Towards Understanding Code-Mixed Social Media Text

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

Masters of Science (by Research)
in
Computational Linguistics

by

Sakshi Gupta
201125087
sakshi.gupta@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, India
August 2016

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Towards Understanding Code-Mixed Social Media Text" by Sakshi Gupta, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Dr. Radhika Mamidi

For my grandfather

Acknowledgments

Thank you, Dr. Radhika Mamidi for being my advisor, and motivating me to pursue research in a really passionate manner. Your suggestions have shaped my thoughts, and given me direction when I most needed it. Our discussions have inspired me to explore the wonderful world of computational linguistics. I also truly appreciate the freedom you gave me as a student to work at my own time, and set my own expectations. They allowed me to create a work-life balance for myself, which is a skill that I hope to improve over the coming years. You have been extremely patient with me, and for that I sincerely respect you. You have been a guide even outside the domain of linguistics.

I thank Prof. Dipti Misra Sharma for all her constructive feedback and guidance. Your lectures have been very instrumental in inculcating a spirit of curiosity in me. Your support has been most valuable in this work. I remember you suggesting me to work on Code-Mixed data a very long time back, and reflected on your far-sightedness much later. Thank you for being the kind, helpful and enthusiastic professor that you are!

Special thanks to Dr. Manish Srivastava for guiding me time and again, and making sure that my work was moving at a good pace. Despite not being my research guide, I have often had discussions with him on what to work on, and how to concretise my ideas.

I thank all the members of Language Technologies and Research Center for being there whenever I needed any advice. My heartfelt thanks to Piyush Bansal for carrying out elaborate discussions and experiments with me that have shaped my understanding. Thanks to Arnav Sharma and Raveesh Motlani for constantly coming up with ideas for improving our projects, and for your suggestions at various point of time during my research. I honestly respect you three for all the help that you provided - my research would not be complete with your companionship. I will cherish those numerous hours in the lab when we discussed ideas and how to improve our models, ran all kinds of experiments and celebrated acceptances and submissions.

I thank all my friends who have shared the curiosity, and fascination towards intelligent machines with me. I'd like to thank all the amazing people who've proofread the drafts, and helped me create the datasets - especially Diksha Yadav, Nishant Krishan, Achira Maloo and Aditi Gupta. It was with their hard effort that I could create and tune huge chunks of data. Thank you, friends!

From the bottom of my heart, I thank my mom and dad, for your love, support and advice. You have always made every possible effort to give me greater comfort and joy in life, and it is because of your

decades of effort that I find enjoyment in my work. Thank you for making me curious, excited, joyful and for showing me how fulfilling work can be.

Finally, I thank my sister for always being a constant support, for consoling me over failed experiments, for motivating me to work even when I felt hopeless, and for guiding me through my research and my life. Thank you, Swati didi. You are the most important mentor I've had. Your love and excitement for research despite immense pressure makes me proud, and I hope I can do as much as you have towards the work you love.

Abstract

The Web 2.0, through its different platforms such as blogs, social networks, microblogs, or forums allows users to freely write content on the Internet, with the purpose to provide, share and use information. It is known that these type of platforms are among the top visited websites[1], and their interest is growing more and more. Given the important role that social media usage is increasingly playing in daily life, a growing body of literature has emerged in the research community that aims to mine social media content, or to evaluate the linguistic aspects of that content in order to better understand its dynamics (Golder et al., 2011) [22].

This user-generated content has a huge drawback, which is the informal nature of the language used. Non-standard abbreviations, contracted spelling variations, casual grammatical structure are just some of the aspects of social media language.

Over the past few decades, sociolinguists have been interested in a phenomena called "code mixing", which has been observed in social media data. Code-Mixing refers to the embedding of linguistic units such as phrases, words or morphemes of one language into an utterance of another language. It is frequently seen in user generated content on social media like Facebook and Twitter, especially by multilingual users. Apart from the inherent linguistic complexity, the analysis of code-mixed content poses complex challenges owing to the presence of spelling variations, transliteration and non-adherence to a formal grammar.

Due to the presence of such data all across social media, there is also a need to understand it. For any downstream Natural Language Processing task, tools that are able to process and analyse code-mixed data are required. The first steps to understanding this data are language identification and word normalisation systems, so that we obtain the standard form of a crude sentence from social media.

In this thesis, we have developed a system for language identification and word normalisation for Hindi-English code-mixed social media text (CMST). We have provided annotation guidelines for our system, after analysing the complex nature of the dataset used.

Using this system, we have released a dataset of 1446 code-mixed Hindi-English sentences along with the associated language and normalisation labels. To the best of our knowledge, our work is the

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

first attempt at the creation of an annotated linguistic resource for this language pair, which is also made public.

We have also performed experiments with shallow parsing, in an attempt to build a complete pipeline from raw data to shallow parsed data. Our pipeline consists of 4 modules - Language Identification, Normalisation, POS Tagging, and Shallow Parsing. As far as we understand, we are the first to attempt shallow parsing on code-mixed social media text. This system has been released online.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Multilingual speakers often switch back and forth between languages when speaking or writing, mostly in informal settings. This language interchanging involves complex grammar and the terms "code switching" or "code mixing" are used to describe it (Lipski et al., 1978) [33]. Code-mixing refers to the use of linguistic units from different languages in a single utterance or sentence, whereas code-switching refers to the co-occurrence of speech extracts belonging to two different grammatical systems (Gumperz et al., 1982) [23]. As both phenomena are frequently observed on social media platforms in similar contexts, we use code-mixing to refer to both the scenarios in this work. However, before delving further into code-mixed data, it is important to first address the complications in social media data itself. In the following sections, we explain how code-mixed data is gaining importance, especially in the social media context. It is also crucial to understand the motivation behind chosing the Hindi-English language pair.

1.1 Effect of Social Media on Language

Social media has greatly altered language. Crystal et al. (2001) claimed that new forms of communication caused changes to the traditional spoken and written languages genres [17]. These new forms were a result of the internet and social media in particular - Facebook [1], Twitter [2], and so on. Written language tends towards a formal tone and a more defined structure whereas spoken language is usually more casual and context dependent. With the advent of technology, this distinction between written and spoken languages blurred (Herring et al., 2001) [28].

[1] www.facebook.com
[2] www.twitter.com

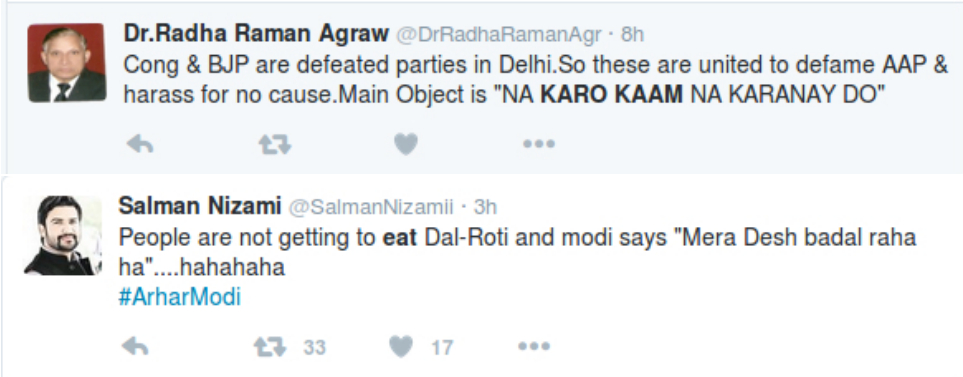

Figure 1.1: Examples of code-mixed Hindi-English tweets

Scholars observed that the language across the internet, especially in a synchronous communication (such as an online chat room), resembled spoken communication. It was observed to be less formal, simpler and very similar to speech (Danet et al., 2007) [18]. Crystal et al. (2001) further supported this argument by suggesting that colloquial features of spoken language such as repetition of phrases, use of reaction signals (such as "you know", "you see") are found in social media texts [17]. This is the result of the conversion of casual vernacular to written form. People use words and symbols to express emotions, which results in inconsistent language generation across users since there is no defined structure for this usage. For instance, the phrase 'oh my god' can be written as 'OH MY GOD!!', 'oh my god', 'OMIGOD', 'OMG', and so on. Also, given that language always undergoes economisation, word shortenings are observed to a great extent on social media. Vowels are dropped, repeated alphabets are merged - which make analysis of this data a rather complex task.

1.2 Complexities of Code-Mixed Social Media Text (CMST)

Multilingual speakers are known to use a mix of the languages that they are fluent in, especially on social media platforms as is seen in the work of Cardenas et al. (2007) [9] and Danet et al. (2007) [18]. Figure 1.1 shows some examples of the kind of data found on social media, collected from public Twitter pages.

For this work, we have chosen the Hindi-English language pair for code mixing related experiments. This is a relevant pair because Hindi is one of the most commonly spoken languages in India, and English is the official court language of the country. As a result, social media across India exhibits a lot of Hindi-English code mixing.

English	Hindi
b	ब, भ
s, c	स
d	ड, द, ड़, ढ, ध

Table 1.1: Examples of Character Mapping between Hindi and English

Hindi and English, being very different languages, present added complexities when generating code-mixed text. Two interesting problems are:

1. The script for Hindi is Devanagiri, but social media code mixed Hindi-English data has text written in Roman script (Choudhury et al., 2010) [12]. It is thus harder to have a standard spelling for a Hindi word, which increases problems in word normalisation. Moreover, there is no 1-1 mapping between characters of English and Hindi. Some examples of the complex mapping between Hindi and English characters, as seen on social media text, are given in Table 1.1. This creates a problem in trying to back-transliterate Romanised Hindi words to Devanagiri.

2. The position of a word in a sentence is a good measure of understanding the role played by that word in the sentence (subject, object, and so on). Hindi sentences follow a Subject-Object-Verb (SOV) ordering, whereas English uses Subject-Verb-Object (SVO) ordering. In a code-mixed sentence, this ordering is lost; and thus an important feature for sentence analysis is lost.

To give an example of how code-mixed data is complex, here are a few variations of the sentence "I went to her house yesterday and ate some rice-pudding" that one may write while code-mixing (with Hindi-English) on social media:

- **Sentence :** I went to her house yesterday and ate some kheer.

  Gloss : I went to her house yesterday and ate some rice-pudding (kheer).

  Hindi words : kheer.

  In this sentence, English is the parent or matrix language, and there is only 1 Hindi word. The word order is SVO (English).

- **Sentence :** Kal main uske ghar gayi and ate some kheer.

Gloss : Yesterday(kal) I (main) her(uske) house(ghar) went(gayi) and ate some rice-pudding (kheer).

Hindi words : kal, main, uske, ghar, gayi, kheer.

This sentence is more complex - it has 2 phrases, separated by “and”. The first phrase is entirely formed by Hindi words, and follows the word ordering for Hindi (SOV); whereas the second phrase has English as the matrix language, follows the syntax for English(SVO) and contains a Hindi word. Thus, the entire syntax has a mix of Hindi-English syntactic rules.

- **Sentence :** Ystrdy I went 2 her house aur kheer khayi.

  Gloss : Yesterday(ystrdy) I went to(2) her house and(aur) rice-pudding(kheer) ate(khayi)

  Hindi words : aur, kheer, khayi.

  Here, the first phrase ("Ystrdy I went 2 her house") is entirely in English, and the second phrase is entirely in Hindi, following their respective grammar(SVO, SOV).

As is evident from the examples above, handling Hindi-English data has a lot of hard problems associated with it, and attempting to solve these is the motivation behind this thesis.

1.3 Contributions of the thesis

Bali et al. (2014) state that any future work on code-mixed in social media content would have to involve a deeper analysis at the intersection of structural and discourse linguistics [4]. In this thesis, we have developed a Language Identification and Word Normalisation system for code-mixed social media text (CMST), choosing Hindi-English as our language pair. While developing this system, we felt a lack of publicly available annotated datasets for Hindi-English CMST. In order to fix this, we have also created a data-set of 1446 Hindi-English code-mixed sentences. We have performed language identification and word normalisation on these sentences with the help of systems that we developed for the 2 tasks. The errors in the annotation of the created dataset due to the inaccuracies of the system were then manually corrected. For each word in a sentence, the word, language ID, and its normalised form are made available in a public dataset available online[3].

We also implemented a shallow parsing pipeline as an experiment to understand the further uses of our normalisation system, consisting of 4 subsystems - language identification and normalisation modules (mentioned above), along with a POS tagger and shallow parsing module. This pipeline was specifically designed for CMST and was the first of its kind to the best of our knowledge, and has

[3] https://github.com/sakshigupta93/normalisation-dataset

been released online[4]. The motivation behind creating this system is to build tools for automated text processing for this language pair.

1.4 Thesis Organisation

This thesis is divided into 7 chapters. In Chapter 2, we give a detailed study of work previously done in this field. Chapter 3 talks about the process of performing language identification and normalisation of CMST for Hindi and English language, along with annotation guidelines for the 2 processes. This chapter describes the different system experiments performed, the features used along with the system accuracies. This is followed by a shallow parsing pipeline described in Chapter 4 - which describes 2 new modules for POS tagging and Shallow Parsing, along with the features and annotation guidelines used to build this system.

In Chapter 5, we describe the dataset created, along with the observations from its annotation process; finally concluding with future work and conclusion in Chapters 6 and 7.

[4] http://bit.ly/csmt-parser-api

Chapter 2

Related Work

A lot of work has been done on social media data and code-mixed data over the past decades. Code-mixing being a relatively newer phenomena has gained attention of researchers only in the past 2 decades. On the other hand, Language Identification has been considered to be a solved problem by McNamee et al. (2009) [35], but new complications were added to this task in the context of code-mixed social media data. Similarly, Word Normalisation has been extensively studied, but there is little work done on the Hindi-English language pair.

2.1 Code-Mixed Data

One of the earliest works on code-switching for online data was done by Warshauer et al. (2002) [47]. They looked at English and Arabic language use in email communication by a group of young professionals, and concluded that English was more frequently used both when searching the Internet and in formal email communications. Interestingly, they also found that a Romanised version of Egyptian Arabic was more frequently used in informal emails, chats and even to express personal content, as opposed to the classical Arabic.

Hidayat et al. (2012) showed that Facebook users tend to mainly use inter-sentential switching over intra-sentential, and report that 45% of the switching was instigated by real lexical needs, 40% was used for talking about a particular topic, and 5% for content clarification [29]. Dey et al. (2014) also investigated the rules for code-switching in Hindi-English data by interviewing bilingual students and transcribing their utterances [20]. They found that on average, roughly 67% of each sentence were made up of Hindi words and 33% English words.

2.2 Language Identification

Previous work on text has mainly been on identifying a language from documents of several languages, such that even when evidence is collected at word level, evaluation is at document level (Prager et al., 1999 [41]; Singh et al., 2007 [42]; Yamaguchi et al., 2012 [50]). Carter (2012) colleced tweets in five different European languages and analysed multi-lingual microblogs for understanding the dominant language in any specific tweet [10]. He then performed post-level language identification, experimenting with a range of different models and a character n-gram distance metric, reporting a best overall classification accuracy of 92.4%. Tratz et al. (2013) on the other hand worked on highly code mixed tweets, with 20.2% of their test and development sets consisting of tweets in more than one language [43]. They aimed to separate Romanised Moroccan, Arabic (Darija), English and French tweets using a Maximum Entropy classifier, achieving F-scores of .928 and .892 for English and French, but only .846 for Darija due to low precision.

Nguyen et al. (2013) worked on language identification at the word level on randomly sampled Turkish-Dutch posts from an online chat forum [37]. They compared dictionary based methods to statistical ones. Their best system reached an accuracy of 97.6%, but with a substantially lower accuracy on post level (89.5%), even though 83% of the posts actually were monolingual. They report on language identification experiments performed on Turkish and Dutch forum data. Experiments have been carried out using language models, dictionaries, logistic regression classification and Conditional Random Fields. They find that language models are more robust than dictionaries and that contextual information is helpful for the task.

Furthermore, Barman et al. (2014) investigated language identification at word level on Bengali-Hindi-English code-mixed social media text [5]. They annotated a corpus with more than 180,000 tokens and achieved an accuracy of 95.76% using statistical models with monolingual dictionaries.

2.3 Normalisation and further work

Owing to massive growth of SMS and social media content, text normalisation systems have gained attention where the focus is on conversion of these tokens into standard dictionary words.

The first Chinese monolingual chat corpus was released by Wong et al. (2008) [48]. They also introduced a word normalisation model, which was a hybrid of the Source Channel Model and phonetic mapping model.

Wang et al. (2009) work with abbreviations for spoken Chinese rather than for English text messages [46]. They first perform an abbreviation generation task for words and then reverse the mapping in a look-up table. They use conditional random fields as a binary classifier to determine the probability of

removing a Chinese character to form an abbreviation. They rerank the resulting abbreviations by using a length prior modeled from their training data and co-occurrence of the original word and generated abbreviation using web search.

A commonly accepted research methodology is treating normalisation as a noisy channel problem. Choudhary et al. (2007) explain a supervised noisy channel framework using HMMs for SMS normalisation [14]. This work was then extended by Cook et al. (2009) [16] to create an unsupervised noisy channel approach using probabilistic models for common abbreviation types and choosing the English word with the highest probability after combining the models. Beaufort et al. (2010) combine a noisy channel model with a rule-based finite-state transducer and got reasonable results on French SMS, but did not test their method on English text [6]. Xue et al. (2011) adopted the noisy-channel framework for normalisation of microtext and proved that it is an effective method for performing normalisation [49].

The work done by Han et al. (2011) determines whether a given out-of-vocabulary (OOV) word needs to be expanded or is some other type of properly-formed OOV [25]. For those predicted to be ill-formed, a confusion set of possible candidate words is generated based on a combination of lexical and phonetic edit distance and the top word is chosen in context using LM and dependency parse information. Han et al. (2012) attempts to automatically build a normalisation dictionary offline using the distributional similarity of tokens combined with their string edit distance [26].

Vyas et al. (2014) worked on POS tagging for Hindi-English data [45]. For Hindi normalisation, they used the system built by Gella et al. (2013) [21] but they did not normalise English text as they used the Owoputi Twitter POS Tagger (2013) [39] in the next step, which does not require normalised data.

Chapter 3

Towards Normalising CMST

In this chapter, we explain the language identification and normalisation systems developed in detail. We have also elaborated on the annotation guidelines followed for both the systems, and reasoned behind the decisions made in the guidelines.

The pipeline flowchart has been explained in Figure 3.1.

3.1 Dataset Used

The code-mixed data was manually selected from the data shared for Subtask 1 of FIRE-2014 Shared Task on Transliterated Search [13]. The dataset was annotated with language tags, along with normalisations for Hindi words. We broke down each post into sentences, finally selecting 858 sentences for our dataset. We used their language tags but annotated the data with normalised forms, POS tags and Shallow Parsing labels ourselves. A small change was also made to their language tags. They had used 5 tags - en (English), hi (Hindi), ne (Named Entities), acro (Acronyms), univ (Universal). We merged the tags “acro”, “ne” and “univ” to “rest”.

3.1.1 Data Statistics

Table 3.1 and Table 3.2 show the distribution of the dataset at sentence and token level respectively. The language of 63.33% of the tokens in code-mixed sentences is Hindi. Based on the distribution, it is reasonable to assume that Hindi is the matrix language (Azuma et al., 1993 [3]; Myers et al., 1997 [36]) in most of the code-mixed sentences.

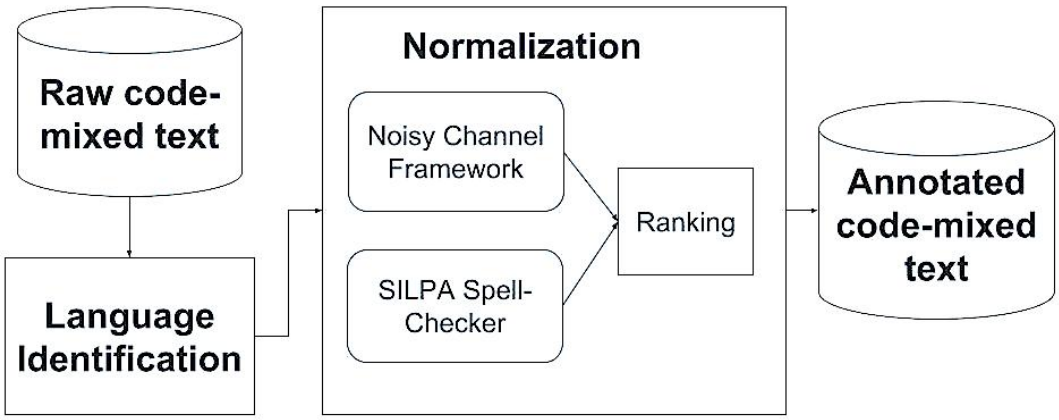

Figure 3.1: Schematic Diagram of the Normalisation Pipeline

Lang.	Sentences
English	141 (16.43%)
Hindi	111 (12.94%)
Code-mixed	606 (70.63%)
Total	858

Table 3.1: Data Distribution at Sentence Level

Lang.	All Sentences	Only CM Sentences
Hindi	6318 (57.05%)	5352 (63.34%)
English	3015 (27.22%)	1886 (22.32%)
Rest	1742 (15.73%)	1212 (14.34%)
Total	11075	8450

Table 3.2: Data Distribution at Token Level

Of the 3015 English tokens, 2010 were already in their normalised form, whereas of the 6318 Hindi tokens, 3387 tokens would have been in their normalised form if we used a back transliterator [1].

We also computed the average code-mixing index or CMI (Das and Gambäck, 2014)[19], using Equation 3.1. This parameter is used to determine the amount of code-mixing found in an utterance/discourse.

$$CMI = \begin{cases} 100 \times [1 - \frac{max[w_i]}{n-u}] & : n > u \\ 0 & : n = u \end{cases} \tag{3.1}$$

The CMI value for this dataset, computed as an average over every sentence was found to be 17.46, and the CMI value computed as an average over code-mixed sentences alone was found to be 24.61.

This whole dataset was annotated by eight Hindi-English bilingual speakers. Two other annotators reviewed and cleaned it. To measure inter-annotator agreement, another annotator read the guidelines and annotated 25 sentences (334 tokens) from scratch. The inter-annotator agreement calculated using Cohen's κ [15] came out to be 0.97 for language identification.

3.1.2 Dataset examples

1. hy... try fr sm gov job jiske forms niklte h...
   Gloss: Hey(hy)... try for(fr) some(sm) government(gov) job which(jiske) forms give(niklte) out(h)...
   Translation: Hey... try for some government job which gives out forms...

2. To tum divya bharti mandir marriage kendra ko donate karna
   Gloss: So(to) you(tum) divya bharti temple(mandir) marriage center(kendra) to(ko) donate do(karna)
   Translation: So you donate to divya bharti temple marriage center

   The dataset is comprised of sentences similar to example 1 and 2. Example 1 shows code-switching as the language switches from English to Hindi whereas example 2 shows code-mixing as some English words are embedded in a Hindi utterance. Spelling variations (`sm` - `some`, `gov` - `government`), ambiguous words (`To` - `So` in Hindi or `To` in English) and non-adherence to a formal grammar (out of place ellipsis - `...`, no or misplaced punctuation) are some of the challenges evident in analysing the examples above.

[1] `https://github.com/ltrc/python-irtrans`

3.1.3 Annotation Guidelines

Due to the complexities discovered in the data (see example in Section 3.1.2), annotation guidelines play an important role in our dataset preparation process. Here, we explain these guidelines for the Language Identification and Normalisation processes. These guidelines were given to the annotators who manually tagged the data.

3.1.3.1 Language Identification

Similar to Barman et al.(2014)[5], we treated language identification as a three class ('hi', 'en', 'rest') classification problem. Every word was given a tag out of three - **en**, **hi** and **rest** to mark its language. Words that a bilingual speaker could identify as belonging to either Hindi or English were marked as 'hi' or 'en', respectively. The label 'rest' was created in order to accomodate words that did not strictly belong to any language, described below:

1. Symbols, emoticons and punctuation

2. **Named Entities :** Named Entities are language independent in most cases. For instance, 'Jack' would be represented by equivalent characters in Hindi and English. Exceptions rise in cases such as 'India' in English, where 'Bhaarat' would be written in Hindi to refer to 'India', and is rightly a Hindi word. For simplicity, we have considered all such ambiguities as 'rest'.

3. **Acronyms :** This includes SMS acronyms such as 'LOL' (Laugh out loud), and established contractions such as 'USA' (United States of America). These also include foreign acronym like QED (Latin, *quod erat demonstrandum*). Acronyms are very interesting linguistic units, and play an important role in social media text. They represent not just entities but also phrases and reactions. We wanted to keep their analysis separate from the rest of the language; and hence they were categorised as 'rest' in our dataset.

4. **Foreign words :** A word borrowed from a language except Hindi and English has been treated as 'rest' as well. This does not include commonly borrowed Urdu words in Hindi; they are treated as a part of Hindi language.

5. **Sub-lexical code-mixing :** Any word with word-level code-mixing has been classified as 'rest', since it represents a more complex morphology. For instance, `chapattis` (Gloss: flat-roasted Indian bread) which is a Hindi word (*chapatti*) following English morphology (plural marker -*s*). Another example is `pencilein` (Gloss: pencils) which is an English word following Hindi morphology (Hindi plural marker -*ein*).

3.1.3.2 Normalisation

Words with language tag 'hi' in Roman script were labeled with their standard form in the native script of Hindi, Devanagari, i.e. a back-transliteration was performed. Words with language tag 'en' were labeled with their standard spelling. Words with language tag 'rest' were kept as they are.

Following are some case-specific guidelines:

1. In case a token consists of two words (due to an error in typing the space), the tokens are separated and written in their original script. For instance, 'whatis' would be normalised to 'what is', with the language ID as English.

2. If the first word of any sentence is in English, its first letter is capitalised.

3. In cases where multiple spellings of a word are considered acceptable, we have allowed both spelling variations to exist as the standard spellings. For instance, in 'color' and 'colour', 'dialogue' and 'dialog', both spellings are valid. This decision was taken so that the dataset and model are not restricted.

4. Contractions such as 'don't' and 'who's' have been left undisturbed. The dataset thus contains both variations - 'don't' and 'do not', depending on the original chat text. The rationale behind this was to not tamper with the original chat text beyond a particular extent.

5. Hindi has evolved through the past decades, and often we see variations in spelling of a single word. We observed the variation patterns and resolved standard spellings for our dataset, based on which words were more common in usage.

3.2 Language Identification System

While language identification at the document level is a well-established task (McNamee et al., 2005) [35], identifying language in social media posts has certain challenges associated to it. Spelling errors, phonetic typing, use of transliterated alphabets and abbreviations makes this interesting. When further combined with the problem of transliterated code-mixing, it becomes even harder to disambiguate words of one language from another.

Model	Accuracy
Linear SVM	92.12%
Logistic Regression	91.58%
Random Forest	85.44%
Decision Tree	83.49%

Table 3.3: Model Accuracies for Language Identification

3.2.1 System Description

Initially, we ran an experiment to understand which machine learning model would best suit our language identification task. We treated the process as a 2-class classification problem, using "en" and "hi" tags. Table 3.3 shows the accuracies for the different models that we worked with.

Following this experiment, we ran two experiments for the language identification module, which involved comparing a trained CRF and SVM model, now treating this problem as a 3-class classification ("en", "hi" and "rest"). The following features were used:

1. **BNC**: The normalised frequency of a word in British National Corpus (BNC)[2], a computer corpus of 100 million words of British English, written and spoken. (Aston and Burnard, 1998) [2]

2. **LEXNORM**: A binary feature indicating presence of the word in the lexical normalisation dataset released by Han et al. (2012) [26]. We have used this since spelling variations are expected in social media data.

3. **HINDI_NORM**: A binary feature indicating presence of the word in a list of 30,823 transliterated [3] Hindi words as released by Gupta et al. (2012) [24]. This list contains multiple spelling variations for common Hindi words in Roman script.

4. **HINDI_DICT**: A dictionary of Hindi words, released by Biemann et al. (2007) [8]. This dictionary contains 1,17,789 words in the Devanagiri script, which were converted to Roman script using a transliteration system [4].

5. **NGRAM**: We have taken into account the N-grams (Cavnar et al., 1994) [11] formed from the given word, since it is an extremely well-rewarding and common approach followed by many language identification researchers. We have experimentally found that the optimum N-gram lengths are 2-4.

[2] http://www.natcorp.ox.ac.uk/

[3] Transliterated: Words from one language written in another script, in this case Hindi words that were written in the Roman script

[4] https://github.com/irshadbhat/python-irtrans

6. **AFFIXES**: Hindi and English being morphologically rich show a great amount of inflectional morphology. Inflection on a word is defined as the modification of the word using certain morphemes, to express its morphology. A lot of such morphemes do not exist independently, but only as a suffix or prefix to a given word, and thus patterns can be identified within the affixes of these two languages as important distinguishing features. Table 3.4 contains some common suffixes used in Hindi and English.

7. **CAPTILISATION**: It was expected that capitalisation would be a valid feature for this ML task, but we found that our accuracies went down after adding this feature. This feature was added in an attempt to segregate "Rest" words such as Named Entities, Proper Nouns, etc, but it was found that social code-mixed data is too casual and capital letters are used more often for emphasis than for named entities.

Purpose	**Language**	**Affix & Example**	**Gloss**
Pluralisation	English	‘s’ as in ‘books’; ‘es’ as in ‘mangoes’	-
	Hindi	‘ein’ as in ‘kitabein’	Books
Doer (verb to noun)	English	‘er’ as in ‘painter’; ‘or’ as in ‘actor’	-
	Hindi	‘era’ as in ‘looterea’	Thief
	Hindi	‘aiya’ as in ‘gavaiya’	Singer

Table 3.4: Common Suffixes Used in English and Hindi

Using these features and including context from n previous and n next words (by appending the corresponding feature vectors to the current word‘s feature vector), we trained a linear SVM [44].

In another experiment we modeled language identification as a sequence labeling task, where we employed CRF [31] into usage. The idea behind this was that code-mixed text has some inherent structure which is largely dictated by the matrix language of the text.

Features	**Accuracy**
BNC	61.26
+LEXNORM	71.43
+HINDI_DICT	77.50
+HINDI_NORM	79.13
+CAPT	73.18
+NGRAM	93.18
+AFFIXES	93.98

Table 3.5: Feature Ablation for Language Identification Module (CRF)

3.2.2 System Accuracy

The approach using CRF had a greater accuracy, which validated our hypothesis and also proved that context is crucial in this process. The results of this module are shown in Table 3.5.

3.3 Normalisation

Once the language identification task was complete, there was a need to convert the noisy non-standard tokens (such as Hindi words inconsistently written in many ways using the Roman script) in the text into standard words. To fix this, a normalisation module that performs language-specific transformations, yielding the correct spelling for a given word was built, which is described in Section 3.3.1.

3.3.1 System Description

Two language specific normalisers, one for Hindi and other for English, had two sub-normalisers each, as described below. 'Rest' words were left undisturbed. Both sub-normalisers generated normalised candidates which were then ranked, as explained later in this subsection.

1. **Noisy Channel Framework** A generative model was trained to produce noisy (unnormalised) tokens from a given normalised word. Using the model's confidence score and the probability of the normalised word in the background corpus, *n*-best normalisations were chosen. First, we obtained character alignments between noisy Hindi words in Roman script (H_r) to normalised Hindi words-format(H_w) using GIZA++ [38] on 30,823 Hindi word pairs of the form (H_w <-> H_r) (Gupta et al., 2012) [24]. Next, a CRF classifier was trained over these alignments, enabling it to convert a character sequence from Roman to Devanagari using learnt letter transformations. Using this model, noisy H_r words were created for H_w words obtained from a dictionary of 1,17,789 Hindi words (Biemann et al., 2007) [8]. Finally, using Equation 3.2, we computed the most probable H_w for a given H_r.

$$\begin{aligned} H_w &= argmax_{H_{w_i}} p(H_{w_i}|H_r) \\ &= argmax_{H_{w_i}} p(H_r|H_{w_i}) p(H_{w_i}) \end{aligned} \tag{3.2}$$

where $p(H_{w_i})$ is the probability of word H_{w_i} in the background corpus.

The accuracy of this sub-module was 58.51% for English and 69.65% for Hindi language words.

P@n for various n	English	Hindi	Overall
n = 1	69.98%	78.25%	74.48%
n = 3	72.23%	81.98%	77.51%
n = 5	75.10%	86.45%	81.76%

Table 3.6: P@n for Various Values of 'n'

2. **SILPA Spell Checker** This subnormaliser uses SILPA libindic spell-checker [5] to compute the top 10 normalised words for a given input word. This spell checker is specifically designed to handle Indian languages and their spelling variations. We used a back transliterator to feed the unnormalised Devanagiri words to this spell checker.

   The accuracy of this module was 51.89% for English and 62.42% for Hindi.

So, the first step of the process is to transliterate a given word to Devanagiri script. The candidates then obtained from these two systems are ranked on the basis of the observed precision of the systems. Candidates generated from a system having higher precision are ranked above the candidates from the other system. The top-k candidates from each system are selected if they have a confidence score greater than an empirically observed Λ. A similar approach was used for English text normalisation, using the English normalisation pairs from Han et al. (2012) [26] and Liu et al. (2012) [34] for the noisy channel framework, and Aspell[6] as the spell-checker. Words with language tag 'rest' were left unprocessed.

3.3.2 System Accuracy

The accuracy of this system is explained by Precision@n for various n in Table 3.6. The accuracy for the Hindi normaliser is higher than that for English.

3.4 Conclusion

To recap this chapter, we have created a language identification and a word normalisation system for code-mixed Hindi-English data. In the next chapters, we describe how these systems were used. We created a shallow parsing pipeline, described in Chapter 4, using the same public dataset that was used in this work. The pipeline was created as a step to further understanding Hindi-English CMST data.

[5] https://github.com/libindic/spellchecker
[6] http://aspell.net/

While working on the public Hindi-English CMST dataset, we realised that there was a need for more datasets for this language pair, which were annotated with language and word normalisations. To solve this problem, we created and released a dataset, decribed in Chapter 5.

Chapter 4

Developing a Shallow Parsing Pipeline

To understand the usage of normalisation and test our normalisation module, we worked on a shallow parsing pipeline - where the pipeline is fed raw code-mixed sentences, and it returns shallow parsed sentences. Each of these sentences go through language identification, normalisation, POS tagging and ultimately, shallow parsing. We describe the pipeline procedure in the following sections.

4.1 Data Preparation and Annotation

The data used for this experiment was the same that has been described in Chapter 3. The annotation guidelines for Language Identification and Normalisation have also been specified in the previous chapter. Here are the annotation guidelines for the POS tagging process and Shallow Parsing process.

4.1.1 Part-of-Speech Tagging (POS)

The universal POS tagset (Petrov et al., 2011) [40] was used to label the POS of each word as this tagset is applicable to both English and Hindi words, and it contained a level of coarseness that suited our goals. The tags used were:

- **ADJ** : Adjective
- **ADP** : Adposition
- **ADV** : Adverb
- **AUX** : Auxiliary Verb
- **CONJ** : Conjunct

- **DET** : Determiner
- **INTJ** : Interjection
- **NOUN**
- **NUM** : Numeral
- **PART** : Particle
- **PRON** : Pronoun
- **PROPN** : Proper Noun
- **PUNCT** : Punctuation
- **SCONJ** : Subjugate Conjunct
- **VERB**

The following case-specific guidelines were also observed :

1. Sub-lexical code-mixed words were annotated based on their context, since POS is a function of a word in a given context.

2. Words embedded in a sentence of another language were tagged as per context of the matrix language, irrespective of the POS tag of the word in its original language. For example, an English verb used as a noun in Hindi context is labeled as a noun, as in the example sentence given below.

    - **Sentence** : Uska wait kar lo.
    - **Gloss** : Uska(for her) wait kar_lo (do)
    - **Translation** : Wait for her
    - **POS Tagging** : Uska (PRON) wait (NOUN) kar (VERB) lo (AUX).

    Here, the word “wait” is a verb in English, yet is labeled as a noun in the sentence.

4.1.2 Chunking

A chunk tag comprises of chunk label and chunk boundary. The chunk label tagset used is a coarser version of AnnCorra tagset (Bharati et al., 2006) [7]. Unlike AnnCorra, only one tag is used for all verb chunks in our tagset.

The tags used were:

- **NP** : Noun Phrase
- **VG** : Verb Phrase
- **CCP** : Conjunct Phrase
- **FRAGP** : Fragmented Phrase, used to connect interruptions in a phrase, usually seen in verb phrases for hanging auxiliary verbs.
- **BLK** : Used for punctuation.
- **RBP** : Adverb Phrase
- **JJP** : Adjective Phrase

Chunk boundary is marked using BI notation where 'B-' prefix indicates beginning of a chunk and 'I-' prefix indicates that the word is inside a chunk.

Punctuations are treated as a part of the preceding chunk. Opening quotes and opening braces are a part of the next chunk.

Similar to the previous chapter, the inter-annotator agreement was calculated using Cohen's κ [15] and came out to be 0.83 and 0.89 for POS tagging and shallow parsing respectively.

4.2 Shallow Parsing Pipeline

Shallow parsing is the task of identifying and segmenting text into syntactically correlated word groups (Abney et al., 1992 [1]; Harris et al., 1957 [27]). Shallow parsing is a viable alternative to full parsing as shown by Li et al. (2001) [32]. Our shallow parsing pipeline is composed of four main modules, as shown in Figure 4.1. These modules, in the order of their usage, are *Language Identification*, *Normalisation*, *POS Tagger* and *Shallow Parser*.

Our pipeline takes a raw utterance in Roman script as input on which each module runs sequentially. Twokenizer[1] (Owoputi et al., 2013) [39] which performs well on Hindi-English CSMT (Jamatia et al.,

[1] `http://www.ark.cs.cmu.edu/TweetNLP/`

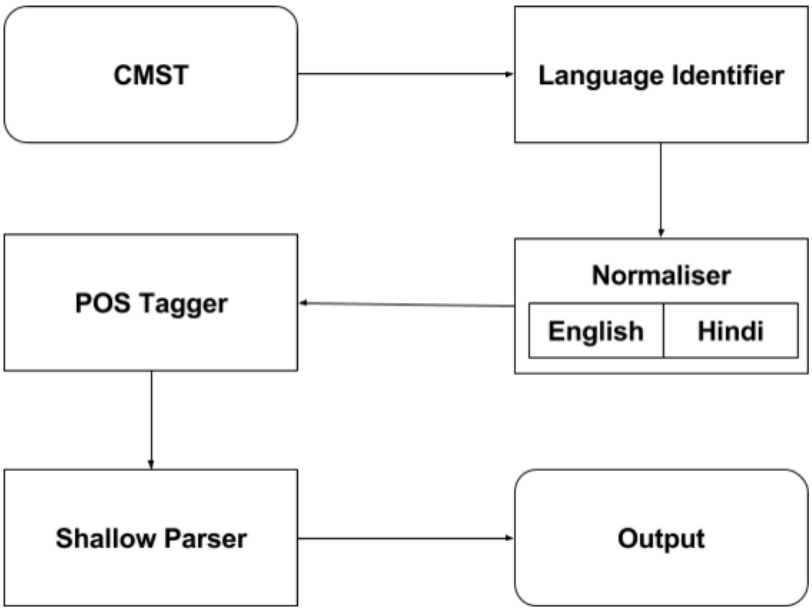

Figure 4.1: Schematic Diagram of the Complete Shallow Parsing Pipeline

2015) [30] was used to tokenise the utterance into words. The *Language Identification* module assigns each token a language label. Based on the language label assigned, the *Normaliser* runs the Hindi normaliser or the English normaliser. 'Rest' words are left untouched. The *POS tagger* uses the output of the normaliser to assign each word a POS tag. Finally, the *Shallow Parser* assigns a chunk label with boundary.

Now, we explain in detail how each system functions.

4.2.1 Language Identification and Normalisation Systems

The systems described in the previous chapter were used as the first two tasks of this pipeline. First, the raw data was fed to the Language Identifier and then, along with its automated language tag, sent to the Normalisation module. The words tagged 'Hindi' were sent to the Hindi normaliser and 'English' words were sent to the English normaliser. 'Rest' words were left as they were.

4.2.2 Part-Of-Speech Tagging System

The next step in the pipeline is understanding the Part-of-Speech (POS) tagging, which provides basic level of syntactic analysis for a given word or sentence. It was modeled as a sequence labeling task using CRF[31].

The feature set comprised of:

1. **Basic word features**: Word based features such as affixes, context and the word itself.
2. **LANG**: Language label of the token, obtained from the Language Identification system. This can have the values - 'en', 'hi' or 'rest'.
3. **NORM**: Lexical features extracted from the normalised form of the word. These include linguistic features such as bound and free morphemes, suffixes, prefixes.
4. **TPOS**: Output of Twitter POS tagger (Owoputi et al., 2013) [39] for the given word.
5. **HPOS**: Output of IIIT's Hindi POS tagger[2] for the given word.
6. **COMBINED**: HPOS for Hindi words and TPOS for English and Rest.

To obtain the Hindi POS tag output, the output from the normalisation module was used by transliterating Romanised Hindi words into WX-notations. Hindi POS tags were obtained using the IIIT Hindi POS tagger. This POS Tagger is trained on WX-notation, thus English and 'Rest' words were transliterated to WX-notation. These transliterations along with the Hindi normalised data was sent to the ILMT POS tagger and final POS tag was obtained.

4.2.2.1 System Accuracy

The feature ablation table for the POS Tagger are shown in Table 4.1. Each feature was added only if it showed a positive increase in the system accuracy.

4.2.3 Shallow Parsing System

To the best of our knowledge, this is the first system to focus on shallow parsing for not just code-mixed data, but social code-mixed data. A chunk comprises of two aspects - the chunk boundary and the chunk label. Shallow Parsing was modeled as three separate sequence labeling problems: **Label**, **Boundary** and **Combined**, for each of which a CRF model was trained.

The feature set comprised of :

[2] `http://ltrc.iiit.ac.in/showfile.php?filename=downloads/shallow_parser.php`

Features	Accuracy
Baseline	69.27
+LANG	70.44
+NORM	72.61
+TPOS	73.18
+HPOS, -TPOS	73.55
+COMBINED	75.07

Table 4.1: Feature Ablation for POS Tagger

Features	L	B	C
POS Tag	88.01	78.75	76.64
+POS Context [W5]	87.92	81.36	78.09
+POS_LEX	88.18	81.46	78.58
+NORMLEX	88.25	82.17	78.73

Table 4.2: Feature Ablation for Shallow Parser

1. **POS**: POS tag of the word, obtained from the POS tagging system.
2. **POS Context**: POS tags in the context window of length 5, i.e., the two previous tags, current tag and next two tags. We have used this feature to retain word-ordering information of the sentence.
3. **POS_LEX**: A special feature made up of concatenation of POS and LEX.
4. **NORMLEX**: The word in its normalised form, obtained from the Normalisation system.

4.2.3.1 System Accuracy

The feature ablation table for this module are shown in Table 4.2.

4.3 Complete Pipeline Results

The best performing model was selected from each module and was used in the pipeline. Table 4.3 tabulates the step by step accuracy of the pipeline calculated using 10 fold cross-validation.

The pipeline developed has been made available to the research community with the goal of enabling better text analysis of Hindi English CSMT. The pipeline is accessible online [3].

[3] `http://bit.ly/csmt-parser-api`

		Actual Pipeline	Gold Pipeline	Error Propagation
Language Identification		93.98	93.98	NA
Normalisation		70.32	74.48	4.16
POS Tagging		68.25	75.07	6.82
Shallow Parsing	Label	75.73	88.25	12.52
	Boundary	74.96	82.17	7.21
	Combined	61.95	78.73	16.78

Table 4.3: Pipeline Accuracy and Error Propagation

4.4 Conclusion

We have developed a system for Hindi-English CSMT data that can identify the language of the words, normalise them to their standard forms, assign them their POS tag and segment them into chunks. We released the system in January, 2016.

This system was an experiment in exploring the uses of our normalisation system, and a step towards understanding Hindi-English code-mixed social media text.

Chapter 5

Dataset Creation

After performing experiments on the publicly available dataset, we felt that there was a lack of annotated Hindi-English CMST datasets. In order to aid further research for this language pair, we created and released a dataset of 1446 Hindi-English code-mixed sentences, collected from social media.

In this chapter, we explain the dataset creation process. We have used our language identification system to annotate the data, explained in Chapter 3. We analysed error patterns in our normalisation module, and made improvements to the dictionaries or word lists used; this improved normalisation module was then used to annotate the created dataset. We provide an explanantion for the improvements made in the following subsections. The final normalised words are written in the original language scripts, i.e. Roman script for English words and Devanagiri script for Hindi words.

5.1 Data Collection

Taking into account the claim that code mixing is frequent among speakers who are multilingual and younger in age (Cardenas-Claros et al., 2009) [9], we choose an Indian student community between the 20-30 year age group as our data source.

In order to create the dataset, 12 multilingual speakers who were fluent in Hindi and English were selected. They were divided into 4 Facebook chat groups and asked to converse on topics related to daily life. Due to the usage of Facebook as the underlying crowd sourcing engine, the data generated was highly conversational and had reasonable amount of social-media lingo. From their interactions, 1446 code-mixed sentences were selected. The names of the participants are anonymised.

5.2 Data Statistics

The size of the original chat data was 60132 sentences. After removing sentences with less than 5 words, 45589 sentences were left. Of this, 1446 (3.1%) are code-mixed, containing a total of 14519 tokens. Of these tokens, 8048 (55.43%) are Hindi words, 5263 (36.24%) are English words and 1208 (8.32%) are marked as 'Rest'.

Of the 5263 English tokens, 4031 (76.59%) were already in their standard form. Of the 8048 Hindi tokens which are in Roman script, 4400 (54.67%) tokens would have been in their standard Devanagiri spelling using a transliteration system [1].

We computed the Code-Mixing Index (CMI) (Das and Gamback, 2014) [19] for our dataset using the formula given in Equation 3.1. The value was computed as an average of the CMIs for every sentence, and found to be 27.53.

5.3 Improvements to our Normalisation system

Here are the key points of difference:

1. In the earlier system, we have used the frequencies of the Leipzig Corpora (Biemann et al., 2007) [8] to create a Hindi dictionary of 1,17,789 words. The purpose of this dictionary is to create a large mapping of normalised-unnormalised pairs, enabling a CRF to learn the transformations required for normalisation. On observing the low-frequency words in the dictionary, we found a lot of misspelled words, and words from other languages merely transliterated in Hindi. These were pruned to create a more accurate Hindi dictionary, totaling to 95,000 words. Similar changes were also made to the SILPA Spell Checker dictionary for Hindi used in the earlier system.

2. Our previous system used Hindi-English transliteration pairs from Gupta et al. (2012) [24], collected from matching parallel Hindi song lyrics from different websites to learn the transformations from a normalised word to an unnormalised word through a CRF classifier. On analysing these transliteration pairs, a number of transliteration pairs were found to be bad pairings for our purpose, and were pruned. Several other pairings were added as well, to help the CRF learn these transformations better.

3. For the English normaliser, our earlier system performed the exact same approach as the Hindi normaliser, and the accuracy for the English normaliser was observed to be much lower than the Hindi normaliser. A closer look suggested that one major reason behind the low English accuracy

[1] `https://github.com/irshadbhat/python-irtrans`

is the nature of the shortening/transformations we see from unnormalised to normalised words in the two languages. While in Hindi, the transformations are simpler (such as vowel drop, similar sounding syllable contractions), English exhibits extreme contractions (such as 'brb' = 'be right back', 'gm' = 'good morning'). These English transformations are hard for a CRF to learn. To help with this, we created a list of 100 commonly used English abbreviations (such as 'brb' = 'be right back'). This list helps us post-process the normalisation results to derive better results.

5.4 Pipeline Accuracy

5.4.1 Language Identification Accuracy

The accuracy of this module was computed by the following equation.

$$Accuracy = \frac{n(CorrectIdentifications)}{n(TotalTokens)} \tag{5.1}$$

where $n(X)$ represents the count of X, and was observed to be 93.88%.

5.4.2 Normalisation Accuracy

The accuracy for Hindi and English normalisation has been computed only over the words that were incorrectly spelled in the dataset. Therefore, of the 1232 incorrectly spelled words (23.41%), the English normaliser was able to correct 750 words (60.85%), followed by a further 4.05% accuracy gain in post-processing, giving an overall accuracy of 91.80% across all English words in the dataset. For Hindi, out of the 3648 incorrectly written words (44.57%), the normaliser corrected 2743 words (75.19%), resulting in an overall accuracy of 88.75% across all Hindi words in our dataset.

Thus, our normalisation module resulted in 11924 words correctly standardised, out of 13311 total words of Hindi and English in the dataset. The overall module accuracy is 89.94%. The earlier system gave an overall accuracy of 83.54% for our dataset, thus our changes yielded an accuracy gain of 6.45%.

5.5 Manual Annotation Correction

Two annotators manually checked the results of the Language Identifier and Normaliser and marked their predictions for each word of the 1446 sentences. Each annotator was given the 1446 sentences in a randomised order.

For Language Identification, inter-annotator agreement calculated using Cohen's κ was found out to be 0.98. The inter-annotator agreement in case of Normalisation was found to be 0.87. This is attributed to the presence of ambiguities in the case of Normalisation due to the lack of context in the sentences. These are further explained in Section 5.6.2.1.

5.6 Error Analysis

During the course of our study, a few observations stood out that highlight the errors made by our automated annotation pipeline, and suggest the importance of manual annotation in dataset creation, especially for code-mixed social media text. We discuss these error patterns, and develop an understanding of the cause of these errors in this section.

5.6.1 Language Identification Process

5.6.1.1 Auto-correction

A vast portion of social media text is created by users accessing these platforms from mobile devices. As a side-effect of using Facebook messenger for this study, our dataset exhibits the characteristic properties of text content created from mobile devices. We observed that the auto-correct on mobile devices led to a Hindi word being auto-corrected to a similarly spelled English word. The following is an example from our dataset.

- **Sentence**: So I thought maybe tu yard kar rahi hogi

  Gloss: So I thought maybe you(tu) remembering(yaad_kar_rahi) were(hogi).

  Translation: So I thought maybe you were remembering me.

In the above example, the correct Hindi word for 'remember' is 'yaad', but it has been auto-corrected to 'yard', which is an English word for a unit of distance. This word was identified as an English word by the Language Identifier.

5.6.2 Normalisation Process

5.6.2.1 Ambiguity in Hindi normalisation

For certain characters pairs in Hindi and English, there is a one-to-many mapping from English to Hindi. One such example is 'd', which may map to five Hindi characters, as shown previously in

haaan hi हाँ
hum hi हम
sab hi सब
tmhare hi तुम्हारे
wing en wing
mein hi में
padne hi [पड़ने or पढ़ने]
wale hi वाले
hain hi हैं

Figure 5.1: Example of Ambiguity in Hindi Normalisation

Table 1.1. In a sentence with insufficient context, this mapping creates ambiguity even for a human. In the example sentence annotation presented in Figure 5.1, the annotators were unable to decide the normalisation for the word 'padne', because the context of the sentence was not enough to disambiguate the two given possibilities. In the figure, the 3 columns represent original word, language ID and normalised word, respectively from left to right. Finally, the ambiguity was resolved to the former of the two given possibilities after using discourse context. This indicates that we need to go beyond sentence-level context for word disambiguation.

5.7 Conclusion

To summarise this chapter, we have released a dataset which contains code-mixed Hindi-English socila media text. It consists of 1446 sentences, and each word is annotated with its language and standardised form.

We also discussed some issues in language identification and word normalisation, and compared the effectiveness of noisy channel techniques over English and Hindi normalisation.

Chapter 6

Conclusion

In this thesis, we have focussed on creating tools for enabling further research on Hindi-English code mixed social media text. We have attempted to build language identification and normalisation systems for this language pair, which we hope would result in better data-mining and sentiment analysis across the Indian subcontinent.

The language identification and normalisation systems follow supervised machine learning and report final accuracies of 93.88% and 83.54% for our dataset, respectively.

We have also developed a complete shallow parsing pipeline, which consists of a POS tagging system and a shallow parsing system, in addition to the language identification and normalisation systems. To the best of our knowledge, this system is the first of its kind. We have released this system online and also provided a public API to access it.

A dataset of 1446 code-mixed Hindi-English sentences has also been released, to further facilitate research in this direction. This dataset has been annotated by our language identification and normalisation systems. The final errors in the dataset due to the innacuracy of the systems were manually corrected.

Chapter 7

Future Work

During the course of this study, we realised that there are numerous avenues where research can be pursued, picking up from where this study ends. This work lays the ground-level work for complete analysis of natural language by a machine. Using this work, one can build upon various tasks such as sentiment analysis, language translations, question answering systems, discourse analysis, and so on!

Some of the scope of future directions for this work is as follows:

- **Building more tools**

  Our work on code-mixed data has just begun, and there are many directions to move forward in, with respect to building new tools for CMST. A positive direction to explore would be creating a full parser, which is the next step from our shallow parser. This would allow processing of code-mixed sentences, opening up opportunities to work on anaphora resolution, discourse analysis and many more problems on Hindi-English code mixed data.

- **Building larger datasets** A lot of our research was restricted due to the absense of a large annotated dataset for Hindi-English. We hope to create bigger datasets in the future, continuously improving our annotation models to make the process seamless.

- **Taking context for normalisation**

  Currently, our normaliser standardisess each word to a single word, irrespective of the context the word is used in. We believe that adding context would definitely increase our accuracy. We hope that by provide enough data to our machine learning model, we would enable it to understand contextual information.

- **Adding more language classes**

As of now, we use three language classes - Hindi, English and Rest. 'Rest' consists of any named entities, acronyms, sub-lexical code-mixed words, foreign language words, symbols and emoticons. We wish to analyse each of these by creating further subclasses within 'Rest', especially for acronyms and named entities. This will allow us to leverage new information for sentiment analysis, personalisation, named entity recognition, and so on.

- **Work on more languages**

  There is little work that has been done for various Indian languages. Tools need to be built for regional language pairs as well, such as Bengali-Hindi-English, Gujarati-Hindi-English, and many more.

Related Publications

- Arnav Sharma, Sakshi Gupta, Raveesh Motlani, Piyush Bansal, Manish Shrivastava, Radhika Mamidi, Dipti M. Sharma : Shallow Parsing for Hindi-English Code-Mixed Social Media Text. In Proc. of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2016), San Diego, USA.
- Sakshi Gupta, Piyush Bansal, Radhika Mamidi : Resource Creation of Hindi-English Code Mixed Social Text. In Proc. of the 4th International Workshop on Natural Language Processing for Social Media (SocialNLP), in conjunction with IJCAI 2016 @ July 11, 2016, New York City, USA.

Bibliography

[1] S. P. Abney. *Parsing by chunks*. Springer, 1992.

[2] G. Aston and L. Burnard. *The BNC handbook: exploring the British National Corpus with SARA*. Capstone, 1998.

[3] S. Azuma. The frame-content hypothesis in speech production: Evidence from intrasentential code switching. *Linguistics*, 31(6):1071–1094, 1993.

[4] K. Bali, Y. Vyas, J. Sharma, and M. Choudhury. "am i borrowing ya mixing?": an analysis of english-hindi code mixing in facebook. *Proceedings of the First Workshop on Computational Approaches to Code Switching, EMNLP*, 2014.

[5] U. Barman, A. Das, J. Wagner, and J. Foster. Code mixing: A challenge for language identification in the language of social media. *EMNLP 2014*, page 13, 2014.

[6] R. Beaufort, S. Roekhaut, L.-A. Cougnon, and C. Fairon. A hybrid rule/model-based finite-state framework for normalizing sms messages. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 770–779. Association for Computational Linguistics, 2010.

[7] A. Bharati, R. Sangal, D. M. Sharma, and L. Bai. Anncorra: Annotating corpora guidelines for pos and chunk annotation for indian languages. *LTRC-TR31*, 2006.

[8] C. Biemann, G. Heyer, U. Quasthoff, and M. Richter. The leipzig corpora collection-monolingual corpora of standard size. *Proceedings of Corpus Linguistic*, 2007.

[9] M. S. Cárdenas-Claros and N. Isharyanti. Code-switching and code-mixing in internet chatting: Between 'yes','ya', and 'si' - a case study. *The Jalt Call Journal*, 5(3):67–78, 2009.

[10] S. Carter, W. Weerkamp, and M. Tsagkias. Microblog language identification: Overcoming the limitations of short, unedited and idiomatic text. *Language Resources and Evaluation*, 47(1):195–215, 2013.

[11] W. B. Cavnar, J. M. Trenkle, et al. N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175, 1994.

[12] M. Choudhury, K. Bali, T. Dasgupta, and A. Basu. Resource creation for training and testing of transliteration systems for indian languages. *LREC*, 2010.

[13] M. Choudhury, G. Chittaranjan, P. Gupta, and A. Das. Overview of fire 2014 track on transliterated search, 2014.

[14] M. Choudhury, R. Saraf, V. Jain, A. Mukherjee, S. Sarkar, and A. Basu. Investigation and modeling of the structure of texting language. *International Journal of Document Analysis and Recognition (IJDAR)*, 10(3-4):157–174, 2007.

[15] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 134:37–46, 1960.

[16] P. Cook and S. Stevenson. An unsupervised model for text message normalization. In *Proceedings of the workshop on computational approaches to linguistic creativity*, pages 71–78. Association for Computational Linguistics, 2009.

[17] D. Crystal. *Language and the Internet*. Cambridge University Press, 2001.

[18] B. Danet and S. C. Herring. Multilingualism on the internet. *Language and communication: Diversity and change. Handbook of applied linguistics*, 9:553–592, 2007.

[19] A. Das and B. Gambäck. Identifying languages at the word level in code-mixed indian social media text. 2014.

[20] A. Dey and P. Fung. A hindi-english code-switching corpus. In *LREC*, pages 2410–2413, 2014.

[21] S. Gella, J. Sharma, and K. Bali. Query word labeling and back transliteration for indian languages: Shared task system description. *FIRE Working Notes*, 3, 2013.

[22] S. A. Golder and M. W. Macy. Diurnal and seasonal mood vary with work, sleep, and daylength across diverse cultures. *Science*, 333(6051):1878–1881, 2011.

[23] J. J. Gumperz. *Discourse Strategies*. Oxford University Press, 1982.

[24] K. Gupta, M. Choudhury, and K. Bali. Mining hindi-english transliteration pairs from online hindi lyrics. In *LREC*, pages 2459–2465, 2012.

[25] B. Han and T. Baldwin. Lexical normalisation of short text messages: Makn sens a# twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 368–378. Association for Computational Linguistics, 2011.

[26] B. Han, P. Cook, and T. Baldwin. Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 421–432. Association for Computational Linguistics, 2012.

[27] Z. S. Harris. Co-occurrence and transformation in linguistic structure. *Language*, pages 283–340, 1957.

[28] S. C. Herring, S. Barab, R. Kling, and J. Gray. An approach to researching online behavior. *Designing for virtual communities in the service of learning*, 338, 2004.

[29] T. Hidayat. An analysis of code switching used by facebookers.

[30] A. Jamatia, B. Gambäck, and A. Das. Part-of-speech tagging for code-mixed english-hindi twitter and facebook chat messages. *Proceedings of Recent Advances in Natural Language Processing*, page 239, 2015.

[31] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the eighteenth international conference on machine learning, ICML*, volume 1, pages 282–289, 2001.

[32] X. Li and D. Roth. Exploring evidence for shallow parsing. In *Proceedings of the 2001 workshop on Computational Natural Language Learning-Volume 7*, page 6. Association for Computational Linguistics, 2001.

[33] J. Lipski. Code-switching and the problem of bilingual competence. *Aspects of bilingualism*, 250:264, 1978.

[34] F. Liu, F. Weng, and X. Jiang. A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 1035–1044. Association for Computational Linguistics, 2012.

[35] P. McNamee. Language identification: a solved problem suitable for undergraduate instruction. *Journal of Computing Sciences in Colleges*, 20(3):94–101, 2005.

[36] C. Myers-Scotton. *Duelling languages: Grammatical structure in codeswitching*. Oxford University Press, 1997.

[37] D.-P. Nguyen and A. S. Dogruoz. Word level language identification in online multilingual communication. Association for Computational Linguistics, 2013.

[38] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

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

[40] S. Petrov, D. Das, and R. McDonald. A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*, 2011.

[41] J. M. Prager. Linguini: Language identification for multilingual documents. *Journal of Management Information Systems*, 16(3):71–101, 1999.

[42] A. K. Singh and J. Gorla. Identification of languages and encodings in a multilingual document. In *Building and Exploring Web Corpora (WAC3-2007): Proceedings of the 3rd Web as Corpus Workshop, Incorporating Cleaneval*, volume 4, page 95. Presses univ. de Louvain, 2007.

[43] S. Tratz, D. Briesch, J. Laoudi, and C. Voss. Tweet conversation annotation tool with a focus on an arabic dialect, moroccan darija. *LAW VII & ID*, 135, 2013.

[44] V. N. Vapnik and V. Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

[45] Y. Vyas, S. Gella, J. Sharma, K. Bali, and M. Choudhury. Pos tagging of english-hindi code-mixed social media content. In *EMNLP*, volume 14, pages 974–979, 2014.

[46] M. Wang, C. Yang, and C. Cheng. The contributions of phonology, orthography, and morphology in chinese–english biliteracy acquisition. *Applied Psycholinguistics*, 30(02):291–314, 2009.

[47] M. Warschauer, G. R. E. Said, and A. G. Zohry. Language choice online: Globalization and identity in egypt. *Journal of Computer-Mediated Communication*, 7(4):0–0, 2002.

[48] K.-F. Wong and Y. Xia. Normalization of chinese chat language. *Language Resources and Evaluation*, 42(2):219–242, 2008.

[49] Z. Xue, D. Yin, B. D. Davison, and B. Davison. Normalizing microtext. *Analyzing Microtext*, 11:05, 2011.

[50] H. Yamaguchi and K. Tanaka-Ishii. Text segmentation by language using minimum description length. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 969–978. Association for Computational Linguistics, 2012.