Learning Representations for Text Classification of Indian Languages

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

Master of Science
in
Computational Linguistics
by Research

by

NURENDRA CHOUDHARY
201325186
nurendra.choudhary@research.iiit.ac.in

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

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "**Learning Representations for Text Classification of Indian Languages**" by Nurendra Choudhary, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Dr. Manish Shrivastava

To my teachers, family and friends for their guidance, love and support.

Acknowledgments

This thesis is the outcome of guidance, support, encouragement and innovative opinions I received from several people.

First and foremost, I thank my parents for always being there for me. Their support throughout the pursuit of my goals cannot be understated. Their love and encouragement is the reason I am able to pursue research with such passion and enthusiasm.

I thank Dr. Manish Shrivastava for being my advisor and helping me develop the aptitude and interest towards research. He provided the perfect amount of counseling and autonomy in deciding the my research problem and direction. The thesis would not be possible without his continuous guidance and insights. Working under his supervision was an enlightening experience and his dedication to work and research provided a perfect ideal to work towards. His courses of Natural Language Processing and Topics in Information Retrieval opened up new methods to work towards. They introduced new challenges and innovative solutions which helped me condition my own thought-process.

I thank Dr. Dipti Misra Sharma and Dr. Radhika Mamidi for building my foundation in linguistics. This base helped me throughout the research path. Linguistics helped shape my thought and build formidable solutions to the problems. Their depth of understanding in languages boggles the mind. I consider myself lucky that I was taught by such great teachers. I also thank all the professors of IIIT for their dedication towards innovation and the support and encouragement provided to students in pursuit of knowledge.

My ardent thanks to Rajat Singh for helping and accompanying me throughout the research journey. It would be impossible to handle the burden and stress of research without him at my side. Our collaboration made a formidable team which is the major factor behind most of the innovative solutions discussed in the thesis. My heartfelt thanks to Ishita Bindlish and Abhirup Dikshit for supporting me throughout the college life. The amount of focus I could develop in my research is all thanks to them. Also, thanks to them for keeping me sane throughout my journey at IIIT. Thanks to Aditya Sreekar for arbitrary discussions ranging from politics to movies and most importantly ground-breaking innovations in technology. My earnest thanks to Sandeep Sricharan for being my guide and mentor. He aided me through all the essentials needed for a research life at IIIT. His guidance expedited my integration into research life.

I would also like to thank the important people who took the time out of their busy schedules to proofread my drafts and help me improve my technical writing skills - especially Vijjini Anvesh Rao and Bocki Soujanya. Thanks to my wingmates, friends and well-wishers - Anurag Deshmukh, Pratik Jacob, Gautam Vepa, Abhinav Prasad, Rajat Singla, Nitish Jain and Kanishk Jain for the innovative opinions and discussions which helped me in both my research and life.

Abstract

Text Classification is among the primary challenges of natural language processing with multitude of applications including but not limited to sentiment analysis, emoji prediction and topic modeling. Several approaches ranging from rule-based systems to machine learning models have been applied to solve text classification. Neural network models, especially, have shown promising results. However, the models are limited by their reliance on the amount of annotated data.

Machine learning models, primarily, operate on numerical data that represent the features of its input. In the case of text classification, the models need extraction of text features relevant to the given problem. This requirement led to the advent of a more specific area analyzing feature engineering called representation learning. Current approaches to learning sentence representations has predominantly been centered around the semantics of a sentence. However, semantic features are often inconsequential in text classification problems. Thus, to bolster task-specific representation learning, models need to learn text features according to the problem, rather than adopting a pre-compiled representation model.

With the proliferation of Internet and its penetration into multilingual societies, the linguistic diversity of the real world is now reflected in online communities. Limiting solutions to a set of major languages is no longer viable. Nevertheless, the proliferation is relatively recent and hence the amount of available data in many widely-spoken languages is inadequate. Manual annotation of data is a humongous task and often expensive. Additionally, morphologically rich languages' property of inflection and agglutination causes data sparsity problems. The possibility of utilizing resource-rich languages as leverage to enhance the performance of text classification in resource-poor languages is fascinating.

To this end, the thesis presents a twin Bidirectional Long Short Term Memory (Bi-LSTM) network with shared parameters consolidated by a contrastive loss function (based on a similarity metric). Fundamentally, the aim is to classify text into multiple categories based on its features. The model jointly learns the text features (or representations) of resource-poor and resource-rich languages in a mutually shared space by utilizing the similarity between their assigned categories. Shared parameters of the Siamese network enable the projection of sentences into a common space and the similarity metric ensures the correctness of the projection.

Essentially, the model projects sentences with similar categories closer to each other and sentences with different categories farther from each other. Additionally, this enables the leveraging of resource-rich languages to enhance text classification of resource-poor languages. The reason is that the projection is based on the language-agnostic annotation tags. So, pairing resource-poor and resource-rich sentences together as input to the Bi-LSTMs pair will result in their projection into a shared space. Hence, the resource-rich sentences aid the classification of resource-poor sentences by composing more training samples.

Experiments, specifically, on the text classification tasks of Multilingual Sentiment Analysis and Multilingual Emoji Prediction with validation against large scale standard datasets reveal that the model significantly outperforms the state-of-the-art approaches in the case of both resource-poor and resource-rich languages. Furthermore, it is also observed that jointly training resource-poor and resource-rich languages exhibit significant performance enhancement over its single resource-poor language counterparts.

Contents

List of Figures

List of Tables

Chapter *1*

Introduction

Text classification assigns a class or category to documents/sentences based on their content. Among the prominent applications of natural language processing that involve text classification are sentiment analysis, topic modeling, emoji prediction and spam filtering.

1.1 Applications of Text Classification

The relevance of particular phrases/tokens in a sentence is decided by the task. In sentiment analysis, words that induce sentiments like *good*, *bad*, *victory* take precedence over other words. Topic modeling emphasizes on nouns, verbs and other content words whereas the task of spam filtering gives prominence to syntactical structures. Following is an introduction to the problems dealt with in the thesis.

1.1.1 Sentiment Analysis

"Sentiment analysis, also called opinion mining, is the field of study that analyzes people' s opinions, sentiments, evaluations, appraisals, attitudes, and emotions towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes."[33]

Sentiment analysis, primarily, is a classification problem that deals with the prediction of emotions, opinions or sentiments conveyed by text. It is a widely studied discipline in both Natural Language Processing and Information Retrieval, with numerous academic and commercial applications that deal with text feature extraction. Sentiment analysis and Opinion mining are used interchangeably in the literature, but refer to the same problem.

The thesis discusses sentiment analysis of standard languages - Hindi, Telugu, English, Spanish and also the code-mixed variety of Hindi-English. Some examples sentences are given in the Table 1.1.

Sentence	Polarity
Happiness is the worst.	NEGATIVE
शिमला बहुत शांतिपूर्ण जगह है। (shimla bahut shantipurn jagah hai) (Shimla is a very peaceful place.)	POSITIVE
నేను చదువుకోవాలి. (nenu chaduvukovaali) (I need to study.)	NEUTRAL
mujhe padhna pasand nahi hai (I do not like to study.)	NEGATIVE

Table 1.1 Examples of sentences in Sentiment Analysis

1.1.2 Emoji Prediction

Social media continues to grow exponentially since its inception and has now become a forum filled with people's expression, opinions and sentiments. To better capture the text's sentimental context, users adopted *emojis*. Fundamentally, emojis are special characters (or pictures) adopted to communicate context inexpressible by standard text. Emojis are ideograms and smileys used in electronic messages and web pages. Originally meaning pictograph, the word *emoji* comes from Japanese e (絵,picture) + moji (文字,character)[55]. Despite immense linguistic diversity, emojis and their definitions remain almost identical across all the major languages. Emojis capture a more mutually shared medium of communication, especially, in case of related cultures.

A frequent usage of social media platforms is microblogging. These microblogs comprise of limited text with an emoji that represents the emotions related to that text. Thus, establishing a general correlation between the text and the emoji. This correlation enables the tagging of text with their corresponding emoji in the microblog. Utilizing this aspect of emojis, the assertion is that sentences with similar corresponding emojis in different languages carry similar semantic features. This assertion aids the development of a text classification architecture to predict corresponding emojis of sentences or phrases. Examples sentences for the task of emoji prediction is given in Table 1.2

Sentence	Emoji
Ahhhhh .. the fulfilling joys of work .. started at 7 am .. that is 7 am yesterday Friday .. and just back .. at 5 am on Saturday ..	😜
जन्मदिन की बहुत बधाई एवं शुभकामनाएँ (janmdin ki bahut badhai evam shubhkaamnaye) (Many congrats and best wishes on your birthday.)	😊
తారకరామారావు అను నేను. (taarakaramaravu anu nenu) (Me called Tarakarama Rao.)	😍
According to my mom The biggest mistake was Mujhe phone khareed dena (According to my mom, the biggest mistake was to buy me a phone.)	😂

Table 1.2 Examples of sentences in Multilingual Emoji Prediction

1.2 Motivation

Text classification is one of the primary challenges in computational linguistics. A legitimately competent solution to the problem implies that machines are capable of understanding the patterns of language and differentiate between their structures.

The advent of deep learning approaches has contributed several architectures that provide efficient solutions to the problem. These architectures, however, focus on major languages and require immense amounts of annotated data. But, the proliferation of internet has created plenty of analyzable unannotated data for minor languages too, establishing the motive for development of language-agnostic systems. There have been several attempts at expanding or annotating the new available data. But, the process is expensive and arduous. To this end, an approach that utilizes resource-rich languages as aid to the performance of resource-poor languages is further advantageous. Development of such method also indicates the presence of task-specific features that are universal to languages. These features help in further establishment of language-agnostic spaces for learning representations in different tasks.

India is a multilingual community with an exponential growth in internet penetration. Also, the country is an attractive emerging economy for multinational firms. This motivates the development of text classification technologies that are capable of handling the massive amount

of user-generated content and generate suitable statistical data for further analysis by business organizations. The work here focuses on resource-poor Indian languages and their code-mixed variety.

The direct impact of text classification is in the tasks of:

- Sentiment Analysis: Sentiment analysis helps in the analysis of brands, political campaigns and advertisements.
- Emoji Prediction: Emoji prediction helps in development of faster modern keyboards and efficient backend emoji analysis of social media platforms.

Text classification, also, indirectly impacts:

- Recommendation Systems: Amidst the labyrinth of recommendation systems' components is user analysis. It is the process of differentiating between users' preferences based on his activity.
- Question Answering: Text classification is applied to classify the domain and narrow down the search for answers according to the question.
- Dialogue Systems: To narrow down the possible answers to a sentence based on the previous dialogues.
- Sarcasm/Abusive Language Detection: Text classification helps in learning the sentiment to further enhance detection of sarcastic/abusive phrases in the text.

1.3 Contribution of this Thesis

The thesis contributes towards advancement in the task of text classification in Indian languages. The research, primarily, focuses on the languages Hindi, Telugu and Hindi-English code-mixed variety. As stated previously, research in these languages is limited due to unavailability of annotated resources. Also, the code-mixed varieties of text present a plethora of new challenges such as word variations, spelling errors and out-of-vocabulary words.

To solve the new problems presented by code-mixing/code-switching of languages, the work proposes two algorithms. The algorithms are given below.

- Vowel Dropping: Primary observations on the word variations produced by code-mixing showed that consonants of parent word and its variations are constant whereas vowels are varying. The eventual solution is proposed based on this observation.
- Clustering-based Approach: Several cases of exception to the rules given in the previous algorithm led to the conception of this approach. Instead of generalizing rules, the approach enables a clustering of the language's semantic vectors decide the possibility of a word being another word's variation.

The major contribution of the thesis is a cross-lingual task-specific representation learning architecture called Siamese Bidirectional Long Short Term Memory(Bi-LSTM) Network Architecture (SBNA). SBNA provides an effective mechanism to cross-lingual text classification by utilizing an architecture that comprises of a twin Bi-LSTM network united by a contrastive loss function. The contrastive loss function is based on a similarity metric (cosine similarity). The architecture takes a pair of sentences as input. In case of resource-poor languages, a pair of resource-poor and resource-rich language is input. The Bi-LSTMs project the sequence of character trigrams (sentence) into a particular language space. The language spaces produced by the two Bi-LSTMs are merged together into a language-agnostic space by the similarity metric. This is feasible because the two Bi-LSTMs are identical and have shared parameters. All the updates in the networks are mirrored for both the networks. Thus, also, leading to a shared space of projection. The computational complexity is also reduced by directly using character-level representations of sentences instead of using word based representations as the number of unique character trigrams is less than unique words. Furthermore, this helps in managing spelling errors, out-of-vocabulary (OOV) words in documents and agglutination.

1.4 Thesis Organization

The thesis is divided into 5 chapters. Chapter 2 presents the previous work relevant to the task of Text classification or its subproblems - Sentiment analysis and Emoji prediction. Chapter 3 explains the problem statement and solutions to the task of handling language typology and code-mixed word variations. Chapter 4 describes the overall architecture of SBNA and its applications to the tasks of Sentiment analysis and Emoji prediction. The chapter also provides the details of the baselines, datasets and experimental setup structure to test and analyze the efficiency of the architecture. Chapter 5 summarizes the observations and concludes the thesis.

Chapter *2*

Related Work

Text classification is a widely studied field with various solutions proposed recently. This section presents a collection of relevant previous methodologies for the task. The emphasis is mainly on the specific tasks of sentiment analysis and emoji prediction.

2.1 Distributional Semantic Vectors Approach

Classic Bag of Words approach does little to capture the dependencies of the words of the sentence on each other. (Mukku et al., 2016)[38] assigns sentiment polarity to words. The sentence is then assigned the polarity of its constituents. The information of the sequence of the words is lost and this leads to incorrect classification. e.g; In *"I am not happy"*, *"not"* carries a *negative* sentiment and *"happy"* carries a *positive* sentiment. The combination gives a *neutral* sentiment, whereas the sentence is truly *negative*. This effect is limited by using bag of n-grams instead, but it does not get rid of the problem completely.

Continuous Bag Of Words (CBOW) and Skip-gram vectors [34] endure a similar problem. Both the approaches work on the principle - "You shall know a word by the company it keeps" (John Rupert Firth:1957). Essentially, in CBOW model, the context of the word is utilized to predict the given word. Similarly, in the case of skip-gram vectors, the word is utilized to predict the context. This approach captures the overall semantic value of the sentence but does not maintain information of the word order. The overall performance of the system based on these vectors, primarily, depends on the method of combination of the word vectors to construct sentence vector. The approach, also, is not able to store information about the intra-sentence relations of the words. Another fatal flaw in the context of the current problem is the dependence on training vocabulary. Social media text contains several spelling errors and out-of-vocabulary words. The Skip-gram model is inept at handling these exceptions.

2.2 Previous Approaches in Sentiment Analysis

The approaches are described according to the available resources of considered languages. Resource-rich languages include English and Spanish. Indian languages of Hindi and Telugu are considered in the resource-poor category, because of their relatively low resource-abundance for the task. Code-mixed text is assigned a special subsection because the problem is relatively recent and approaches to the problem differ significantly.

2.2.1 Resource-rich Languages

(Turney,2002)[56] tries to classify documents into a single polarity. The authors utilize a unsupervised learning algorithm based on the semantic orientation of the phrases with adjectives and adverbs. (Pang et al.,2002)[44] train classifiers (Naive Bayes, Maximum Entropy classification and Support Vector Machines) using different language features such as n-grams and part-of-speech.

Another line of research [61, 28, 62, 60, 25] focuses on sentences rather than documents. The approaches analyze the individual phrases and extract features of the sentence to predict its sentiment.

Sentiment analysis significantly depends on the domain. The reason is that every domain has its own properties, features and challenges with respect to the task. Hence, there were studies conducted on specific domain that present improvement in performance over generic models for the task.

- Social Media Text: Social media platforms are informal and encompass various challenges such as word variations and sub-standard grammar. Therefore, additional approaches are developed to handle the challenges. Twitter is a popular platform for this analysis and several validation datasets are compiled based on tweets. (Mozetivc et al.,2016)[37] and SemEval-2016 Task 4 Twitter dataset are among the popular ones. (Pak et al.,2010)[41] utilize twitter corpus for ternary classification of tweets into positive, negative and neutral. (Go et al.,2009)[20] apply distant supervision to solve sentiment analysis on twitter data.

- Movie/Product Reviews Text: Automated sentiment analysis of movies and products provides important statistics that helps in understanding the overall perspective of users towards the product. Cornell Movie review dataset[44] is certainly the largest available corpus of reviews. (Glorot et al.,2011)[19] modeled a deep learning architecture to adapt the technique according to domain of the data. However, the approach requires a large dataset to learn relevant information.

- News Corpus: News corpus is formal text that follows standard vocabulary and grammar. Sentiment analysis of news corpus has been shown to improve the performance of credi-

bility analysis systems [50]. (Godbole et al., 2007)[21] have performed sentiment analysis over large corpus of news and blogs.

Matrix Vector Recursive Neural Networks (MV-RNN) [51] provides a solution that considers both individual meaning of a word and its semantic relation with other words in the sentence. The model assigns a vector and a matrix to each word which represent its semantic value and relation with other words respectively. The model effectively captures the semantic information of a sentence and its constituents. However, it fails in case of inadequate training data making it impractical for resource-poor languages.

Adaboost-based Convolutional Neural Network (Ada-CNN) architecture [17] utilizes multiple CNN classifiers of different filter sizes. The filters capture information at different n-gram levels. Adaboost combines the different classifiers in a weighted combination. The weights signify the importance of n-grams' differing levels in the overall prediction. Furthermore, Indian languages, often, have long term word relations, whereas, n-grams only provide information relevant to short term word relations. This creates another exception to the considered languages.

2.2.2 Resource-poor Languages

A recent surge is seen in the natural language processing research of Indian languages. This trend is due to the increasing amount of data generated in online medias and development of standard corpora.

(Joshi et al.,2010)[26] designed Hindi-SentiWordNet as a fall-back strategy to classify documents into sentiments based on a majority scoring strategy. (Balamurali et al.,2012)[3] utilizes linked wordnets to perform cross-lingual sentiment analysis. The idea is to link resources to enhance the models together. These techniques are highly accurate, but are susceptible to the problems of spelling errors and improper sentences. And these problems are very frequent given any informal text including tweets.

(Arora et al.,2013)[1] developed and compiled several resources pivotal to the growth of sentiment analysis in Hindi. The resources include annotated blogs, reviews and subjective lexicon for the Hindi language.

(Mittal et al.,2013)[35] uses rule-based approaches based on Negation and Discourse Relation to perform sentiment analysis. (Bakliwal et al.,2012)[2] developed a graph-based method for expansion of Hindi-WordNet to create a more complete subjective lexicon. The initial seed is also expanded using word relations such as antonymy and synonymy.

Furthermore, (Sarkar et al.,2015)[46] train a Multinomial Naive Bayes model on annotated Hindi tweets to solve the problem. (Das et al.,2010)[14] provide a sentence-level emotions labelled Bengali text utilizing Ekman's[16] six emotion classes. (Das et al.,2010)[12] developed the SentiWordNet for diverse number of Indian Languages using computational methods ranging from lexicon-based approaches to Wordnet based and generative approaches.

Additionally, there have been efforts by researchers [39] to generate more annotated resources by utilizing available raw corpus. The authors employ the availability of different domains to construct a Multi-arm Active Transfer Learning(MATL) algorithm to label raw samples and continuously add them to the original dataset. The parameters of the algorithm are updated at each step using reinforcement learning with a reward function. The above approach is proven to work well for the domains taken into consideration in the work - Sports, Movies and Politics. These domains are structured with a formal vocabulary and grammar. Tweets do not follow this trend. Hence, the model is not suitable for unstructured tweets. The generated resources are also limited in terms of the dynamic nature of natural language. The new resources depend on the domains of the already available resources, which is risky, especially in the case of tweets that do not comply to any certain domain.

2.2.3 Hindi-English Code-mixed Text

Code-mixed text is a very recent phenomena. Hence, the previous work done in this area is limited.

A standard dataset for this task is developed by (Joshi et al.,2016)[27]. The dataset consists of annotated Hindi-English code-mixed social media (specifically from Facebook) text. In the paper, the authors also utilize character level LSTMs to learn sub word level information of social media text. This information then classifies the sentences using an annotated corpus. The model presents an effective approach for embedding sentences. However, the limitation in the approach here is the requirement of abundant data.

Additionally, (Sharma et al.,2016)[48] have made advances in the development of shallow parsers for Hindi-English code-mixed text. (Vyas et al.,2014)[57] have developed a system for Parts-of-Speech tagging of Hindi-English code-mixed data. Further improvement of theses fields should significantly aid the performance of sentiment analysis tools in the area.

2.3 Previous Approaches in Emoji Prediction

Emoji prediction has its applications in the development of faster keyboards and shallow analysis of social media. However, it is an under-researched area in context of Indian languages. The previous methodologies are discussed for resource-rich languages. They are applicable for resource-poor languages as well. However, their reliance on data deters their performance in Indian languages.

(Barbieri et al.,2016)[5] tries to identify the semantic significance of emojis using semantic vectors based on skip-gram model. BiLSTM model[4] provides a solution to the problem of maintaining the sentence's sequence, by using recurrent neural network to embed sentences. They propose two types of embeddings based on words and characters. This approach presents

an effective model for capturing the content and sequence in the sentence. However, the approach requires immense amount of data to train and hence will fail in case of languages with fewer resources. Recent approaches[5] proposed a multi-modal analysis approach to solve the problem.

2.4 Approaches in Handling Variations

The rise in the number of non-native English speakers has given significance to the processing of code-mixed text. (Sharma et al.,2015)[49] segregated Hindi and English words using language identification to normalize the code-mixed text. The phenomena of Hindi-English (Hi-En) code-mixing is the result of a growth in the number of bilingual speakers of the both the languages. The new fusion of these languages is popular because of its ease in communication and more varied expressiveness. The writing of code-mixed language leads to adoption of the script of one language (Roman script in this case) in both the languages. Romanization is the process of transliteration, resulting from code-mixing, which refers to converting one writing system to Roman script. This transliteration, however, does not follow any standard rules. Hence it makes formal rules irrelevant, leading to more complexities in the NLP tasks pertaining to CSMT data, highlighted by (Chittaranjan et al.,2014;Vyas et al.,2014;Barman et al.,2014)[10, 57, 6].

Shared tasks [47, 53] have initiated the process of CSMT data analysis. (Sridhar et al.,2015)[54] proposed a distributional representation approach that normalizes spelling errors and out-of-vocabulary words that result from informal use of English in social media forums.

(Zhang et al.,2015;Socher et al.,2013)[63, 52] demonstrated deep learning based solutions for various NLP tasks. (Sharma et al.,2015)[49] provides a methodology for normalization of these variably spelled romanized words in a rule-based manner. However, giving an automated unsupervised machine learning model enables the system to become language-agnostic. This is, especially, critical to other Indian Languages that exhibit similar code-mixing and romanization behavior as Hindi.

Chapter *3*

Language Typology and Word Variations

Hindi possesses the property of inflection and Telugu demonstrates agglutination. There is a significant proliferation of social media platforms in multilingual societies. This, further, induces portmanteau of South Asian languages with English. The blend of multiple languages as code-mixed data has recently become popular in research communities. Code-mixed data consists of anomalies such as grammatical errors and spelling variations. These properties are elaborately explained in the subsequent sections.

3.1 Agglutination in Telugu

Agglutination is a linguistic phenomena relevant to morphology, where, complex words are formed by attaching simpler words or morphemes without any significant alteration in spelling or phonetics. It is not a binary property, but rather exists in languages to a certain extent. Languages with this property are called agglutinative languages. Telugu and other Dravidian languages are considered highly agglutinative language. Table 3.1 demonstrates some examples of agglutination in Telugu.

Agglutinated Word	Constituents
ఈదుకుంటువేళ్ళాడు (eedukuntuvelladu) (he went away swimming)	ఈదు (eedu) (swim) కుంటు (kuntu) (-ing) వేళ్ళాడు (velladu) (went + male case-marker)
తాగడానికిచ్చిన (taagadanikicchina) (given for drinking)	తాగ (taaga) (drink) డా (da) (-ing) నిక్ (nik) (for) ిచ్చిన(icchina) (given)

Table 3.1 Word Segmentation in Telugu

Root Word	Possible Inflections
खा kha (eat)	खाऊँगा (khaunga) (will eat + first person + male case-marker) खाएगा (khayega) (will eat + third person + male case-marker) खाएगी (khayegi) (will eat + third person + female case-marker)
लिख (likh) (write)	लिखो (likho) (write + second person) लिखना (likhna) (to write) लिखिए (likhiye) (write + imperative modality)

Table 3.2 Word Segmentation in Hindi

3.2 Inflection in Hindi

Fusional or Inflectional languages tend to employ a single inflectional morpheme to represent diversified grammatical, syntactic and semantic features to a word. Hindi, English and several other Indo-European languages exhibit this property of inflection. Analogous to agglutination, inflection is also not a binary feature, but rather exists in varying degrees in different languages. For example, Hindi is considered more inflectional than English because of the additional variance in Hindi's inflection types. Table 3.2 presents some examples of inflection in Hindi.

3.3 Word Variations: Resultant of Code-Mixing/Code-Switching

The phenomenon of code-mixing is the embedding of linguistic units such as phrases, words or morphemes of one language into the utterance of another language, whereas code-switching refers to the co-occurrence of speech extracts belonging to two different grammatical systems. Prevalent use of social media platforms by multilingual speakers leads to an increase in the phenomenon of code-mixing and code-switching [22, 40, 15, 23]. Text created due to code-mixing and code-switching cannot be processed by tools of any individual language. Therefore, this necessitates the creation of novel techniques to handle this phenomenon. Figure 3.1 depicts some examples of code-mixed text found on social media platforms. Additionally, a majority of Hindi and Telugu native speakers type in Roman script. *Telugu lipi* and *Devanagari* (scripts of Telugu and Hindi respectively) are both phonetic scripts. Hence, their transliteration into non-phonetic Roman script leads to spelling variations and out-of-vocabulary words. Examples of such variations are given in Table 3.3.

3.3.1 Types of Word Variations

Word variations created by code-mixing are an under-researched area of study. Therefore, a new nomenclature for the domain is proposed below for promoting further discussion and provide a shared discourse.

Figure 3.1 Example Code-Mixed text on Twitter

Vocabulary Word	Meaning	Variations		
खूबसूरत	beautiful	khoobsurat	khubsurat	khubsoorat
मेहरबानी	clemency	meherbani	meharbaani	meharbani
చేస్తున్నారు	doing	chestunaaru	cestunaru	chestuunAru
కలవాలి	should meet	kalavali	kalvaali	kalvAli

Table 3.3 Word Variations due to Roman transliteration

- **Informal Transliteration:** Absence of a transliteration standard suggests that choices of choosing vowels and different sounds depend altogether on the writer/user. This occurs in the event of transliterating phonetic script languages like Hindi and Telugu to non-phonetic scripts like Roman script. The types of variations caused due to this are explained below.

  ⇒ **Long Vowel Transliteration:** Users/Writers demonstrate numerous styles of representing vowel length in long vowels. For example, the word खाया is transliterated to *khaaya* (with vowel repetition) or as *khAya* (vowel upper casing) or just *khaya* (no indication). More examples of such variations are:

  1. साल: *saal*, *sAl*, *sal*;
  2. मेरा: *meraa*, *merA*, *mera*;
  3. आजा: *aajaa*, *AjA*, *aja*, *aaja*;
  4. आपका: *aapka*, *apka*, *Apka*

  ⇒ **Double Consonants:** Analogous to long vowels, Hindi words, often, contain repeated consonants and this sound is emphasized in regular speech. This leads to another aberration where users create variants with and without repeating the consonant. Some examples are given below:

1. इज्जत: *izzat*, *izat* with stress on *z*.
2. स्वछता: *swachhta*, *swachta* with stress on *ch*.

⇒ **Borrowed Words' Transliteration:** Indian languages are relatively old and their evolution has been influenced by several other languages. In case of Hindi, Sanskrit, Persian and recently English have played a significant role in the development of its modern version. However, borrowing of words is not always perfect. Often, the borrowing language does not contain phonemes needed in the borrowed words. This is, especially, precarious in case of languages with phonetic script because writing these borrowed words perfectly also becomes a problem. In Hindi, new letters are formally introduced to handle such cases. For example, ज़ was introduced to handle /z/ phoneme and ॉ to handle /ɒ/ phoneme. Nevertheless, the alterations require time to transmit and majority of the native speakers rely on old phonemes to handle such borrowed words. This creates multiple word variations in written text. For example, इज़्ज़त is transliterated as *ijjat* and *izzat*. Similarly, आज़ाद is transliterated to *azad*, *ajad* and ज़िंदाबाद to *zindabad*, *jindabad*. This is also observed in English to Hindi transliteration. For example, *school* may be written as स्कूल (/sku:l/) or इस्स्कूल (/isku:l/).

⇒ **Accent and Dialect Phonetics:** Dialect, accent and such environmental properties of a speaker extensively impact his transliteration. Majority of the transliterations reflect the user's utterance of the word. For example, श्री is transliterated as both *shree*, *sree*. Similarly शपत is transliterated into *shapat* or *sapat*. Moreover, certain accents of Hindi lack aspiration. This is observed when transliterating words with aspirated characters such as खाया to *khaya* or *kaya*.

⇒ **Non-Phonetic Writing:** Transliteration of same sounds also leads to multiple variations in case of non-phonetic target script. The reason is that same characters represent different phonemes according to their context. For example, *k* and *q* have the same sound in *king* and *queen* respectively. Therefore, transliteration of वक्त creates the variations *waqt* and *wakt*. Similarly, *i* and *ee* have the same pronunciation in *ski* and *spree* respectively, leading to the variations of श्री as *shri* and *shree*.

- **Informal Speech:** This type explains the variations caused by speech in informal setting such as social media platforms or regular speech. The subtypes of this phenomena are explained below.

  ⇒ **Elongation:** In this case, the context needs the correct spelling and there is no transliteration ambiguity. But, users/writers choose to stretch the words, often, to express interjectional sentiments. This deviates the word from its original version, thus, creating variations. Some examples of this type of variation - *cooooooool*, *soooooo goood* and *noooo* convey emphasis on respective correct spellings.

⇒ **Brief Words:** This phenomena results from the adoption of limited character services like Short Message Service (SMS) and Twitter. It started with early SMS applications and was later adopted by social media communities. An observable pattern found in phrases thus formed is the retention of consonants and vowel drops focusing more on word inference than correctness. Examples of such phrases are *gud nite*, *plz dnt do ths* or *ista kb arhn hn?*(in romanized Hindi).

⇒ **Abbreviations:** Abbreviated forms of words are adopted by speakers in informal speech, which often translates to informal writing. Examples of such variations are *lappy* (laptop), *mac* (macbook) and others.

3.4 Morphology Analyzer - Handling Language Typology for Word Segmentation

Morphological Analysis is the process of separating the morphemes (lemma and suffixes) of an agglutinated or inflected word and analyzing the suffixes. This is a necessary preprocessing step for languages high in agglutination or inflection like Hindi and Telugu. The three primary approaches employed to capture morphemes are given below.

- **Morpheme-based morphology or Item-and-Arrangement Approach**: This approach considers morphemes to be the basic immutable element of a language and assumes that words are constructed by a set of morphemes. For example, बदनामी (badnaami) (defamation) is made with the root word नाम (naam) and prefix बद (bad) and suffix ी (i). According to the theory, words are a string made of morpheme (immutable) beads. So, the items (morphemes) are arranged to form a given word. Hence, the approach is also termed as item-and-arrangement.

- **Lexeme-based morphology or Item-and-Process Approach**: This approach considers lemma as the primary item, which goes through a linguistic process to create the final inflected or agglutinated (compounded) word. For example, लड़का goes through the process of pluralization (-ा +े) to form लड़के. As the item (lemma) goes through linguistic process to form the final word, the approach is also termed as item-and-process.

- **Word-based morphology or Word-and-Paradigm Approach**: This theory is centered around paradigms. In this approach, instead of stating rules for individual words, generalizations (paradigms) are stated that hold true for class of words. Table 3.4 is an example paradigm table defined using this approach [7]. The approach generalizes a paradigm over a class of words. Hence, the approach is termed as word-and-paradigm.

Number	Case	
	Direct	Oblique
Singular	लड़का	लड़के
	कपड़ा	कपड़े
Plural	लड़के	लड़कों
	कपड़े	कपड़ों

Number	Case	
	Direct	Oblique
Singular	(0,ϕ)	(1,◌े)
Plural	(1,◌े)	(1,◌ों)

Table 3.4 Word forms of लड़का, कपड़ा and the corresponding generalized paradigm table for the class. In the paradigm table, the numbers denote the length of suffix to be removed from the root, the character denotes the new suffix to be added after deletion.

3.5 Approach to Handle Word Variations

Code-mixing is an important phenomena of languages in multilingual societies. It has recently become unignorable due to the exponential growth of internet penetration. Additionally, major usage of this code-mixed variety is in informal forums like regular speech or social media platforms. This creates more anomalies like out-of-vocabulary words and variations of the same word. The following subsections describe two of our approaches to handling word variations.

3.5.1 Perpetual Consonant Fluctuating Vowel

In Table 3.3, it is observed that in all variations of a word, consonants are constant whereas the vowels alter in each case. Word variations depend on the writers/users of the word. Each writer has a unique dialect and speaking environment. These word variations tend to depend mainly on such factors. For example, बहुत is pronounced /bəˈɦʊt̪/ by some speakers and /boɦət̪/ by others. Further, in case of English too, *are* is dropped to "r" in informal forums of communication. So, the assumption is consonants remain constant but vowels vary according to the various factors influencing a speaker. Thus, based on given observations, the following methodology is developed.

1. Store the frequency of all the words.

2. Drop the vowels of all the words.

3. Group the words with same strings after dropping vowels into classes.

4. Replace all the words in a class with the most frequent word belonging to the class.

A run-through of the methodology is given in Figure 3.2. The detailed algorithm is described as Algorithm 1.

The algorithm provides a simple and effective method of preprocessing to handle variations. However, it suffers from significant flaws.

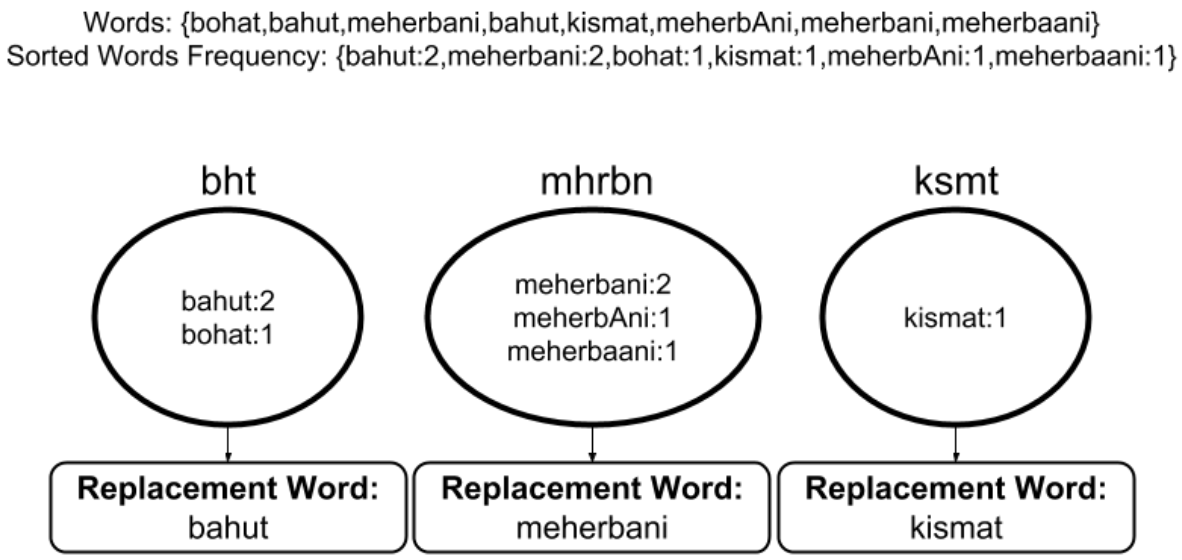

Figure 3.2 Run-through of methodology for handling word variations

Algorithm 1 Replace words for Handling Word Variations

INPUT WF={list of words and their frequency (word,frequency)}; vowels={a,e,i,o,u};
$hashmap(string, tuple)\ VT = \{\}$;
while $i = 1 \rightarrow length(WF)$ **do**
    $word_{voweldrop} \leftarrow WF[i]_{word} - vowels$; # Drop vowels from words
    if $VT[word_{voweldrop}]_{frequency} < WF[i]_{frequency}$ **then**
        $VT[word_{voweldrop}] \leftarrow WF[i]$;
    $i \leftarrow i + 1$.
done
$hashmap(string, string)\ replacement = \{\}$;
while $i = 1 \rightarrow length(WF)$ **do**
    $word_{voweldrop} \leftarrow WF[i]_{word} - vowels$;
    $replacement[WF[i]_{word}] \leftarrow VT[word_{voweldrop}]_{word}$;
    $i \leftarrow i + 1$.
done
OUTPUT replacement;

- Collisions: The vowels here are dropped without considering any of the word's properties. Hence, there are several cases where different words lead to the same consonant string. For example, *hue* (हुए), *hai* (है) and *ho* (हो) get dropped to the same string *h*.

- Loss of inflections: Many important inflections are formed using vowels. So, dropping vowels results in the loss of valuable information. For example, *meherbaan* (मेहरबान) and *meherbaani* (मेहरबानी) are both dropped to *mhrbn*.

- Invariant Consonant: The consonants remain unaffected in this algorithm. Hence, types of variations dealing with consonants (Double Consonants, Accent and Dialect Phonetics, Non-Phonetic Writing) are not handled.

To avoid the above flaws, it is important to consider the context and properties of the words. The next section describes such an approach based on distributional semantics.

3.5.2 Clustering of Word Variations - A Distributional Semantics approach

Word variations inherit the properties of the parent vocabulary word. These properties include the context of the word's occurrence. Thus, the assumption is that multiple variations of the same word share similar context. These words are also the derivatives of formal vocabulary words. Hence, variations have minimal deviation from its parent word in terms of spelling (character sequence). Therefore, the conclusion, based on these assumptions, is that the closeness of a variation to parent vocabulary word is a function of the similarity between their distributional vectors and similarity in spelling.

The methodology developed for this purpose consists of three major steps.

1. Distributional Semantic Modeling and Candidate Clustering

2. Candidate Selection

3. Candidate Substitution

The overall architecture is given in Figure 3.3. Individual steps of the architecture are explained in the subsequent subsections.

3.5.2.1 Distributional Semantic Modeling and Candidate Clustering

This step maps the the words and all variations in the data to a semantic space using distributional semantic vectors. The vectors are then clustered using a similarity metric modeled to capture spelling similarity and vector similarity. Word embeddings of 300-dimensions are computed by training skip-gram-10 model [34] on the data. Specifically, python-gensim module [45] learns the semantic word representations from the data. These words also include all the

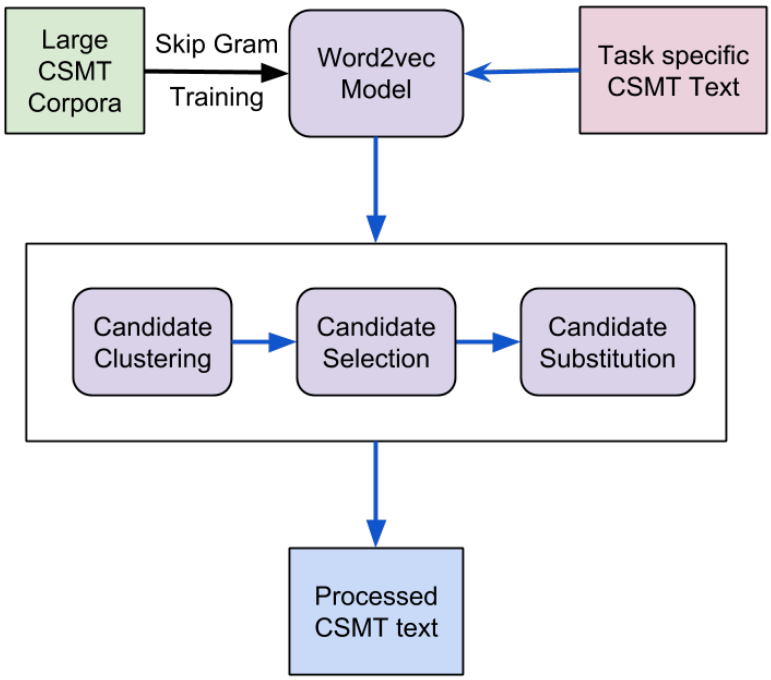

Figure 3.3 Methodology for Handling Word Variations

variations. Hierarchical softmax reduces the vocabulary count during training and minimum vocabulary count is set to 5. This is done to capture only popular variations.

Skip-gram vectors give the contextual word representations in a semantic space. Thus, similarity between these vectors is proportional to the similarity between the context of words. Additionally, capturing the spelling deviation between possible parent word and its variation is also necessary. To this end, Levenshtein distance [32] (or Edit Distance) is applied to capture the deviation as a function of difference between strings in terms of insertions and deletions. A threshold ϵ is selected empirically to determine validity of word variations in a cluster. The similarity metric formed to cluster the words and variations is formally defined in Equation 3.1.

$$f(v1, v2) = \begin{cases} sim(vec(v1), vec(v2)) & \text{if EditDistance(v1,v2)} \leq \epsilon \\ 0 & \text{else} \end{cases} \tag{3.1}$$

where $f(v1, v2)$ is the clustering similarity function between variations $v1$ and $v2$ (all words are considered variations for clustering), sim is a vector similarity metric (cosine similarity is better because it gives values between 0 and 1), vec returns the skip-gram vectors of given words and the threshold of Levenshtein distance ϵ is set according to the task.

Results of experiments performed with different values of ϵ is given in Table 3.5. Smaller values of ϵ lead to smaller clusters with better accuracy of variations, whereas, larger values

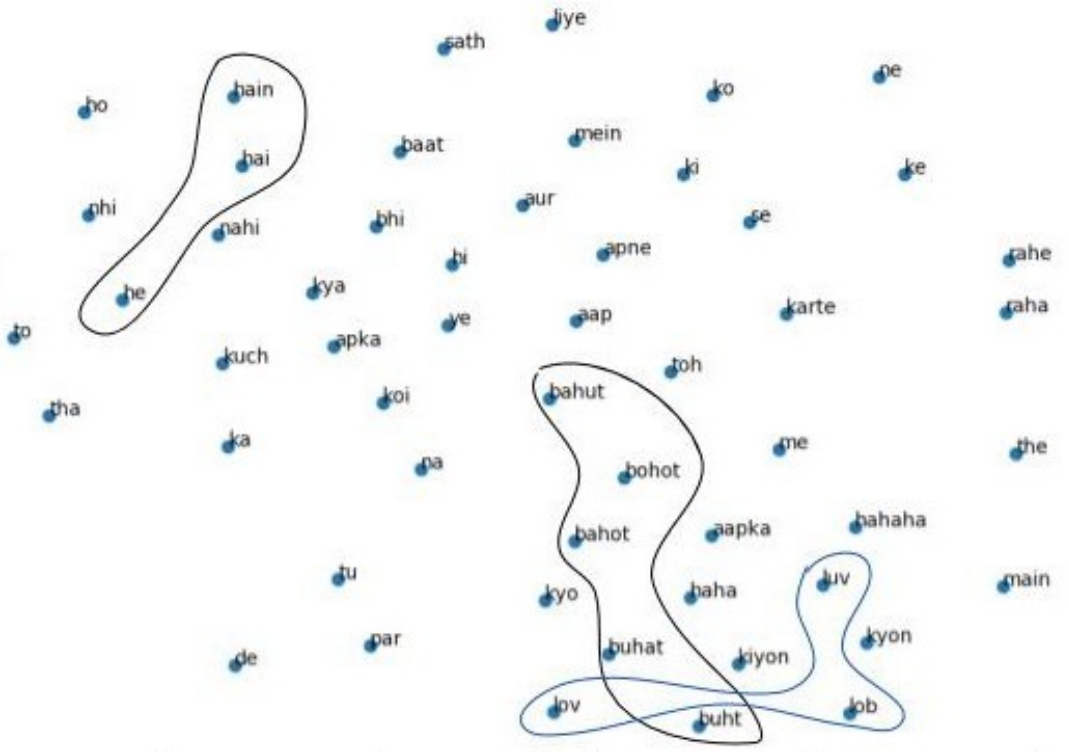

Figure 3.4 Clusters from Hindi-English Code-Mixed data with $\epsilon = 1$.

ϵ	**error %**	**Average #Words per cluster**
0	1.2	3.7
1	8.3	4.5
2	28.1	7.2
3	47.7	11.4

Table 3.5 Error analysis of clusters using 500 random sample of words

of ϵ lead to larger clusters with more coverage but with lesser emphasis on accuracy. Clusters formed using the 50 most frequent words in Hindi-English code-mixed data is illustrated in Figure 3.4.

3.5.2.2 Candidate Selection

The previous steps outputs clusters of variations. But, there is still no concrete method of determining the actual parent word to the variations. One solution is to utilize a well-defined dictionary for the vocabulary. However, standard dictionaries or phonetic transcriptions do not exist for code-mixed data. A variation's probability of being the candidate is directly proportional to its adoption in the community. Hence, the parent word is defined as the most

popular variation. Thus, the most frequent word is chosen as the candidate or parent word for each of the cluster.

3.5.2.3 Candidate Substitution

In this final step, the candidate word of a cluster substitutes all the words belonging to that cluster in the data. This method results in significant reduction in the total number of unique tokens and increase in the frequency of individual tokens. The removal of noisy word variations from the data increases emphasis on individual code-mixed words. Thus, increasing the words' reliability leading to significant performance improvements in various tasks. The experiments depicting these improvements are explained in the next section.

3.5.3 Experimental Setup to test the Clustering Approach

The approach is tested on the following tasks that operate on Code-mixed Social Media Text (CSMT).

- Sentiment Analysis on CSMT: Sentiment Analysis is a text classification task, applied in numerous NLP applications. Thus, the task presents a good area to study the improvements of clustering-based preprocessing. In the paper [27], the authors propose a method to capture sub-word level information of a CSMT sentence with Subword-LSTM architecture. The paper proposes character and subword level networks to classify given sentences into three classes - positive, negative and neutral.

  For the experiment, the HECM dataset from [27] is adopted for proper comparative analysis. Significant improvements in the performance of the model is observed on the same dataset after adding the clustering-based preprocessing step. The results of the experiment are presented in Table 3.6.

- POS-tagging on CSMT: POS-tagging is a sequence labeling task, that form the foundation of several NLP applications such as machine translation, named entity recognition and shallow parsing. Thus, the preprocessing step is tested on the area as well to check the significance of improvements. In (Ghosh et al.,2016) [18], Conditional Random Fields (CRF) capture patterns of code-switching/mixing to accurately mark each word with its corresponding part-of-speech tag.

  The datasets, adopted for the experiment, were released for a shared task conducted by 12^{th} International Conference on Natural Language Processing (ICON-2015) [1]. Addition of clustering-based preprocessing step as a precursor to the algorithm for the same dataset shows an improvement in performance of POS-tagging model. The results of the experiment are presented in Table 3.7.

[1] `http://ltrc.iiit.ac.in/icon2015/`

Method	Without Preprocessing		Clustering-based Preprocessing	
	Accuracy	**F-score**	**Accuracy**	**F-score**
NBSVM(Unigram)[59]	59.15%	0.5335	**60.38%**	**0.5421**
NBSVM(Uni+Bigram)[59]	62.5%	0.5375	**63.43%**	**0.5566**
MNB(Unigram)[59]	66.75%	0.6143	**67.23%**	**0.6325**
MNB(Uni+Bigram)[59]	66.36%	0.6046	**68.85%**	**0.6367**
MNB(Tf-Idf)[59]	63.53%	0.4783	**65.76%**	**0.5025**
SVM(Unigram)[43]	57.6%	0.5232	**58.23%**	**0.5325**
SVM(Uni+Bigram)[43]	52.96%	0.3773	**55.63%**	**0.4238**
Lexicon Lookup[49]	51.15%	0.252	**53.28%**	**0.2745**
Char-LSTM[27]	59.8%	0.511	**60.82%**	**0.5285**
Subword-LSTM[27]	69.7%	0.658	**71.38%**	**0.6621**

Table 3.6 Comparison of Clustering-based Preprocessing in Sentiment Analysis Task.

	Without our approach		Proposed Preprocessing	
Language Pair	**Baseline (Stanford Model)**	**CRF Model**	**Baseline (Stanford Model)**	**CRF Model**
Bengali-English	60.05%	75.22%	**62.27%**	**76.14%**
Hindi-English	50.87%	73.2%	**53.45%**	**75.24%**
Tamil-English	61.02%	64.83%	**63.38%**	**65.97%**

Table 3.7 Comparison of Clustering-based Preprocessing on POS-tagging task

3.5.4 Analysis of the Experiment Results

The method is validated on two of the primary tasks in natural language processing - text classification (Sentiment Analysis) and sequence labeling (POS tagging). The results for the tasks (given in Table 3.6 and 3.7) depict performance improvements (around 1-3%) in the state-of-the-art approaches of both the tasks. The experiments demonstrate that word variations are closer to each other in the semantic space and their clustering is possible. Clustering of words increases reliability and frequency of individual tokens, leading to better context vectors. These advantages prove effective in various NLP tasks involving code-mixed text.

3.5.5 Critique of the Clustering Approach

This approach is based on the assumption that word variations have similar context. The assumption helps in generalizing rules for word variations, that are inherently dynamic. Clustering is able to provide an adequate amount of coverage over the data. The handling of variations immensely improves the reliability of tokens and effectively improves performance metrics of several tasks relying on word embedding.

However, the method relies on the parameter ϵ that needs to be empirically tuned according to the task. Also, the context being considered for the variations is noisy. Hence, this noisiness

may cause improper context for the primary skip-gram model training. An iterative process to continuously cluster and improve data would be better to properly tune ϵ and also enhance context of the variations simultaneously.

This model contributes to the initial step towards building a more generalized model for automated text processing of Code-mixed Social Media Text (CSMT). Code-mixed Social Media is a dynamic phenomena and affects all multilingual communities. So, the model needs to cover a wider range of languages. Clustering-based preprocessing is language-agnostic but inherently assumes features specific to Indian languages.

Chapter 4

Cross-Lingual Task-Specific Representation Learning for Text Classification

Text classification is a primary challenge in natural language processing. The problem has various applications including but not limited to sentiment analysis [29], topic modeling [58], emoji prediction [4], named entity recognition [30] and community question-answering [13].

Several approaches, ranging from rule-based systems to machine learning algorithms, have been studied to tackle the problem. Deep learning approaches, especially, show exceptional effectiveness in the task. However, three major problems limit them.

- Text representation: Deep learning architectures are predominantly mathematical models that essentially work with features. This leads to a major problem of capturing the important task-specific features. The leading approaches capture the generic semantic and syntactic features of the text but do not consider several features relevant to the particular task. This results in a model centered around the semantic features rather than the actual problem.

- Reliance on data: Number of data samples greatly affects the performance of deep learning architectures. Low amount of data results in a model over-fitted on the small dataset. The amount of data required for training a competent model is limited in majority of the languages (also includes Indian languages).

- Language-specific feature extraction: Deep neural network architectures are capable of automatically learning features. Thus, eliminating the need for any manual feature engineering. But, particular architectures deal with only a certain category of features. For example, Convolutional Neural Networks (CNN) exhibit effective feature extraction in the case of languages with short distance word dependency (English and Spanish), but are inadequate for languages with long distance word relations (Indian Languages). Hence, languages relevant in the problem have to be analyzed before selecting an appropriate deep learning architecture.

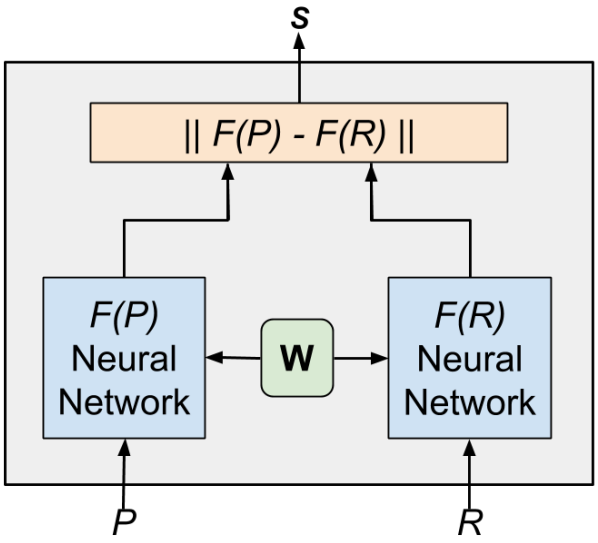

Figure 4.1 Siamese Networks

This chapter describes the conception and motivation behind Siamese Bi-LSTM Network Architecture (SBNA), a unified architecture developed to harness resource-rich languages for the task of representation learning of a resource-poor language for a given task. The proposed architecture consists of two Bi-LSTM networks with mutually shared parameters consolidated by a contrastive loss function. The adopted energy function suits discriminative training for energy based models [31].

4.1 Siamese Networks

Siamese networks were first introduced by (Bromley et al.,1994) [9] for the task of signature verification. Further, (Chopra et al.,2005)[11] applied the networks with a discriminative loss function for face verification. Recently, (Das et al.,2016)[13] utilized siamese networks to solve community question-answering. The networks display exceptional performance improvement in problems involving similarity or relation computation between comparable items such as image similarity. As illustrated in Figure 4.1, Siamese networks consist of identical networks with shared parameters united together by a contrastive loss function. Identical network implies that both of them have the same configuration of weights and parameters. The weights' update in the back-propagation is mirrored in both the networks. The working of siamese network is explained below.

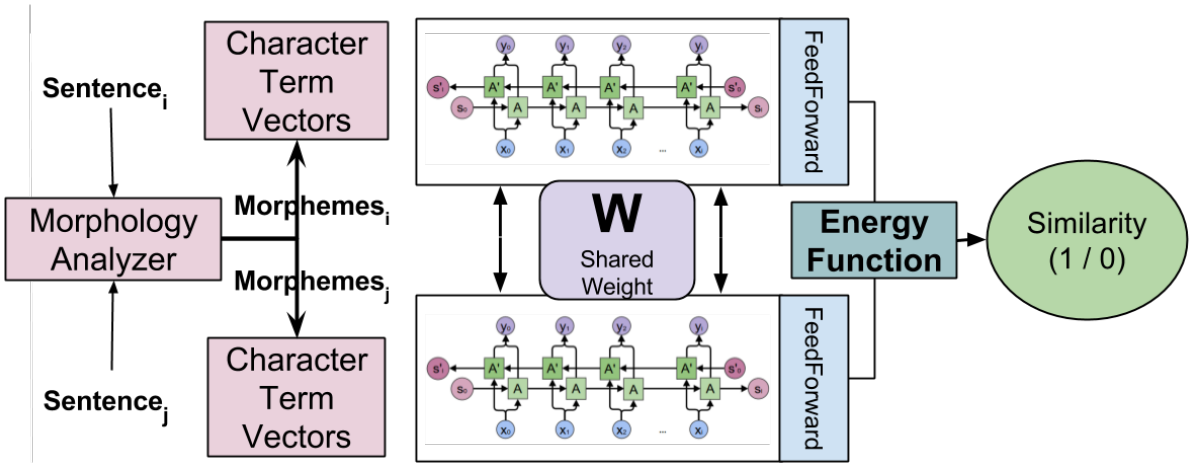

Figure 4.2 Overall Architecture of SBNA

Let, $F(X)$ be the family of functions with parameters/weights W. $F(X)$ is differentiable with respect to W. Siamese network seeks a value of the parameter W such that the symmetric similarity metric is small if $X1$ and $X2$ belong to the same category, and large if they belong to different categories. The scalar energy function $S(P, R)$ that measures the relatedness of sentiments between resource-poor (P) and resource-rich (R) language's tweets is formalized as:

$$S(P, R) = ||F(P) - F(R)|| \tag{4.1}$$

In SBNA, tweets are input from both the languages to the network. The loss function is minimized so that $S(P, R)$ is small if the R and P carry the same sentiment and large otherwise.

4.2 Architecture of SBNA

As depicted in Figure 4.2, the architecture consists of a Bi-LSTM pair connected to a dense feed-forward layer at the end. The Bi-LSTMs capture the sequence and content of the character trigrams in the sentence and project them into the problem space. The Bi-LSTMs pass these projections to a layer that computes the similarity between them. The contrastive loss function combines the similarity and the sample's label. Back-propogation through time computes the loss function's gradient with respect to the weights and biases shared by the sub-networks. The weights are then updated to rectify the error in the epoch.

4.2.1 Primary Representations

Informal data consists of spelling errors, rare words and multiple spelling of the same word. The style of writing a word may also convey a feature (e.g; "Hiiii" conveys a positive sentiment whereas "Hi" is a neutral sentiment). Hence, character trigrams are a better choice to embed

the sentence than words. This approach properly addresses the spelling errors and rare words because a partial match exists in the character trigrams. Character trigrams gain the information of all the word's inflections, thus, restraining the problem of agglutination and inflection to a certain extent. Also, this method captures the sentiment of different writing styles as information is attained on a character-level. To further address the problem, clustering-based preprocessing step (explained in Chapter 3 Section 3.5.2) is applied to handle variations. This also helps in the computational complexity as the number of character trigrams is less than the number of complete words. The approach represents a sentence using a one-hot vector with number of dimensions equal to the number of unique character trigrams in the training dataset.

Character-based term vectors of both the languages' sentences and a label are input to the twin networks of the model. The label indicates whether the samples are nearer or farther to each other in the problem space. For positive samples (nearer in the problem space), the twin networks are fed with term vectors of sentences with the same tag. For negative samples (far away from each other in the problem space), the twin networks are fed with term vectors of sentences with different tags.

4.2.2 Bidirectional Long Short Term Memory (Bi-LSTM)

Each sentence-pair is mapped into $[l_i^1, l_i^2]$ such that $l_i^1 \in \mathbb{R}^m$ and $l_i^2 \in \mathbb{R}^n$, where m and n are the total number of character trigrams in both the languages respectively.

Recurrent Neural Network (RNN) suffer the problem of vanishing and exploding gradients. In deep networks with several layers, gradients accumulate over time and cause large updates in the weights during training. The problem of vanishing gradients occurs in the early layers of a network. The parameter's gradients are too small to cause any significant change in the output. This impedes the learning of certain parameter leading to the problem of vanishing gradient.

Long Short Term Memory (LSTM) [24] units are introduced as a solution to the problems of vanishing and exploding gradients in RNNs. The idea of an LSTM is that instead of a simple hidden cell with a activation function, special kind of memory cell that stores information is adopted. Standard RNNs accept the previous hidden state and current input to output the new hidden state. In LSTMs, the old hidden state of particular cell is also taken into equation to output the new state. As illustrated in Figure 4.3, LSTM consists of three major gates.

1. Forget Gate: A function of the previous hidden state and the new input passes through the forget gate. The forget gate decides the relevance of the previous hidden state. High output values of forget gate (close to 1) represent that the previous state is relevant and low values (close to 0) represent that the previous state is irrelevant.

2. Input Gate: A function of inputs passes through the input gate and gets added to the cell state to update it.

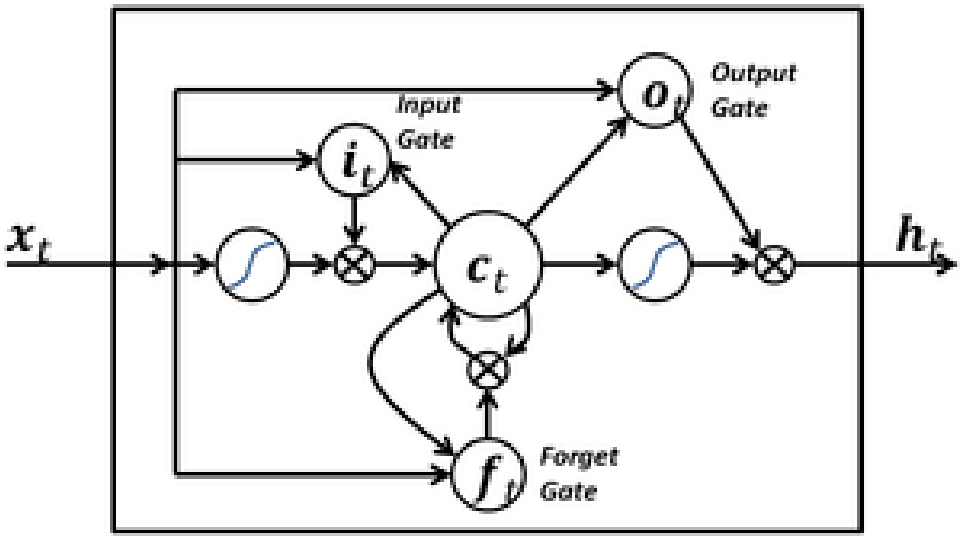

Figure 4.3 Basic LSTM cell

3. Output Gate: The gate decides which values of the cell gates should be added to output of the hidden state.

At timestep t, the gates of LSTM cell are updated according to the following steps. Following is the definition of parameters used in the update equations[1].

- x_t is the input to the LSTM memory cell at timestep t.
- W_i, W_f, W_c, W_o are weight matrices of the input, forget, cell and output gate that interact with the input parameter x_t.
- U_i, U_f, U_c, U_o are weight matrices of the input, forget, cell and output gate that interact with the hidden state of the previous timestep h_{t-1}.
- b_i, b_f, b_c, b_o are bias vectors of the input, forget, cell and output gate.
- V_o is the matrix that represents the weights of an intermediate cell needed to update the output gate.
- $\odot$ represents element-wise multiplication.

The value of input gate at time step t is:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{4.2}$$

$$\tilde{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{4.3}$$

[1] `http://deeplearning.net/tutorial/lstm.html`

The equation for update of the forget gate at timestep t is given by:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{4.4}$$

Given i_t, f_t and $\tilde{C}_t$, the update value of memory cell's C_t is computed as:

$$C_t = i_t \bigodot \tilde{C}_t + f_t \bigodot C_{t-1} \tag{4.5}$$

The final updates to the weights of the output cell and LSTM cell are computed as:

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + V_o C_t + b_o) \tag{4.6}$$

$$h_t = o_t \bigodot \tanh(C_t) \tag{4.7}$$

Bi-LSTM model encodes the sentence twice, once in the original order (forward) of the sentence and then in the reverse order (backward). Back-propagation through time [8] calculates the weights for both the orders independently. The algorithm works in the same way as generic back-propagation, except in this case the back-propagation occurs over all the hidden states of the unfolded timesteps.

Element-wise Rectified Linear Unit (ReLU) is applied to the output encoding of the BiLSTM. ReLU is defined as: $f(x) = max(0, x)$. The choice of ReLU simplifies back-propagation, causes faster learning and avoids saturation.

The architecture's final dense feed forward layer converts the output of the ReLU layer into a fixed length vector $s \in \mathbb{R}^d$. In SBNA, the value of d is empirically set to 128. The overall model is formalized as:

$$s = max\{0, W[fw, bw] + b\} \tag{4.8}$$

where W is a learned parameter matrix (weights), fw is the forward LSTM encoding of the sentence, bw is the backward LSTM encoding of the sentence, and b is a bias term.

4.2.3 Training Phase

SBNA trains on pairs of sentences in both the languages to capture the similarity in their classes. The architecture differs from other deep learning counterparts due to its property of parameter sharing. Training the network with a shared set of parameters not only reduces the number of parameters (thus, save many computations) but also ensures that the sentences of both the languages project into the same problem space (shown in Figure 4.4). The network learns the shared parameters to minimize the distance between the sentences with the same classes and maximize the distance between the sentences with different classes.

Given an input l_i^1, l_i^2 where l_i^1 and l_i^2 are sentences from both the languages respectively and a label $y_i \in \{-1, 1\}$, the loss function is defined as:

$$loss(l_i^1, l_i^2) = \begin{cases} 1 - cos(l_i^1, l_i^2), & y = 1; \\ max(0, cos(l_i^1, l_i^2) - m), & y = -1; \end{cases} \tag{4.9}$$

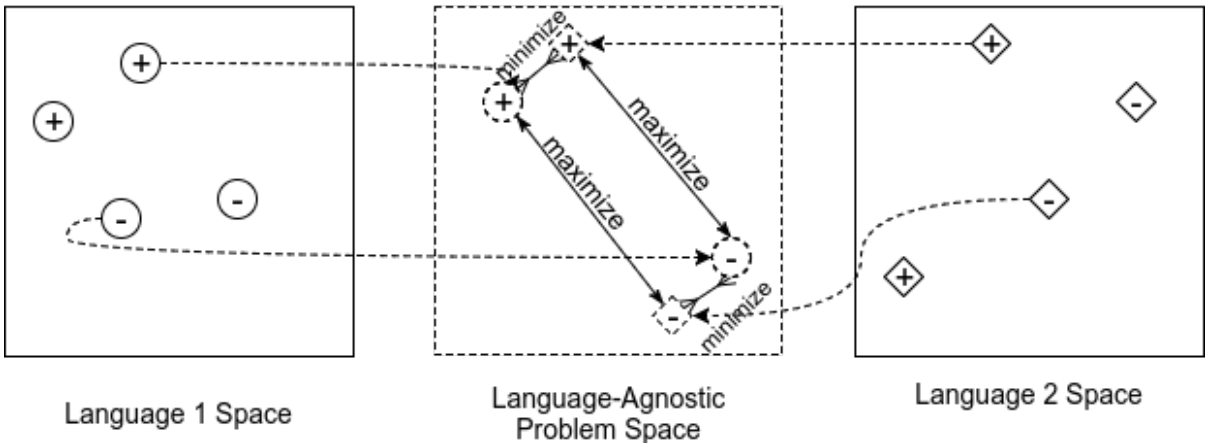

Figure 4.4 Projecting sentences of different languages into language-agnostic space.

where m is the margin that decides the distance by which dissimilar pairs should be moved away from each other. It varies between 0 to 1. The loss function is minimized such that pair of sentences with the label 1 (same class) are projected nearer to each other and pair of sentences with the label -1 (different class) are projected farther from each other in the problem space.The model is trained by minimizing the overall loss function in a batch. The objective is to minimize:

$$L(\Lambda) = \sum_{(l_i^1, l_i^2) \in C \cup C'} loss(l_i^1, l_i^2) \tag{4.10}$$

where C contains the batch of same sentiment sentence pairs and C' contains the batch of different sentiment sentence pairs. Back-propagation through time (BPTT) updates the parameters shared by the Bi-LSTM sub-networks.

4.2.4 Testing Phase

The network computes the similarity between sentences according to the task. So, the classification tasks are reduced to similarity problem. Here, the similarity metric provides the probability of unknown sentence belonging to the known sentence's class. So, a combination of random sampling and weighted sum results in the final class of the given unknown sentence.

A certain number (100 in this case) of sentences are randomly sampled for each class R_{class} from the language corpus with higher amount of data. For every input, the trained model is applied to get the similarity between the input and all corresponding R_{class}. The metric is absolute cosine similarity and hence the result is between 0 and 1. Finally, the R_{class} with the most matches with the input as the correct tag is selected.

For sentence with unknown tag u, given SBNA function as $sim(l_1, l_2)$ and 100 sample sentences with known tag of class $(s_{c1}^1, s_{c1}^2 ... s_{c1}^{100}), (s_{c2}^1, s_{c2}^2 ... s_{c2}^{100}), ..(s_{cn}^1, s_{cn}^2 ... s_{cn}^{100})$, the test proce-

dure to predict class of unknown sentence $class_u$ is formalized as:

$$class_u = \arg\max_{class} \sum_{i=0}^{100} (sim(u, s^i_{class})) \tag{4.11}$$

In case the correlated data available for both the languages are not annotated, one language's abundant resources are utilized to construct a state-of-the-art model for the task ([17] for sentiment analysis and [4] for emoji prediction). The sentiment analysis model in conjunction with the correlation data obtained from our model aids the resource-poor language's prediction.

4.3 Experimental Setup for Multilingual Sentiment Analysis

Sentiment Analysis is among the most prominent and widely researched text classification tasks. The task has a clear problem statement and an exhaustive set of datasets to validate the hypothesis.

Problem Statement: Given a sentence, predict the sentiment conveyed by the sentence. The languages of the sentence are formal or informal varieties of Hindi, Telugu, English and Spanish. The sentiments to be predicted are ternary (Positive, Negative and Neutral) or four-class (Very Positive, Positive, Negative, Very Negative).

Hypothesis: The proposal is that a Siamese Bi-LSTM Network Architecture (SBNA) solves the problem by projecting languages with similar sentiment closer and different sentiment farther from each other. This overall architecture should result in a performance better than the state-of-the-art because of parameter sharing and increment in data (training and testing samples) due to cross-lingual representation learning.

4.3.1 Datasets

The datasets for the different languages considered in the experiments are given below:

- **English - Movie Review Dataset:** The dataset[42] consists of 5006 movie reviews annotated into 3 classes (positive, neutral and negative) and 4 classes (very positive, positive, negative and very negative).
- **English - Twitter Dataset:** The dataset[37] consists of 103035 tweets annotated into 3 classes - positive, neutral and negative.
- **Spanish - Twitter Dataset:** The dataset[37] consists of 275589 tweets annotated into 3 classes - positive, neutral and negative.
- **Hindi - Product Review Dataset:** The dataset[36] consists of 1004 product reviews annotated into 3 classes - positive, neutral and negative.

		3 classes			4 classes			
Datasets	Sentence Length	Pos	Neg	Neu	V.Pos	Pos	Neg	V.Neg
English - Movie Reviews	429	38%	24%	38%	17%	40%	31%	12%
English - Twitter	12	29%	26%	45%	-	-	-	-
Spanish - Twitter	14	48%	13%	39%	-	-	-	-
Hindi - Reviews	15	33%	31%	36%	-	-	-	-
Telugu - News	13	27%	27%	46%	-	-	-	-

Table 4.1 Distribution of the datasets considered in the experiments. Pos, Neg, Neu, V. Pos and V.Neg stand for Positive, Negative, Neutral, Very Positive and Very Negative respectively. 4 classes are available only in Movie Review dataset.

- **Telugu - News Dataset:** The dataset[39] is an annotated corpus of news data tagged into 3 classes - positive, neutral and negative.

The sentiment tags' distribution in the above datasets is given in Table 4.1.

4.3.2 Baselines

The approaches vary based on the language in consideration. Hence, baselines are also defined below accordingly. English, Japanese and Spanish enjoy the highest share of data on Twitter[2]. English and Spanish are considered here because of their structural and typological similarity (both are SVO). The baselines considered for resource-rich languages - English and Spanish are:

- **Average Skip-gram Vectors (ASV):** Word2Vec skip-gram model [34] is trained on a corpus of 65 million raw sentences in English and 20 million raw sentences in Spanish. Word2Vec provides a vector for each word. Average of the words' vectors results in the sentence's vector. So, each sentence vector is defined as:

$$V_s = \frac{\sum_{w \in W_s} V_w}{|W_s|} \tag{4.12}$$

  where V_s is the sentence's vector s, W_s is the set of the words in the sentence and V_w is the vector of the word w.

  After obtaining each message's embedding, an L2-regularized logistic regression (with ϵ equal to 0.001) is trained for classification.

- **Matrix Vector Recursive Neural Network (MV-RNN):** The model[51] assigns a vector and a matrix to every node of a syntactic parsed tree. The vector represents the node's semantic value and the matrix represents its relation with the neighboring words. A recursive neural network model is then trained using backpropagation through structure to define the nodes' weighted contribution to the sentence's sentiment.

[2]The Many Tongues of Twitter - MIT Technology Review

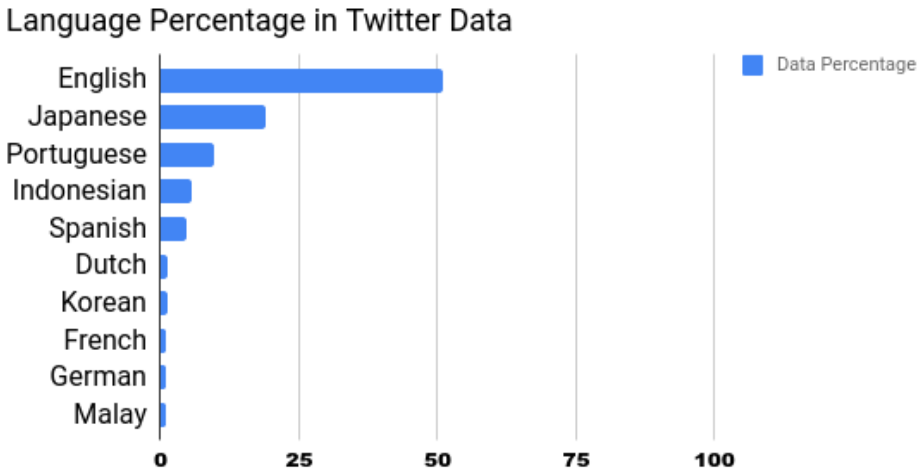

Figure 4.5 Twitter share of the top languages

- **Adaboost based Convolutional Neural Network (Ada-CNN):** CNN sentence classifier models[29] with filter sizes 3,4 and 5 are trained on the datasets. These filter sizes capture the 3-gram, 4-gram and 5-gram tokens' contribution to the overall sentiment respectively. Adaboost then attains a weighted combination of these classifiers. This weighted combination of the classifiers assigns the overall sentiment tag. This helps in giving a weighted emphasis to the information provided by 3-grams, 4-grams and 5-grams in the sentence.

Hindi and Telugu are the 3^{rd} and 17^{th} most spoken language in the world respectively. But they hold a relatively low share of Twitter data (they are not among the top languages in Twitter depicted in Figure 4.5). The speakers of Hindi and Telugu on Twitter primarily use the roman transliterated form of the language. This also further translates to a limited availability of annotated corpus for these languages. The baselines for these languages are:

- **Domain Specific Classifier (Telugu) (DSC-T):** Word2Vec model is trained on a corpus of 700,000 raw Telugu sentences provided by Indian Languages Corpora Initiative (ILCI). Random Forest (RF) and Support Vector Machines classifier (SVM) (given by [38]) are trained on the Telugu News dataset to construct our baseline for Telugu language.

- **Multinomial Naive Bayes Model (Hindi) (MNB-H):** Multinomial Naive Bayes model (given by [46]) is trained on the Hindi Review dataset to form our baseline for Hindi language.

Method	Accuracy	Precision	Recall	F-score
ASV	52.59%	0.49	0.52	0.50
MV-RNN	79.0%	0.77	0.75	0.76
Ada-CNN	81.2%	0.82	0.80	0.81
DSC-T (RF)	67.17%	0.67	0.66	0.66
MNB-H	62.14%	0.61	0.58	0.59
SBNA(Eng-Eng)	**82.25%**	**0.83**	**0.80**	**0.81**
SBNA(Spa-Spa)	81.5%	0.83	0.80	0.81
SBNA(Hin-Hin)	70.2%	0.72	0.69	0.70
SBNA(Tel-Tel)	69.2%	0.70	0.69	0.69
SBNA(Hin-Eng)	**80.5%**	**0.82**	**0.79**	**0.80**
SBNA(Tel-Eng)	**80.3%**	**0.82**	**0.79**	**0.80**

Table 4.2 Comparison between different language pairs of our model and previous methodologies for sentiment analysis experiment.

4.3.3 Experiment

The experiment is a text classification task. English and Hindi sentiment datasets (Eng-Hin) are adopted to align each Hindi sentence with English sentences of the same sentiment (positive samples) and the pair is labeled 1. Similarly, the same number of English tweets with different sentiment (negative samples) are randomly sampled for each Hindi tweet and the pair is labeled -1. Similarly, the experiment is repeated for all the different pairs of datasets possible.

For an appropriate comparative study, baselines defined in Section 4.3.2 are trained on the same datasets. In case of resource-rich languages (English and Spanish), the baselines ASV, MV-RNN and Ada-CNN are trained on their respective sentiment datasets. We train the baseline MNB-H on Hindi sentiment dataset and baseline DSC-T on Telugu sentiment dataset. The baselines of English compare to the (Eng-Eng) pair's performance. For each of the other languages *lang*, the baselines compare with the performance of (Eng-*lang*) and (*lang*-*lang*) pair.

Table 4.2 demonstrates the results of sentiment analysis experiments. SBNA(lang1-lang2) denotes Siamese Bi-LSTM Network Architecture trained on pair of lang1 and lang2. ASV (Average Skip-gram Vectors), MV-RNN (Matrix Vector - Recurrent Neural Network) and Ada-CNN (Adaboost Convolutional Neural Network) compare to SBNA(Eng-Eng). DSC-T (RF) is the Random Forest based Domain Specific Classifier approach for Telugu. It compares with SBNA(Tel-Eng) and SBNA(Tel-Tel). MNB-H is the Multinomial Bayes model for Hindi and compares to SBNA(Hin-Hin) and SBNA(Hin-Eng).

4.4 Experimental Setup for Multilingual Emoji Prediction

Recently, the task of multilingual emoji prediction has gained significance due to proliferation of internet in multilingual communities.

Lan	😊	❤️	😍	😐	😠	😊	😘	😢	🤔	😶	😭	😡	😄	😜	😉	💕	💙	💜
Eng	17.1	15.3	10.0	5.7	4.9	4.7	3.8	3.7	3.6	3.5	3.3	3.2	3.1	3.0	2.8	2.7	2.7	2.6
Spa	9.7	10.8	13.1	3.1	2.7	6.5	6.3	6.8	6.0	3.4	6.1	4.0	4.6	5.4	3.4	2.8	3.4	2.0
Hin	9.7	6.8	7.5	3.9	2.8	9.5	4.9	1.5	6.7	2.9	4.3	9.4	8.6	7.7	7.2	1.9	2.3	2.6
Tel	22.7	5.7	16.6	1.4	0.3	10.8	3.9	0.3	3.0	1.6	0.9	13.4	5.1	6.7	4.2	2.3	0.9	0.5

Table 4.3 Distribution of emojis in languages' tweets. [3]

Problem Statement: Given a sentence or phrase, predict the emoji that most appropriately matches the sentence. The languages of the sentence are social media text varieties of Hindi, Telugu, English and Spanish.

Hypothesis: The proposal is that a Siamese Bi-LSTM Network Architecture (SBNA) solves the problem by projecting languages with similar emoji closer and different emoji farther from each other. This overall architecture is expected to result in a performance better than the state-of-the-art because of parameter sharing and increment in data (training and testing samples) due to cross-lingual representation learning.

4.4.1 Datasets

The twitter datasets for different languages are given below:

- **English:** Tweets from the ids given by [4]. The dataset consists of the tweets with 18 most frequent emojis, which is, 500,000 tweets.
- **Spanish:** Tweets containing the most frequent emojis present in English tweets, which is, 100,000 tweets.
- **Hindi:** Tweets containing the most frequent emojis present in English tweets, which is 15000.
- **Telugu:** Tweets containing the most frequent emojis present in English tweets, which is, 6000.

Table 4.3 demonstrates the distribution of the emojis in the above datasets.

4.4.2 Baselines

Following are the baselines defined for resource-rich languages - English and Spanish.

- **Average Skip-gram Vectors (ASV): Average Skip-gram Vectors (ASV):** Word2Vec skip-gram model [34] is trained on a corpus of 65 million raw sentences in English and 20

[3]The hearts in the table are of different colors. The most frequent one is *red*, the second most frequent one is *blue* and the last one is *purple heart*

million raw sentences in Spanish. Word2Vec provides a vector for each word. Average of the words' vectors results in the sentence's vector. So, each sentence vector is defined as:

$$V_s = \frac{\sum_{w \in W_s} V_w}{|W_s|} \tag{4.13}$$

where V_s is the sentence's vector s, W_s is the set of the words in the sentence and V_w is the vector of the word w.

After obtaining each message's embedding, an L2-regularized logistic regression (with ϵ equal to 0.001) is trained for classification.

- **Bidirectional LSTM (Bi-LSTM):** There are two approaches - word based and character based Bi-LSTM embeddings. The architecture is modeled as described in (Barbieri et al.,2017)[4]. The Bi-LSTMs capture character-level information. This is specially helpful for social media text (variations and spelling errors are common).

Following are the baselines defined for resource-poor languages - Hindi and Telugu.

- **Domain Specific Classifier (Telugu) (DSC-T):** Word2Vec model is trained on a corpus of 700,000 raw Telugu sentences provided by Indian Languages Corpora Initiative (ILCI). Random Forest (RF) and Support Vector Machines classifier (SVM) (given by [38]) are trained on the Telugu twitter dataset to construct our baseline for Telugu language.
- **Multinomial Naive Bayes Model (Hindi) (MNB-H):** Multinomial Naive Bayes model (given by [46]) is trained on the Hindi tweets dataset to form our baseline for Hindi language.

4.4.3 Experiments

Contrastive learning is performed on SBNA using data made by aligning each English tweet with a set of positive Hindi tweet samples (with the same emoji) with label 1 and a set of negative Hindi tweet samples (with different emoji) of the same size with label -1. The experiment is conducted thrice taking 5 most frequent emojis (5 classes), 10 most frequent emojis (10 classes) and all the emojis (18 classes). Similarly, the experiment is repeated for all the different pairs of datasets possible.

For an appropriate comparative study, the baselines defined in section 4.4.2 are also trained on the twitter-emoji datasets. In case of resource-rich languages (English and Spanish), the baseline Bi-LSTM model is trained on the respective twitter-emoji datasets. We train the baseline MNB-H on Hindi twitter-emoji dataset and baseline DSC-T on Telugu twitter-emoji dataset. The baselines of English compare to the (Eng-Eng) pair's performance. For each of the other languages *lang*, their baselines are compared with the performance (Eng-*lang*) and (*lang*-*lang*) pair.

Method	5			10			18		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
ASV	0.59	0.60	0.59	0.44	0.47	0.45	0.32	0.34	0.35
Bi-LSTM(W)	0.61	0.61	0.61	0.45	0.45	0.45	0.34	0.36	0.35
Bi-LSTM(C)	0.63	0.63	0.63	0.48	0.47	0.47	0.42	0.39	0.40
DSC-T(RF)	0.34	0.35	0.34	0.31	0.32	0.31	0.24	0.23	0.23
MNB-H	0.45	0.49	0.46	0.42	0.43	0.42	0.38	0.36	0.37
SBNA(Eng-Eng)	**0.74**	**0.73**	**0.73**	**0.62**	**0.60**	**0.61**	**0.49**	**0.54**	**0.51**
SBNA(Spa-Spa)	0.71	0.72	0.71	0.58	0.59	0.58	0.42	0.42	0.42
SBNA(Hin-Hin)	0.54	0.58	0.56	0.45	0.47	0.46	0.39	0.33	0.35
SBNA(Tel-Tel)	0.47	0.49	0.48	0.39	0.41	0.40	0.31	0.35	0.33
SBNA(Hin-Eng)	**0.68**	**0.70**	**0.69**	**0.52**	**0.56**	**0.54**	**0.46**	**0.43**	**0.44**
SBNA(Tel-Eng)	**0.63**	**0.66**	**0.64**	**0.49**	**0.47**	**0.48**	**0.41**	**0.44**	**0.42**

Table 4.4 Comparison between different language pairs of our model and previous methodologies for emoji prediction experiment. 5,10 and 18 are the number of most frequent emojis considered in that experiment. P,R,F1 are the Precision, Recall and F-scores respectively.

Table 4.4 demonstrates the results of the emoji prediction experiments. ASV is Average Skipgram Vectors, Bi-LSTM(W) and Bi-LSTM(C) refer to Word and Character based Bi-LSTM models. They are baselines for English and compare to SBNA(Eng-Eng). DSC-T is Domain Specific Classifier for Telugu and compares to SBNA(Tel-Eng) and SBNA(Tel-Tel). MNB-H refers to Multinomial Bayes Model for Hindi and compares to SBNA(Hin-Eng) and SBNA(Hin-Hin).

4.5 Experiment Setup for Code-mixed Sentiment Analysis

Code-mixing/Code-switching is a recent phenomena observed due to the proliferation of social media forums into multilingual communities.

The problem statement and hypothesis of the task are same as in the case of Multilingual Sentiment Analysis (Section 4.3). The difference is that in this case sentences are a code-mixed variety of Hindi and English called Hindi-English Code-mixed (HECM) text.

4.5.1 Datasets

The datasets are utilized for testing the architecture on both code-mixed data (Hindi-English) and social media text of a standard language (English). Following are the datasets considered in our experiments.

- **Hindi-English Code-Mixed (HECM):** The dataset, proposed in [27], consists of 3879 annotated Hindi-English Code-Mixed sentences.

Datasets	Words	Char-trigrams	Positive	Neutral	Negative
HECM	43725	12842	35%	50%	15%
English-Twitter	337913	197649	28%	46%	26%
English-SemEval'13	97280	52011	40%	40%	20%

Table 4.5 Properties of the datasets.

- **English - Twitter:** The dataset, proposed in [37], consists of 103035 annotated English tweets.
- **SemEval 2013:** The dataset, used for SemEval 2013 Task 2B[4], consists of 11338 annotated English tweets.

All the datasets are annotated with three classes - positive, negative and neutral. Table 4.5 demonstrates the distribution of classes in the above datasets.

4.5.2 Baselines

Following are the baselines defined according to relevant previous approaches.

- **Average Skip-gram Vectors (ASV):** Word2Vec [34] provides a vector for each word. The vectors are averaged to get the sentence's vector. So, each sentence vector is defined as:

$$V_s = \frac{\sum_{w \in W_s} V_w}{|W_s|} \tag{4.14}$$

  where V_s is the vector of the sentence s, W_s is the set of the words in the sentence and V_w is the vector of the word w.

  After obtaining each message's embedding, a L2-regularized logistic regression (with ϵ equal to 0.001) is trained.

- **Subword LSTM (SWLSTM):** The approach, proposed in (Joshi et al.,2016)[27], is adopted as the baseline for Hindi-English Code-Mixed data. Character embeddings of the sentence are input and Convolutional Neural Networks capture sub-word level information from the sentence. These embeddings of the tweets further classify them into different sentiment classes.

4.5.3 Experiment

The experiment is a text classification task. English and HECM sentiment datasets (Eng-HECM) are utilized and each HECM sentence is aligned with English sentences of the same sentiment (positive samples) and labeled 1. Similarly, the same number of English tweets are

[4] `https://www.cs.york.ac.uk/semeval-2013/task2/index.html`

Model	Accuracy	Precision	Recall	F-score
ASV	57.6%	0.5132	0.5336	0.5232
SWLSTM	69.7%	0.646	0.671	0.658
SBNA(HECM-HECM)	71.3%	0.68	0.665	0.672
SBNA(HECM-Eng)	**77.3%**	**0.766**	**0.753**	**0.759**
Improvement	7.6%	0.12	0.082	0.101

Table 4.6 Comparison of SBNA with the baselines. ASV and SWLSTM denote the Average Skip-gram vector and Sub-Word LSTM model respectively.

sampled with different sentiment (negative samples) for each HECM Tweet and labeled -1. Similarly, the experiment is repeated for the HECM-HECM pair too.

For an appropriate comparative study, the baselines defined in section 4.5.2 are trained on the same HECM dataset.

Table 4.6 demonstrates the performance of these models compared to the baselines.

4.6 Analysis of the Experiment Results

The analysis study of SBNA is presented from two perspectives - Qualitative Analysis and Quantitative Analysis. Qualitative Analysis studies the improvement in performance compared to previous methods for the same amount of data. Quantitative Analysis studies the extent of correlation between the amount of data and the performance of the model.

4.6.1 Qualitative Analysis

The results of the experiments on sentiment analysis and emoji prediction task (given in Tables 4.2, 4.4 and 4.6) portray that for same language pairs (Eng-Eng,Spa-Spa,Hin-Hin,Tel-Tel, HECM-HECM), SBNA model outperforms its counterparts significantly on both the tasks despite the amount of data being same. The reason is that SBNA is centered around the similarity metric. Bi-LSTMs learn representations of the sentences according to the similarity metric. This helps the model in capturing task-specific features along with necessary semantic features. The shared parameters of the network enable the sentences' projection to the same problem space based on the similarity metric.

4.6.2 Quantitative Analysis

From the overall results (given in Tables 4.2, 4.4 and 4.6), it is observed that the model's performance in both the tasks is directly proportional to the amount of data available in the language. Also, in the cases where resource-poor languages are paired with relatively resource-rich languages (Eng-Hin, Eng-Tel, Eng-HECM), a significant improvement in performance is

Sentence	True Label	SBNA Label
It is perfect for the mall , where teenagers experiencing brain drains can feel good about being mindless and not have to apologize that this is an american film.	NEGATIVE	NEGATIVE
मैं इस उत्पाद से बहुत खुश हूँ। यह आराम दायक है। यह खरीदने लायक है। (I am very happy with the product. It is comfortable. It is worth buying.)	POSITIVE	POSITIVE
దేశంపట్ల కానీ, ప్రజలపట్ల కానీ బాధ్యత కనిపించదు. (He does not see the reponsibility of the country or the people.)	NEGATIVE	NEUTRAL
Jan Jan ka viswas kam ho na jai hind. (People's belief should not reduce Jai hind.)	NEUTRAL	NEUTRAL

Table 4.7 Excerpts of SBNA output in Sentiment Analysis

observed compared to their monolingual performance. The Bi-LSTMs and their shared parameters allow the system to project sentences into the same problem space. This allows the problem space to be language-agnostic. Hence, the abundant resources of other languages are able to promote significant performance improvements in resource-poor languages.

4.7 Critique of SBNA Architecture

SBNA architecture leverages resource-rich languages to enhance text classification in resource-poor languages. The model solves the problem by learning representations of input sentences into a shared language-agnostic problem space. The model employs twin Bi-LSTM networks with shared parameters to capture a task-specific representation of the sentences. These representations are utilized in conjunction with a similarity metric to group sentences with similar classes together. The qualitative analysis showed that SBNA outperforms the state-of-the-art methodologies with training on same language pairs. The quantitative analysis presented a significant enhancement in SBNA's prediction by training the resource-poor language in conjunction with resource-rich language. Experiments conducted on large-scale standard datasets for the tasks of sentiment analysis and emoji prediction revealed that our model significantly outperforms the state-of-the-art approaches. Excerpts of SBNA's outputs for the task of sentiment analysis and emoji prediction is given in Table 4.7 and 4.8 respectively.

Sentence	True Label	SNBA Label
First of all, Park Jimin’ s smile fills my soul with happiness and life.	❤️	❤️
सलमान की जगह केजरीवाल होते तो अभी तक हिरणों से माफ़ी मांगकर फिर अगले शिकार पर निकल जाते। (If it were Kejriwal instead of Salman, then he would asked the buck’s forgiveness and gone for another hunt.)	😜	😉
ఇంటెర్ ఫైల్.ఇంత అవగాహన ఎలా అందుకే చదువు కుంటే ఒస్తుంది కొంటే కాదు అనేది!! (Inter fail. How is this awareness possible? Hence, goes the saying study gets it not buying.)	😊	😊
Neend nhi arhi h ..Kya karu.... (I am not sleepy. What to do?)	😐	😐

Table 4.8 Excerpts of SBNA output in Emoji Prediction

However, the model presents a case for over-fitting. If the amount of data available in the resource-poor language is significantly less, then there is a possibility that the model does not generalize for the language rather just over-fits on the domain of the dataset. Also, SBNA utilizes Bi-LSTM for both the networks in the architecture, but CNNs (Convolutional Neural Network) have proved more effective in case of languages with short term relation dependencies such as English and Spanish. Bi-LSTM are adopted in SBNA because it handles both long-term and short-term dependency languages equally well and the aim was to develop a language-agnostic architecture.

Chapter *5*

Conclusions

Text classification is among the primary challenges of natural language processing. The prominent tasks of text classifications such as sentiment analysis and emoji prediction are some of the most active research areas in the field. The problem lies at the juncture of Information Retrieval and Natural Language Understanding. This is the reason for the extensive amount of attention and studies conducted in the area. A legitimate solution to the problem implies that machines are capable of understanding human-generated text and identify differentiating patterns in the language. Text classification is, especially, challenging in the case of Indian languages that do not nearly boast the amount of resources as some of the relatively better researched languages such as English and Spanish. Also, Indian languages have long term word relations and are morphologically rich. Romanization of the scripts of these languages lead to word variations. This phenomena of code-mixing introduces new challenges and exceptions to the previous efficient text classification systems.

The thesis provides a two-part solution to solving the problem of text classification. First off, the problems due to morphological variety and code-mixing are tackled. This is done using a combination of morphology analyzer and a clustering-based distributional semantic vectors approach. Then, for text classification on the preprocessed data, a deep learning is proposed. The deep learning architecture is Siamese Bi-LSTM Network Architecture (SBNA). The model comprises of an identical twin Bi-LSTM pair with a contrastive loss function on top. The Bi-LSTMs aid the projection of sentences (as sequences of character trigrams) into a language-specific task space. The similarity metric works in coordination with the Bi-LSTMs to merge the language-specific spaces into one.

5.1 Handling Morphological Diversity and Word Variations

Chapter 3 demonstrates that morphology analysis of words, in the case of Indian languages, show effective improvement in the final result. Furthermore, addition of a clustering-based approach for automatic capturing and substitution of different word variations in Code-mixed

Social Media Text (CSMT) proves very efficient in handling the exceptions due to code-mixing of languages. The approach exploits the property that words and their variations share similar context in large noisy text. Thus, a vector space model of the variations provides valuable insights that are later worked upon to arrive at the final clustering-based solution. The final algorithm is validated by using the methodology on POS-tagging and sentiment analysis of CSMT text which showed improvements in the state-of-the-art models over the previous unprocessed data. It is demonstrated that informal word variations in CSMT data belong to the same clusters of semantic space and the clustering approach helps in improving the efficiency of finding the candidate words for the substitution phase. This work contributes to the initial step towards building a generic model for automated preprocessing of CSMT data that is valid across large corpora and multiple language pairs.

5.2 SBNA to solve Text Classification

The thesis introduces a solution to leverage resource-rich languages and enhance text classification in resource-poor languages. The proposed solution SBNA solves the problem by projecting language-pairs into the same problem space. The model employs twin Bi-LSTM networks with shared parameters to capture a task-specific representation of the sentences. The weight updates in the network are mirrored through both the networks in each iteration. These representations are utilized in conjunction with a similarity metric to group sentences with similar classes together. Combining of the language-specific spaces into one aids us in utilizing the resources of a resource-rich languages (English, Spanish) to improve performance in relatively resource-poor languages (Hindi,Telugu).

The qualitative analysis of the results show that SBNA outperforms the state-of-the-art methodologies with training on same language pairs. This result justifies the addition of a similarity metric to produce a similarity-centric model instead of representation/semantics-centric model. Additionally, the computation time of the model is reduced because of the Siamese network's shared parameters. The quantitative analysis presented a significant enhancement in the model's prediction by training the resource-poor language in conjunction with resource-rich language. This outcome proves the assertion that merging the language spaces using siamese networks leads to a significant performance improvement.

Experiments conducted on large-scale standard datasets for the tasks of sentiment analysis and emoji prediction on both standard languages (Hindi, Telugu, English and Spanish) and code-mixed variety (Hindi-English) revealed that SBNA significantly outperforms the state-of-the-art approaches.

5.3 Future Work

Future extensions of this work include testing on diverse NLP tasks on CSMT data, as well as across many more language pairs, especially exploring the challenges faced by working on language pairs beyond languages of the Indian subcontinent. This also makes a case for testing the current model on more applications based on learning similarity like question-answering, conversation systems and semantic similarity.

Related Publications

1. Nurendra Choudhary, Rajat Singh, Ishita Bindlish and Manish Shrivastava.
   Emotions are Universal: Learning Sentiment Based Representations of Resource-Poor Languages using Siamese Networks, 19^{th} International Conference on Computational Linguistics and Intelligent Text Processing, CICLing, 2018.

2. Nurendra Choudhary, Rajat Singh, Ishita Bindlish and Manish Shrivastava.
   Contrastive Learning of Emoji-based Representations for Resource-Poor Languages, 19^{th} International Conference on Computational Linguistics and Intelligent Text Processing, CICLing, 2018.

3. Nurendra Choudhary, Rajat Singh, Ishita Bindlish and Manish Shrivastava.
   Sentiment Analysis of Code-Mixed Languages leveraging Resource Rich Languages, 19^{th} International Conference on Computational Linguistics and Intelligent Text Processing, CICLing, 2018.

4. Rajat Singh, Nurendra Choudhary and Manish Shrivastava.
   Automatic Normalization of Word Variations in Code-Mixed Social Media Text, 19^{th} International Conference on Computational Linguistics and Intelligent Text Processing, CICLing, 2018.

5. Sandeep Sricharan Mukku, Nurendra Choudhary and Radhika Mamidi.
   Enhanced Sentiment Classification of Telugu Text using ML Techniques, In Proceedings of the 4^{th} International Workshop on Sentiment Analysis where AI meets Psychology(SAAIP 2016), International Joint Conference on Artificial Intelligence (IJCAI 2016), New York City, USA, July 10, 2016.

6. Nurendra Choudhary, Rajat Singh, Vijjini Anvesh Rao and Manish Shrivastava.
   Twitter Corpus of Resource-Scarce Languages for Sentiment Analysis and Multilingual Emoji Prediction, 27^{th} International Conference on Computational Linguistics, COLING, 2018.

Bibliography

[1] P. Arora. Sentiment analysis for hindi language. *MS by Research in Computer Science*, 2013.

[2] A. Bakliwal, P. Arora, and V. Varma. Hindi subjective lexicon: A lexical resource for hindi polarity classification. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC)*, pages 1189–1196, 2012.

[3] A. Balamurali, A. Joshi, and P. Bhattacharyya. Cross-lingual sentiment analysis for indian languages using linked wordnets. *Proceedings of COLING 2012: Posters*, pages 73–82, 2012.

[4] F. Barbieri, M. Ballesteros, and H. Saggion. Are emojis predictable? *arXiv preprint arXiv:1702.07285*, 2017.

[5] F. Barbieri, F. Ronzano, and H. Saggion. What does this emoji mean? a vector space skip-gram model for twitter emojis. In *LREC*, 2016.

[6] U. Barman, A. Das, J. Wagner, and J. Foster. Code mixing: A challenge for language identification in the language of social media. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pages 13–23, 2014.

[7] A. Bharati, V. Chaitanya, R. Sangal, and K. Ramakrishnamacharyulu. *Natural language processing: a Paninian perspective*. Prentice-Hall of India New Delhi, 1995.

[8] M. Boden. A guide to recurrent neural networks and backpropagation. *the Dallas project*, 2002.

[9] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a" siamese" time delay neural network. In *Advances in Neural Information Processing Systems*, pages 737–744, 1994.

[10] 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.

[11] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 539–546. IEEE, 2005.

[12] A. Das and S. Bandyopadhyay. Sentiwordnet for indian languages. In *Proceedings of the Eighth Workshop on Asian Language Resouces*, pages 56–63, 2010.

[13] A. Das, H. Yenala, M. Chinnakotla, and M. Shrivastava. Together we stand: Siamese networks for similar question retrieval. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 378–387, 2016.

[14] D. Das and S. Bandyopadhyay. Labeling emotion in bengali blog corpus–a fine grained tagging at sentence level. In *Proceedings of the Eighth Workshop on Asian Language Resouces*, pages 47–55, 2010.

[15] L. Duran. Toward a better understanding of code switching and interlanguage in bilinguality: Implications for bilingual instruction. *The Journal of Educational Issues of Language Minority Students*, 14(2):69–88, 1994.

[16] P. Ekman and W. V. Friesen. A new pan-cultural facial expression of emotion. *Motivation and emotion*, 10(2):159–168, 1986.

[17] Y. Gao, W. Rong, Y. Shen, and Z. Xiong. Convolutional neural network based sentiment analysis using adaboost combination. In *Neural Networks (IJCNN), 2016 International Joint Conference on*, pages 1333–1338. IEEE, 2016.

[18] S. Ghosh, S. Ghosh, and D. Das. Part-of-speech tagging of code-mixed social media text. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 90–97, 2016.

[19] X. Glorot, A. Bordes, and Y. Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 513–520, 2011.

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

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

[22] J. J. Gumperz. *Discourse strategies*, volume 1. Cambridge University Press, 1982.

[23] M. Gysels. French in urban lubumbashi swahili: Codeswitching, borrowing, or both? *Journal of Multilingual & Multicultural Development*, 13(1-2):41–55, 1992.

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

[25] 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.

[26] A. Joshi, A. Balamurali, and P. Bhattacharyya. A fall-back strategy for sentiment analysis in hindi: a case study. *Proceedings of the 8th ICON*, 2010.

[27] A. Joshi, A. Prabhu, M. Shrivastava, and V. Varma. Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text. In *COLING*, pages 2482–2491, 2016.

[28] S.-M. Kim and E. Hovy. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1367. Association for Computational Linguistics, 2004.

[29] Y. Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[30] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

[31] Y. LeCun and F. J. Huang. Loss functions for discriminative training of energy-based models. In *AIStats*, volume 6, page 34, 2005.

[32] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[33] B. Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167, 2012.

[34] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[35] N. Mittal, B. Agarwal, G. Chouhan, N. Bania, and P. Pareek. Sentiment analysis of hindi reviews based on negation and discourse relation. In *Proceedings of the 11th Workshop on Asian Language Resources*, pages 45–50, 2013.

[36] A. Mogadala and V. Varma. Retrieval approach to extract opinions about people from resource scarce language news articles. In *Proceedings of the First International Workshop on Issues of Sentiment Discovery and Opinion Mining*, page 4. ACM, 2012.

[37] I. Mozetič, M. Grčar, and J. Smailović. Multilingual twitter sentiment classification: The role of human annotators. *PloS one*, 11(5):e0155036, 2016.

[38] S. S. Mukku, N. Choudhary, and R. Mamidi. Enhanced sentiment classification of telugu text using ml techniques. In *SAAIP@ IJCAI*, pages 29–34, 2016.

[39] S. S. Mukku, S. R. Oota, and R. Mamidi. Tag me a label with multi-arm: Active learning for telugu sentiment analysis. In *International Conference on Big Data Analytics and Knowledge Discovery*, pages 355–367. Springer, 2017.

[40] P. Muysken. *Bilingual speech: A typology of code-mixing*, volume 11. Cambridge University Press, 2000.

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

[42] B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 115–124. Association for Computational Linguistics, 2005.

[43] B. Pang, L. Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008.

[44] 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.

[45] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.

[46] K. Sarkar and S. Chakraborty. A sentiment analysis system for indian language tweets. In *International Conference on Mining Intelligence and Knowledge Exploration*, pages 694–702. Springer, 2015.

[47] R. Sequiera, M. Choudhury, P. Gupta, P. Rosso, S. Kumar, S. Banerjee, S. K. Naskar, S. Bandyopadhyay, G. Chittaranjan, A. Das, et al. Overview of fire-2015 shared task on mixed script information retrieval. In *FIRE Workshops*, volume 1587, pages 19–25, 2015.

[48] A. Sharma, S. Gupta, R. Motlani, P. Bansal, M. Srivastava, R. Mamidi, and D. M. Sharma. Shallow parsing pipeline for hindi-english code-mixed social media text. *arXiv preprint arXiv:1604.03136*, 2016.

[49] S. Sharma, P. Srinivas, and R. C. Balabantaray. Text normalization of code mix and sentiment analysis. In *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*, pages 1468–1473. IEEE, 2015.

[50] R. Singh, N. Choudhary, I. Bindlish, and M. Shrivastava. Neural network architecture for credibility assessment of textual claims. *arXiv preprint arXiv:1803.10547*, 2018.

[51] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1201–1211. Association for Computational Linguistics, 2012.

[52] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[53] T. Solorio, E. Blair, S. Maharjan, S. Bethard, M. Diab, M. Ghoneim, A. Hawwari, F. AlGhamdi, J. Hirschberg, A. Chang, et al. 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*, pages 62–72, 2014.

[54] V. K. R. Sridhar. Unsupervised text normalization using distributed representations of words and phrases. 2015.

[55] C. Taggart. *New Words for Old: Recycling Our Language for the Modern World*. Michael O'Mara Books, 2015.

[56] P. D. Turney. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 417–424. Association for Computational Linguistics, 2002.

[57] Y. Vyas, S. Gella, J. Sharma, 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)*, pages 974–979, 2014.

[58] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 448–456. ACM, 2011.

[59] 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*, pages 90–94. Association for Computational Linguistics, 2012.

[60] J. M. Wiebe, R. F. Bruce, and T. P. O'Hara. Development and use of a gold-standard data set for subjectivity classifications. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 246–253. Association for Computational Linguistics, 1999.

[61] T. Wilson and J. Wiebe. Annotating opinions in the world press. In *Proceedings of the Fourth SIGdial Workshop of Discourse and Dialogue*, 2003.

[62] H. Yu and V. Hatzivassiloglou. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 129–136. Association for Computational Linguistics, 2003.

[63] X. Zhang and Y. LeCun. Text understanding from scratch. *arXiv preprint arXiv:1502.01710*, 2015.