speech information of the word and makes the use of lexicons unnecessarily complicated.

3.3.2 Vocabulary Generation

We assign a unique ID to all words occurring in our data. All the hashtags we encounter in the data are hashed to a single string place holder and a single unique ID is assigned to it, as opposed to different Ids for different hashtags. Hashtags are mostly formed by concatenation of multiple words without any space in between, and therefore, unless hashtags are segmented into meaningful chunks, raw hashtags seldom add any semantics to the sentence. Hence, we do not distinguish between the different hashtags and consider them as a single unit. Similarly, we hash all mentions of the kind @user1 and @user2 to a single string placeholder and assign a single ID to it. This is because these words prefixed by @ are all named entities and do not contribute anything to the semantic meaning or towards the polarity of a tweet.

3.3.3 Feature Engineering

The task required us to classify a tweet into positive, negative and neutral polarity categories. This can essentially be treated as a 2-step process:

- Classify each tweet into subjective (positive/ negative) and objective(neutral) classes.
- Classify subjective tweets into positive and negative ones.

We keep this philosophy in mind, but do not explicitly model the problem as two sub-problems. We treat them as a single step, but we select features such that some of the features are best suited for

distinguishing between subjective and objective classes, while some others are engineered to be able to tell a positive tweet from a negative one. The problem with treating the problem as a pipeline of two steps is that we would have to deal with the propagation of errors from one step to the other. If a subjective tweet is mis-classified as an objective one, we rob that tweet of its opportunity of being classified any further in the next step and immediately label them as neutral. This might be detrimental in cases where certain features lead us to believe a tweet is objective, while a combination of all features might rightly lean them towards positive or negative polarities. We take aid from both extrinsic features like emoticons and grapheme stretching as well as intrinsic ones like unigrams and so on.

The list of features we employed is as follows:

Unigrams For each word in a tweet, we look up the vocabulary we generated in the previous step. If the word is present in the vocabulary, we determine its position in the feature vector from the unique ID assigned to it and put 1 in its position. All other positions are 0 by default. Unigram features contribute to understanding both the distinction between subjective and objective tweets as well as between positive and negative tweets. Number of hashtags Inspection of the data lead us to believe that the more the number of hashtags in the tweets, the more the authors involvement with it and hence more the subjectivity.

Presence of URLs A factual tweet is often accompanied by a URL as a proof of its validity or for more enthusiastic of the authors followers to go and explore the news/fact further. Hence, presence of URLs is likely to indicate that the sentence is objective/neutral. For example- Jose Iglesias / Igleisas started at shortstop Wednesday night for the second http://t.co/Gkpx9Blu and Today In History November 02, 1958 Elvis gave a party at his hotel before going out on maneuvers. He sang and... http://t.co/Za9bLTcE

Presence of exclamation marks From our observation, a subjective tweet is much more likely to be ended by exclamation marks than a purely factual or objective tweet. Further, positive tweets are more prone to contain exclamation than negative ones.

Presence of question marks and wh-words Tweets containing question marks or wh-words like why and where are seldom objective. Statistics tells us that this feature can act as a strong cue for not only subjectivity, but also of negativity.

Number of positive/negative emoticons An emoticon is a representation of a facial expression such as a smile or frown, formed by various combinations of keyboard characters and used heavily in tweets to convey the writers feelings or intended tone. Quite intuitively, positive emoticons accompany positive tweets and negative emotions juxtapose negative tweets. More the number of emoticons, it leans with more confidence towards the corresponding polarity. However, a lot of sarcastic tweets also contain positive emoticons, but we do not explicitly handle sarcasm and hence ignore such possibility.

Number of named entities Just like URLs, named entities act as another indication for factual and hence neutral sentences. For example, Remember this? Santorum: Romney, Obama healthcare mandates one and the same http://t.co/sIoG48TO #TheRealRomney @userX @userY. We extract named entities using a python wrapper for the Stanford NER tool (Finkel et al., 2005). However, ablation experiments done after the submission of system in the competition revealed that this feature actually ended up degrading performance by more than 2

Grapheme Stretching Words with characters repeated multiple times (at least twice) herald strong subjectivity, most often positive. For example, Not only is @userZ home from China, shes in LA...I called her and screamed Mandyyyyyyyyyyyyy...Im gonna hug her for 2 hrs tomorrow! and daniel radcliffe was sooo attractive in the 3rd and 5th films omg im in love. We used the number of grapheme stretched words as a feature for our sentiment classifier.

Number of words with unusual capitalization Words with characters made upper-case or lower-case out of turn might potentially convey subjectivity. This feature also proved to slightly degrade performance, during post-competition ablation experiments.

Number of words with all the characters capitalized Strongly positive or strongly negative tweets often have words in all caps in order to convey the excitement that normally the loudness of a voice intones.

Presence of numbers Numbers are used profusely in factual tweets. For example- 13:58 Steven Pourier, Jr. (OLC) MADE the 2nd of the 2 shot Free Throw. DaSU leads 90 - 36 in the 2nd Half. #NAIAMBB. Hence the presence of numbers can be used as an useful feature to distinguish between subjective and objective tweets.

Lexicon features We use 15 lexicon features extracted from publicly available lexicons, which prove to be one of the most powerful features in our features list. Social media data, specially tweets, have a style of language use that is quite different from other text data. We included lexicons which are specially tailored to handle social media data, like Sentiment140 and NRC Hashtag Lexicon.

From Sentiwordnet [1], we extract number of positive tokens, number of negative tokens, total positive sentiment score, total negative sentiment score and maximum sentiment score. From Bing Liu's Opinion Lexicon [30], MPQA Subjectivity Lexicon [74] and NRC Emoticon Association Lexicon [46] we extract Number of positive tokens and number of negative lexicons). We also used Sentiment 140 Lexicon [25] and NRC Hashtag Lexicon [76] to extract sum of sentiment score and maximum sentiment score for each term. Thus we are able to utilize curated lexicon information as additional feature to our classifiers.

3.3.4 Classifier

Once we have extracted all the features, we train a linear SVM using Python based Scikit Learn library [55] for the purpose of classification. We experimentally ascertained the optimal value of the parameter C to be 0.025. In order to cope with the slight class imbalance in the data, we automatically adjust weights inversely proportional to class frequencies.

3.4 Evaluation and Results

This system was built specifically for SemEval 2015 task for Sentiment Analysis on Twitter. The dataset was provided for the shared task - for which training and development data sets were provided apriori, but test dataset was provided at the time of result submission.

The results presented in Table **??** show a comparison of performance of removing various features on SemEval 2015 Twitter Sentiment Analysis dataset. We observe that using all the features specified in previous sections yields an F1 score of 0.567 while the best results are obtained by neglecting the number of entities.

We also tested the performance of our system on other Twitter sentiment analysis datasets along with Live Journal 2014 dataset and SMS 2013 dataset. Table **??** shows the F1 score of results produced by our system in comparison with top performing systems for those particular datasets.

3.5 Conclusion

In this chapter, we studied the problem of sentiment analysis in social media and built a sentiment classifier that also encompasses the features specific to social media. The model presented here can be tweaked to domain specific problems by effective feature extraction. The system is fairly simple and performs slightly below state of art systems.

Chapter 4

Aspect Based Sentiment Analysis

Aspect Based Sentiment Analysis (ABSA) systems receive as input a set of texts (e.g., product reviews or messages from social media) discussing a particular entity. The systems attempt to detect the main aspects (features) of the entity and to estimate the average sentiment of the texts per aspect. [cite ipavlopoulos thesis]. Standard sentiment analysis task tries to detect the overall polarity of a sentence or document regardless of the entities or aspects discussed by the user. ABSA systems are concerned with identifying the specific aspects towards which sentiments are displayed.

For example, an ABSA system might receive product reviews from an e-commerce website like Amazon.

The example on Fig. **??** shows a users review of a camera they brought from Amazon. It clearly discusses the aspects picture quality and battery life. User is positive towards both of these. Meanwhile in the last sentence, user also discusses an implicit aspect, the weight of the camera towards which the review displays a negative aspect.

Some Aspect based sentiment analysis systems also try to determine a category of the mentioned aspects from business perspective. For an e-commerce website review, a sentence can discuss about categories like product quality, packaging, delivery etc. An overview of the components of an ABSA system is shown in Fig. **??**. Each sub-task of the problem is handled separately in various stages. These are explained in details in the later sections.

ABSA systems can help identify fine grain sentiments in other domains like restaurant reviews or political discussions. Fig. **??** shows a review from Zomato, a restaurant review website.

This review discusses a restaurant and comments about the ambience of the rooftop restaurant and food. The user intends to present a positive sentiment towards veg starters and slightly negative sentiment towards non-veg starters. There is explicit mention of food item (kali dal) and the review closes with positive words towards desserts and service.

4.1 Definitions

Lets define the important terms that will be mentioned in this chapter several times.

Aspect Term

Aspect term is a word of phrase present in the review about which an opinion has been expressed. Aspect term is often a feature or facet of the product or entity that has been referred. Aspect has also been called as target in the past literature. Aspects can also be implicit where an aspect term is not present in the sentence but sentiment is expressed towards a concept. In Figure **??**, veg starters, non veg starters, kali dal etc are some of the aspect terms.

Aspect Category

Aspect category is a business category of verious aspect term that can be mentioned in context of the main entity being talked about (restaurant or product). The review in Figure **??** talks about Ambience, Food and Service.

Sentiment Polarity

Sentiment Polarity is specific direction of the sentiment towards an term or entity. In this thesis, sentiment polarity can be positive, negative, or neutral.

Opinion

An opinion consists of two key components: a target g and a sentiment s on the target, i.e., (g, s), where g can be any entity or aspect of the entity about which an opinion has been expressed, and s is a positive, negative, or neutral sentiment, or a numeric rating score expressing the strength/intensity of the sentiment (e.g., 1 to 5 stars). Positive, negative and neutral are called sentiment (or opinion) orientations (or polarities). For example, the target of the opinion in sentence (2) is Canon G12, and the target of the opinion in sentence (3) is the picture quality of Canon G12. Target is also called topic in the literature.

4.2 Challenges

Along with usual challenges that are present in most of the sentiment analysis problems, Aspect Based Sentiment Analysis presents several challenges on it's own.

Multi-Word Aspect

Aspect terms often can be comprised of more than one word. e.g. In our example in Figure **??**, *kali dal* is such an aspect. There can be more complex aspects where the names of dishes are combination of multiple phrases like *Chicken Pizza with Mushroom Toppings* or use conjunctions within the name like *Pumpkin Golden Buns Peking Duck Dinner*.

Implicit Aspects

Many reviews contain reference to aspects which might not be present as a word in the review. e.g. *Very crowded.* talks about the ambience, but the whole sentence is only an adjective phrase and doesn't contain any mention of a noun. Similarly *I ordered dimsums and they were very quick*. Here *they* refers to the waiters but is present only as a pronoun in the sentence.

Unseen Aspects

In the domains like restaurants, there are many aspects which might not have been seen before in the training data. Due to this nature of text in the domain, the methods that rely on word list fail drastically and need NLP or Machine Learning based solutions.

4.3 Past Works

(Hu and Liu, 2004b) [30] initiated works on aspect identification in product reviews using an association rule based system. In his book (Liu, 2012) [38] specifies four methods for aspect extraction, namely, frequent phrases, opinion and target relations, supervised learning and topic models.

Sasha Blair-Goldensohn [5] looked at the problem of summarizing opinions of local services. The designation included restaurants and hotels, but is equally applicable to reviews for a wide variety of entities such as hair salons, schools, museums, retailers, auto shops, golf courses, etc. They created a general system that can handle all services with sufficient accuracy to be of utility to users. They employed standard architecture for aspect-based summarization. For every queried service S, they identify all sentiment laden text fragments in the reviews. Then they proceed to Identify relevant aspects for the sentences that are mentioned in these fragments. They finally aggregate the sentiment over each aspect based on sentiment of mentions in a manner similar to (Bing Liu, 2004) [30]. The first component is a string-based dynamic extractor that looks for frequent nouns or noun compounds in sentiment laden text. They identified aspects as short strings which appear with high frequency in opinion statements, using a series of filters which employ syntactic patterns, relative word frequency, and the sentiment lexicon.

4.4 Aspect Category Detection

The Aspect Category Detection task involves identifying every entity E and attribute A pair E#A towards which an opinion is expressed in the given review.

We take a supervised classification approach where we use C one-vs-all Random Forest Classifiers, for each of the C entity,attribute pairs or aspect categories in the training data, with basic bag of words based approach. We have also tried other features that we explain shortly, but surprisingly the bag of

words approach yielded us the best performance. As a part of the pre-processing procedure, we did the following:

- Removed stop words, except pronouns, because we observed that the category SERVICE#GENERAL can easily be distinguished from other categories by using pronouns as cues
- Stemmed All Words using Porter Stemmer
- Removed pronunciation
- Normalized all numbers by replacing them by zeros, with the motivation that the exact figures do not hold any semantic meaning and are not of importance to us.

Following is the list of features we experimented with:

- Unigrams: For each word in a review, we mark its corresponding position True if it is present in the vocabulary. Presence of number We check if a review sample contains numbers or not, with the motivation reviews talking about the PRICES attribute are more likely to have numbers in them.
- Presence of word in Food and Drinks list [1]: The motivation behind using this feature is, sentences talking about say, FOOD#PRICES and DRINKS#PRICES are likely to use similar words like cheap, expensive, value for money, dollars, etc., but we need to be able to distinguish between the two (FOOD and DRINKS). Hence we use look-up lists for food and drinks with the hope that the customers would explicitly use names of food and drinks items in reviews, wherever applicable.
- WordNet synsets: WordNet is a large lexical database of English. In Wordnet, synonyms or words that denote the same concept and are interchangeable in many contexts, are grouped into unordered sets called synsets. Word forms with several distinct meanings are represented in as many distinct synsets, and hence this feature is useful for capturing semantic information. For each word we find its corresponding synset and use it as a feature for our classifier in a bag of words fashion.
- TF-IDF: Instead of using binary values to denote absence or presence of a word in the sentence, we put its corresponding TF-IDF score pre-computed from the train data. Normally for document classification tasks, TF-IDF performs better than n-grams because the former rightly penalizes common words that are not helpful in distinguishing one topic from the other. Although our Aspect Detection task is very similar to document classification task, this feature did not help much, probably because of the small size of the data set.

[1] Food list compiled from http://eatingatoz.com/food-list/ and Drinks list compiled manually

- Word2Vec: The Word2Vec is an efficient implementation of skip-gram and continuous bag of words architectures that takes a text corpus as input and produces the word vectors of its constituent words as output. We trainedWord2Vec on a corpus comprising Yelp Restaurant reviews data, SemEval 2014 data, SemEval 2015 train data. Let the vector dimension to be D. For each word in a review sentence, we get a vector representation of dimension D. We take an average over all words and end up with a single D-dimensional vector. We experiment with the value of D, which is essentially an optimization over time required to train, and the performance and finally set it to be 30. However, vectors averaged over all words in a sentence are not very good representations for the sentence, which is possibly why this feature did not add much value to our system.

For train and test data were pre-processed and their features extracted in the same way. As for the Random Forest Classifier, we used 50 decision tree estimators using Gini index criterion and at each step we consider only S features when looking for the best split, where S is the square root of the total number of features.

We had also tried hierarchical 2-level classification, i.e. first classifying a review sentence into one of the entities and then classifying them further into one of the pre-defined attributes. However, this 2-level classification technique, with the same set of features mentioned above, yielded poorer performance.

So we decided to not make any distinction between entities and attributes, and consider an entity-attribute pair together as an aspect category. This task required us to categorize reviews into very fine-grained and inter-related categories, with hierarchical dependencies among themselves. This might have been one of the reasons why many of the popular features used for regular document classification did not perform as good as they promised to.

Another challenge was the small number of training examples, as compared to the large number of categories to be classified into, which was not the case in any of the previous works to the best of our knowledge.

4.5 Aspect Term Extraction Methods

4.5.1 Word List Generation

Our initial method used a word list generated from training data and expanded using popular lexicons like Wordnet and WiBi. The end goal of this approach is to generate a powerful word-list during the learning phase, which serves as a lexicon.

Word List Expansion

To prepare the expanded word list, we collect seed aspect terms from the training data. We search for individual terms in publically available lexicons, namely, WordNet and Wibi. For each term, we

extract it's synset. All the words in the synset are synonyms of the current word. We then look for it's neighbors upto a distance of two nodes which give us another terms related to it. e.g. Figure **??** shows that if the word *pasta* is present in our seed list, it can be used to expand to other related terms like *noodles*, *spaghetti* and *macaroni and cheese* by traversing it's neighbors upto two levels.

4.5.2 Dependency Parsing

Dependency parsing is the analysis of grammatical structure of a sentence in order to attain underlying semantics of the sentence. It establishes relationships between *head* words and words which modify those heads. Figure **??** shows a dependency parse of a short sentence.

Here it is observable that such a tool is highly helpful for extracting aspect phrases along with their modifiers which give us the phrases that highlight the opinion of the user. In the example in Figure **??**, pizza is a subject noun which is connected with the adjectives crispy and tasty. We can manually process through a large number of such examples to create a list of rules involving Part of Speech tags and Dependency lables of the edges connecting such words. Such combinations can be used directly with coordination of a sentiment lexicon to provide a simple solution for Aspect Based Sentiment Analysis. Our approaches utilizes such dependencies in conjunction with other heuristics and sequence labelling approaches to provide more reliable results.

4.5.3 Hybrid Method using Sequence Labelling

Given a review sentence, the aim of this task is to find the Opinion Target Expressions (OTE), that is, the particular attribute of the entity the user expresses his/her sentiment about. Aspects may either be explicitly explained in the review as in the sentence The service was really quick and I loved the fajitas. Here service and fajita are explicit aspects. In a sentence like Dont go. Really horrible, the user didnt use any individual term but still gives an impression of her sentiment. In such cases, the slot takes the value NULL. Our system uses a sequence labelling approach to tackle this problem by the use of Conditional Random Fields. The tagger from Mallet toolkit, is trained to identify three possible tags, namely BEG and INT for beginning and intermediate target words and OTH for other words.

Considered Features are as follows:

- Word: The lowercase form of the word itself
- POS: Part of speech tag of the word
- Dependency: We use two kinds of dependency features the dependency label on incoming edge on the word, and the first dependency label on outgoing edge. This proved to be a very important feature.
- Capitalization: If the first character of the word is in capital, mark it as capital.
- Punctuation: If the word contains any nonalphanumeric character, we mark it as punctuation.

- Seed: The word is marked as a seed if it was present in the seed-list created by collecting all the OTEs in the training data, splitting them by word and removing all the stop words.
- Brown Cluster: Brown Cluster ID is obtained by first training Brown Clustering on the same corpus we described for Word2Vec features in Subtask 1. Brown clustering is a form of hierarchical clustering of words based on the context in which they occur. The intuition behind the method is that a class-based language model where probabilities of words are based on the clusters of previous words, can overcome the data sparsity problem inherent in language modeling. From brown clustering, for each word in the corpus we get the cluster ID to which it has been assigned. We generate 100 clusters.
- Presence in Expanded List: We curated an expanded seed list from the original seed list explained above. We utilized WiBi, which is a taxonomy of Wikipedia pages and categories. We traversed the WiBi Page graph and collected the pages located next to the words (if present) in the seed list. The new list was again split by spaces and punctuation, and stop words were removed. This feature is marked if the term is present in the expanded list.
- Stop Word: This feature is marked if the current term is a stop word in English language.
- Seed Stem: This feature contains the stemmed form of the original word as obtained from Porter Stemmer.

4.5.4 Sentiment Polarity Classification

In this subtask, the input consists of a review sentence and the set of aspect categories it belongs to. The expected output is a polarity label for each of the associated aspect categories. We have first extracted Bag of Words and Wordnet Synset features from both train and test data. Then we run a variety of classifiers (like Stochastic Gradient Descent, SVM, Adaboost) multiple times and store the confidence scores obtained from decision functions of each of these classifiers. Finally we build a linear SVM classifier that uses the scores obtained from the classifiers in the 1st level as features, along with 15 other hand-crafted lexicon features as explained in Section 5.2. This is also known as stacking, a form of ensemble learning. It is essentially stacking of classifiers inside a classifier. Stacking typically yields performance better than any single one of the trained models, and this is what we wanted to leverage. However, since we need polarity labels per aspect category, we need to identify the segments in the sentence that deals with each of the categories and then treat those segments as individual samples for polarity detection. For example, if there are three aspect categories associated with a sentence, we want to break it down into 3 sentence,category pairs:

$$sent1, \{cat1, cat2, cat3\} \rightarrow \{sent1, cat2\}, \{sent2, cat2\}, \{sent3, cat3\}$$

For each {sentence, category} pair, we find a word in the sentence that is the best representative of the category, which we call as centroid. Then we take a window of n words surrounding the centroid and

consider that window to be the segment of interest for that category. So in this example sentence, we need to have three centroids and hence three segments, not necessarily disjoint:

$$\{sent1, cat1\}, \{sent2, cat2\}, \{sent3, cat3\} \rightarrow \{sent1, cat2\}, \{sent2, cat2\}, \{sent3, cat3\}$$

We experimented with the window size, and decided upon using a window size of 3 words to the left and to the right of the centroid. It is interesting to note that among sentences that have more than one category, the average length of a review sentence is 15 words in train data and 17 words in test data. After we get these segments, we extract the following features from these segments for polarity detection:

- Bag of Words
- Grapheme Stretching i.e. words with repeated characters. For example, words like Tooooo goooood indicates strong subjectivity and therefore is less likely to belong to Neutral class.
- Presence of exclamation also signals subjectivity, usually positivity.
- Presence of wh-words and conditional words like why, what,if, etc. Observation tells us that such presence are mostly characteristic of sentences with negative polarity.
- Wordnet Synsets, as explained before

While bag of words features include statistical information, WordNet synsets help incorporate semantic information. These two complementary features help us in making the maximal discrimination among the target classes.

4.5.5 Extracting Centroid for a {Sentence, Category} Pair

We automatically generate a set of seed words for each of the aspect categories by the following technique: From the train data, we consider all sentences labelled with a single category as a single document. As a result, for 13 possible categories in the train data, we have 13 documents. Now for each document (corresponding to each category), we compute the TF-IDF scores of all the words and consider words having TF-IDF greater than a certain threshold as seed words for that category. We ascertain the optimal value of the threshold to be 0.2 through experimentation. We generate a co-occurrence matrix of words from three datasets SemEval 2015 train data, SemEval 2014 train and test data. Typically, it is considered that two words co-occur if they are present as bigram in the corpus. However, we define co-occurrence as occurring in the same review sentence, rather than occurring as a bigram as it is less likely to find repetition of co-occurring bigrams in a smaller corpus. This co-occurrence matrix stores the frequency of co-occurrence of two words in the corpus. For N words in the vocabulary, we have a N N co-occurrence matrix.

Given a {sentence, category} pair, for each word in the review sentence, we find the Point wise Mutual Information (P.M.I.) between that word and each word in the seed list of the assigned category

and take their average for that word. We do the same for all words in the sentence. The word in the sentence having the maximum average P.M.I. score is defined as the centroid for the {sentence, category} pair. P.M.I is defined as the ratio of the probability of occurrence of two words together in the corpus to the product of the probabilities of occurrence of the two words independently in the corpus. We derive the co-occurrence frequencies from the co-occurrence matrix we built in the previous step.

4.5.6 Ensemble Learning - Stacking Classifiers

After feature extraction, we train 3 kinds of classifiers — Linear Support Vector Machines, Stochastic Gradient Descent and Adaboost, for each of the features — Bag of words and Wordnet Synsets. We repeat the process K times where $K \in \mathbb{Z}$. We have experimentally chosen K to be 30 which controls trade off between the time taken to train the model and the performance improvements. As we increased K over 30, the improvement in performance started to diminish. For each test sample, we obtain 3 scores (corresponding to three classes — positive, negative, neutral) from the decision function of each classifier.

We use these confidence scores as features along with 15 other hand-crafted lexicon features for a linear Support Vector Machine classifier. We employ features such as number of positive tokens, number of negative tokens, total positive sentiment score, total negative sentiment score, sum of sentiment scores, maximum sentiment score, etc. from Sentiwordnet [1], Bing Lius opinion lexicon [30], MPQA subjectivity lexicon [74], NRC Emotion Association lexicon [46], Sentiment140 lexicon [25], and NRC Hashtag Lexicon [76]. From Sentiwordnet [1], we extract number of positive tokens, number of negative tokens, total positive sentiment score, total negative sentiment score and maximum sentiment score. From Bing Liu's Opinion Lexicon [30], MPQA Subjectivity Lexicon [74] and NRC Emoticon Association Lexicon [46] we extract Number of positive tokens and number of negative lexicons). We also used Sentiment 140 Lexicon [25] and NRC Hashtag Lexicon [76] to extract sum of sentiment score and maximum sentiment score for each term. Thus we are able to utilize curated lexicon information as additional feature to our classifiers.

For the final linear SVM classifier, we experimentally ascertain the optimal value of the parameter C to be 0.024. The linear SVM classifiers, in the first level of stacking, had a default value of 1.0 for parameter C. We did not have enough time to tune them, as we had many classifiers inside the main SVM classifier. The Ada Boost Classifier uses 100 decision tree estimators and a default learning rate of 1. We have used Scikit Learn for building all the classifiers. Although we employ several classiers, the time taken is negligible. This is because the different classifiers in the first stage of stacking can be trained in parallel quite easily.

4.6 Results

We submitted unconstrained systems for the Restaurants dataset. We did not run our system for other domains mainly due to lack of time during the competition. Table **??**, Table **??** and Table **??** present the results of ablation experiments carried out for category identification, aspect extraction and sentiment polarity detection respectively. We show the effect of varying the size of the context window surrounding the centroid, on F1-score in Figure **??**. Finally Table **??** compares our ensemble system with a baseline system trained on a single linear SVM with only lexicon features.

4.7 Conclusion

This chapter explains the problem of Aspect Based Sentiment Analysis and discusses our approach towards various components of the problem. The final system was submitted for SemEval 2015 Task 12. For all the three subtasks, our system performs quite well, ranking between 4th and 8th. We experimented with Ensemble Learning technique using stacking of classifiers, which can be further explored and improved. Future work would adapt the system to other domains, or progress towards building a domain independent system.

Chapter 5

Sentiment Analysis on Code Mixed Text

Code Mixing is a natural phenomenon of embedding linguistic units such as phrases, words or morphemes of one language into an utterance of another [47, 17, 28]. Code-mixing is widely observed in multilingual societies like India, which has 22 official languages most popular of which are Hindi and English. With over 375 million Indian population online, usage of Hindi has been steadily increasing on the internet.

This opens up tremendous potential for research in sentiment and opinion analysis community for studying trends, reviews, events, human behaviour as well as linguistic analysis. Most of the current research works have involved sentiment polarity detection [23, 38, 52] where the aim is to identify whether a given sentence or document is (usually) positive, negative or neutral. Due to availability of large-scale monolingual corpora, resources and widespread use of the language, English has attracted the most attention.

5.1 Languages in the Social Web

Seminal work in sentiment analysis of Hindi text was done by [34] in which the authors built three step fallback model based on classification, machine translation and sentiment lexicons. They also observed that their system performed best with unigram features without stemming. [2] generated a sentiment lexicon for Hindi and validated the results on translated form of Amazon Product Dataset [6]. [16] created Hindi SentiWordNet, a sentiment lexicon for Hindi.

Sentiment Analysis in Code-mixed languages has recently started gaining interest owing to the rising amount of non-English speaking users. [65] segregated Hindi and English words and calculated final sentiment score by lexicon lookup in respective sentient dictionaries.

Hindi-English (Hi-En) code mixing allows ease-of-communication among speakers by providing a much wider variety of phrases and expressions. A common form of code mixing is called as *romanization* [1], which refers to the conversion of writing from a different writing system to the Roman script. But this freedom makes the task for developing NLP tools more difficult, highlighted by [9, 71, 3].

[1] https://en.wikipedia.org/wiki/Romanization

Initiatives have been taken by shared tasks [64, 66], however they do not cover the requirements for a sentiment analysis system.

The *romanized* code mixed data on social media presents additional inherent challenges such as contractions like ”between” $\rightarrow$ ”btwn”, non-standard spellings such as ”cooolll” or ”bhut bdiya” and non-grammatical constructions like ”sir hlp plzz naa”. Hindi is phonetically typed while English (Roman script) doesn't preserve phonetics in text. Thus, along with diverse sentence construction, words in Hindi can have diverse variations when written online, which leads to large amount of tokens, as illustrated in Table **??**. Meanwhile there is a lack of a suitable dataset.

5.2 Dataset Preparation

We collected user comments from public Facebook pages popular in India. We chose pages of Salman Khan, a popular Indian actor with massive fan following, and Narendra Modi, the current Prime Minister of India. The pages have 31 million and 34 million facebook user likes respectively. These pages attract large variety of users from all across India and contain lot of comments to the original posts in code-mixed representations in varied sentiment polarities. We manually pre-processed the collected data to remove the comments that were not written in roman script, were longer than 50 words, or were complete English sentences. We also removed the comments that contained more than one sentence, as each sentence might have different sentiment polarity.

Then, we proceeded to manual annotation of our dataset. The comments were annotated by two annotators in a 3-level polarity scale - positive, negative or neutral. Only the comments with same polarity marked by both the annotators are considered for the experiments. They agreed on the polarity of 3879 of 4981 (77%) sentences. The Cohen's Kappa coefficient [13] was found to be 0.64. We studied the reasons for misalignment and found that causes typically were due to difference in perception of sentiments by individuals, different interpretations by them and sarcastic nature of some comments which is common in social media data. The dataset contains 15% negative, 50% neutral and 35% positive comments owing to the nature of conversations in the selected pages.

The dataset exhibits some of the major issues while dealing with code-mixed data like short sentences with unclear grammatical structure. Further, *romanization* of Hindi presents an additional set of complexities due to loss of phonetics and free ordering in sentence constructions as shown in Table **??**. This leads to a number of variations of how words can be written. Table **??** contains some of the words with multiple spelling variations in our dataset, which is one of the major challenges to tackle in Hi-En code-mixed data.

5.3 Learning Compositionality

Our target is to perform sentiment analysis on the above presented dataset. Most commonly used statistical approaches learn word-level feature representations. We start our exploration for suitable algorithms from models having word-based representations.

5.3.1 Word-level models

Word2Vec[45] and Word-level RNNs (Word-RNNs) [70] have substantially contributed to development of new representations and their applications in NLP such as in Summarization [7] and Machine Translation [10]. They are theoretically sound since language consists of inherently arbitrary mappings between ideas and words. Eg: The words person(English) and insaan(Hindi) do not share any priors in their construction and neither do their constructions have any relationship with the semantic concept of a person. Hence, popular approaches consider lexical units to be independent entities. However, operating on the lexical domain draws criticism since the finite vocabulary assumption; which states that models assume language has finite vocabulary but in contrast, people actively learn & understand new words all the time.

Excitingly, our dataset seems suited to validate some of these assumptions. In our dataset, vocabulary sizes are greater than the size of the dataset. Studies on similar datasets have shown strong correlation between number of comments and size of vocabulary [63]. This rules out methods like Word2Vec, N-grams or Word-RNNs which inherently assume a small vocabulary in comparison to the data size. The finite vocabulary generally used to be a good approximation for English, but is no longer valid in our scenario. Due to the high sparsity of words themselves, it is not possible to learn useful word representations. This opens avenues to learn non-lexical representations, the most widely studied being character-level representations, which is discussed in the next section.

5.3.2 Character-level models

Character-level RNNs (Char-RNNs) have recently become popular, contributing to various tasks like [35]. They do not have the limitation of vocabulary, hence can freely learn to generate new words. This freedom, in fact, is an issue: Language is composed of lexical units made by combining letters in some specific combinations, i.e. most of the combinations of letters do not make sense. The complexity arises because the mappings between meaning and its construction from characters is arbitrary. Character models may be apriori inappropriate models of language as characters individually do not usually provide semantic information. For example, while " $King - Man + Women = Queen$" is semantically interpretable by a human, "$Cat - C + B = Bat$" lacks any linguistic basis.

But, groups of characters may serve semantic functions. This is illustrated by $Un+Holy = Unholy$ or $Cat+s = Cats$ which is semantically interpretable by a human. Since sub-word level representations can generate meaningful lexical representations and individually carry semantic weight, we believe

that sub-word level representations consisting composition of characters might allow generation of new lexical structures and serve as better linguistic units than characters.

5.3.3 Sub-word level representations

Lexicon based approaches for the SA task [69, 65] perform a dictionary look up to obtain an individual score for words in a given sentence and combine these scores to get the sentiment polarity of a sentence. We however want to use intermediate sub-word feature representations learned by the filters during convolution operation. Unlike traditional approaches that add sentiment scores of individual words, we propagate relevant information with LSTM and compute final sentiment of the sentence as illustrated in Figure **??**.

Hypothesis: We propose that incorporating sub-word level representations into the design of our models should result in better performance. This would also serve as a test scenario for the broader hypothesis proposed by Dyer et. al. in his impressive ICLR keynote [2] - Incorporating linguistic priors in network architectures lead to better performance of models.

Methodology: We propose a method of generating sub-word level representations through 1-D convolutions on character inputs for a given sentence. Formally, let C be the set of characters and T be an set of input sentences. The sentence $s \in T$ is made up of a sequence of characters $[c_1, ..., c_l]$ where l is length of the input.

Hence, the representation of the input s is given by the matrix $Q \in R^{d \times l}$ where d is the dimensionality of character embedding that corresponding to $[c_1, ..., c_l]$. We perform convolution of Q with a filter $H \in R^{d \times m}$ of length m after which we add a bias and apply a non-linearity to obtain a feature map $f \in R^{l-m+1}$. Thus we can get sub-word level (morpheme-like) feature map. Specifically, the i^{th} element of f is given by:

$$f[i] = g((Q[:, i : i + m - 1] * H) + b) \tag{5.1}$$

where $Q[:, i : i + m - 1]$ is the matrix of $(i)^{th}$ to $(i + m - 1)^{th}$ character embedding and g corresponds to ReLU non-linearity.

Finally, we pool the maximal responses from p feature representations corresponding to selecting sub-word representations as:

$$y_i = max(f[p * (i : i + p - 1)]) \tag{5.2}$$

Next, we need to model the relationships between these features $y^i[:]$ in order to find the overall sentiment of the sentence. This is achieved by LSTM[27] which is suited to learning to propagate and 'remember' useful information, finally arriving at a sentiment vector representation from the inputs.

We provide f_t as an input to the memory cell at time t. We then compute values of I_t - the input gate, $\tilde{C}_t$ - the candidate value for the state of the memory cell at time t and f_t - the activation of the forget

[2] Available at: http://videolectures.net/iclr2016_dyer_model_architecture/

gate, which can be used to compute the information stored in memory cell at time t. With the new state of memory cell C_t, we can compute the output feature representation by:

$$O_t = \sigma(Wy_t + Uh_(t-1) + V(C_t + b) \tag{5.3}$$

$$h_t = O_t tanh(C_t) \tag{5.4}$$

where W,U and V are weight matrices and b_i are biases. After l steps, h_l represents the relevant information retained from the history. That is then passed to a fully connected layer which calculates the final sentiment polarity as illustrated in the Figure **??**.

Figure **??** gives schematic overview of the architecture. We perform extensive experiments to qualitatively and quantitatively validate the above claims as explained in the next section.

5.4 Experimental Setup

Our dataset is divided into 3 splits- Training, validation and testing. We first divide the data into randomized 80-20 train test split, then further randomly divide the training data into 80-20 split to get the final training, validation and testing data.

As the problem is relatively new, we compare state of the art sentiment analysis techniques [72, 52] which are generalizable to our dataset. We also compare the results with system proposed by [65] on our dataset. As their system is not available publicly, we implemented it using language identification and transliteration using the tools provided by [4] for Hi-En Code Mixed data. The polarity of thus obtained tokens is computed from SentiWordNet [22] and Hindi SentiWordNet [16] to obtain the polarity of words, which are then voted to get final polarity of the sentence.

The architecture of the proposed system (Subword-LSTM) is described in Figure **??**. We compare it with a character-level LSTM (Char-LSTM) following the same architecture without the convolutional and maxpooling layers. We use Adamax [36] (a variant of Adam based on infinity norm) optimizer to train this setup in an end-to-end fashion using batch size of 128. We use very simplistic architectures because of the constraint on the size of the dataset. As the datasets in this domain expand, we would like to scale up our approach to bigger architectures. The stability of training using this architecture is illustrated in Figure **??**.

5.4.1 Validation of proposed hypothesis

We obtain preliminary validation for our hypothesis that incorporating sub-word level features instead of characters would lead to better performance. Our Subword-LSTM system provides an F-score of 0.658 for our dataset, which is significantly better than Char-LSTM which provides F-score of 0.511.

Since we do not have any other dataset in Hi-En code-mixed setting of comparable to other settings, we performed cross-validation of our hypothesis on SemEval'13 Twitter Sentiment Analysis dataset.

We took the raw tweets character-by-character as an input for our model from the training set of 7800 tweets and test on the SemEval'13 development set provided containing 1368 tweets. The results are summarized in Table **??**. In all the cases, the text was converted to lowercase and tokenized. No extra features or heuristics were used.

5.5 Observations

In the comparative study performed on our dataset, we observe that Multinomial Naive Bayes performs better than SVM[52] for snippets providing additional validation to this hypothesis given by [72].

We also observe that unigrams perform better than bigrams and Bag of words performs better than tf-idf in contrast to trends in English, as the approaches inducing more sparsity would yield to poorer results because our dataset is inherently very sparse. The lexicon lookup approach [65] didn't perform well owing to the heavily misspelt words in the text, which led to incorrect transliterations as shown in Table **??**.

We obtain preliminary validation for our hypothesis that incorporating sub-word level features instead of characters would lead to better performance. Our Subword-LSTM system provides an F-score of 0.658 for our dataset, which is significantly better than Char-LSTM which provides F-score of 0.511.

Since we do not have any other dataset in Hi-En code-mixed setting of comparable to other settings, we performed cross-validation of our hypothesis on SemEval'13 Twitter Sentiment Analysis dataset. We took the raw tweets character-by-character as an input for our model from the training set of 7800 tweets and test on the SemEval'13 development set provided containing 1368 tweets. The results are summarized in Table **??**. In all the cases, the text was converted to lowercase and tokenized. No extra features or heuristics were used.

5.5.1 Visualizing character responses

Visualizations in Figure **??** shows how the proposed model is learning to identify sentiment lexicons. We see that different filters generally tend to learn mappings from different parts, interestingly showing shifting trends to the right which maybe due to LSTM picking their feature representation in future time steps. The words sections that convey sentiment polarity information are captured despite misspelling in example (i) and (ii). In example (iii), starting and ending phrases show high response which correspond to the sentiment conveying words (party and gift). The severe morpheme stretching in example (iv) also affects the sentiment polarity.

5.6 Conclusion

We introduce Sub-Word Long Short Term Memory model to learn sentiments in a noisy Hindi-English Code Mixed dataset. We discuss that due to the unavailability of NLP tools for Hi-En Code

Mixed text and noisy nature of such data, several popular methods for Sentiment Analysis are not applicable. The solutions that involve unsupervised word representations would again fail due to sparsity in the dataset. Sub-Word LSTM interprets sentiment based on morpheme-like structures and the results thus produced are significantly better than baselines.

Further work should explore the effect of scaling of RNN and working with larger datasets on the results. In the new system, we would like to explore more deep neural network architectures that are able to capture sentiment in Code Mixed and other varieties of noisy data from the social web.

Chapter 6

Conclusions

In this thesis, we have explored approaches for three sentiment analysis sub-problems. We presented our system for sentiment analysis on twitter feed, where we extracted several features which help us compute the overall sentiment of a subjective message from social media. Such features are specifically targeted towards noisy text from social media. We built a SVM based classifier with these features. Experiments on the SemEval 2015 dataset for Twitter Sentiment Analysis showed that the proposed approach achieves significant performance gains over the baseline at F1 measure of 58.68%. We also showed the results of our system with other standard sentiment analysis data sets specific to social media.

We then discussed the problem of Aspect Based Sentiment Analysis. The ABSA task is divided into three major subtasks: (1) Aspect Term Extraction, (2) Aspect Category Identification, and, (3) Aspect Sentiment Prediction. Aspect Based Sentiment Analysis addresses a number of limitations faced by traditional sentiment analysis techniques. It solves the problem of review-level sentiment prediction that identifies the sentiments at aspect level, and generalizes them to the business category it is related to.

For Aspect Term Extraction, we presented a sequence labelling based approach that uses Conditional Random Fields (CRF). We used various features including presence of a word in word lists, POS tags of words, information of clusters the term is present in, and some other syntactic information. Using these features we obtained an F1 score of 57% which stood at 8th rank in SemEval Aspect Based Sentiment Analysis task.

For Category Identification, we used C one-vs-all Random Forest Classifiers,for each of the C entity,attribute pairs or aspect categories in the training data, with basic bag of words based approach. For sentiment computation, we proposed an ensemble classification method that stacks multiple classifiers that have individually presented better results. The confidence scores of individual classifiers are used as final features to a meta-classifier which finally classifies an input. We have obtained an F1 Score of 57.0% and 71.6% respectively.

Our third task revolves around a relatively new problem in Sentiment Analysis. Rather than text present in a particular language, we attempt to deal with text using lexical units of one language written in another script, which is referred as Code Mixing. We generated a dataset from Facebook comments which were labelled for sentiment polarity. Each sentence in the dataset was annotated by two annotators, and only those sentences which had same polarity marked by both were considered for the experiment.

In initial experiments, we worked with standard baseline methods for sentiment analysis, and tried to normalize the spelling variations. The presented method using Subword-LSTM surpasses other methods by a good margin with F-Score at 65.8%.

Further Solutions

Sentiment Analysis is a category of problems which can be helpful in several ways. Another path that seems very promising to explore is working with aggregate data on a specific entity. Rather than computing sentiment for a given sentence *(coarse grain)*, or find sentiment for an aspect in the text *(fine grain)*, we can rather attempt to summarize the overall sentiments regarding an entity.

The field of processing Code Mixed text has recently started gaining attention, especially since SAIL Code Mixed Shared Task [54] in ICON 2017. Several teams presented novel ideas for the shared task on sentiment analysis of code-mixed data pairs of Hindi-English and Bengali-English collected from the different social media platform. As a part of this shared task, IIITNBP combined the idea of leveraging word embeddings with tfidf of the word and character nGrams and proceed to ensemble voting. JU_KS used nGrams and sentiment lexicons for words in English and Bengali and used MNB for classification. Another recent work (Pratapa et al. 2018) [59] proposed the idea of obtaining bilingual embeddings for NLP tasks in code mixed text.

6.1 Future Work

The purpose of this thesis is to explore various methods for recent problems in the area of sentiment analysis. In this section, we will discuss some of the ideas for extending the experiments and extending the research.

Generation of Data Sources

In chapter 5, we present a dataset that we built specifically for the problem of sentiment analysis in Code Mixed text. To explore and come up with new solutions, we need much larger datasets. Just like English-Hindi code mixing, we can experiment with other such language combinations used in the web.

Deep Learning Techniques for Aspect Extraction

The aspect extraction methods presented in this thesis in Chapter 3 evolved from the standard methods involving heuristics and learning based approaches. However recently deep learning solutions have been found highly helpful for sequence labelling tasks as well. This would require substantial amount of work but would highly improve the results. This system was built in Python using keras for deep learning solutions.

With the rise in intelligent consumer products, sentiment analysis plays a major role in various mediums of interaction - text, speech or a combination of these. User generated content is rapidly increasing - and as we have discussed through-out this thesis, sentiment analysis and opinion mining is present as a tool for decision making process. The enhancements from theoretical, as well as applications point of view will be required in the coming years.

Related Publications

Aditya Joshi, Ameya Prabhu, Manish Shrivastava, Vasudeva Varma. (2016, Dec). **Towards Sub-Word Level Compositions for Sentiment Analysis of Hindi-English Code Mixed Text.** In Proceedings of 26th International Conference on Computational Linguistics, (COLING 2016), December 11-16, 2016, Osaka, Japan

Aditya Joshi, Satarupa Guha, Vasudeva Varma. (2015) **Sentibase: Sentiment Analysis in Twitter on a Budget**. In SemEval@NAACL-HLT 2015: 590-594, Denver, Colorado, USA

Aditya Joshi, Satarupa Guha, Vasudeva Varma. (2015) **SIEL: Aspect Based Sentiment Analysis in Reviews.** SemEval@NAACL-HLT 2015: 759-766, Denver, Colorado, USA

Bibliography

[1] S. Baccianella, A. Esuli, and F. Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*, volume 10, pages 2200–2204, 2010.

[2] A. Bakliwal, P. Arora, and V. Varma. Hindi subjective lexicon: A lexical resource for hindi polarity classification. In *Proceedings of International Conference on Language Resources and Evaluation*, 2012.

[3] U. Barman, A. Das, J. Wagner, and J. Foster. Code-mixing: A challenge for language identification in the language of social media. In *In Proceedings of the First Workshop on Computational Approaches to Code-Switching*, 2014.

[4] I. A. Bhat, V. Mujadia, A. Tammewar, R. A. Bhat, and M. Shrivastava. Iiit-h system submission for fire2014 shared task on transliterated search. In *Proceedings of the Forum for Information Retrieval Evaluation*, FIRE '14, pages 48–53, New York, NY, USA, 2015. ACM.

[5] S. Blair-Goldensohn, K. Hannan, R. McDonald, T. Neylon, G. A. Reis, and J. Reynar. Building a sentiment summarizer for local service reviews. In *WWW workshop on NLP in the information explosion era*, volume 14, pages 339–348, 2008.

[6] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *In ACL*, pages 187–205, 2007.

[7] Z. Cao, F. Wei, L. Dong, S. Li, and M. Zhou. Ranking with recursive neural networks and its application to multi-document summarization. In *AAAI*, pages 2153–2159, 2015.

[8] D. Chen and C. D. Manning. A fast and accurate dependency parser using neural networks. In *EMNLP*, pages 740–750, 2014.

[9] G. Chittaranjan, Y. Vyas, K. Bali, and M. Choudhury. Word-level language identification using crf: Code-switching shared task report of msr india system. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pages 73–79, 2014.

[10] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[11] F. Chollet et al. Keras. https://github.com/fchollet/keras, 2015.

[12] R. B. Cialdini. Influence: Science and practice. *Boston: Allyn & Bacon*, 2001.

[13] J. Cohen. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46, Apr. 1960.

[14] CoreNLP. http://stanfordnlp.github.io/CoreNLP/. [Online; accessed 25-Dec-2016].

[15] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, Sep 1995.

[16] A. Das and S. Bandyopadhyay. Sentiwordnet for indian languages. In *Proceedings of the Eighth Workshop on Asian Language Resouces*, 2010.

[17] L. Duran. Toward a better understanding of code switching and interlanguage in bilinguality: Implications for bilingual instruction. *Journal of Educational Issues of Language Minority Students*, 14:69–87, 1994.

[18] A. G. Emory. A first look at communication theory, 1997.

[19] A. Esuli and F. Sebastiani. Determining the semantic orientation of terms through gloss classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 617–624. ACM, 2005.

[20] A. Esuli and F. Sebastiani. Determining term subjectivity and term orientation for opinion mining. In *EACL*, volume 6, page 2006, 2006.

[21] A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422. Citeseer, 2006.

[22] A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC06*, pages 417–422, 2006.

[23] R. Feldman. Techniques and applications for sentiment analysis. *Commun. ACM*, 56(4):82–89, Apr. 2013.

[24] J. R. Fontaine, K. R. Scherer, E. B. Roesch, and P. C. Ellsworth. The world of emotions is not two-dimensional. *Psychological science*, 18(12):1050–1057, 2007.

[25] A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1:12, 2009.

[26] N. Godbole, M. Srinivasaiah, and S. Skiena. Large-scale sentiment analysis for news and blogs. *ICWSM*, 7(21):219–222, 2007.

[27] A. Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

[28] M. Gysels. French in urban lubumbashi swahili: Codeswitching, borrowing, or both. *Journal of Multilingual and Multicultural Development*, 13:41–55, 1992.

[29] J. W. Y. Ho et al. Code-mixing: Linguistic form and socio-cultural meaning. *The International Journal of Language Society and Culture*, 21, 2007.

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

[31] T. Jaakkola, M. Meila, and T. Jebara. Maximum entropy discrimination. In *Advances in neural information processing systems*, pages 470–476, 2000.

[32] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001. [Online].

[33] K. S. Jones. A statistical interpretation of term specificity and its application in information retrieval. *Journal of Documentation*, 28(1):11–21, 1972.

[34] A. Joshi, R. Balamurali, and B. Pushpak. A fall-back strategy for sentiment analysis in hindi: a case study. In *Proceedings of the 8th ICON*, Stroudsburg, PA, 2010. Association for Computational Linguistics.

[35] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush. Character-aware neural language models. *CoRR*, abs/1508.06615, 2015.

[36] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[37] P. Langley, W. Iba, and, and K. Thompson. An analysis of bayesian classifiers. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, AAAI'92, pages 223–228. AAAI Press, 1992.

[38] B. Liu. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2012.

[39] E. Loper and S. Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[40] H. P. Luhn. A statistical approach to mechanized encoding and searching of literary information. *IBM J. Res. Dev.*, 1(4):309–317, Oct. 1957.

[41] W. C. Mann and S. A. Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281, 1988.

[42] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014.

[43] A. K. McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[44] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[45] T. Mikolov, W.-t. Yih, and G. Zweig. Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751, 2013.

[46] S. M. Mohammad, S. Kiritchenko, and X. Zhu. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the seventh international workshop on Semantic Evaluation Exercises (SemEval-2013)*, Atlanta, Georgia, USA, June 2013.

[47] P. Muysken. *Bilingual Speech: A Typology of Code-mixing*. Cambridge University Press, 2000.

[48] K. Nigam. Using maximum entropy for text classification. In *In IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61–67, 1999.

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

[50] C. E. Osgood, G. J. Suci, and P. H. Tannenbaum. The measurement of meaning. *Urbana: University of Illinois Press*, 1957.

[51] A. Pak and P. Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10, pages 1320–1326, 2010.

[52] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135, Jan. 2008.

[53] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[54] B. Patra, D. Das, and A. Das. Sentiment analysis of code-mixed indian languages: An overview of sail$_code-mixedsharedtask@icon - 2017.032018$.

[55] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

[56] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830, Nov. 2011.

[57] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 27–35. Citeseer, 2014.

[58] R. Prasad, N. Dinesh, A. Lee, E. Miltsakaki, L. Robaldo, A. K. Joshi, and B. L. Webber. The penn discourse treebank 2.0. In *LREC*. Citeseer, 2008.

[59] A. Pratapa, M. Choudhury, and S. Sitaram. Word embeddings for code-mixed language processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, 2018.

[60] D. T. Robinson, L. Smith-Lovin, and A. K. Wisecup. Affect control theory. In *Handbook of the sociology of emotions*, pages 179–202. Springer, 2006.

[61] S. Rosenthal, P. Nakov, S. Kiritchenko, S. M. Mohammad, A. Ritter, and V. Stoyanov. Semeval-2015 task 10: Sentiment analysis in twitter. *Proceedings of SemEval-2015*, 2015.

[62] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A bayesian approach to filtering junk e-mail. Technical report, PAPERS FROM THE 1998 WORKSHOP, AAAI, 1998.

[63] H. Saif, M. Fernandez, Y. He, and H. Alani. Evaluation datasets for twitter sentiment analysis: A survey and a new dataset, the sts-gold. *1st Interantional Workshop on Emotion and Sentiment in Social and Expressive Media: Approaches and Perspectives from AI (ESSEM 2013)*, 2013.

[64] R. Sequiera, M. Choudhury, P. Gupta, P. Rosso, S. Kumar, S. Banerjee, S. K. Naskar, S. Bandyopadhyay, G. Chittaranjan, A. Das, and K. Chakma. Overview of FIRE-2015 shared task on mixed script information retrieval. In P. Majumder, M. Mitra, M. Agrawal, and P. Mehta, editors, *Post Proceedings of the Workshops at the 7th Forum for Information Retrieval Evaluation, Gandhinagar, India, December 4-6, 2015.*, volume 1587 of *CEUR Workshop Proceedings*, pages 19–25. CEUR-WS.org, 2015.

[65] S. Sharma, P. Srinivas, and R. C. Balabantaray. Text normalization of code mix and sentiment analysis. In *2015 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2015, Kochi, India, August 10-13, 2015*, pages 1468–1473, 2015.

[66] T. Solorio, E. Blair, S. Maharjan, S. Bethard, M. Diab, M. Gohneim, A. Hawwari, F. AlGhamdi, J. Hirschberg, A. Chang, and P. Fung. Overview for the first shared task on language identification in code-switched data. In *Proceedings of The First Workshop on Computational Approaches to Code Switching, held in conjunction with EMNLP 2014.*, pages 62–72, Doha, Qatar, 2014. ACL.

[67] P. J. Stone, D. C. Dunphy, and M. S. Smith. The general inquirer: A computer approach to content analysis. 1966.

[68] R. Sussman and R. Gifford. Be the change you want to see modeling food composting in public places. *Environment and Behavior*, 45(3):323–343, 2013.

[69] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede. Lexicon-based methods for sentiment analysis. *Comput. Linguist.*, 37(2):267–307, June 2011.

[70] M. thang Luong, R. Socher, and C. D. Manning. Better word representations with recursive neural networks for morphology. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics*, pages 104–113, 2013.

[71] Y. Vyas, S. Gella, Jatin, K. Bali, and M. Choudhury. Pos tagging of english-hindi code-mixed social media content. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 974–979, 2014.

[72] S. Wang and C. D. Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pages 90–94, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[73] B. Webber, M. Egg, and V. Kordoni. Discourse structure and language technology. *Natural Language Engineering*, 18(04):437–490, 2012.

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

[75] S. Zhu, X. Ji, W. Xu, and Y. Gong. Multi-labelled classification using maximum entropy method. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 274–281. ACM, 2005.

[76] X. Zhu, S. Kiritchenko, and S. M. Mohammad. Nrc-canada-2014: Recent improvements in the sentiment analysis of tweets. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 443–447. Citeseer, 2014.