Better Understanding of Code-Mixed Social Media Data via Information Extraction

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

*Master of Science in **Computer Science and Engineering** by Research*

by

Sumukh S
20171404
sumukh.s@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
October 2023

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Better Understanding of Code-Mixed Social Media Data via Information Extraction" by Sumukh S, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Adviser: Prof. Manish Shrivastava

To the pursuit of knowledge and embracing the spirit of curiosity.

Acknowledgments

Since I joined IIIT-Hyderabad, I can think of so many people I am grateful for. It has been an eventful and fulfilling journey, and I am thankful for all those who have influenced me positively in some way or another, even if they are not currently a part of my life.

I want to express my gratitude towards my family, who have been immensely supportive at every step of the process, especially Amma, with their love and encouragement.

I want to thank my advisor, Professor Manish Shrivastava. Starting from my admission interview and then as my research advisor, I have been lucky to have him as a pillar of support throughout. I was given the freedom to explore the field of NLP without any constraints, allowing me to explore multiple domains and commit mistakes I could learn from. I am truly grateful for his patience through this journey and for being someone who I could talk to about anything. He taught me to look at long-term gains over instant gratifications while not succumbing to short-term struggles, which helped me get through the lows of the process.

I am thankful to have worked with Suhan, Ujwal, and Alok, who introduced me to the field of event identification in NLP. I would like to thank Prashant Kodali for his detailed feedback on my work. I want to thank Abhinav Appidi for his help with code-mixed Kannada-English data.

Apart from my advisor, research peers and family, there are numerous people who have played important roles in this journey. Jayitha, thank you for being an amazing friend from the first day of college who has helped me grow as a person. I have immense respect for you for maintaining a high standard of friendship throughout! Mohsin and Nikhil, for the thousands of conversations we have had in the corridors, mess, trips, and during lockdown. You guys have helped me in so many ways all these

years. Pathak, thank you for your overwhelming support, especially in the last couple of years. Also, your work ethic always motivates me to do better. Vighnesh, for being such a warm and welcoming guy all the time. The entire group from OBH 3rd floor, E Block - I always felt lucky to be a part of this group. The environment that we had where help and support were always there while celebrating every small win. In no particular order - Gulati, Chand, Sahu, Vashist, Prusty, Joshi, Prathyakshun, Kunal, Bakshi, Harish, Aditya, Shyam, Unni and Subrahamanyam. Also - Patil, Aaron, Amal, Trunapushpa, Roopal, Ganesh, Kritika, Hiranmai, Ramkishore, and many more people. Thank you all! Last but most importantly, I would like to thank Samiksha. In the last few years, you have been nothing but supportive and helpful in this journey. Thank you for helping me work through the problems, especially when things looked bleak and for all those conversations on philosophy, ethics, and life.

Abstract

Code-switching or code-mixing occurs when ”lexical items and/or grammatical features from two languages appear in one sentence”. With the rising popularity of social media platforms such as Twitter, Facebook, and Reddit, the volume of texts on these platforms has also grown significantly. Twitter alone has over 500 million text posts (tweets) per day. India, a country with over 300 million multilingual speakers, has over 23 million users on Twitter as of January 2022, and code-switching can be observed heavily on this social media platform.

Code-mixed social media posts present unique challenges due to the mixing of languages, slang, and informal expressions. Extracting valuable information from code-mixed data enables a better understanding of user sentiments, preferences, and trends across different language communities. It helps identify named entities, detect events, and extract relevant information from multilingual conversations. By effectively extracting and analyzing code-mixed content, businesses, researchers, and language processing systems can gain valuable insights into diverse language patterns, linguistic phenomena, and cultural nuances, contributing to more accurate language processing, sentiment analysis, and targeted communication strategies. Extracting valuable information from social media content has numerous benefits that impact various domains. This has led to a growing interest in Information Extraction (IE) as an active research area in artificial intelligence.

However, research efforts are often hindered by the lack of automated NLP tools to analyse massive amounts of code-mixed data, especially for resource-poor languages. There have been significant efforts towards understanding some languages, such as English, German, French, and Spanish, by creating annotated resources for various tasks, while some languages have not received the same focus, making

them resource-poor. One such language is Kannada, which has over 58 million native and secondary-language (L2) speakers worldwide. Like in most places around the world, Kannada speakers tend to use code-mixed language, mixed with English, in informal settings including social media and texting platforms, but the same has not received much focus from researchers. In this thesis, we work towards information extraction from Kannada-English code-mixed data from social media. We work on the problems of Named Entity Recognition (NER) and Event Detection.

Named Entity Recognition (NER) and Event Detection on social media code-mixed data are crucial for understanding and analyzing user-generated content in diverse languages. They enable the identification of named entities, such as names of people, organizations, and locations, as well as the detection of events, allowing for deeper insights into multilingual conversations, cross-cultural trends, and effective information retrieval from code-mixed social media posts.

We have collected Kanglish code-mixed data from social media, Twitter, and curated annotated datasets for both NER and Event Detection tasks while mentioning the annotation guidelines for the same. We have analysed the challenges that are unique to Kannada-English code-mixed data and have provided annotation guidelines for the same. We have also proposed a few supervised approaches towards these tasks on our dataset with careful feature selection and critically analysed the results in hopes to promote more focus from the research community on such low-resource languages.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Globally, the usage of social media and the internet has skyrocketed in recent years, enabling individuals to engage with one another and exchange ideas, thoughts, and opinions, among other things. Every day, social media platforms such as Twitter[1], Facebook[2] and Reddit[3] facilitate the extremely rapid and widespread distribution of large amounts of data. Twitter alone has over 6,000 text posts (tweets) every second, as of 2022[4]. In mathematical terms, this works out to 350 thousand tweets each minute, 500 million per day, and 200 billion tweets annually[5]. India, a country with over 300 million multilingual speakers, has over 23 million users on Twitter as of January 2022[6], and code-switching can be observed heavily on this social media platform [64].

1.1 Code Mixing

Code-switching or code-mixing occurs when lexical items and/or grammatical features from two languages appear in one sentence [51]. The terms ”code-mixing” and ”code-switching” are used interchangeably by many researchers, and we also use these terms interchangeably. Multilingual society

[1]http://www.twitter.com/
[2]https://www.facebook.com/
[3]https://www.reddit.com/
[4]https://www.internetlivestats.com/twitter-statistics/
[5]https://www.demandsage.com/twitter-statistics/
[6]https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/

speakers often tend to switch back and forth between languages when speaking or writing, mostly in informal settings.

India has the second largest English-speaking population, second only to the United States of America, and most people in India are multi-lingual. Like many other non-English speakers throughout the world, Indians tend to avoid utilising a single code in informal settings. Code-mixing of one's native language and English is the most common in India, especially on social media, where sentences usually follow the syntax of the native language but borrow some vocabulary from English.

In recent years, the internet, particularly social media sites, has played an important role in this spread. To look more edgy, radio hosts and TV hosts purposely mix English terms with a slew of local tongue expressions. CM can happen for a multitude of causes. It depends on who is speaking and what the speaker's goal is. In the multilingual Indian setting, code-mixing is highly natural and successful. It is most commonly used to combine English and local language codes.

An example of code-mixed Kannada-English sentence is presented below:

> **Ka-En CM:** Rajesh/*NE* next/*En* year/*En* Arunachal/*NE* visit/*En* madtane/*Ka* #excited/*En* :D/*Univ*
> **Translation:** *Rajesh will visit Arunachal next year #excited :D*

Please note that *NE* refers to a named entity token, *Univ* refers to a universal token, *En* refers to an English language token and *Ka* refers to a Kannada language token.

1.1.1 Types of Code-Mixing

According to a study by Zirker[93], there are mainly two types of code-mixing that happens - intrasentential and intersentential.

- **Intrasentential code-mixing**: Intrasentential code mixing refers to the phenomenon where multiple languages are mixed within a single sentence. It occurs when a speaker incorporates words, phrases,

or even grammatical structures from different languages in a seamless manner, without switching languages between sentences or clauses. An example of this would be the following-

> **Ka-En CM:** Saanu/*NE* next/*En* month/*En* Gujarat/*NE* visit/*En* madtale/*Ka* #excited/*En* :D/*Univ*
> **Translation:** *Saanu will visit Gujarat next month #excited :D*

We will mostly deal with this type of code-mixing in this thesis as we are working with Kannada-English code-mixed data where much of code-mixing happens at word level. We shall discuss more on this in Section 1.3.

- **Intersentential code-mixing**: Inter-sentential code-mixing, also known as code-switching, refers to the phenomenon where speakers alternate between different languages at sentence or clause boundaries. It involves using one language for one sentence or clause and then switching to another language for the next sentence or clause. The following is an example of this-

  > **Ka-En CM:** I don't know what to do this weekend. Manege hoogi aaram madtini.
  > **Translation:** I don't know what to do this weekend. I'll go home and relax.

1.2 Motivation

Code mixing is intriguing for a number of reasons, including the fact that it occurs naturally and frequently in multilingual communities, the difficulties it poses for Natural Language Processing (NLP) tasks, the potential insights it may offer into language interaction and processing, the practical ramifications it may have in a variety of fields, and its relationship with emotional expression [67] and identity.

In many regions of the world, it is ubiquitous to speak two or more languages fluently, and switching between them on a regular basis is common. At several linguistic levels, such as within specific words, phrases, and sentences, code-mixing can take place. It may represent a speaker's identity, their degree of

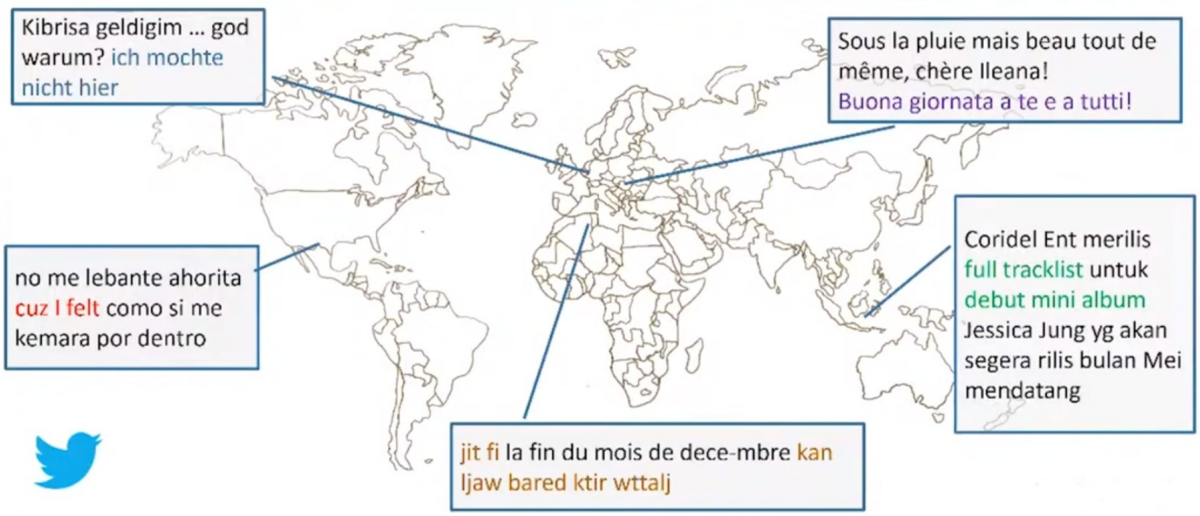

Figure 1.1: Code mixing around the globe, as seen on Twitter
(source: Microsoft Research slides [64])

language skill, and the social setting in which they are talking, as well as social and cultural components of communication.

Furthermore, the study of code mixing can reveal how various languages interact and impact one another. Code mixing can result in the production of new linguistic structures, such as hybrid words and syntactic patterns, which reflect the fluidity of language usage. Understanding code mixing can also help us understand how the brain processes language and how bilingualism influences cognitive functions like attention and memory.

Bilingual speakers, for example, may have higher cognitive control and attentional flexibility than monolingual speakers, according to study [54]. Bilingualism may also impact how non-linguistic inputs, such as music and visual stimuli, are processed. Code mixing research can assist researchers in better understanding these cognitive processes and their consequences for language usage and processing.

To address these challenges, researchers have created a variety of tools and methodologies for analysing code-mixed data, including rule-based, statistical, and machine learning-based approaches. These techniques entail creating models that can recognise and categorise language-specific elements such as grammar, syntax, and vocabulary inside code-mixed data. These tools and approaches have the

potential to accelerate the development of NLP systems that can handle several languages and increase communication across linguistic communities.

1.2.1 Motivation to study Kannada-English code-mixed data

In the recent past, we have seen interest in this field by researchers across different languages in working towards a better understanding of code-mixed data. This can be seen even for a resource-poor Dravidian language like Kannada where researchers have worked on creating and benchmarking Kannada-English code-mixed social media data for some of the basic NLP tasks like language identification (LID)[77], part-of-speech (POS) tagging[3], and emotional analysis[4]. Though these are small datasets compared to the humongous corpora that are present for monolingual languages across different tasks, these have paved the way for some of the downstream tasks in NLP.

Kannada is a Dravidian language spoken majorly in the Indian state of Karnataka (shown in image 1.2) with over 58 million native and second-language (L2) speakers worldwide[7]. Kannada is also one of the six languages designated as a classical language of India by the Indian Government. In the era of ChatGPT[8], GPT-4[9] and other large-scale models that produce human equivalent results which consume hundreds of gigabytes/petabytes of data for training, Kannada is far behind in terms of monolingual data, and code-mixed Kannada-English is further behind.

With these numbers, we can see why it is imperative to have a better understanding of Kannada-English code-mixed data, which is used the most on a daily basis in almost all informal settings between Kannada speakers.

[7] https://www.ethnologue.com/language/kan/
[8] https://openai.com/blog/chatgpt
[9] https://openai.com/gpt-4

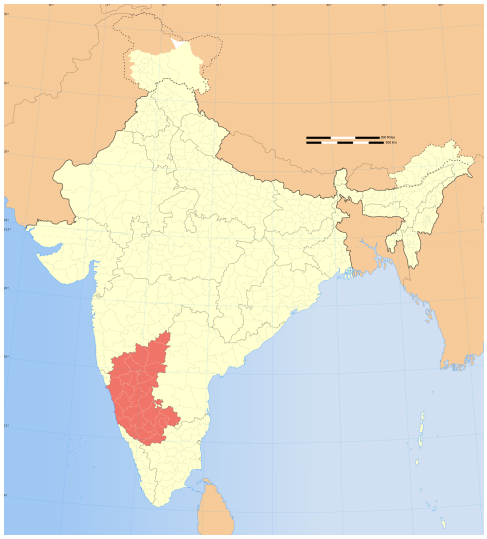

Figure 1.2: Karnataka, the state in India where most Kannada language speakers reside (image source:Wikipedia)

1.3 Challenges

For natural language processing (NLP) tasks including named entity recognition, sentiment analysis, and machine translation, code-mixing offers considerable challenges. Due to the difficulties inherent in handling multiple languages, code-mixed data can pose challenges for NLP tasks. The following are some of them-

- **Limited NLP tools:** It is incredibly difficult to develop large-scale, reliable, and efficient systems due to the lack of text processing tools designed specifically for code-mixed languages. Tools for even the fundamental NLP tasks like language identification, POS tagging, NER, and transliteration are yet to be developed at a large scale for code-mixed data[9], especially for resource-poor Dravidian languages. Building solutions for multilingual societies is a vicious loop due to the lack of high-quality datasets for these jobs.

- **Filtering code-mixed data:** The coexistence of other languages presents a significant obstacle when working with code-mixed data. It is rare to come across a platform where people exclusively communicate in code-mixed languages. Additionally, the lack of readily available tools for automatic filtering poses a considerable challenge in creating a substantial corpus of code-mixed

data. This process is usually done manually where humans help in identifying and filtering code-mixed data[42][89], making this is a very expensive and time-consuming process.

- **Lack of grammatical standardization:** The difficulty of spelling variance, dialect, and readability arises from the code-mixed languages' lack of grammatical standardisation. It is difficult to enforce standardisation due to the code-mixed languages' unofficial status. It is difficult to use human annotators for annotation work due to the lack of standards. While intrinsic human bias is one of the biggest and most hazardous obstacles for the majority of NLP systems, this contributes to the bias among human annotators. Data that has been code-mixed frequently combines several languages, resulting in differences in vocabulary and syntax. Furthermore, NLP models may have trouble with code-mixed vocabulary, which might include loanwords, borrowed phrases, or language-specific expressions, if they were trained on monolingual data. Different languages may also have different grammatical structures and syntax, which makes it challenging for NLP models to interpret and comprehend the code-mixed text.

- **Variance and similarity between the languages:** For instance, named entity recognition identifies individuals, locations, and organisations within text data. Named entities might exist in multiple languages in code-mixed text, making it challenging for NLP models to correctly recognise and categorise them. Similarly, sentiment analysis, which includes figuring out the text's emotional tone, can be difficult with code-mixed data since different languages have various ways of expressing emotion. For example, the word "*Bhaarath*", one of the names for the country India, is a *location* named entity while the words "*Bharat*" and "*Bhaarti*" are common first names in India which are tagged as *person* named entity.

1.3.1 Challenges specific to Social Media Data

Adding to the above, we should also note that we are working with social media data here which brings its own set of challenges.

- **Social media slang/lingo words:** As this is an informal setting, people tend to get creative with their spellings on social media. For example, a word like *tonight* could be written as *tonight, tonite, tonihgt, ton8, etc.,* which posed a significant challenge while building spelling agnostic models.
- **Spelling mistakes:** Code-mixed data on social media platforms frequently involves spelling variations and errors due to the informal nature of communication. These variations can add to the noisiness of the data.
- **Different forms with Roman characters:** While transliteration words from one language to the other in code-mixing, users tend to use many forms of a word for a single word on social media, and also use colloquial words/slang on social media and have their own preference of native words. For example, *baralilla* is a Kannada word and it can be written with Roman characters as *brlilla*, *barlilla*, etc.
- **Domain-specific challenges:** Different social media platforms and online communities have their own linguistic norms, jargon, and specific language usage patterns. Adapting NLP models to different social media platforms and capturing the domain-specific characteristics of code-mixed data can be demanding, as the models need to be trained on representative and domain-specific data.

1.4 Named Entity Recognition

Named Entity Recognition (NER) is a subtask of natural language processing (NLP) that involves identifying and classifying named entities in text. Named entities are specific words or phrases that refer to individuals, organizations, locations, dates, quantities, and other predefined categories.

Code-mixed social media data is entirely unstructured data. Unstructured data lacks a consistent schema or well-defined patterns, making it challenging to analyze and utilize effectively. Processing unstructured data is important because it unlocks valuable insights, enhances information retrieval, facilitates regulatory compliance, enables automation and efficiency, and enables personalization. Analysing such social media data also helps in identifying user patterns, understanding public opinions, trend

identification, market research and more. In this process, identifying named entities is very important. Information extraction without NER may lead to the extraction of unstructured information from text without specifically identifying and classifying named entities. This means that the extracted information may lack entity-level granularity and context. Without NER, the extracted information may not distinguish between different types of entities (e.g., people, organizations, locations), making it challenging to analyze or utilize the extracted data in a structured manner. Consider the example *"Tim Cook announced that Apple Inc. is going to open a service centre in Ediburgh soon."* In NER, the system would analyze this sentence and identify the named entities as follows - *Tim Cook* as person (PER), *Apple Inc.* as an organisation (ORG), and *London* as location (LOC).

The following sentence 1.4 is an example of a code-mixed Kanglish sentence that has also been translated to English. Named entities have been tagged along with the language tags (*Ka-Kannada, En-English, NE-Named Entity, Univ-Universal*).

> **Ka-En T1:** Saanu/*Person/NE* next/*Other/En* month/*Other/En* Gujarat/*Location/NE* visit/*Other/En* madtale/*Other/Ka* #excited/*Other/En* :D/*Other/Univ*
> **Translation:** *Saanu will visit Gujarat next month #excited :D*

Here we see that we have 2 named entities-

- *Saanu* - person (PER)
- *Gujarat* - location (LOC)

Some downstream tasks after NER in NLP include entity classification, entity linking, relation extraction, sentiment analysis, event extraction, question answering, text summarization, and information retrieval/search.

1.5 Event Annotation and Detection

Most historical definitions follow a pattern in which events are defined from the perspectives of space - objects and entities, and time. Event detection in Natural Language Processing (NLP) and Information Retrieval (IR) refers to the process of identifying and extracting events from text data. An event can be defined as something that happens at a particular time and place, involving one or more participants and having certain properties or attributes. The emphasis is on detecting the presence of events. This information can be useful for various applications, including news analysis by accurate selection of news messages[19], enhanced risk analytics [17], improve traffic monitoring systems [40], forecasting civil unrest [62], social media monitoring, event detection, trend analysis, and knowledge graph construction [91].

Event detection is important on social media because social media platforms generate a vast amount of user-generated content in real-time, which often includes discussions, updates, and reports about various events happening worldwide. Extracting events from social media data can help in understanding current trends, identifying significant occurrences, tracking public sentiment, and detecting emerging topics. It enables the monitoring of social media conversations around events, facilitates crisis management, supports social listening and market research, and provides valuable insights for various domains such as journalism, public opinion analysis, and brand reputation management.

1.6 Contribution of the Thesis

1.6.1 Named Entity Recognition for Kanglish CM Social Media data

In this thesis, we aim to take a few steps toward working on Named Entity Recognition (NER) for Kannada-English code-mixed social media data. We have curated an annotated dataset for Kannada-English code-mixed data from a social media platform, Twitter. Our two annotators who were fluent in both Kannada and English languages annotated the tokens with the named-entity tags (”Person”, ”Location” and ”Organization”). Our annotated dataset consists of 3,759 tweets with a total of 53,385

annotated tokens. We have also presented the results of various baseline classification models along with their analysis on the above-mentioned dataset.

1.6.2 Event Detection in Kanglish CM Social Media data

We have also worked towards the task of event annotation and detection in Kannada-English code-mixed social media data as a part of this thesis. For this task as well, we have curated an annotated dataset for Kannada-English code-mixed data from a social media platform, Twitter. Our two annotators who were fluent in both Kannada and English languages annotated the tokens with the event tag (BIO standard). Our annotated dataset consists of 800 tweets with a total of 7,688 annotated tokens. We have also presented the results of various baseline classification models along with their analysis on the created dataset.

Both the NER dataset and the Event Detection dataset, which to the best of our knowledge, are the first ones to be created for Kannada-English code-mixed social media data, in hopes that this would promote the generation of data resources for the Dravidian language.

1.7 Thesis Outline

The thesis is organised as below-

- In Chapter 1, we discuss the research motivation, challenges, and contribution of this thesis.
- In Chapter 2, we discuss the work that has been previously done related to code-mixed data, and the NLP tasks of Named Entity Recognition (NER) and Event Detection along with annotation guidelines.
- In Chapter 3 and 4, we present the work done as part of the thesis towards NER and Event Detection for Kannada-English code-mixed data respectively. We talk about the corpus creation process, annotation guidelines, experiments, results and analysis here.

- In Chapter 5, we conclude our thesis and discuss the scope of future work.

Chapter 2

Related Work

2.1 Code-Mixing in Indian Languages

In a multilingual country like India, code-mixing (CM) has become the standard rather than the exception. The term ”*Hinglish*”[87] is a combination of the terms ”Hindi” and ”English.” These languages have a code-mixing variety that occurs in both inter-sentential and intra-sentential code-mixing. While the term originated from Hindi, it is not exclusive to that language. It is widely used in India, with English terms blending with Tamil (*Tanglish*), Telugu (*Tenglish*), Kannada (*Kanglish*), Malayalam (*Manglish*), and many more. English and local tongue code-mixed language has become the lingua franca for the majority of Indians, particularly the youth. Although code-mixing is more prevalent in India's urban and semi-urban areas where we see most of the Indian-English speakers, it is rapidly spreading into rural and remote areas across all states.

Several studies on code-switching among Indian speakers have been conducted, including in Hindi movies/Bollywood [70][80], among Hindi-English bilinguals [41], and among bilinguals who speak English along with Kannada and Malayalam[29].

In recent years, the internet, particularly social media sites, has played an important role in this spread. Code-switching is highly common on social networking sites such as Facebook, Twitter, and WhatsApp, according to [2]. According to their findings, 86.40% of students and 81.00 % of teachers

employ code-switching on social media. A team at Microsoft Research[9] analysed Facebook posts generated by Hindi-English bilingual users and confirmed the presence of significant code mixing in them. They also found that CM data was in both English-in-Hindi and Hindi-in-English matrix forms, and showed that CM can happen at different levels with varying degree of mix of the two languages in a sentence. Nagane's study[22] focused on English and Romanized Hindi tweets from multilingual Indians on social-media. According to their findings, cursing in the dominant language (first language) is greatly favoured where emotions are high. They also analysed the causes for code-mixing and the list includes limited vocabulary, habitual expression, attitude expression, identification marker, occupation, displaying respect, and criticism.

As we talked about in Section 1.3, the lack of formal grammar and the use of nonstandard spellings make analysing code-mixed text problematic which only increases when we use the CM data from social media (Section 1.3.1. The first task that needs to be solved in this process is the task of language identification. Language identification is one of the basic steps while processing code-mixing data which involves identifying which language a particular word belong to in a code-mixed sentence. There has been some work done on the same across different CM language pairs. Barman et. al. [10] created a dataset and worked towards Bengali-English CM data for word-level language identification (LID). Nguyen et. al. [53] created an annotated corpus of Turkish-English code-mixed data for the same task. They used models such as dictionary based models and mathematical models to achieve an accuracy of 97.6% with language-model based approach. Hidayatullah et. al. (2022) [30] studied techniques, challenges, and dataset availability with corresponding quality criteria and developed a comprehensive framework for LID of code-mixed social media data. They studied 32 different annotated datasets from resource-poor languages, such as Kannada-English, Malayalam-English and Indonesian-English, to resource-rich languages such as Spanish-English, German-English and Hindi-English.

Another important basic task in NLP is POS tagging. POS tagging, or Part-of-Speech tagging, in code-mixed language refers to the process of assigning appropriate part-of-speech labels to each word in a text. The goal of POS tagging is to identify the grammatical category or part of speech (such as noun, verb, adjective, etc.) of each word in a sentence. POS tagging in code-mixed language is particularly challenging due to the presence of multiple languages and the mixing of grammatical structures and

lexical items from different languages. Vyas et. al. [90] worked on part-of-speech (POS) tagger for Hinglish code-mixed social-media data where they showed that enhancements and a better feature selection can resolve the transliteration issue that we see with social-media data. Singh et. al. [76] created a large annotated corpus for Hindi-English data from Twitter for POS tagging where they shared some baseline models as well. Similar work has been done by researchers recently for Telugu-English code-mixed social media data by Srirangam et. al. [78]. Sharma et. al. [74] addressed the issue of shallow parsing of Hinglish code-mixed data that they collected from social media. They developed a new system for Hinglish code-mixed text that can perform word-level language identification (LID), normalise the tokens to their standard forms, assign part-of-speech (POS) tags, and perform chunking. Irshad et. al. [13] created a tool for automatic language identification for English code-mixed data with 7 Indian languages - Hindi, Punjabi, Urdu, Kannada, Tamil, Gujarati, Telugu, and Malayalam, which greatly helps in the pre-processing and filtering steps in analysing these Indian code-mixed languages.

Tasks like emotional analysis [67][4] have also been explored for some language pairs. Joshi et. al. [39] curated and analysed Hinglish data for sentiment analysis. Translation of the identified words from a resource-poor language to a resource-rich language like English has been explored in tasks like question answering [61] where they could make use of the advantages of a resource-rich language and its tool for the complex task.

While we might have a good amount of work done for most of the popular monolingual languages and some of the code-mixed language pairs, many code-mixed language pairs have received very little interest or attention, including Kannada-English CM data. However, it has garnered some interest in the recent past. For Kannada-English CM data, Sowmya Lakshmi and Shambhavi [77] have proposed an automatic word-level Language Identification (LID) system for sentences from social media posts. Abhinav et. al. [3] reported a work on annotating CM Kannada-English data collected from Twitter and creating POS tags for this corpus. Hande et. al. [28] released a corpus for sentiment analysis and offensive language detection for Kanglish YouTube[1] comments along with computational models.

[1] www.youtube.com

2.2 Named Entity Recognition

Named Entity Recognition (NER) is a significant step forward in information extraction, with the goal of identifying and classifying crucial information in unstructured texts, i.e. named entities [52]. There has been a considerable amount of research done for named entity recognition (NER) in research-rich monolingual languages and newswire data such as such as English [25], German [81], and Spanish [21]. Morwala et. al. [50] used Hidden Markov Model (HMM) for their language-independent approach where they exhibited results for Indian languages such as Hindi, Urdu and Punjabi. There has been work done for multilingual NER models as well recently. Some of the recent examples of multi-lingual NER models are Tedeschi et. al. [79] where they propose a new methodology for producing high-quality training corpora for NER that is based on an effective mix of knowledge-based techniques and neural models, as well as a unique domain adaption strategy, and Shaffer et. al. [73] where they present a simple strategy for automatically finding language clusters in this embedding space by leveraging embeddings from a pre-trained masked language model.

NER in code-mixed language has picked up some pace as well in the last decade. In the CALCS workshop[2], Aguilar et al. [1] presented a shared challenge on named entity recognition. English-Spanish (En-Sp) and Arabic(modern)-Egyptian (Ar-Eg) were the language pairings employed. They created a new dataset for code-switched NER benchmarks using Twitter data and nine entity types. For the job, the participating teams employed models such as LSTM, CNN, and CRF, as well as custom word embeddings. Fetahu et. al. [24] worked towards NER in code-mixed web queries where L1 was English and L2 were Dutch, Russian, Korean, Farsi and Turkish. They compared the results with that of monolingual data for the same 6 languages and noisy CM data as well to simulate real-life scenarios. Singh et. al. [75] proposed an automatic NER pipeline for Hindi-English code-mixed data. Similarly, Srirangam et. al. [78] presented an annotated corpus for Telugu-English code-mixed social media data. A hybrid model where dictionary-based approach along with supervised classification approaches was proposed for Hindi-English and Tamil-English social-media text [12].

[2]https://code-switching.github.io/

To promote research work on the NER task for code-mixed data, there are workshops and shared tasks conducted periodically by different organisers[3][4].

2.3 Event Annotation and Detection

The study of events dates back quite a long time, and pre-linguistic definitions of events sought to describe and recognise events as "change in aspects of the perceived sense." Before we start with event detection, we should be first clear on what constitutes an event in a sentence. The guidelines for annotation of events have been published in English in 2006 [72], while we have recently published guidelines for event annotation in monolingual Kannada data - *Detection and Annotation of Events in Kannada* (2020) [60].

Now that we have the data annotated with the guidelines published, our main task is to identify those events automatically. Automated event mention detection in an open domain setting is a keystone for various information extraction tasks. This task was first brought to light during SemEval-2007, where the shared task *Task 15: TempEval Temporal Relation Identification* [85] was added as a new task with a focus on identification of temporal constructions. One of the six proposed tasks was concerned with the detection of events mention extent in the text. Participants in these tasks were provided with an event annotated corpora, annotated in accordance with the TimeML guidelines in several languages as well as an evaluation framework.

Previous research on automatic event detection has been conducted in two contexts: closed-domain and open-domain. The aim in a closed-domain context is to reliably detect event nuggets in text and assign an "event type" to each discovered event. "Event types" are predefined for each domain by a specific event ontology. The ACE corpus (2004) [23] has been the focus of most closed-domain event detection research. The purpose in an open-domain context is to identify all probable events in a given text without considering any specific event kinds or confining events to a single domain or type. This is the type of event detection that we will be focussing on as the text on social media is open-domain even

[3]https://dlnlp.ai/st/wojood/

[4]https://codalab.lisn.upsaclay.fr/competitions/11750

if we try to extract domain-related text alone. The subsections that follow showcase previous work on open-domain event detection for several languages.

Arnulphy's study [6] provides a machine learning-based strategy to detecting events in English by combining a k-Nearest Neighbour (kNN) classifier with a Conditional Random Field (CRF). While CRFs may learn the sequential connections between words on their own, they are generally poor at handling quantitative information in text and synonyms. To address these issues, a kNN is combined with a CRF. After the CRF has computed all of the possible labels and their probabilities, the kNN computes a similarity between each candidate (every potential event discovered by the CRF, regardless of probability) and all of the training cases. Using n-gram language modelling, the similarity is calculated. The authors employ Laplacian smoothing to handle unseen unigrams and adapt a back-off method from unseen bigrams to unigrams. In addition, the authors use a lexical resource [7] to provide information to polysemic terms. The authors train and test their model on the English corpus from the *SemEval-2010 Task 13: TempEval-2* [86]. They reported an F1 score of 0.79 with CRF model and 0.86 with the CRF-kNN model.

Arnulphy's study [6] also provides a way to automatically detecting events in French, where they employ CRF-kNN with two extra lexicons - if a word belongs to the VerbAction Lexicon [5] and if the word is The Alternate Noun Lexicon [15]. They train and test with the same shared task corpus [86] as in the case of English, reporting an F1 score of 0.78 with CRF and 0.83 with CRF-kNN.

UzZaman et. al. [83] introduced the TIPSem-B-Freeling system, which automates event extraction. This system is the Spanish counterpart of TIPSem [46], with the key distinction being that it lacks semantic roles. Furthermore, it employs Freeling [55] to automatically extract language aspects. TIPSem is a CRF model that has been supplemented with lexical, syntactic, and semantic data from the training data. The authors used the Spanish Timebank 1.0 corpus published by Saur and Badia [71] and reposed an F1 score of 0.88.

With a deep learning approach, a language invariant neural event detection (ALINED) architecture was proposed by Suhan et. al. [59] which makes use of a combination of sub-word level characteristics as well as lexical and structural data. Convolution over character embeddings is used with recurrent

layers over contextual word embeddings to achieve this. They show that their model recovers significant elements for identifying event spans without depending on language-specific features. They beat the then state-of-the-art model by 1.65 points and also introduced automatic annotation of events in Hindi.

When it comes to social-media, the data is the most up-to-date and inclusive stream of information on the current happenings but the data is noisy to say the least. However, researchers have explored event detection from social media platforms like Twitter in the last decade [58]. Ritter et. al. proposed Twical [65] which is a system to extract events from tweets and also categorise them automatically. Bhattu et. al. [14] used a two-fold approach where they first detect events using statistical models and then extract them using CRF, along with POS tags for dealing with the noisiness in social media data. Based on this approach, Burramsetty et. al. [16] have created a corpus for event detection from Telugu-English code-mixed dataset and provided baseline results achieving their best with Maximum Entropy model for event detection, with an F1 score of 0.79, and CRF for event detection. There have been attempts at event detection from social media streams[33], but we will not be working on those as part of this thesis.

Chapter 3

Named Entity Recognition

3.1 Overview

In section 1.4, we have defined the task of named entity recognition (NER) in NLP and its applications. We have then discussed the related work done in the field of NER in monolingual data, social media and code-mixed data in Section 2.2. In this chapter, we formally define the task of Named Entity Recognition (NER) in Kannada-English social-media data. We will also discuss the annotation methodology for this task in Section 3.3. We will also be sharing the approaches for this task in Section 3.5 and evaluating the experiments at the end in Section 3.9.

3.2 Task Definition

The task we are trying to solve is identifying named entities in a Kannada-English code-mixed sentence. Given that there is no annotated corpus available for the same, a said corpus for the said NER tasks has to created from a social media platform like Twitter. The collected data, after cleaning/preprocessing, would be annotated with the named entity (NE) tags decided ($\mathrm{t1}, \mathrm{t2}, ..., \mathrm{tn}$, where *n* refers to the number of NE tags) with B-Begin, I-In and O-Other format. Once we have the annotated corpus, an ideal solution would be to build a model that would predict a tag for each token of a code-mixed Kannada-English sentence.

3.3 Annotation Methodology

In this section, we shall discuss the method that we have used to annotate our corpus. We label each tokens with the following three named entity tags - 'Person', 'Organisation', 'Location' using the inside-outside-beginning (IOB) format [63] for annotating.

- *Per* - Refers to the 'Person' tag, which indicates names of people. Some examples are 'Sumukh', 'George', 'Aamir Khan'. It can also refer to twitter handles/usernames such as '@ssumukh'. We also include common nicknames in this type.

  Ka-En: Bollywood/*B-Org/NE* alli/*Other/Ka* ,/*Other/Univ* Aamir/*B-Per/NE* Khan/*I-Per/NE* tumba/*Other/Ka* chennagi/*Other/Ka* acting/*Other/En* madtane/*Other/Ka* #awesome/*Other/En* :D/*Other/Univ*

  Translation: *In Bollywood, Aamir Khan acts really well #awesome :D*

  Here, we see the "Per" named entity - *Aamir Khan*.

- *Org* - Refers to 'Organisation' named entities. Some examples are "IIIT Hyderabad", "NVIDIA", "Kwality Walls". They can also be names of political parties such as 'JDS', 'Aam Aaddmi Party'.

  Ka-En: Hoogi/*Other/Ka* work/*Other/En* maadu/*Other/Ka* Kwality/*B-Org/NE* Walls/*I-Org/NE* company/*Other/En* alli/*Other/Ka*

  Translation: *Go work in Kwality Walls company*

  We can see the 2 worded organisation named entity - *Kwality Walls*.

- *Loc* - Refers to the 'Location' tag which indicates the name of a place. Some examples are 'Yelenahalli', 'Santa Clara', 'North Korea', 'Moscow', etc. As we are annotating social media, we will also consider hashtags of location named entities such as '#India' as a 'Loc' tag.

  Ka-En: Namma/*Other/Ka* #Bangalore/*B-Loc/NE* roads/*Other/En* alli/*Other/Ka* tumba/*Other/Ka* traffic/*Other/En* jam/*Other/En* aagatte/*Other/Ka* #annoyed/*Other/En*

  Translation: Traffic jam happens often in #Bangalore roads #annoyed

Here, we have the name of a city - *#Bangalore* with a hashtag being tagged as a location named entity.

Please note that the structure of the sentences was changed in some of the examples above. This is because of the difference in grammatical structure between Kannada and English. The sentences where the structure was changed in the translation were originally following the Kannada language structure while swapping some Kannada words with English words.

As stated, we will be using the Inside-Outside-Beginning (IOB) format, so the total number of named entity tags becomes 6 - B-Per, I-Per, B-Org, I-Org, B-Loc and I-Loc. *B-* tags refer to the beginning of a named entity and *I-* tags refer to the intermediate tokens of an entity if the particular entity has mutliple words. We use the 'O' (Other) tag for all tokens that do not come under the mentioned named entities. This makes a total of 7 tags that will be used for our annotation task.

The following is an instance of the annotation guidelines above where all the mentioned named entities are present-

Ka-En: Tomorrow/*Other* ,/*Other* Chandu/*B-Per* Reddy/*I-Per* avru/*Other* Mysore/*B-Loc* alliro/*Other* NVIDIA/*B-Org* Graphics/*I-Org* office/*Other* visit/*Other* madtaare/*Other* !/*Other*

Translation: *Tomorrow, Chandu Reddy will visit NVIDIA Graphics office in Mysore!*

3.3.1 Dealing with Ambiguous Tokens

In case a word has different means in the two languages we are working with and the words refers to a named entity in one language while it does not in the other language, we tag the token with whatever seems appropriate based on the context of the sentence.

For example, the token '*Bali*' refers to the name of the capital city of Indonesia, but in Kannada, the word is used to mean '*near*'. In sentence T1 below, we tag the token with a *'Other'* while in sentence T2 we tag the token with a *B-Loc*.

Ka-En T1: Naale, Rohan Mysore **bali** iruva vinyard visit maadtaane

Translation: Tomorrow, Rohan will visit the vinyard **near** Mysore

Ka-En T2: Naale, Rohan **Bali** visit maadtaane

Translation: Tomorrow, Rohan will visit **Bali**

The annotators shall make such context-based decisions for any other ambiguity that they might come across as some words can have multiple meanings, in the same language or across languages. Such ambiguity is one of the challenges in working with Code-Mixed social media data, as we discussed in Section 1.3 and Section 1.3.1.

3.4 Corpus and Statistics

3.4.1 Data collection

Data collection is a vital step while dealing with any problem with any neural-network based approaches [66]. As there are only a few sources for code-mixed low-resource language data, this would be challenging as it is difficult to build supervised models.

The corpus that we created from Twitter[1] for Kannada-English code-mixed tweets contains tweets from December 2020 to August 2022. We used hashtags related to city names where Kannada is widely spoken, politics, movies, events, and trending hashtags in collecting the corpus. We also manually identified some of the Twitter accounts that posted often with code-mixing between Kannada and English languages.

Using the Twitter API, we retrieved around 222,124 tweets. The following types of tweets were identified and removed-

- Tweets having only English or only Kannada.

[1] http://twitter.com/

Label	Count of tokens
Kannada	20,380
English	19,701
Named Entities	8,096
Universal	5,208
Total number of tokens	53,385
Avg. tweet length	14.2
Total tweets	3,759

Table 3.1: NER Corpus statistics

Tag	Count of tokens
B-Per	3,729
I-Per	787
B-Org	1,338
I-Org	750
B-Loc	1,137
I-Loc	355

Table 3.2: NER tag statistics

- Tweets having only URLs, emojis or hashtags.

- Tweets with less than 5 tokens.

After manually filtering the data with the steps mentioned above, we were left with 3,759 code-mixed Kannada-English tweets. We tokenized these sentences and removed URLs from the same in an effort to reduce the noise. For the language identification part, we used the tool that was developed by Irshad et. al. [13] but it needed manual checks as social media data differs from the standard language (Section 1.3.1.

3.4.2 Data statistics

The corpus has a total of 53,385 tokens which were tagged for the 7 tags mentioned in the Section 3.3. The corpus statistics and the tag statistics can be seen in Table 3.1 and Table 3.2 respectively.

Tag	Cohen Kappa score
B-Per	0.97
I-Per	0.96
B-Org	0.97
I-Org	0.91
B-Loc	0.96
I-Loc	0.94

Table 3.3: Inter Annotator Agreement

3.4.3 Inter Annotator Agreement

Annotation of the dataset for NE tags in the tweets was carried out by 2 human annotators having linguistic background and proficiency in both Kannada and English based on the methodology in Section 3.3. In order to validate the quality of annotation, we calculated the inter-annotator agreement (IAA) between the 2 annotation sets of 3,759 code-mixed tweets having 53,385 tokens using Cohen's Kappa [20]. Table 3.3 shows the results of agreement analysis.

We find that the agreement is significantly high. Furthermore, the agreement of 'I-Loc' and 'I-Org' annotation are relatively lower than that of 'I-Per', and this is because of the presence of uncommon/confusing words in these entities along with the ambiguities that we discussed in Section 3.3.1.

Disagreements about the tags were resolved through discussions between the annotators to reach a mutual agreement.

3.5 Supervised Approaches for Named Entity Recognition

Supervised machine learning is a category of machine learning where a model is trained on labeled training data, where each data point has both input features and corresponding output labels. The goal of supervised learning is for the model to learn the mapping between input features and output labels so that it can make accurate predictions on unseen data.

In the context of Named Entity Recognition (NER) for Natural Language Processing (NLP), supervised machine learning is commonly used to build NER models.

Supervised learning for Named Entity Recognition (NER) has been used for several decades, and its introduction predates the specific term ”NER” itself. While the concept of supervised learning has a long history, the application of supervised learning techniques to NER can be traced back to the early days of Natural Language Processing (NLP) research.

One influential work in the early development of supervised learning for NER is the paper titled ”A Maximum Entropy Approach to Named Entity Recognition” by Andrew McCallum and Dayne Freitag, published in 2000 [49]. The paper introduced a probabilistic approach using maximum entropy modeling for NER. The authors formulated NER as a sequence labeling task and trained models on labeled data to predict the named entity tags for each word in a given sentence. This work laid the foundation for using supervised learning algorithms, particularly conditional random fields (CRFs) and maximum entropy models, for NER. It provided a framework for training models on labeled data, extracting relevant features, and making predictions for named entities in text.

Since then, supervised learning techniques for NER have evolved and improved with advancements in machine learning and deep learning. Researchers and practitioners have explored various models, including recurrent neural networks (RNNs), long short-term memory (LSTM) networks, Bidirectional LSTMs and more. These models have achieved state-of-the-art performance in NER by leveraging large labeled datasets and powerful computational resources. We have benchmarked our dataset with Random Forests, CRF, Bi-LSTM and BiLSTM+CRF.

However, supervised machine learning for NER also has some limitations, including the need for labeled training data, potential biases in the annotated data, and difficulties in handling out-of-vocabulary or rare entities. Despite these limitations, supervised machine learning remains a popular and effective approach for building NER models in NLP, enabling accurate identification and classification of named entities in text for a wide range of applications

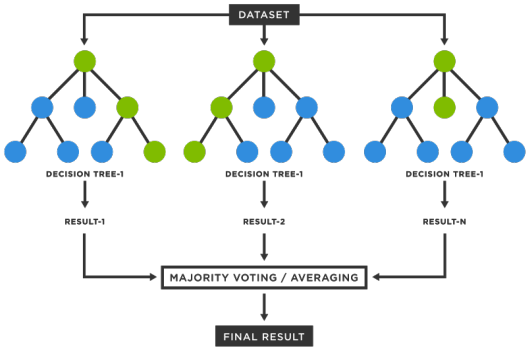

Figure 3.1: Visual representation of Random Forest classifier

3.5.1 Random Forest

Random Forest is an ensemble learning algorithm that combines multiple decision trees to make predictions. Each decision tree in the Random Forest is trained on a random subset of the training data and uses a random subset of features for splitting at each node, originally proposed in 1995 by Ho [31].

Here's a brief explanation of how Random Forest works:

1. **Random Sampling:** Random subsets of the training data are created through a process called bootstrapping, where data points are randomly selected with replacement. This ensures diversity in the training data for each decision tree.

2. **Feature Randomness:** At each node of a decision tree, only a random subset of features is considered for splitting. This randomness reduces correlation among trees and introduces variability, leading to better generalization.

3. **Tree Construction:** For each decision tree, the splitting of nodes is performed recursively. At each node, the algorithm selects the best feature and threshold to split the data based on a specified criterion, such as Gini impurity or information gain. Information gain focuses on reducing entropy and works well with categorical target variables, while Gini impurity quantifies misclassification probability and is suitable for continuous target variables and imbalanced class distributions. Both measures are effective in practice, and the choice between them depends on the specific problem

and dataset characteristics. The process continues until a stopping condition is met, such as reaching a maximum depth or having a minimum number of samples in each leaf node.

4. **Ensemble Prediction:** Once all the decision trees are constructed, predictions are made by aggregating the predictions of each individual tree. For classification tasks, as in our case of NER, the most common prediction among the trees is taken as the final prediction.

By using random subsets of data and features, Random Forests reduce overfitting and improve generalization compared to individual decision trees. It also handles noise and outliers well due to the averaging effect of multiple trees[57].

3.5.2 Conditional Random Fields

Conditional Random Fields (CRFs) are probabilistic models used for sequence labeling tasks, such as Named Entity Recognition (NER) in Natural Language Processing. CRFs model the conditional probability of a sequence of labels given an input sequence, taking into account the dependencies between labels.

Here's an explanation of CRFs:

1. **Input Sequence:** Consider an input sequence of tokens $x = (x_1, x_2, \ldots, x_n)$. These tokens could represent words, characters, or any other units of the sequence.

2. **Label Sequence:** We aim to assign labels to each token in the input sequence. Let's denote the label sequence as $y = (y_1, y_2, \ldots, y_n)$, where each y_i represents the label assigned to the corresponding token x_i.

3. **Conditional Probability:** CRFs model the conditional probability $p(y|x)$, which represents the probability of a label sequence given the input sequence. The goal is to find the most probable label sequence given the input.

4. **Feature Functions:** CRFs use feature functions to capture the dependencies between labels and the input sequence. These feature functions, denoted as $\phi(y_i, y_{i-1}, x)$, quantify the compatibility of a particular label assignment y_i with the neighboring labels y_{i-1} and the input features x. These features can include word representations, part-of-speech tags, or any other relevant information.

5. **Modeling Probability:** The conditional probability of a label sequence given the input sequence in CRFs is defined as:

$$p(y|x) = \frac{1}{Z(x)} \prod_{i=1}^{n} \phi(y_i, y_{i-1}, x)$$

   Here, $\prod_{i=1}^{n}$ denotes the product over all positions in the sequence, and $Z(x)$ is a normalization factor known as the partition function. The partition function ensures that the probabilities sum up to 1 over all possible label sequences for a given input sequence.

6. **Inference:** During inference or prediction, the most likely label sequence $\hat{y}$ is obtained by maximizing the conditional probability:

$$\hat{y} = \arg\max_{y} p(y|x)$$

   This can be efficiently computed using dynamic programming algorithms like the Viterbi algorithm, which finds the label sequence that maximizes the product of the feature functions' values.

CRFs are trained using optimization techniques such as maximum likelihood estimation (MLE) to estimate the model parameters. The parameters are learned to maximize the log-likelihood of the training data.

By considering the dependencies between labels and the input sequence, CRFs can capture contextual information and make coherent predictions for sequence labeling tasks like NER. They have been widely used in various NLP applications where sequence structure plays a crucial role. In named entity recognition (NER) using IOB standard annotation for the tags, the *I-Org* cannot follow *I-Per* tag [82]. CRFs are appropriate and offer better performance metrics for the NER challenge since we are concentrating on the sentence level rather than the token level.

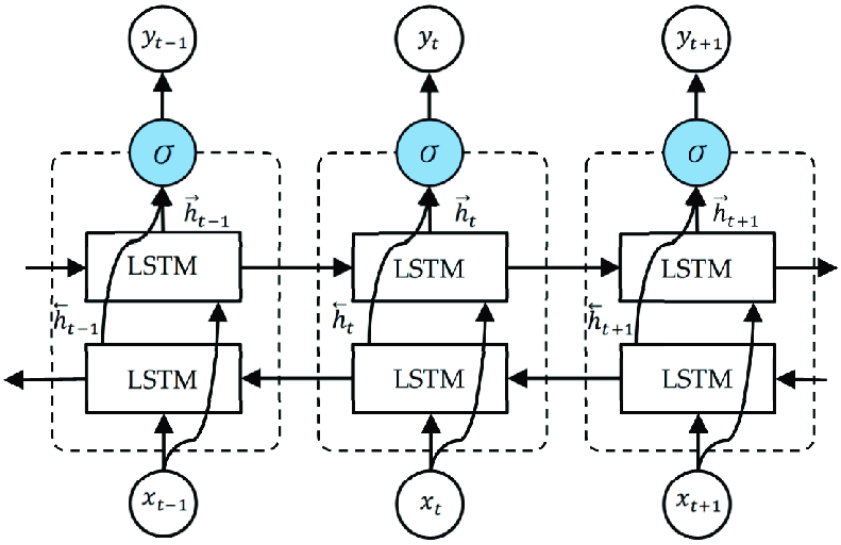

Figure 3.2: The unfolded architecture of Bidirectional LSTM (BiLSTM) with three consecutive steps[45]

3.5.3 Bi-directional Long Short-Term-Memory (Bi-LSTM)

Bidirectional Long Short-Term Memory (BiLSTM)[27] is a type of recurrent neural network (RNN)[38] architecture commonly used in Natural Language Processing (NLP) tasks, including sequence labeling tasks like Named Entity Recognition (NER). BiLSTM combines the forward and backward information of a sequence to capture contextual information from both directions with Long Short-Term Memory (LSTM)[32] networks.

A visual representation of the BiLSTM can be seen in the Figure 3.2Here's an explanation of BiLSTM:

1. **Sequential Input:** BiLSTM takes a sequential input, such as a sequence of words or characters. Each element in the sequence (e.g., word or character) is represented as an input vector.

2. **Forward and Backward Processing:** BiLSTM consists of two LSTM layers: one processes the sequence in the forward direction (from the beginning to the end), and the other processes the sequence in the backward direction (from the end to the beginning). Each LSTM layer maintains an internal state to capture information from past elements and updates it as new elements are processed.

3. **Hidden States:** During processing, each LSTM layer generates a sequence of hidden states. The hidden states capture the contextual information of the input sequence up to that position, considering both preceding and succeeding elements.

4. **Concatenation:** The hidden states from both the forward and backward LSTMs are concatenated at each position, combining the information from both directions. The concatenated hidden state represents the contextual information of the input sequence at that position.

5. **Output:** The concatenated hidden states can be further processed or used directly for downstream tasks. In NER, for example, the BiLSTM output can be passed through a fully connected layer or a softmax layer to obtain the predicted labels for each element in the input sequence.

The key advantage of BiLSTM is that it captures information from both past and future elements, allowing it to consider the complete context around each input element. This is especially useful in NLP tasks like NER, where the labeling decision for an element often depends on the surrounding words or characters.

By using both forward and backward information, BiLSTM improves the model's ability to capture long-range dependencies and contextual cues. It has been shown to be effective in various NLP tasks, including sequence labeling, sentiment analysis, machine translation, and more.

3.5.4 Bidirectional Long Short-Term Memory network with Conditional Random Fields (BiLSTM-CRF)

Bidirectional Long Short-Term Memory Conditional Random Fields (BiLSTM-CRF) combines the context-capturing abilities of Bidirectional Long Short-Term Memory (BiLSTM) with the sequential modeling and label dependencies of Conditional Random Fields (CRF) [34][44], as shown in figure 3.3.

We understood the working of CRFs and BiLSTMs in Section 3.5.2 and Section 3.5.3 respectively. The output of the BiLSTM layer, which represents the contextual information of the input sequence, is fed into the CRF layer. The CRF layer takes the BiLSTM outputs as input and models the conditional

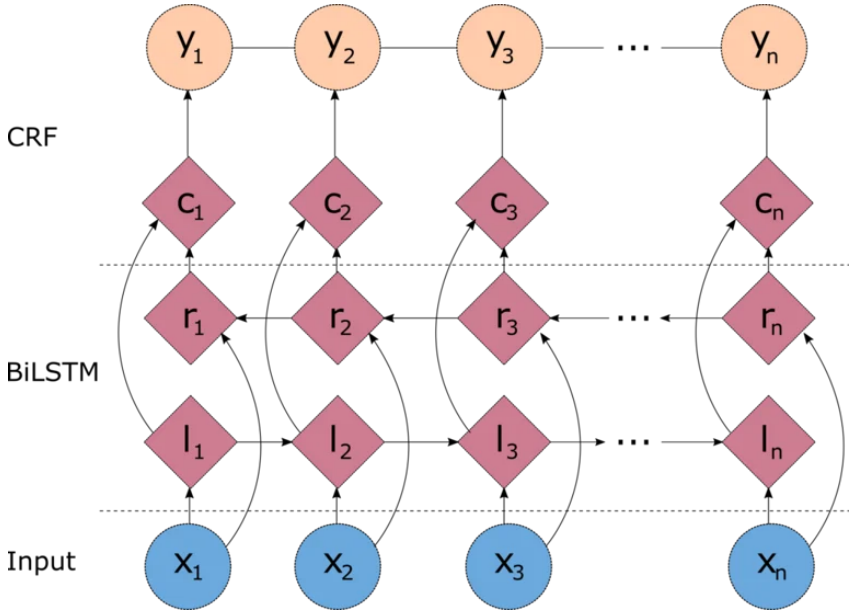

Figure 3.3: Architecture of a BiLSTM-CRF model

probability of label sequences given the input sequence. The CRF layer uses the transition probabilities between labels and considers the compatibility between labels and input features.

During training, the BiLSTM-CRF model is optimized to maximize the log-likelihood of the correct label sequences in the training data. The model learns the parameters that define the transition probabilities and feature compatibility. During inference or prediction, the Viterbi algorithm is commonly used to find the most likely label sequence given the input sequence, considering both the BiLSTM outputs and the label dependencies modeled by the CRF.

The combination of BiLSTM and CRF in BiLSTM-CRF addresses the limitations of using BiLSTM alone, as it does not explicitly model label dependencies. The CRF component allows the model to make predictions based on the overall label sequence's coherence and consistency.

BiLSTM-CRF has achieved state-of-the-art results in various sequence labeling tasks, including NER, part-of-speech tagging, and named entity chunking. It effectively captures contextual information, handles label dependencies, and produces accurate and coherent label predictions for sequential data. Due to it's proven effectiveness in NER[56], we use this model as well.

3.6 Feature Selection

The input to our supervised machine learning models includes lexical, word-level, and character-level features such as char N-Grams (bigram and trigram) to get information from emojis, mentions, suffixes in social media like '#,' '@,' digits in the string, numerals, and punctuation. Contextual features are derived from nearby tokens.

1. **Capitalization:** People on social media frequently use capital letters when referring to the names of people, organisations, and places[88]; at times, they put the full name in caps to emphasise significance or to signify hostility. This results in a few binary characteristics. One characteristic indicates whether the first letter of a word is capitalised, while the other indicates if the entire word is capitalised.

2. **Mentions and Hashtags:** On social media, users often tag by using the '@' symbol with the usernames to refer to people or organisations, and they use '#' hashtags to highlight something or make the topic trend. Thus, the inclusion of these two increases the likelihood that the word is a named thing that falls within the category of proper nouns.

   For example, consider the following sentence-

   > **Ka-En:** @rakshit nim movies andre tumba ishta, namma #Sandalwood industry improve maadi!.
   > **Translation:** @rakshit your movies I like a lot, improve our #Sandalwood industry!

   The token "@*rakshit*" is referring to a person (B-Per tag) and "*#Sandalwood*" is the name of the Kannada film industry (B-Org tag). They are identified by the symbols @ and #. It is important to note that not all hashtags will be a named entity, so we need to understand the word context to correctly classify.

3. **Word N-Grams:** Bag of Words (BoW) is a simple and popular technique used in Natural Language Processing (NLP) for representing text data. It disregards grammar and word order and focuses solely on the presence and frequency of words within a document or a corpus. The BoW model

represents a document as a collection (or ”bag”) of words, without considering the sequence or context.

While BoW has limitations in capturing word order and contextual information, it remains a widely used and effective technique for many NLP applications, thanks to its simplicity and ability to transform text data into numerical representations suitable for machine learning algorithms. In tasks like NER, the bag of words has been the standard for languages other than English [36]. As a result, we train our model as our word N-Grams using adjacent tokens as a feature vector. These are also referred to as contextual features. In our work of NER for Kannada-English code-mixed task, we have used trigrams.

4. **Character N-Grams:** Character n-grams are subword units formed by grouping consecutive characters together, where ”n” represents the number of characters in each group. For example, with the word ”example” and a character trigram (n=3), the resulting n-grams would be ”exa”, ”xam”, ”amp”, ”mpl”, ”ple”. Character N-Grams have been shown to be efficient in the task of text classification and are language-independent [48]. They are useful when there are typos in the text [18] [35] [47]. A group of characters helps in capturing morphological, syntactic and semantic information. They provide a finer-grained representation of text and can help overcome the out-of-vocabulary problem.

   Character N-Grams are especially useful in circumstances when there is free usage of terms that differ greatly from the normal Kannada-English vocabulary, such as coding mixed language especially when we are dealing with social media data.

5. **Common Symbols:** It has been noted that currency symbols, as well as braces (such as square, curly, etc.), are generally followed by numbers or other unimportant statement. As a result, these are a good sign that the words following or before them are not a NE. Though this might not hold true for all cases, our ablation study of the results points that this is a useful feature.

6. **Numbers in String:** In the context of social media content, it has been observed that users often use alphanumeric representations of legitimate vocabulary words to save typing effort, shorten message length, or express their style. Examples of such representations include ’n8’ for ’night’

and 'b4' for 'before'. Based on the analysis of a corpus, it has been noticed that alphanumeric words are generally not named entities (NEs). Therefore, they can serve as effective indicators for negative examples, helping in distinguishing NEs from non-NEs in tasks like Named Entity Recognition (NER) or information extraction from social media data.

3.7 Experimental Setup

In this section, we conducted a series of experiments to assess the impact of different features and model parameters. Our goal was to understand the effect of individual features that we discussed in Section 3.6 and explore the influence of various model settings that we discussed in the Section 3.5.

To achieve this, we performed experiments using different combinations of features and systems. We experimented with using a subset of features together and all features simultaneously. Additionally, we varied the parameters of the models. For example, we adjusted the criterion parameter ('Information gain' or 'gini') and the maximum depth of the decision tree model. For the CRF model, we explored different regularization parameters and optimization algorithms such as 'L2 regularization', 'Avg. Perceptron', and 'Passive Aggressive'. Similarly, we experimented with optimization algorithms and loss functions for the LSTM model.

To evaluate the performance of our classification models, we employed 5-fold cross-validation. This approach helped us validate the models by partitioning the data into five subsets, training the models on four subsets, and evaluating their performance on the remaining subset. We repeated this process five times, each time using a different subset for evaluation.

For the implementation of the algorithms, we utilized the 'scikit-learn' and 'keras' libraries in Python, which provide efficient and user-friendly tools for machine learning tasks.

In terms of data splitting, we allocated 60% of the data for training, 10% for validation, and 30% for testing. This division allowed us to train our models on a substantial portion of the data, validate their performance on a separate subset, and finally assess their generalization ability on a dedicated testing set.

By conducting these experiments, we aimed to gain insights into the impact of different features and model parameters, enabling us to make informed decisions about the best configuration for our classification models. The following are the model parameters that worked best for us-

- Random Forest (RF): A random forest with a max depth of 32, with Gini index as the criterion yielded the best results.
- Conditional Random Field (CRF): The CRF model that performed the best had L1 regularization parameter as 0.1 and L2 regularization as 0.1 with 'Average Perceptron' as the optmisation algorithm.
- Bidirectional Long Short-Term Memory network (BiLSTM): By selecting 'softmax' as the activation function, 'adam' as the optimizer, and 'sparse categorical cross-entropy' as the loss function, the BiLSTM model achieved the best performance in terms of accuracy or other evaluation metrics, as assessed during the experiments.
- BiLSTM-CRF: By utilizing the 'softmax' activation function, 'rmsprop' optimizer, categorical cross-entropy loss function, and random initialization of embedding vectors, the BiLSTM-CRF model achieved the best results in terms of our evaluation metrics, as observed during the experiments.

3.8 Evaluation Metrics

To evaluate the performance of our models for event nugget detection, we employ three commonly used metrics: Precision, Recall, and F-Measure (F1).

Precision measures the proportion of correct positive predictions made by the model. It is calculated as the ratio of True Positives (the number of correctly identified event nuggets) to the sum of True Positives and False Positives (the total number of event nuggets identified by the system).

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives}+\text{False Positives}}$$

Recall, also known as sensitivity or true positive rate, measures the proportion of actual event nuggets correctly identified by the model. It is calculated as the ratio of True Positives to the sum of True Positives and False Negatives (the event nuggets missed by the system).

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

F-Measure (F1) is the harmonic mean of precision and recall and provides a balanced measure of a model's performance. It combines precision and recall into a single metric that takes both false positives and false negatives into account.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

These metrics are commonly used in information retrieval and classification tasks, including event nugget detection. They help assess the effectiveness and accuracy of the model in identifying event nuggets correctly while considering both false positives and false negatives. By examining these metrics, we can evaluate the overall performance of the models and make informed decisions about their effectiveness in event nugget detection.

3.9 Results and Analysis

Table 3.4 captures performance of all models for our dataset. Our best model is the BiLSTM-CRF which achieved a weighted average F1-score of 0.94 with '*softmax*' activation function, '*rmsprop*' optimiser, '*categorical cross-entropy*' loss function and random initialisations of embedding vectors. As BiLSTM-CRF can efficiently use both past and future input features from BiLSTM and sentence level tags from CRF, we see that the accuracy is enhanced.

Table 3.5 shows results of our ablation study after removing each particular feature. We can see that the N-grams features have the most impact on our F1-scores, and this is understandable as char n-grams are helpful when there are misspellings and capturing semantic information when there is free use of words which vary significantly from standard word of Kannada and English words.

Tag	RF	CRF	BiLSTM	BiL-CRF
B-Per	0.32	0.82	0.81	0.84
B-Org	0.70	0.63	0.65	0.63
B-Loc	0.37	0.70	0.82	0.81
I-Per	0.35	0.55	0.57	0.62
I-Org	0.23	0.52	0.46	0.55
I-Loc	0.30	0.46	0.41	0.45
Other	0.95	0.97	0.96	0.97
Wtd avg	0.89	0.93	0.92	0.94

Table 3.4: F1-scores for CRF, BiLSTM and BiLSTM-CRF respectively with the weighted average at the end.

On analysing some of the results from the model, we see that the intermediate tags of location and organisation is lower than that of a name. This can be explained with the fact that there are uncommon/confusing words in the oraganisation and location names. For example, the word "*Bhaarath*", one of the names for the country India, is "*B-Loc*" while the words "*Bharat*" and "*Bhaarti*" are common first names in India which are tagged as "*B-Per*". Furthermore, there are confusing words like "*Bali*" which is a city in Indonesia, but in Kannada, it means "*near*". This can be seen in the example provided in Table 3.6 where the word "*Bharat*" is referring to a person with that name while our model is predicting that the word is a location, referring to the country India.

We tested a random tweet with the BiLSTM-CRF model that we trained, and here is the model predicted tags along with the ground truth tags in the Table 3.7. We noticed that the *I-Per* is predicted incorrectly for the Kannada word *puttanige* (an endearment word for kids) as this word is very similar to some of the common last names in southern part of India such as *Puttanna* and *Puttagere*. The low scores for intermediate tags (*I-per, I-Org* and *I-Loc*) can be attributed to these reasons along with the "noisiness" of the social media data which tends to have misspelled words and colloquial forms of words. This gets more difficult with Kannada-English code-mixed data as mixing happens at word-level, mostly for Kannada language prepositions and named entities or English language nouns (Section 1.3).

Complex named entities, like creative work titles such as movie, book, song, or software names, often exhibit characteristics that make them challenging to identify. According to research by Ashwini and Choi (2014) [8] where they worked on NER for social media data, these entities frequently involve

Feature removed	Precision	Recall	F1
Capitalisation	0.74	0.53	0.61
Mentions, hashtags	0.72	0.57	0.63
Char n-gram	0.65	0.41	0.50
Word n-Gram	0.62	0.44	0.51
Common symbols	0.75	0.48	0.58
Numbers in String	0.78	0.56	0.65

Table 3.5: Ablation study: Weighted average scores when a specific feature is removed for the BiLSTM-CRF model.

Word	Truth	Predicted
Banashankari	B-Loc	B-Loc
alliro	Other	Other
BESCOM	B-Org	B-Org
kacheeri	Other	Other
alli	Other	Other
work	Other	Other
siktu	Other	Other
Bharat	B-Per	B-Loc
annavrige	Other	Other
#work	Other	Other

Table 3.6: BiLSTM-CRF example (T1) prediction

Word	Truth	Predicted
Javalli	B-Loc	B-Loc
village	Other	Other
alli	Other	Other
Jnanadeepa	B-Org	B-Org
School	I-Org	I-Org
sersudvi	Other	Other
nan	Other	Other
maga	Other	Other
Suhas	B-Per	B-Per
puttanige	Other	I-Per

Table 3.7: BiLSTM-CRF example (T2) prediction

imperative clauses or resemble regular syntactic elements. As a result, they can be more difficult to recognize compared to simple nouns.

These complicated noun phrases pose challenges in syntactic parsing, and most Named Entity Recognition (NER) algorithms struggle to accurately identify them. We see the same in our experiments as well, where creative work titles tend to be misclassified as one of 'Per', 'Org' and 'Loc' tags.

3.10 Conclusion

In conclusion, our work on Named Entity Recognition (NER) for code-mixed Kannada-English social media data has yielded significant achievements. Here is a summary of our accomplishments:

1. **Annotated Corpus:** We have created a new annotated corpus specifically for code-mixed Kannada-English named entity recognition. To the best of our knowledge, this is the first corpus of its kind, providing valuable resources for future research in this domain.

2. **Research Problem:** We have identified and addressed the challenge of named entity recognition in code-mixed Kannada-English data as a research problem. Code-mixed data presents unique linguistic complexities, and our work contributes to advancing NER techniques in this particular context.

3. **Machine Learning Models:** We conducted experiments using various machine learning models on our annotated corpus. Specifically, we employed Random Forest, CRF (Conditional Random Fields), BiLSTM (Bidirectional Long Short-Term Memory), and BiLSTM-CRF models.

4. **Performance Evaluation:** Our models achieved impressive results in terms of F1-score. We obtained an F1-score of 0.89, 0.93, 0.93, and 0.94 for the Random Forest, CRF, BiLSTM, and BiLSTM-CRF models, respectively. These scores demonstrate the effectiveness of our approaches, especially considering the complexity of the task and the limited research available in the field of code-mixed named entity recognition for low-resource languages.

By creating the annotated corpus, addressing the research problem, and achieving strong performance with the machine learning models, we have laid the foundation for further advancements and opens avenues for future research in this emerging domain for Kannada-English code-mixed social media data.

Chapter 4

Event Annotation and Detection

4.1 Overview

In section 1.5, we have defined the task of Event Annotation and Detection in NLP and its applications. Then we discussed the related work done in the field of event detection in monolingual data, social media and code-mixed data in Section 2.3. In this chapter, we will formally define the task of Event Detection in Kannada-English social-media data while providing annotation guidelines for the same. We will be sharing the approaches for this task in Section 4.5 and evaluating the experiments at the end in Section 4.8.

4.2 Task Definition

The task we are trying to solve is detecting event entities in a Kannada-English code-mixed sentence. Given that there is no annotated corpus available for the same or annotation guidelines for the same, we first define the guidelines and then a corpus for the said Event Detection task has to be created from a social media platform like Twitter. The collected data, after cleaning/preprocessing, would be annotated with the *B-Begin, I-In*, and *O-Outside* tags based on the guidance we have established to create an annotated corpus. Finally, the ideal solution would be to build a model that would predict a tag for each token of a code-mixed Kannada-English sentence.

4.3 Annotation Methodology

In this section, we shall discuss the method that we have used to annotate our corpus. We label each tokens with the inside-outside-beginning (IOB) format [63], where B refers to the beginning of an event, I refers to the token that is part of the event but not the first token and O refers to all other tokens. We propose these principles, which are inspired by TimeML, and are organised by the Part-of-Speech (POS) of the event nugget. Nouns, finite verbs, non-finite verb constructions such as infinitives, and adjectival and adverbial participle constructions are examples of these components of speech. As most of the code-mixed Kannada-English sentences follow the structure of Kannada grammar while swapping language for keys words such as common nouns, we will follow the guidelines that we have proposed for Kannada monolingual event annotation in our paper - Detection and Annotation of Events in Kannada (2020) [60]. For English grammar based sentences, we have the TimeML annotation guidelines [72]. Following are the guidelines that we are using for Kannada language structured sentences from our monolingual work.

4.3.1 Nouns

Events can occasionally be expressed as nouns, although this is less common compared to verbs. Nominal events refer to abstract nouns that relate to a temporal phenomenon and inherently convey a notion of finiteness, such as chunavane (election), pasavu (famine), etc.

> **Ka-En:** Karnataka *chunavane* sheeghra agutte antha namme home minister announce madidru
>
> **Transation:** Our home minister announced that Karnataka *election* will be soon

Here we see an example of a noun - *chunavane* (election) being used as an event. We also have the event of '*announced*,' but we are not highlighting it as we are trying to understand nouns as an event.

4.3.2 Verbs

Due to the flexible nature of word order, verbs are often morphologically marked, making it easier to identify events. Events can occur at various positions within a sentence, and their identification can be achieved by identifying the main verb [84].

4.3.2.1 Finite Verbs

Finite verbs are categorized as events because they denote actions that bring about a change in the state of the world. They possess tense and aspect information, which inherently conveys a notion of temporality.

Ka-En: Prashant avna resignation letter annu *kalisidaane*

Translation: Prashant *sent* his resignation letter

4.3.2.2 Non-finite Verbs

In Kannada syntax, there is a rule that allows only one finite verb per sentence. All other verb forms in Kannada are classified as participles, which can function as adjectives or adverbs. Additionally, Kannada does not have a gerund verb form, but it does include infinitives and subjunctives.

1. **Adjectival Participle Construction:** Adjectival participle constructions in Kannada involve converting the verb into an adjective to describe the noun involved in the main verb through its previous actions. These constructions exhibit semantics of sequentiality in relation to the main verb and convey a sense of finiteness in the action. The adjectival participle is also inflected with tense, aspect, and modality, indicating its event-like nature.

   Ka-En: avnu *oduva* shoes annu wear madida

   Translation: He wore his *running* shoes

2. **Adverbial Participle Construction:** Similarly, adverbial participle forms in Kannada are used to represent verbs performed by or associated with a noun in the dative or accusative case. Unlike adjectival constructions, there is no direct sequentiality associated with the main verb and the adverbial participle.

   Ka-En: *malkondiruva* deer annu Rakesh shoot maadida

   Translation: Rakesh shot the *sleeping* deer

3. **Infinitives:** In Kannada, infinitive verbs are identified by the characteristic inflective ending of 'lu'. These infinitive forms of the verb are also considered as events in linguistic annotation.

   Ka-En: Samiksha eega computer alli *oodalu* hoguttale

   Translation: Samiksha will now *go read* on computer

4. **Subjunctives:** The subjunctive is an uncommon verb form that expresses desires or imagined situations. It is used to indicate events that are uncertain or not guaranteed to happen. As a result, subjunctive verbs are also labeled as events in linguistic annotation. Subjunctives can undergo morphological inflections for tense, aspect, and modality, allowing for the expression of different temporal and modal nuances within the desired or imagined events.

   Ka-En: ninage olle aarogya irali anta *bayasutteene*

   Translation: *I wish* you good health

Please note that the structure of the sentences was changed in some of the examples above for translation. This is because of the difference in grammatical structure between Kannada and English. The sentences where the structure was changed in the translation were originally following the Kannada language structure while swapping some Kannada words with English words.

As stated, we will be using the Inside-Outside-Beginning (IOB) format, so the total number of tags becomes 3 - B, I, and O.

The following are some examples of Kannada-English code-mixed sentences after annotation-

Ka-En 1: avanu/O Mysore/O alliro/O international/O school/O ge/O hoogi/B science/O odustaane/B

Translation: He *goes* to the international school in Mysore and *teaches* science

Ka-En 2: Naanu/O Sudeep/O sir/O movie/O naanu/O 2/O times/O nodidini/B

Translation: I have *watched* Sudeep sir movie 2 times

4.3.3 Dealing with Ambiguous Tokens

As we saw in the previous Chapter 3 in Section 3.3.1, the ambiguities are a part of the code-mixed data. We have also discussed in Section 1.3 and Section 1.3.1 that such ambiguity is one of the challenges in working with Code-Mixed social media data.

We shall deal with them now in the same way we dealt with them earlier. In case a word has different means in the two languages we are working with and the words refers to an action/event in one language while it does not in the other language, we tag the token with whatever seems appropriate based on the context of the sentence.

The annotators shall make such context based decisions for any other ambiguity that they might come across as some words can have multiples meanings, in the same language or across languages.

4.4 Corpus and Statistics

4.4.1 Dataset

As we intend to use Part-of-Speech (POS) tags in our work, we have used a subset of the annotated dataset created by Appidi et. al. [3] for POS tagging of Kannada-English code-mixed data. The corpus was created from Twitter[1].

[1] http://twitter.com/

Quantity	Value
Total number of tokens	7,688
Avg. tweet length	9.61
Total tweets	800

Table 4.1: Corpus statistics

Event type	Count of occurances
Singe word events	695
Multi-word events	376

Table 4.2: Event types statistics

We annotated 800 of these code-mixed Kannada-English tweets. For language identification part, we used the tool that was developed by Irshad et. al. [13] but it needed manual checks as social media data differs from the standard language (Section 1.3.1.

The corpus has a total of 7,688 tokens which were tagged for the 3 tags mentioned in the Section 4.3. The corpus statistics and the event types statistics can be seen in Table 4.1 and Table 4.2 respectively.

4.4.2 Inter Annotator Agreement

Annotation of the dataset for event tags in the tweets was carried out by 2 human annotators having linguistic background and proficiency in both Kannada and English based on the methodology in Section 4.3. In order to validate the quality of annotation, we calculated the inter annotator agreement (IAA) between the 2 annotation sets of 800 code-mixed tweets having 7,688 tokens using Cohen's Kappa [20] which came up to 0.88 which is fairly high given the challenges we have with the task at hand, discussed in Section 4.3.3, and the complexity of the annotation guidelines.

Disagreements about the tags were resolved through discussions between the annotators to reach a mutual agreement.

4.5 Supervised Approaches for Event Detection

Supervised machine learning is a category of machine learning where a model is trained on labeled training data, where each data point has both input features and corresponding output labels. The goal of supervised learning is for the model to learn the mapping between input features and output labels so that it can make accurate predictions on unseen data.

As we have a limited amount of data for our task of event detection (800 sentences), we will only explore probabilistic models. They have been described in the following sub-sections.

4.5.1 Hidden Markov Model

Hidden Markov Models (HMMs), that was first introduced by Baum et. al. [11], have been used for event detection in NLP, particularly in scenarios where the focus is on sequential data [92] [43] [37]. HMMs are probabilistic models that involve hidden states and observable outputs, making them suitable for modeling the sequential nature of the text.

The HMM is an extension of the Markov chain, a model that provides information about the probabilities of sequences of random variables called states, where each state can have values from a specific set. Hidden Markov Models (HMMs) are finite stochastic automata consisting of two stochastic processes. The first process is a Markov chain with transition probabilities and hidden states. The second process generates observable emissions based on a state-dependent probability distribution. In our context, the emission probability refers to the likelihood of a token being assigned a B, I, or O tag.

In figure 4.1, we see an example of probabilistic parameters of a Hidden Markov Model, where

- X — states
- y — possible observations
- a — state transition probabilities
- b — output probabilities

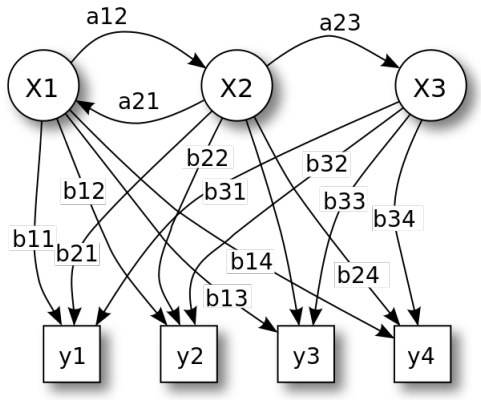

Figure 4.1: An example of Probabilistic parameters of a Hidden Markov Model (HMM)

Here's an overview of how HMMs can be applied to event detection in NLP:

1. **Define states:** In the context of event detection, states represent different event classes or categories. These states are hidden, meaning they are not directly observable from the input text.

2. **Define observations:** Observations are the words or features that are observable in the input text. Each observation is associated with a specific state or event class.

3. **Training:** The HMM is trained using labeled data, where the input text is annotated with event labels. The HMM learns the transition probabilities between states and the emission probabilities of observations given the states.

4. **Viterbi algorithm:** After training, the Viterbi algorithm is applied to the HMM. Given a new input text, the Viterbi algorithm calculates the most likely sequence of hidden states (event classes) that generated the observed words.

5. **Event detection:** Once the most likely sequence of hidden states is obtained, events can be extracted from the text by identifying segments where the hidden state corresponds to an event class.

It's important to note that HMMs make the simplifying assumption of the Markov property, which assumes that the current hidden state depends only on the previous state. While this assumption may not always hold in complex NLP tasks, HMMs can still be effective for event detection in certain scenarios.

Overall, HMMs provide a framework for modeling sequential data and can be utilized for event detection in NLP when the focus is on capturing the sequential nature of events in text.

4.5.2 Conditional Random Fields

The Conditional Random Fields have been explained in detail in the previous chapter (Chapter 3 Section 3.5.2), as this is a sequence labeling task as well, like NER.

CRF is effective for event detection and sequence labeling tasks because it captures dependencies between adjacent labels, considering the contextual information from neighboring words. It enables global optimization by finding the most likely label sequence that maximizes the joint probability, leading to more coherent and accurate predictions. CRF's ability to model the structure of the sequence enhances its performance in identifying event segments within text.

4.6 Feature Selection

In "A Semantico-Syntactic Approach to Event-Mention Detection and Extraction In Hindi", by Goud et. al. [26], they have used specific features for HMMs and some additional features for CRF. We shall use those features along with an additional feature - language identifier (LID) which will either be Kannada (Ka), English (En), Named Entity (NE) or a Universal (Univ) token.

The list of features picked for the HMM in Goud et. al. [26] are:

1. Word Identity (WI) : This would be the annotated event tag of the token.

2. Part-of-Speech (POS) : This would be the relevant POS tag of the token.

3. Beginning Of Sentence (BOS) : This would be a binary to mark if a token is the first token of a sentence.

4. Capitalization (C) : This is to identify if the token is capitalisized as capitalisation signifies nouns most of the time if not emphasis.

5. Language Identifier (LID) : This is binary feature that is important for code-mixed data as we need to know which language it belongs to - Kannada or English.

The list of features picked for CRF are the following:

1. Word Identity (WI) : This would be the annotated event tag of the token.

2. Part-of-Speech (POS) : This would be the relevant POS tag of the token.

3. Bi-gram features : Adjacent 2 token feature.

4. Tri-gram features : Adjacent 3 token feature.

5. Beginning Of Sentence (BOS) : This would be a binary to mark if a token is the first token of a sentence.

6. Previous word's POS : This feature would help in context understanding of the present token.

7. Previous word's WI : If the previous token's tag was B, then the present token would either be a I or an O tag. This helps in contextual understanding for a CRF.

8. Next word's POS : Similar to previous token's POS tag, next token's POS tag helps understand the present token better.

9. Next word's WI : Similar to previous token's event tag, next token's event tag helps understand the present token better.

10. Language Identifier (LID) : This is binary feature that is important for code-mixed data as we need to know which language it belongs to - Kannada or English.

4.7 Experimental Setup

In this section, we conducted a series of experiments to assess the impact of different features and model parameters. Our goal was to understand the effect of individual features that we discussed in Section 4.6 and explore the influence of various model settings that we discussed in the Section 4.5.

To achieve this, we performed experiments using different combinations of features and systems with rigorous hyper-paparemeter tuning, for both HMM and CRF using GridSearch. We experimented with using a subset of features together and all features simultaneously.

To evaluate the performance of our classification models, we employed 5-fold cross-validation. This approach helped us validate the models by partitioning the data into five subsets, training the models on four subsets, and evaluating their performance on the remaining subset. We repeated this process five times, each time using a different subset for evaluation.

For the implementation of the algorithms, we utilized the data handling libraries in Python for HMM and CRF++[2] tool for CRF, which are efficient and user-friendly tools for our tasks.

In terms of data splitting, we allocated 60% of the data for training, 10% for validation, and 30% for testing. This division allowed us to train our models on a substantial portion of the data, validate their performance on a separate subset, and finally assess their generalization ability on a dedicated testing set.

By conducting these experiments, we aimed to gain insights into the impact of different features and model parameters, enabling us to make informed decisions about the best configuration for our classification models. The following are the model parameters that worked best for us-

We use the same evaluation metrics as those explained in the previous chapter (Chapter 3 Section 3.8).

[2]https://taku910.github.io/crfpp/

Model	Precision	Recall	F-1 score
HMM	0.38	0.42	0.39
CRF	0.46	0.63	0.53

Table 4.3: Evaluation of HMM and CRF models for Event Detection

4.8 Results and Analysis

Table 4.3 captures the performance of our models for our dataset. Our best model is the CRF (window size 3) which achieved a weighted average F1-score of 0.53 compared to 0.39 for HMM. The performance of the HMM is in line with expectations. This limitation can be attributed to their ineffectiveness in capturing contextual details and their reliance on a restricted set of features during training, resulting in sparse information availability.

As CRFs are trained to develop a local model at the sentence level for event identification, as we observe improvements in the accuracy of the CRF's local model, it indicates that the engineered features effectively capture all the available information within the sentence's local structure.

Even then, the results are nowhere near perfect but the purpose of the models was just to provide an exploratory baseline for the dataset created with the annotation guidelines.

We should also note that the grammatical structure of the sentences are not standard, making prediction harder. This gets more difficult with Kannada-English code-mixed data as mixing happens at word-level, mostly for Kannada language prepositions and named entities or English language nouns (Section 1.3).

4.9 Conclusion

In conclusion, our work on Event Detection for code-mixed Kannada-English social media data has yielded the following:

1. **Annotation Guidelines:** We have provided guidelines for annotating code-mixed Kanglish social media data for event detection.

2. **Annotated Corpus:** We have created a new annotated corpus specifically for code-mixed Kannada-English Event Detection. To the best of our knowledge, this is the first corpus of its kind, providing valuable resources for future research in this domain.

3. **Research Problem:** We have identified and addressed the challenge of event detection in code-mixed Kannada-English data as a research problem. Code-mixed data presents unique linguistic complexities, and our work contributes to advancing event detection and other information extraction techniques in this particular context.

4. **Machine Learning Models:** We conducted experiments using probabilistic machine learning models on our annotated corpus. Specifically, we employed Hidden Markov Model and Conditional Random Fields (CRF) models which could be employed as baseline models for further exploration.

By creating the annotated corpus, addressing the research problem, and achieving strong performance with the machine learning models, the work done as part of this thesis has made contributions to the field of event detection in code-mixed Kannada-English data. Our work lays the foundation for further advancements and opens avenues for future research in this emerging domain.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

Code-switching or code-mixing occurs when lexical items and/or grammatical features from two languages appear in one sentence. With the rising popularity of social media platforms such as Twitter, Facebook, and Reddit, the volume of texts on these platforms has also grown significantly. Twitter alone has over 500 million text posts (tweets) per day. India, a country with over 300 million multilingual speakers, has over 23 million users on Twitter as of January 2022, and code-switching can be observed heavily on this social media platform. With over 58 million Kannada languages speakers (L1 and L2), it is important that we understand them on social media and in other informal settings where they are likely to use code-mixed language.

We have tried to work on information extraction on Kanglish social media data by working on problems of named entity recognition and event detection. Named Entity Recognition (NER) and Event Detection on social media code-mixed data are crucial for understanding and analyzing user-generated content in diverse languages. They enable the identification of named entities, such as names of people, organizations, and locations, as well as the detection of events, allowing for deeper insights into multilingual conversations, cross-cultural trends, and effective information retrieval from code-mixed social media posts. Towards these efforts of NER and event detection, we have created annotated datasets for this low-resource code-mixed language for each of these tasks and provided detailed annotation

guidelines for the same to overcome some of the challenges specific to Kanglish code-mixing and social media noisy data.

We have collected Kanglish code-mixed data from social media, Twitter, and curated annotated datasets for both NER and Event Detection tasks. We have analysed the challenges that are unique to Kannada-English code-mixed data and have provided annotation guidelines for the same. We have also proposed a few supervised approaches towards these tasks on our dataset with careful feature selection and critically analysed the results in hopes to promote more focus from the research community on such low-resource languages. With these datasets, we have thoroughly analysed these datasets by providing some baselines with supervised machine learning approaches with careful feature selection and rigorous experimentation.

One of the main limitations in our work looks to be the size of the annotated datasets. For neural approaches, we need larger annotated datasets but the process of annotation is very time-consuming/expensive. Data augmentation can be explored next to deal with this challenge. By increasing the size of the dataset, we also reduce any bias that tends to happen with smaller datasets. We should explore possible ways of cross-lingual transfer learning, synthetic data generation, or explore multilingual learning as more Indian code-mixed language pairs are gaining interest from the research community, especially in the languages that are grammatically closer to Kannada such as Tamil, Telugu, Konkani, and Malayalam. As we increase the size of the dataset, it would be interesting to see how some of the state of the neural-network based approaches perform on these tasks. More importantly, we would also need active engagement within the research community to share findings and participate in related conferences and workshops which would help in gaining insights and receiving feedback. We hope that the work done as part of this thesis increases interest and promotes further work in this domain.

Related Publications

- **Sumukh S** and Manish Shrivastava. 2022. "Kanglish alli names!" Named Entity Recognition for Kannada-English Code-Mixed Social Media Data. *In Proceedings of the Eighth Workshop on Noisy User-generated Text (W-NUT 2022)*, pages 154–161, Gyeongju, Republic of Korea. International Conference on Computational Linguistics (COLING), Oct. 2022. [69]

- Suhan Prabhu, Ujwal Narayan, Alok Debnath, **Sumukh S**, and Manish Shrivastava. 2020. Detection and Annotation of Events in Kannada. *In 16th Joint ACL - ISO Workshop on Interoperable Semantic Annotation PROCEEDINGS*, pages 88–93, Marseille, France. Language Resources and Evaluation Conference (LREC), June 2020. [60]

- **Sumukh S** and Manish Shrivastava. 2023. Event Annotation and Detection in Kannada-English Code-Mixed Social Media Data. *In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2023)*, pages 1007-1014, Varna, Bulgaria. [68]

Bibliography

[1] G. Aguilar, F. AlGhamdi, V. Soto, M. Diab, J. Hirschberg, and T. Solorio. Named entity recognition on code-switched data: Overview of the calcs 2018 shared task. *arXiv preprint arXiv:1906.04138*, 2019.

[2] M. A.-E. N. Al-Qaysi. Codeswitching usage in social media: A case study from oman. *International Journal of Information Technology and Language Studies*, 16:27–46, 2017.

[3] A. R. Appidi, V. K. Srirangam, D. Suhas, and M. Shrivastava. Creation of corpus and analysis in code-mixed Kannada-English social media data for POS tagging. In *Proceedings of the 17th International Conference on Natural Language Processing (ICON)*, pages 101–107, Indian Institute of Technology Patna, Patna, India, Dec. 2020. NLP Association of India (NLPAI).

[4] A. R. Appidi, V. K. Srirangam, D. Suhas, and M. Shrivastava. Creation of corpus and analysis in code-mixed Kannada-English Twitter data for emotion prediction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6703–6709, Barcelona, Spain (Online), Dec. 2020. International Committee on Computational Linguistics.

[5] B. Arnulphy. A weighted lexicon of french event names. In *Proceedings of the Second Student Research Workshop associated with RANLP 2011*, pages 9–16, 2011.

[6] B. Arnulphy, V. Claveau, X. Tannier, and A. Vilnat. Supervised machine learning techniques to detect timeml events in french and english. In *International Conference on Applications of Natural Language to Information Systems*, pages 19–32. Springer, 2015.

[7] B. Arnulphy, X. Tannier, and A. Vilnat. Automatically generated noun lexicons for event extraction. In *Volume 7182*, March 2012.

[8] S. Ashwini and J. D. Choi. Targetable named entity recognition in social media. *ArXiv*, abs/1408.0782, 2014.

[9] K. Bali, J. Sharma, M. Choudhury, and Y. Vyas. "I am borrowing ya mixing ?" an analysis of English-Hindi code mixing in Facebook. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 116–126, Doha, Qatar, Oct. 2014. Association for Computational Linguistics.

[10] 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.

[11] L. E. Baum and T. Petrie. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *The Annals of Mathematical Statistics*, 37(6):1554 – 1563, 1966.

[12] R. Bhargava, B. V. Tadikonda, and Y. Sharma. Named entity recognition for code mixing in indian languages using hybrid approach. In *FIRE*, 2016.

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

[14] S. N. Bhattu, N. S. Krishna, and D. V. L. N. Somayajulu. Event extraction from social media text using conditional random fields. In *Fire*, 2017.

[15] A. Bittar, P. Amsili, P. Denis, and L. Danlos. French timebank: An iso-timeml annotated reference corpus. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 130–134. Association for Computational Linguistics, 2011.

[16] S. S. Burramsetty, N. P. Gonugunta, S. Uppu, and S. K. Nunna. Event extraction from telugu-english code mixed social media text. In *2022 7th International Conference on Communication and Electronics Systems (ICCES)*, pages 946–951, 2022.

[17] P. Capet, T. Delavallade, T. Nakamura, A. Sandor, C. Tarsitano, and S. Voyatzi. A risk assessment system with automatic extraction of event types. In Z. Shi, E. Mercier-Laurent, and D. Leake, editors, *Intelligent Information Processing IV*, pages 220–229, Boston, MA, 2008. Springer US.

[18] W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. In *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, 1994.

[19] P. Cimiano and S. Staab. Learning by googling. *SIGKDD Explor. Newsl.*, 6(2):24–33, dec 2004.

[20] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37 – 46, 1960.

[21] J. L. Copara Zea, J. E. Ochoa Luna, C. Thorne, and G. Glavaš. Spanish NER with word representations and conditional random fields. In *Proceedings of the Sixth Named Entity Workshop*, pages 34–40, Berlin, Germany, Aug. 2016. Association for Computational Linguistics.

[22] N. Dhanaji. Code-mixing as a need of post-modern indian society. *Pune Research*, December 2015.

[23] G. R. Doddington, A. Mitchell, M. A. Przybocki, L. A. Ramshaw, S. Strassel, and R. M. Weischedel. The automatic content extraction (ace) program-tasks, data, and evaluation. *Linguistic Data Consortium, Philadelphia*, 2004.

[24] B. Fetahu, A. Fang, O. Rokhlenko, and S. Malmasi. Gazetteer enhanced named entity recognition for code-mixed web queries. In *SIGIR 2021*, 2021.

[25] J. R. Finkel, T. Grenager, and C. D. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting of the association for computational linguistics (ACL'05)*, pages 363–370, 2005.

[26] J. S. Goud, P. Goel, A. Debnath, S. Prabhu, and M. Shrivastava. A semantico-syntactic approach to event-mention detection and extraction in hindi.

[27] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks : the official journal of the International Neural Network Society*, 18 5-6:602–10, 2005.

[28] A. Hande, R. Priyadharshini, and B. R. Chakravarthi. Kancmd: Kannada codemixed dataset for sentiment analysis and offensive language detection. In *Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, 2020.

[29] M. Hegde, D. Alva, S. G. Oommen, and S. Bhat. Discourse functions of code-switching among normal kannada-english and malayalam-english bilinguals—a pilot study. *Asia Pacific Journal of Speech, Language and Hearing*, 14(4):211–216, 2011.

[30] A. F. Hidayatullah, A. Qazi, D. T. C. Lai, and R. A. Apong. A systematic review on language identification of code-mixed text: Techniques, data availability, challenges, and framework development. *IEEE Access*, 10:122812–122831, 2022.

[31] T. K. Ho. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pages 278–282 vol.1, 1995.

[32] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[33] A. H. Hossny and L. Mitchell. Event detection in twitter: A keyword volume approach. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1200–1208, 2018.

[34] Z. Huang, W. Xu, and K. Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[35] S. Huffman. Acquaintance: Language-independent document categorization by n-grams. In *TREC*, 1995.

[36] F. Jahangir, W. Anwar, U. I. Bajwa, and X. Wang. N-gram and gazetteer list based named entity recognition for Urdu: A scarce resourced language. In *Proceedings of the 10th Workshop on Asian Language Resources*, pages 95–104, Mumbai, India, Dec. 2012. The COLING 2012 Organizing Committee.

[37] S. Jiampojamarn, G. Kondrak, and T. Sherif. Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. In *Human Language Technologies 2007: The Conference of the North*

American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference, pages 372–379, 2007.

[38] M. I. Jordan. Serial order: a parallel distributed processing approach. technical report, june 1985-march 1986. 5 1986.

[39] A. Joshi, A. Prabhu, M. Shrivastava, and V. Varma. Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2482–2491, 2016.

[40] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi. Traffic monitoring and accident detection at intersections. *IEEE Trans. Intell. Transp. Syst.*, 1(2):108–118, 2000.

[41] A. Klingler. Changes in code-switching behaviour among hindi-english bilinguals in northern india. *Lifespans and Styles*, 31(1):40–50, 2017.

[42] R. Kumar, A. N. Reganti, A. Bhatia, and T. Maheshwari. Aggression annotated corpus of hindi-english code-mixed data. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[43] J. Kupiec. Robust part-of-speech tagging using a hidden markov model. *Computer Speech & Language*, 6(3):225–242, 1992.

[44] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California, June 2016. Association for Computational Linguistics.

[45] Y. Li, L. N. Harfiya, K. Purwandari, and Y.-D. Lin. Real-time cuffless continuous blood pressure estimation using deep learning model. *Sensors*, 20, 09 2020.

[46] H. Llorens, E. Saquete, and B. Navarro. Tipsem (english and spanish): Evaluating crfs and semantic roles in tempeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 284–291. Association for Computational Linguistics, 2010.

[47] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels.

[48] P. Majumder, M. Mitra, and B. B. Chaudhuri. N-gram: a language independent approach to ir and nlp, 2002.

[49] A. McCallum and D. Freitag. A maximum entropy approach to named entity recognition. In *Proceedings of the 7th Conference on Natural Language Learning*, pages 289–292, 2000.

[50] S. Morwal, N. Jahan, and D. Chopra. Named entity recognition using hidden markov model (hmm). *International Journal on Natural Language Computing (IJNLC)*, 1(4):15–23, 2012.

[51] P. Muysken. *Bilingual speech*. 01 2000.

[52] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[53] D. Nguyen and A. S. Dogruoz. Word level language identification in online multilingual communication. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 857–862, 2013.

[54] M. Noort, E. Struys, P. Bosch, L. Jaswetz, B. Perriard, S. Yeo, P. Barisch, K. Vermeire, S.-H. Lee, and S. Lim. Does the bilingual advantage in cognitive control exist and if so, what are its modulating factors? a systematic review. *Behavioral Sciences*, 9:27, 03 2019.

[55] L. Padro and E. Stanilovsky. Freeling 3.0: Towards wider multilinguality. In *LREC2012*, 2012.

[56] R. Panchendrarajan and A. Amaresan. Bidirectional lstm-crf for named entity recognition. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*, 2018.

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

[58] A.-M. Popescu, M. Pennacchiotti, and D. Paranjpe. Extracting events and event descriptions from twitter. In *Proceedings of the 20th International Conference Companion on World Wide Web*, WWW '11, page 105–106, New York, NY, USA, 2011. Association for Computing Machinery.

[59] S. Prabhu, P. Goel, A. Debnath, and M. Shrivastava. Incorporating sub-word level information in language invariant neural event detection. In *Proceedings of the 16th International Conference on Natural Language Processing*, pages 36–44, International Institute of Information Technology, Hyderabad, India, Dec. 2019. NLP Association of India.

[60] S. Prabhu, U. Narayan, A. Debnath, S. S, and M. Shrivastava. Detection and annotation of events in Kannada. In *16th Joint ACL - ISO Workshop on Interoperable Semantic Annotation PROCEEDINGS*, pages 88–93, Marseille, May 2020. European Language Resources Association.

[61] K. C. Raghavi, M. K. Chinnakotla, and M. Shrivastava. Answer ka type kya he?: Learning to classify questions in code-mixed language. In *Proceedings of the 24th International Conference on World Wide Web*, pages 853–858. ACM, 2015.

[62] N. Ramakrishnan, P. Butler, S. Muthiah, N. Self, R. Khandpur, P. Saraf, W. Wang, J. Cadena, A. Vullikanti, G. Korkmaz, C. Kuhlman, A. Marathe, L. Zhao, T. Hua, F. Chen, C.-T. Lu, B. Huang, A. Srinivasan, K. Trinh, L. Getoor, G. Katz, A. Doyle, C. Ackermann, I. Zavorin, J. Ford, K. Summers, Y. Fayed, J. Arredondo, D. Gupta, and D. Mares. 'beating the news' with embers: Forecasting civil unrest using open source indicators, 2014.

[63] L. A. Ramshaw and M. P. Marcus. Text chunking using transformation-based learning. In *Natural Language Processing Using Very Large Corpora*, pages 157–176. Springer, 1999.

[64] S. Rijhwani, R. Sequiera, M. Choudhury, K. Bali, and C. S. Maddila. Estimating code-switching on Twitter with a novel generalized word-level language detection technique. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1971–1982, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[65] A. Ritter, Mausam, O. Etzioni, and S. Clark. Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, page 1104–1112, New York, NY, USA, 2012. Association for Computing Machinery.

[66] Y. Roh, G. Heo, and S. E. Whang. A survey on data collection for machine learning: A big data - ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1328–1347, 2021.

[67] K. Rudra, S. Rijhwani, R. Begum, K. Bali, M. Choudhury, and N. Ganguly. Understanding language preference for expression of opinion and sentiment: What do Hindi-English speakers do on Twitter? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1131–1141, Austin, Texas, Nov. 2016. Association for Computational Linguistics.

[68] S. S, A. Appidi, and M. Shrivastava. Event annotation and detection in Kannada-English code-mixed social media data. In R. Mitkov and G. Angelova, editors, *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*, pages 1007–1014, Varna, Bulgaria, Sept. 2023. INCOMA Ltd., Shoumen, Bulgaria.

[69] S. S and M. Shrivastava. "kanglish alli names!" named entity recognition for Kannada-English code-mixed social media data. In *Proceedings of the Eighth Workshop on Noisy User-generated Text (W-NUT 2022)*, pages 154–161, Gyeongju, Republic of Korea, Oct. 2022. Association for Computational Linguistics.

[70] P. Sailaja. Hinglish: code-switching in indian english. *ELT Journal*, 65(4):473–480, 2011.

[71] R. Sauri and T. Badia. Spanish timebank 1.0, 2012.

[72] R. Saurí, J. Moszkowicz, B. Knippen, R. Gaizauskas, A. Setzer, and J. Pustejovsky. Timeml annotation guidelines version 1.2.1. 01 2006.

[73] K. Shaffer. Language clustering for multilingual named entity recognition. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 40–45, Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics.

[74] A. Sharma, S. Gupta, R. Motlani, P. Bansal, M. Shrivastava, R. Mamidi, and D. M. Sharma. Shallow parsing pipeline - Hindi-English code-mixed social media text. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1340–1345, San Diego, California, June 2016. Association for Computational Linguistics.

[75] K. Singh, I. Sen, and P. Kumaraguru. Language identification and named entity recognition in Hinglish code mixed tweets. In *Proceedings of ACL 2018, Student Research Workshop*, pages 52–58, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[76] K. Singh, I. Sen, and P. Kumaraguru. A Twitter corpus for Hindi-English code mixed POS tagging. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 12–17, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[77] B. S. Sowmya Lakshmi and B. R. Shambhavi. An automatic language identification system for code-mixed english-kannada social media text. In *2017 2nd International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS)*, pages 1–5, 2017.

[78] V. K. Srirangam, A. A. Reddy, V. Singh, and M. Shrivastava. Corpus creation and analysis for named entity recognition in Telugu-English code-mixed social media data. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 183–189, Florence, Italy, July 2019. Association for Computational Linguistics.

[79] S. Tedeschi, V. Maiorca, N. Campolungo, F. Cecconi, and R. Navigli. WikiNEuRal: Combined neural and knowledge-based silver data creation for multilingual NER. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2521–2533, Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics.

[80] S. Thomas. Emergence, usage, and implications of hinglish in bollywood films. In *Proceedings of the 16th International Association of World Englishes Conference*, Vancouver, Canada, 2010.

[81] E. F. Tjong Kim Sang and F. De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003.

[82] E. F. Tjong Kim Sang and J. Veenstra. Representing text chunks. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 173–179, Bergen, Norway, June 1999. Association for Computational Linguistics.

[83] N. UzZaman, H. Llorens, L. Derczynski, J. Allen, M. Verhagen, and J. Pustejovsky. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9, 2013.

[84] R. Veerappan, P. Antony, S. Saravanan, and K. Soman. A rule based kannada morphological analyzer and generator using finite state transducer. *International Journal of Computer Applications*, 27(10):45–52, 2011.

[85] M. Verhagen, R. Gaizauskas, F. Schilder, M. Hepple, G. Katz, and J. Pustejovsky. SemEval-2007 task 15: TempEval temporal relation identification. In *Proceedings of the Fourth International Workshop on*

Semantic Evaluations (SemEval-2007), pages 75–80, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[86] M. Verhagen, R. Sauri, T. Caselli, and J. Pustejovsky. Semeval-2010 task 13: Tempeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 57–62, 2010.

[87] S. Verma. Code-switching: Hindi-english. *Lingua*, 38(2):153–165, Jan. 1976.

[88] P. von Däniken and M. Cieliebak. Transfer learning and sentence level features for named entity recognition on tweets. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 166–171, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics.

[89] 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.

[90] 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.

[91] H. Ye, N. Zhang, H. Chen, and H. Chen. Generative knowledge graph construction: A review. *CoRR*, abs/2210.12714, 2022.

[92] G. Zhou and J. Su. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 473–480, 2002.

[93] K. A. H. Zirker. Intrasentential vs. intersentential code switching in early and late bilinguals. pages 114–133, 2007.