Microblog Processing : Summarization and Impoliteness Detection

by

SANDIP JAYANTILAL MODHA
201221001

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY
to

DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

June, 2019

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Doctor of Philosophy at Dhirubhai Ambani Institute of Information and Communication Technology and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

SANDIP JAYANTILAL MODHA

Certificate

This is to certify that the thesis work entitled MICROBLOG PROCESSING : SUMMARIZATION AND IMPOLITENESS DETECTION has been carried out by SANDIP JAYANTILAL MODHA for the degree of Doctor of Philosophy at *Dhirubhai Ambani Institute of Information and Communication Technology* under my supervision.

Prasenjit Majumder
Thesis Supervisor

Acknowledgments

It is a matter of immense contentment to offer the words of gratitude in the honor of my mentor and advisor Prof. Prasenjit Majumder. His incessant guidance and critical annotations from conceptualization to realization of my doctoral work have been eternally engraved in my well-structured research. I take the privilege to thank Prof. Suman Mitra, Prof. Maniklal Das and Prof.Bhaskar Chaudhry who bejeweled their place in my thesis committee. Their erudite remarks as well as prudent queries and my sincere endeavors to incorporate them have made my work more scalable. I would also like to express my sincere gratitude to my thesis reviewers Prof.Dr. Christa Womser-Hacker and Dr. Namita Mittal for their insightful reviews of my thesis. I am thankful to Prof. Thomas Mandl for mentoring me during my stay at Hildesheim, Germany and valuable feedback throughout my Ph.D. journey.

Throughout my itinerary of doctoral work, I have been comprehensively assisted and even taught by the pretty number of associates of IRLAB. I thank all of them from the bottom of my heart. My stopover at DAIICT during my work has bestowed upon me with prolific acquaintances. I feel delighted to tender the words of appreciation in the respect of all the faculty members and my fellow Ph.D. scholars for their direct and indirect help. I thank Mr. Chintak Mandlia, Daksh Patel, Risab Singla, Surupendu Gangopadhyay, Apurva Parikh, and of course, my senior colleague Mr. Parth Mehta for their stimulating conversations and altercations on various aspects of my doctoral work.

I feel a deep sense of gratitude for my parents for their emotional support and blessings while I was busy with this doctoral work. Behind every successful man, there is a woman: so goes a popular saying. My wife Khushbu has proved this time and again.

Above all, I offer my pranam in the lotus feet of Almighty for all his incessant blessings He has showered on me throughout my life, and I pray to him for his perpetual blessings.

Contents

Abstract

Social Media is an excellent source for studying human interaction and behavior. Sensing social media such as Facebook and Twitter, by the smart autonomous application empower its user community with real-time information unfolds across different part of the world. In this thesis, we study social media text from the summarization and impoliteness perspective.

In the first part of the thesis, Microblog Summarization is explored from the three scenarios. In the first scenario, we present a summarization system, built over the Twitter stream, to summarize the topic for a given duration. Daily summary or digest from Microblog is a way to update social media users what happened today on the subject of her interest. To design a Microblog based summarization system, Tweet ranking is the primary task. After ranking tweets, relevant tweet selection is the crucial task for any summarization system due to the massive volume of tweets in the Twitter stream. In addition, the Summarization system should include novel tweets in the summary or digest. The measure of relevance is typically the similarity score obtained from different text similarity (between user information need and tweets) algorithms. More similar, the higher the score. So, we need to choose a threshold that can minimize false-positive judgments in this case. We have developed various methods by exploiting statistical features of the rank list to estimate these thresholds and evaluated against thresholds determined via grid search. We have used language models to rank the tweets to select relevant tweets where the selection of the smoothing technique and its parameters are critical. Results are also compared with the standard probabilistic ranking algorithm BM25. Learning to Rank strategies are also implemented, which show substantial improvement in some of the result metrics. In the second scenario: we develop a real-time version of the summarization system that continually monitors the Twitter stream and generates relevant and novel real-time push notifications that are delivered to users' cellphones. In the third scenario, the summarization system was evaluated on a disaster-related incident such as an earthquake. We have also performed comprehensive failure analysis on our experiment and identified key issues that can be addressed in the future.

In the second part of the thesis, the social media stream is studied from the impoliteness perspective. Due to an exponential rise in the social media user-base, incidents like Hate speech, trolling, cyberbullying are also increasing, and that has lead to Hate Speech detection problems being reshaped into different research problems such as aggression detection, offensive language detection, factual-post detection. We refer to all such anti-social typology under the ambit of impoliteness. This thesis attempts to study the effectiveness of different text representation schemes on an NLP downstream task such as classification. A set of text representation schemes, based on Bag-of-Word techniques, distributed word representation or word embedding, sentence embedding, are empirically evaluated on traditional classifiers and deep neural models. Experiment results show that on the English dataset, overall, text representation using Googles' universal sentence encoder (USE) performs better than word embedding, and BoW techniques on traditional classifiers such as SVM, while pre-trained word embedding models perform better on classifiers based on the deep neural models. Recent pre-trained transfer learning models like Elmo, ULMFiT, and BERT are fine-tuned for the aggression classification task. However, results are not at par with the pre-trained word embedding model. Overall, word embedding using fastText produces best weighted F_1-score than Word2Vec and Glove. On the Hindi dataset, BoW techniques perform better than word embedding on traditional classifiers such as SVM, while pre-trained word embedding models perform better on classifiers based on the deep neural nets. Statistical significance tests are employed to ensure the significance of the classification results. In the case of lexically different test datasets, other than training dataset, deep neural models are more robust and perform substantially better than traditional classifiers such as SVM, logistic regression, and Naive Bayes classifiers. During the disaster-related incident, Twitter is flooded with millions of posts. In such emergencies, identification of factual posts is vital for organizations involved in the relief operation. We approach this problem as a combination of classification and ranking problems.

Following from this work, the aggression visualization problem is addressed as the last component. We have designed a user interface based on web browser plugins over Facebook and Twitter to visualize the aggressive comments posted by any user. This plugin interface might help the security agencies to keep a tab on the social media stream. The proposed plugin help celebrities to customize their timeline by raising the appropriate flag, which enables them to delete or hide such abusive comments from their timeline. In addition to these, the interface might be helpful to the research community to prepare weakly labeled training data in a few minutes using comments posted by users on celebrity's social media timeline.

List of Tables

List of Figures

CHAPTER 1

Introduction

Social networking websites such as Twitter[1], Facebook [2], Instagram [3] have a significant impact on the social life of the human being. Social Media become the dominant media for the ordinary person for information, social connection, networking, etc. As per the survey conducted by a reputable marketing agency Medikix [4], the common user will spend a substantial amount of its lifetime on social media.

The term Big Data describes three dimensions: volume, velocity, and variety. As per the live statistic reported by website Internetlivestats.com [5] on 23/5/2019, in one second, approximately 8500 tweets are posted on Twitter, 916 photos are uploaded on Instagram, 1533 messages are posted Tumbler. If we project these numbers day-wise, The total no posts are around hundreds of millions. Looking at these numbers, it is quite evident that Microblog like, Twitter, Facebook adhere to the Big Data characteristics. Social media empowered ordinary people to act as a sensor to the world and can disseminate information on a real-time basis. The concept of "social sensors" was introduced by [137, 105]. The assumption made is that each online social user regarded as a sensor and each message as sensory information.

Social Media is an excellent source of studying human interaction and behavior. This thesis focus on two essential problems related to Microblog: (i) Summarization (ii)Impoliteness Detection and Visualization. We have chosen Facebook and Twitter for the experiment.

[1]https://twitter.com
[2]https://Facebook.com
[3]https://instagram.com
[4]http://mediakix.com/how-much-time-is-spent-on-social-media-lifetime/#gs.dp8a6r
[5]https://www.internetlivestats.com/one-second/

1.1 Microblog Summarization

Microblog is a social web where the user can post a short message to disseminate any information or opinion to the community. We have chosen Twitter as Microblog for the experiment. Henceforth, Twitter and Microblog are used interchangeably in the rest of the thesis. The Microblog summarization research area has been inspired by the work carried out in the field of multiple document summarization. Each tweet or post can be analogous to the one document. Massive data volume and velocity are the severe challenges posed by Twitter, which differentiates Microblog summarization from the traditional multiple document summarization. There are mostly two methods for the summarization: abstractive summarization and extractive summarization [94]. This thesis will focus on the extractive real-time summarization (RTS) of the Microblog; There are two cases for real-time summarization from the Microblog [64].

- Email Digest from Microlog
- Real-time Push Notification from Microblog

In addition to these, we study a domain-specific case of summarization: Summarizing Microblog during the disaster.

1.1.1 Email Digest

Twitter is a real-time platform where thousand of topics discussed by the many users across the globe. Many of the issues, such as the refugee crisis in Europe, conflict in middle-east are not covered by print, electronic, or television media daily. Social media users might be interested in getting remain informed about the new developments about these topics regularly. The main objective of this research task is to generate a summary from the Microblog against users' information needs, which is encapsulated as an Interest Profile similar to the topic. At the end of the day, the user might be interested in receiving an email digest which essentially summarizes what happened on that day. Users will expect relevant and novel tweets in summary, while timeliness is not the issue for this research task.

Formal Problem Statement

From the set of topics or interest profiles Q=$\{Q_1, ..Q_n\}$, and set of tweets T=$\{t_1, .., t_n\}$, summarization system is required to generate summary S=$\{s_1, s_2, ..., s_n\}$ from set of relevant tweets RT=$\{rt_1, rt_2...rt_n\}$, where rt_i represents a relevant tweet for a particular Interest profile and RT $\subset$ T. Digest include top n ranked tweets for the given interest profiles for each day. Tweets included in the digest must ensure novelty, i.e., for any two tweets in the digest must have a similarity of less than the specified threshold sim$(rt_1, rt_2) < T_n$. At a high level, there are two basic tasks.

- Tweet Selection,i.e., selection of most representative tweet from the dataset
- Ensure novelty across the tweets

TREC microblog 2015 [62], TREC RTS 2016, 2017 [64, 63] were considered for the experiments.

1.1.2 Real Time Push Notification

Push notification is the real-time case of Microblog summarization. Consider a situation where a user is interested in receiving the latest development about his favorite topic via push notification over his phone. In this case, the system continuously monitors Twitter feed using the streaming API of Twitter. The aim of the system is how fast it can deliver relevant tweets to users' cellphone using push notification. Therefore, timeliness or latency is a crucial issue as opposed to the email digest for this research task.

Formal Problem Statement

The above problem is mainly a real-time filtering task. Given an Interest Profile Q=$\{Q_1, ..Q_n\}$, and the stream of tweets T=$\{t_1, t_2, ..t_n\}$ from public sample stream, the system need to calculate relevance or similarity score between tweets and interest profile

$$R\_score = f(Q, T)$$

Tweets having similarity greater than threshold with respect to the interest profiles moved in the set RT= $\{rt_1, rt_2, ...rt_n\}$. At most, n novel tweets can be pushed to the user mobile phone using push notification in a day. TREC Real Time summarization TREC RTS 2016, 2017 [64, 63] were considered for the experiments.

1.1.3 Summarizing Microblog during Disaster

Many incidents in the past have proved that social media is the first medium through which event related to a disaster like earthquakes reach to the people. Recently, many earthquake incidents have been reported first on Twitter before other media[105]. Twitter can be effectively accessed by an NGO/Government agency to assess the ground reality of the disaster area to assist in their rescue operations. The motivation behind this work is to explore IR methodologies that can be used to extract meaningful information from social media during emergency events. We have reported our system results on SMERP dataset [13] which was created during the first two days after the earthquake shook Italy in August 2016.

1.2 Tracking Impoliteness across Social media

Online Hate speech or inflammatory speech online causes violence and spread hatred across the many parts of the world. Often, the fringe elements of society exploit the right of free speech to spread violence through social media. Discrimination between Hate speech and free speech is a critical issue that needs to be addressed by society. Social scientists and psychologists have confirmed that social media posts and other online speech can inspire violence. In this thesis, We meticulously study Hate speech detection and the related problem like offensive content identification, cyberbullying, aggression Detection, etc. and they are subsumed under the broader phrase impoliteness detection. Popular Website thefreedictionary.com defines impoliteness as follows: a discourteous manner that ignores accepted social usage. In this research work, we study textual aggression, offensive language, and factual contents from social media under the ambit of impoliteness detection.

1.2.1 User aggression Detection

Unfortunately, Social media become the dominant platform to spread online Hate speech across billions of people due to its popularity and easy access. Events such as abuse, trolling, and bullying is happening every day. Online discussion on controversial topics makes people more aggressive on social media. Verbal aggression could be understood as any linguistic behavior which intends to damage the social identity of the target person and lower their status and prestige [52, 27]. At the coarse-grained level, aggression is classified into three labels, namely- ‘Non-aggressive‘ NAG), ‘Covertly Aggressive‘ (CAG), and Overtly Aggressive (OAG). We have explored various methods based on supervised learning to tackle the classification problem. We have focused on multiple text representation techniques based on Bag-of-words(BoW), distributed word representation techniques such as Word2vec[77], Glove [91], fastText [15], sentence embedding techniques like Doc2vec [55], and Contextual pre-trained language model such as ELMOs[92], ULMFiT [49], BERT [31]. The statistical significance tests are performed to ensure the significance of the results. Empirically, we found that pre-trained fastText is the better scheme among all.

1.2.2 Offensive Content Detection

The exponential rise in social media user-base backed by the cutting edge mobile data technologies leads to the inorganic growth in the posts related to hate speech or offensive content. The objective of the research problem by the OffensEval forum [134] is to identify offensive content at multi-level. The research task is divided into three levels. In the first level, the system is required to check whether social media posts contain any offensive or profane content or not, in the second level, offensive tweets are further required to be categorized into two labels, namely: targeted (TIN)-post which contain threat/insult to the targeted entity and untargeted (UNT), respectively. In the third level, targeted posts are further classified into the individual, group, or other categories.

1.2.3 Factual Post Detection

Social Media, specifically Microblog, has proved its importance during the disaster-related incidents like an earthquake, hurricane, and floods [6]. Organizations involved in relief operations actively track posts related to situational information posted on Facebook and Twitter during the disaster. However, At the same time, social media is flooded with lots of prayer and condolence messages. Posts that contain factual information are extremely important for the organization involved in post-disaster relief operations for coordination. Filtering and ranking of the posts containing factual information will be very useful to them. We believe that this is the special problem of the Sentiment analysis/Hate speech task. We consider this problem as a combination of two-class classification problems: factual posts and nob-factual posts plus ranking.

1.3 Impoliteness Visualization

Visualization provides in-depth insight and uncovers the hidden pattern from the massive dataset. In the last part of the thesis, we discuss the different ways to visualize aggression live on Facebook and Twitter. We have deployed our deep learning model over the internet, which filters the aggressive content from the social media and web browser plugin raised appropriate flag based upon the type of aggression predicted by the model. We have developed two interfaces for the visualization (i) Plugin: which run inside the Chrome browser and raised the flag as soon as it model detect aggressive contents (ii) WEB UI[7]: an interactive interface where user can input the text and check the aggression level predicted by the model. .

1.4 Thesis Contributions

In this section, we provided a brief overview of the main contributions of this thesis in the area of Microblog Summarization and Impoliteness Detection.

[6]https://phys.org/news/2018-08-social-media-bad-disaster-zones.html

[7]http://3.16.1.236:8000/

1.4.1 Microblog Summarization

- Tweet ranking and selection is the primary task to summarize event using tweet. The measure of relevance is typically the similarity score obtained from different text similarity (between user information need and tweets) algorithms. More similar, the higher the score. So we need to choose a threshold that can minimize false-positive judgments in this case.**We propose a Summarization algorithm and threshold estimation technique to estimate the silent day threshold T_s and a relevance threshold T_r for the tweet selection from the rank-list.**

- We have performed a comprehensive failure analysis of the summarization system on TREC RTS 2016 and 2017 datasets.

1.4.2 Impoliteness Detection and Visualization

- We have proposed a new multilingual corpus for the Online Hate speech and offensive contents in English, Hindi and German Language. The proposed corpus is sampled from Twitter and Facebook, by and large, on similar topics for each language. The annotation is carried out by multiple annotators of different races and ethnic. Appendix B contains details about the dataset and our upcoming track HASOC [8] (Hate speech and Offensive Content Identification) offered at Forum for Information Retrieval (FIRE)[9]2019.

- We have empirically benchmarked different text representation schemes based on Bag-of-Words, word embedding, sentence embedding, and pre-trained language models on TRAC Dataset [52] by performing exhaustive experiments on various traditional classifiers and deep neural models.

- We have developed unique tools to visualize aggressive content live on Facebook and Twitter. We have developed two interfaces for the visualization (i) Plugin: which run inside the Google Chrome browser and raise the flag as soon as it discovers aggressive contents on Facebook and Twitter (ii) WEB UI: Interactive interface where user can input the text and check the aggression level predicted by the model.

[8]https://hasoc2019.github.io
[9]http://fire.irsi.res.in/fire/2019/home

- We have developed a weakly supervised method to address pure IR ranking research task offered by FIRE IRMiDis track [10]

1.5 Thesis Outline

The rest of the thesis is structured as follows: In Chapter 2, we will present the comprehensive literature survey in the area of Microblog Summarization, Hate speech, and related concepts such as cyberbullying, aggression, offensive content identification. The Impoliteness term is used to refer to these concepts. Various summarization methods based on abstractive and extractive summarization will be reviewed. We summarize the journey of the research carried out in the area of the online Hate Speech. Different Forums and authors have reshaped the online Hate Speech detection problem into various fine-grained interesting classification problems. Furthermore, We will present publicly available Hate speech datasets along with data sampling and annotation methods. Wide range of classification methods studied for the impoliteness detection tasks.

In Chapter 3, we have examined the problem of Microblog summarization. Three variants of Microblog Summarization will be considered for the experiments: Push Notification, Email Digest, Summarizing Microblog during the disaster-related events. Different similarity measures and our proposed threshold estimation techniques to determine relevance and silent days will be discussed. In Chapter 4, we will present a comprehensive failure analysis of the Microblog Summarization system. The various trade-off to determine evaluation metrics are discussed in detail.

Chapter 5 will focus on various forms of impoliteness, such as online Hate speech, aggression, offensive content will be studied in detail. We have investigated different text representation scheme based on Bag-of-Words, Word embedding, and sentence embedding to model the social web text. These schemes are benchmarked on the TRAC dataset using various traditional classifiers and deep neural models. A new generation of contextual pre-trained language model-based text representation schemes such as ELMO, ULMfIT, and BERT are studied and benchmarked on the TRAC dataset [52].

In chapter 6, we will present the problem of aggression visualization live on Facebook

[10]https://sites.google.com/site/irmidisfire2018/

and Twitter. We have developed two interfaces to visualize aggression. The first interface is based on a web browser plugin that communicates with our deep neural model, deployed on the internet, from the web browser, and renders the content according to the prediction made by the classifier model. The second interface is WEB UI: an interactive interface, where the user can input the text and verify the aggression predicted by the model. We have deployed classifier based on deep learning model over the internet, which filter the aggressive content from the social media and Web UI. In chapter 7, we will give our concluding remarks and future directions.

CHAPTER 2

Background and Related Work

In this chapter, a summary of the background work is provided in the area of Microblog summarization and Impoliteness detection. Text summarization is one of the oldest research areas in the field of Information Retrieval and we will limit our scope of study in the domain of Microblog in the first section. Online Hate speech detection and related concept such as aggression, cyberbullying, offensive content are studied under the ambit of Impoliteness detection in the last section.

2.1 Microblog Summarization

The problem of text summarization from the short text such as Microblog or tweet could be formulated as either classification problem if sufficient labeled data made the available or ranking problem. However, the fusion of classification and ranking techniques can also be used as a prospective solution. In literature, at a top-level, there are two basic methods for summarization: abstractive summarization and extractive summarization. Abstractive summary generates a new sentence that is not present in the original tweet dataset, while extractive summarization select most representative tweets from the dataset to create the summary [94]. .

2.1.1 Abstractive Microblog Summarization

Most of the approaches for the Microblog summarization are based on extractive summarization, but there are a few works that are based on abstractive summarization. Sharifi et al. [109] proposed a Phrase Reinforcement algorithm based upon graph using topic keywords or phrase as a root node. Each incoming tweet words are other graph node. The weight of each node is computed using distance to the root and its frequency. The path having the highest weight with root node is selected in summary. Rudra et al. [103] have considered crisis-related events like an earthquake. They anticipated the tweet summarization problem as a classification problem and assign a predefined label like infrastructure damage, missing or found people, the requirement of emergency resource to incoming tweets. To summarize each class tweets, authors have created a bigram based word graph and generated new sentences using integer linear programming based optimization techniques. Chakrabarti et al. [19] have used Hidden Markov Model (HMM) to summarize sports-related event. Authors have argued that HMM could learn the language model of the segment of the event or sub-events. They have proposed a Tweet propagation model which is used to infer dynamic probabilistic distributions over topics. ROUGE-1 and ROUGE-2 score are used as evaluation metrics.

2.1.2 Extractive Microblog Summarization

Microblog track was started in TREC since 2010 with the objective to explore new retrieval techniques on Microblog text. In 2016 TREC Microblog track was merged with temporal summarization track and introduced as TREC Real time Summarization (RTS) which have synergies from both tracks [64]. There are two use cases for Microblog Summarization. The first case is the real-time push notification: How fast the summarization system delivers tweets to the user on his mobile phone through push notification as soon as the system identifies relevant and novel tweets. The second case is an Email Digest. At the end of the day, the system generates daily email digest or tweet summary for the user with respect to its interest profile, which essentially summarizes what happened today.

CLIP [9] was one of the top team in TREC MB 2015 [62]. Authors have done query expansion by training Word2Vec model [75] using the last four years of tweet corpus. The relevance score was calculated using the Okapi BM25 model. They have trained a learning to rank model using a BM25 score and other tweet features like count of the stem, the number of external URLs, number of Hashtags, etc. Tan et al. [116] proposed a simple term matching formula with different weight to title terms and expanded terms (3:1) to calculate the relevance score between tweets and interest profiles. They have expanded title terms with 5 Hashtags and 10 other terms using the Twitter search engine.

Li et al. [59] have used various text features of Interest profiles like title term, expanded term, description, and narrative term matched with tweet's text. They have also considered features like follower count and status count of the user who had posted the tweet. Based on these features, they have trained the SVM model using TREC Microblog 2015 labeled data. In another run, they have design customize linear scoring function with the same features which outperform the SVM model. Tan et al. [115] consider the real-time push notification problem as the long-term optimization problem and proposed a neural network based on the reinforcement learning algorithm which decides to push or not to push the relevant tweet to the user phone. The proposed neural network, having an LSTM layer and three fully connected layers, maximize the long-term reward, which is essentially the evaluation metrics. Word statistical features, temporal features, and semantic features are extracted from the tweet. Statistical features include no of terms in the interest profile and in the tweet, no of hashtags, temporal features include the time of the tweet when it is posted and the time of last pushed tweet. They have used the pre-trained Google Word2Vec model as a word embedding to generate a tweet text vector by averaging the individual word vector of the tweet. In addition to these, the SVM regression score, cosine similarity are used as semantic features. TREC standard metrics like EG-1, EG-0, nCG-1, nCG-0 [117, 64] were used as evaluation metrics, and these are substantially higher than the best run of TREC RTS 2016.

Lu et al. [68] have focused on detecting a silent day for the Interest profile(topic). There are days where relevant tweets are not available for some of the topics. These days are called silent day and the system should not push or include tweets in summary. Authors argued that most of the system assumed that there are relevant tweets in the Twitter stream and focused on the ranking of the tweet with respect to the interest profile. The authors formulated the silent day detection problem as a classification problem and considered features based on collective information from the query terms. Phrase-based weighted information gain (PWIG), similar to weighted information gain [138] is the first feature to decide relevance using the information gain by the occurrence of more than one query terms. If the query terms are closely related in the tweet collection on any day, the probability of the silent day is minimal. The second feature local query term coherence (LQC) based on above intuition. Authors reported results on push notification and email digest. Our work is similar to this, but we detected a silent day problem by setting silent day and relevance thresholds at an appropriate level which are estimated by method based on linear and support level regression. Our results are comparable to them. Chellal et al. [20] considered summarization problem as an optimization problem. Relevance score between query and tweet is calculated using extended boolean model and term weight is estimated using word embedding. To generate the summary, authors have used the integer linear programming method.

Yang et al. [129] have proposed Multitask learning Algorithm for Web-scale Real-time Event Summarization (MARES). The system consists of two tasks: The first task is a relevance prediction classifier based on the deep neural network having hierarchical LSTM. The second task is document filtering based on the reinforcement learning algorithm to maximize the rewards. Both tasks are performed simultaneously. They have reported results only for real-time push notification. Gonçalves et al. [43], consider the text summarization problem as a clustering problem. They have used various clustering algorithms like K-means, NMF, LDA, DTM on Qrels of the dataset. However, they have not mentioned the tweet filtering task, which is very important for Microblog summarization. Meladianos et al. [73] proposes a system which identifies sub-events or important moments within the specific event. Events are divided into time intervals and tweets posted during each time interval represented as a graph. To detect sub-events, they use convex optimization to determine the change in edge weight between two intervals. Standard information retrieval metrics such as precision, recall, and f1-measure are used to measure the performance of the system. Authors have created a dataset from the tweet related to the football match.

Our approach for the Microblog summarization is partially similar to [114], but we have chosen different smoothing parameters. We have employed two-level threshold mechanisms. In addition to this, we have designed a predictive model to estimate these thresholds. Suwaileh et al. [114] had set up similarity thresholds empirically.

2.2 Impoliteness Detection

This section presents a comprehensive summarization of research carried out in the field of Online Hate speech detection and related problem such as offensive content identification, cyberbullying, aggressive content detection. We have provided structure surveys and present the current state of research in these fields. We will describe the different problems formulated under the ambit of Online Hate speech detection. Various shared-task organized at different forums, Multilingual Datasets, along with different annotation methods, will be discussed thoroughly. We will present the numerous approached based on supervised learning, along with features used for the classification, will be summarized. We will discuss the trade-off in the selection of evaluation metrics to measure the classifier performance. We will conclude the discussion with a list of issues that needs to be addressed in the future. Hate speech, aggression, cyberbullying, offensive content are subsumed under the term impoliteness in the rest of the thesis.

Culpeper et al. [27] studied mechanisms of impoliteness through cross-cultural comparisons. The author has defined impoliteness in the context of large data collections and proposed a comprehensive impoliteness model that is not only theoretically informed but data-driven. There are many definitions available on Hate Speech in literature [88, 37]. Even popular website such as Facebook/Twitter/Youtube also has published policy related to the hate speech on their website. As per Nockleby et al. [89], Hate speech is commonly defined as any communication that disparages a person or a group on the basis of some characteristic such as race, color, ethnicity, gender, sexual orientation, nationality, religion, or other characteristics. Fortuna et al. citefortuna2018survey have also attributed some of the characteristics like Hate speech is to incite violence; Hate speech has a target; Hate speech is to attack or diminish.

Online Hate speech detection, as a research area, has been studied by the diverse research communities from the area of natural language processing, sociology, psychology, linguistics, and more recently, the machine learning community. This chapter will concentrate on the technical aspect of hate speech detection like classifiers, text representation features, dataset collection and annotation method, and evaluation metrics. We have come across the two popular survey : [37], [107].

Schmidt et al. [107] define term Hate speech as an umbrella of various anti-social or impoliteness related utterances. These include flames, Abusive/hostile messages, offensive content, cyberbullying. They have focused on different features that can be used for the hate speech classification. They have categorized features in simple surface, sentiment, word generalization, lexical resources, linguistic, Knowledge-based, and meta-information features. Simple surface features include character level N-gram, word N-gram, skip-gram, word generalization features include Brown cluster, word embedding, paragraph embedding. Linguistic features include the Part-of-Speech tag of tokens. General hate speech or profane related word list are categorized into lexical features. Meta-information includes no of a profane word, no of comment on the post, gender of the user who posted the message. Multimodal information includes features other than text like image, video, etc. They have studies various classification methods reported in the literature. However, comparative studies of these methods are not available. They also presented multiple datasets with annotation guidelines and presented challenges for hate speech detection.

Fortuna et al. [37] reported a detailed survey on Hate speech detection from a computer science perspective. They have precisely defined and differentiate hate speech with a related concept like profanity, flame, abusive message, extremism, radicalization. They have presented a search technique to select papers of hate speech detection from the Internet. Authors have reported results along with a set of features of the various well-cited paper. Authors describe different datasets and summarize the research challenges mentioned by the various authors.

This thesis attempts to study a variety of Hate speech detection problems formulated by the various authors and shared task forums. We will study various multilingual datasets offered at different venues such as TRAC [52], Offeneval 2019 [132], GermanEval 2018 [127]. A comparative study of the traditional classifiers with hand-crafted features and deep neural models which learn abstract features from the input data will be presented. We have listed out different approaches used by the top-performing teams at various forums. We also discuss the various trade-off involved in selection evaluation metrics.

2.2.1 Text Classification

Text classification is a fundamental problem in the NLP research field. Li et al. [61] define text classification as the automatic assignment of documents to one or more predefined categories. The application domain for the text classification is very vast, and that triggered researchers to formulate interesting real-time problems under the ambit of text classification. In this section, we will discuss the interesting application of text classification. Akimushkin et al.[2] considered text authorship identification problems in the disputed document using the dynamics of word co-occurrence networks. Authors have used network topology features such as network radius, clique. Authors have used J48, KNN, RBNF, and Naive Bayes classifier to classify a corpus of 80 texts by eight authors. The author has reported accuracy around 88.75 % in the KNN classifier. Amanico et al. [3] consider the problem to distinguish original scientific manuscripts from the artificially generated fake manuscript. Authors have used complex networks based approach for the text representation and experiment with Naive Bayes, KNN, and C4.5 classifier. Authors claim that the proposed method to classify real from fake papers with at least 89 % accuracy. Angelove et. al. [4] proposed a graph-based classification method, with particular emphasis on hyperlinked text documents. Authors claimed that the proposed algorithm outperforms traditional classifiers like SVM and NB on IMDB and Wikipedia dataset.

2.2.2 Hate Speech Detection: Problem Formulation

The research on automatic Hate speech detection started on the way back to 1997. Spertus et al. [112] have used term flames to denote the abusive/hostile message. Authors have built a prototype system called Smokey based on semantic and syntactic features of the sentences. They have categorized flames message into three classes, namely: Flames, Ok, Maybe. Razavi et al. [95] proposed an automatic flame detection method which extracts features at different conceptual levels and applies multilevel classification for flame detection by leveraging a variety of statistical models and rule-based patterns. Authors have classified messages into two class Flame or Ok. They have prepared a dictionary of offensive words and phrases.

Warner et al. [123] were the first who use the term Hate Speech to classify abusive

Table 2.1: Sample Post from each class of Hate Speech Dataset [29]

Text	Class
@MoriTaheripour shut up nigger whore! Hope u get raped by one of those animals. Might change your tune	HATE
I'm tired of people saying I look like my brother ; calling me Deondre' like serious Succ My As* fag a**es	OFFENSIVE
@RiotSupport so I was suspended for a day because of a random lag spikes that force me to close the client and relog and suspended. GG	Neither

messages over the internet. Authors have identified seven categories like anti-Semitic, anti-black, anti-Asian, anti-woman, anti-Muslim, anti-immigrant, or other-hate. They have formulated the above problem as a binary classification problem, i.e., One vs. all. There are a few works in which authors considered domain-specific hate speech problem. Kwok et al. [54] have tried to classify tweets against black or not. They have collected two-class: racist and non-racists of tweets and used the Naive Bayes classifier for binary classification. Djuric et al. [34] also build a binary classifier to classify in between hate speech and "clean" user comments on a website. Burnap et al. [16] explore cyber hate on Twitter. They have collected tweets for the specific domain in a two-week time window. A collection of 450,000 tweets were annotated as hateful or genuine. Gitari et al. [41] abstracted Hate speech into fine-grained labels like race, religion, and nationality. Similarly, Nobata et al. [88] construct binary classification problem on Yahoo news and finance category pages. They have used clean and abusive labels for the classification. Wassem et al. [125] formulated a 3-class classification problem on hate speech using three labels, namely: racist, sexist, and neither of them.

Davidson et al.[29] attempt to discriminate hate speech from offensive content. They have annotated tweets into three classes, namely: OFFENSIVE, HATE, and Neither. Table 2.1 shows sample annotated tweet from the Hate speech dataset created by [29].

Apart from English, few works are also reported in Arabic, Chinese, Dutch, German, Hindi, Slovene languages in the literature. Tulkens et al.[118] considers the racism detection problem in Dutch social media. Authors labeled social media posts in three different categories, namely: racist, non-racist, and invalid. The racist label describes comments that contain negative utterances or insults about someone's ethnicity, nationality, religion, or culture. Posts that did not contain any text or that were written in languages other than Dutch are categorized in Invalid class. Mubarak et al. [85] study abusive language in Arabic social media. They have annotated Arabic tweets into three classes, namely: obscene, offensive, and clean. Ross et al.[100] prepared German Hate speech corpus on the European refugee crisis. they have used two classes: Hate and Non-hate. Fiser et al.[36] propose Hate Speech classification task Slovene language at multiple granularities. At coarse-level, they have identified 2 classes: SUD (Socially Unacceptable Online Discourse), and not SUD. At medium level granularity, authors have categorized post into 4-classes, namely: no elements of Problematic speech, Inappropriate speech, Inadmissible speech, Hate speech. At a fine-grained level, they have proposed eight classes. Su et al. [113] attempted to detect and rephrase profanity in Chinese text instead of masking detected profanity. Authors have developed 29 rephrasing rules after examining sentences on social media.

Cyberbullying

Traditional bullying was a well-studied research area by social scientists. With the advent of the internet and the world wide web, Cyberbully term was coined by the social scientist. Li et al. [60] defined cyberbully as bullying via electronic communication tools such as email, cell phone, Personal Digital Assistant (PDA), instant messaging, or the World Wide Web. Vandebosch et al.[120] studied cyberbullying among the youngsters. After reviewing the profile of cyberbully and victim, authors have concluded that the problem of cyberbullying is not marginal. Dinakar et al. [33] studied cyberbullying on comments posted on Youtube videos. Comments are categorized into Sexuality, Race and Culture, Intelligence. The classification problem was formulated as a 3-class classification and binary classification (one vs. all). Xu et al. [128] claimed that social media, with appropriate NLP, can be a valuable and abundant data source for the study of bullying in the cyber world. Authors have identified several key problems in social media data sources and formulate them as text classification, role labeling, sentiment analysis, and topic modeling. Authors have created the dataset from Twitter. Dadvar et al. [28] had performed a binary classification experiment between the bully and not-bully. Waseem et al. [124] have categorically defined the difference between Cyberbullying and Hate speech. Authors stated that cyberbullying is always directed towards an individual or group. i.e., it is always targeted while this may not be true for the Hate speech.

Sentiment Analysis vs. Hate Speech Detection

Social media provide a new form of communication to share opinion and sentiment about any real-world entity. The contents generated by the heterogeneous user across the world create a new platform to study user behavior. Sentiment analysis is the computational study of sentiment expressed in the text. Liu et al. [65] have defined sentiment analysis as follows: Given a document d which comments on an entity e on feature f, oo represents the orientation oo of the opinion expressed on e. The above can be represented by the quintuple (e, f, oo, h, t). Initially, Sentiment analysis problem is formulated as a binary classification problem for predicting the election results or detecting political opinion [72, 25, 26, 119] on Twitter. Gradually, It turned into a multi-class classification problem with the introduction of the neutral label. SemEval (International workshop on semantic evaluation)[97] is one of the popular competition on sentiment analysis, which has started a sentiment analysis challenge since 2013. They have introduced a sentiment analysis task at two-level granularity: message level and phrase-level granularity.

Research on Sentiment analysis and Hate speech are commenced in parallel, and both problems share many similarities. One can assume that comment with negative sentiment has a higher probability to be in the category of a hate speech than a message with positive sentiment. However, every comment having negative sentiment might not belong to the Hate speech category. Many works on Hate speech detection, use the sentiment as a feature for the classifier. Some of the popular approaches [14] used in sentiment analysis are used in the hate speech detection problem. If one can look at the recent edition of SemEval [1], Sentiment analysis task is replaced by Hate speech or offensive speech detection task. Researchers working in the area of domain-specific sentiment analysis move to the problem of domain-specific or open domain hate or offensive speech detection.

Trolling Aggression and Cyberbullying (TRAC)

The Trolling Aggression and Cyberbullying (TRAC) workshop co-located with and organized under COLING 2018 at Santa Fe, USA in August 2018. This workshop aims

[1]http://alt.qcri.org/semeval2019/index.php?id=tasks

Table 2.2: Sample post for each class in TRAC dataset.

Text	Class
Royal Enfield is icon... its not possible to shake image	NAG
One common thing I noticed in above criticisms is..they all have same color	CAG
BJP idiots dictators, upper caste brahminical mentality. Divide people the policy? Kick those idiots out.	OAG

to study online aggression, a particular case for the hate speech, on social media. Aggression is classified into three labels, namely- ‘Non-aggressive‘ NAG), ‘Covertly Aggressive‘ (CAG), and Overtly Aggressive (OAG). Any speech/text in which aggression is overtly expressed either through the use of specific kinds of lexical items or lexical features which is considered aggressive and or certain syntactic structures is overt aggression [52]. Covertly Aggression contains an indirect attack against the victim and is framed as a polite or sarcastic expression and might not contain any abusive/offensive/controversial word. Non-aggressive posts do not contain any aggression.

Kumar et al. [52] describes aggression as discursive features. Discursive features include role and effect. Authors attributed three discursive roles: attack, defend, and abet. Discursive features can be any of the following aggression: Physical Threat, Sexual Aggression, Identity Threat / Aggression, Gendered Aggression, Geographical Aggression, Political Aggression, Casteist Aggression, Communal Aggression, Racial Aggression. However, the difference between aggression and cyberbullying does not exist in any literature. Table 2.2 shows sample posts from the NAG, CAG, OAG categories.

Kaggle’s Toxic Comment Classification Challenge

This shared task was aimed to classify English Wikipedia comments into six categories, namely: toxic, severe toxic, obscene, insult, identity hate, and threat. These categories are not mutually exclusive. In other words, the shared task is based on a multi-label classification problem. Comments can have more than one label.

GermEval Task 2018 — Shared Task on the Identification of Offensive Language

GermEval [2] presents a series of shared task evaluation which focuses on various NLP task on the German language. GermanEval 2018 offers the two sub-tasks [127]. The first task is binary classification to determine a tweet includes offensive language or not. A fine-grained classification is offered in the second sub-task where offensive tweets should be categorized into three classes, namely: PROFANITY, INSULT, and ABUSE. In Germaneval 2019, third sub-task is included which categorize offensive tweet into explicit and implicit offensive language.

Automatic Misogyny Identification – IBEREVAL 2018

This shared task proposes the automatic identification of misogynous content both in English and in Spanish languages on Twitter data. The AMI shared task was offered in two sub-tasks [35]. Sub-task A primarily focused on the binary classification of the tweets into two categories: misogynistic and non-misogynistic. The sub-task B is a more fine-grained classification where classification system is required to classify misogynistic tweets into misogynistic behavior which is further categorized into Stereotype & Objectification, Dominance, Derailing, Sexual Harassment & Threats of Violence. Furthermore, the system required to classify the target of the misogynistic tweet into active and passive. If the tweet text includes offensive messages purposely sent to a specific target or individual, then the target is categorized into active other passive.

OffensEval:SemEval 2019-Task 6

The OffensEval shared task in SemEval- 2019 was introduced as a 3-level classification task [132] to identify offensive language from the Twitter posts. In the first level, sub-task A, systems are required to classify tweets into two classes, namely: Offensive (OFF) and Non-offensive (NOT). In the second level, sub-task B, offensive tweets are further required to be categorized into two labels, namely: targeted (TIN)-post which contain threat/insult to the targeted entity and untargeted (UNT), respectively. In the sub-task C, a target of insults and threats are further classified to Individual (IND), Group (GRP), or Other (OTH) classes. Table 2.3 shows sample posts for each classes.

[2]https://projects.fzai.h-da.de/iggsa/germeval/

Table 2.3: Sample Post from each class of OLID dataset

Text	**Offensive**	**Targeted**	**Target Type**
I got more common sense than all of my followers	OFF	TIN	GRP
@USER Then your gonna get bitten	OFF	TIN	IND
@USER I would be worse than her l*cking that as*	OFF	UNT	NULL
@USER He is like a cheap plastic version of a real president.	NOT	NULL	NULL

HateEval : SemEval 2019 Task 5

This shared task [11] is aimed to detect Hate Speech against immigrants and women on Twitter for Spanish and English language. The shared task is offered in two sub-tasks. The sub-task A consists of a binary classification where systems are needed to classify a tweet in English or Spanish with a given target is hateful or not hateful [3]. The sub-task B is consists of two binary classifications- Aggressive behavior classification and target classification. In the first classification, systems are required to classify hateful tweets as aggressive or not aggressive, and in the second part, the system needs to identify the target harassed as an individual or group.

2.2.3 Dataset

Standard benchmarked datasets play a critical role in any NLP task to set the baseline for the evaluation and peer comparison. Various authors [29, 100, 125] and shared task forums [132, 52, 35, 127] have developed and published labelled dataset. Unfortunately, none of the datasets could be established as a standard dataset for Hate speech detection. In this section, we will study different datasets created from social media for the Hate-speech and related problems. Various data collection and annotation methods are summarized here. Table 2.4 present the details of the publicly available dataset for future experiments.

[3]https://competitions.codalab.org/competitions/19935

Table 2.4: Hate speech and Related dataset

Dataset and venue	**# Posts**	**Classes**	**Language**
Zampieri et al.[132] Offenseval at Semeval 2019	Train :13240 Test: 860	Level-1: OFF, NOT. Level-2: TIN, UNT Level-3:IND,GRP,OTH	English
Basile et al. [11] HateEval at Semeval 2019	Train-9000 Test:3000	Level-1: Hate, NOT L-2: Aggressive, NOT Level-3:IND,GRP	English Spanish
Kumar et al.[52] at TRAC COLING 2018	Train-15001 & 15001 Test 2173 & 2164	NAG, CAG, OAG	English, Hindi
Fersini et al. [35]AMI 2018	Train-3251 and 3307 Test 726 and 831	misogynous, Stereotype, Dominance, Derailing, Sexual Harassment, Threats	English, Spanish
Wiegand et al.[127] GermanEval 2018	Train-5009 Test 3552	Level-1 : OFF,oth. Level -2 : ABUSE, INSULT, PROFANITY	German
Davidson et al. [29] Hate Speech corpus	14509	Hate speech, offensive and neither	English
Mubarak et al.[85] Hate speech corpus Arabic social media	1100	obscene, offensive, and clean	Arbiac
Fiser et al. [36]Socially Unacceptable Online Discourse corpus	NA	No Problematic, Inappropriate, Inadmissible, Hate	Slovene
Ross et al. [99] German hate speech corpus for the refugee crisis	470	Hate speech, not offensive	German
tulkens et al. [118] Racism Detection in Dutch Social Media	Train 5424 Test 607	Racist, Non-Racist	Dutch
kowk et al. [54] Corpus for Racist comment	24582	Racist, Non-Racist	English

Data collection/Sampling

Twitter, Facebook, YouTube, Yahoo are the popular Hate Speech data source. Most of the Hate Speech datasets are sampled from Twitter [29, 99, 52, 127, 132, 35, 86, 110, 16]. However, some of the datasets are also sampled from Facebook [52, 118], Yahoo [88, 34, 123], and Youtube [33, 28]. Followings are popular ways to sample data from these social media.

- Searching social media, using representative keywords (Bi*tch,F*ck) of hate or offensive language
- Using popular Hash-tags which represent any controversial event in the social media (#Aslyanten, #WehrDich, #Krimmigranten.)
- User timeline of the user who regularly posts hate speech/offensive contents.
- celebrity/politician/feminist timeline where anonymous social media user often post offensive posts
- popular phrase like you are, she is, He is
- Comments posted on popular YouTube video timeline.

Dinakar et al.[33] and Dadvar et al. [28] used comments posted on popular Youtube video to prepare the dataset. Ross et al.[100] created Hate Speech corpus for the German language. They have collected tweets on the European refugee crisis using popular hashtags like #Aslyanten, #WehrDich, #Krimmigranten. Fersini et al.[35] have used several representative keywords (Bi*tch, F*ck), potential victim accounts like a feminist/celebrity and Twitter profile who frequently posts hate speech related concept to sample posts. Mubarak et al. [86] prepared an Arabic tweet corpus from Twitter using offensive Arabic words. Davidson et al. [29] use Hatebase.org, which contain hate speech lexicon containing words and phrases to prepare seed words to sample tweet from Twitter. Kumar et al. [52] have collected tweets using the hashtag, which represents some political controversies. Wiegand et al. [127] created a corpus using the Twitter timeline of 100 different users. Zampieri et al. [132] collected tweet using keywords which often occur in offensive language. these keywords are: she is, ’to:BreitBartNews’, ’you are’.

The size of the available Hate speech datasets is ranging from a hundred to tens of thousands. Most of the datasets were sampled from Twitter. The classification task becomes coarse-grained to fine-grained with the introduction to new labels(Profanity, offensive, Insult, etc.) [127]. Single level binary classification task turning into multilevel binary/multi-class classification [132].

Data Annotation method

Data annotation is a very critical task for the hate speech detection dataset where agreement among the annotator is a significant issue. Annotation platform, annotator profiles, number of annotators per tweet annotation, inter-coder agreement are the major issues in the annotation process.

The annotation method by [54] presented hundreds of potential hate speech tweets to three students of different races with the same age and gender and asked to classify whether a tweet was offensive or not. The calculated percentage of overall agreement was only at 33%, which is very low and indicates difficulty in the classification would be even more difficult. Nobata et al. [88] use Amazon Mechanical Turk(AMT), an online crowdsourcing platform, for the annotation, They achieved agreement rate at 0.86 for the binary classification while for more fine-grained classification agreement rate was reduced at 0.405. Ross et al. [99] have tried to assess the reliability of the hate speech corpus with 51 annotators with two groups with and without Twitter's Hate speech definition. 20 tweets were given to 51 annotators results in 1120 annotations. The author has reported a low agreement of around 0.18 to 0.30. Davidson et al. [29] use Crowdflower, an online platform, to annotate tweets into predefined classes: Hate, Offensive, and None. Each tweet was annotated by more than people, and the inter-coder-agreement score was quite impressive at 92 %. Mubarak et al. [86] have used the same crowdsourcing platform for the annotation and achieved inter-annotator agreement was around 85%.

Kumar et al. [52] initially started annotation process with 4 annotators and achieved $\kappa = 0.49$. Due to lower κ, authors perform an agreement test on the Crowdflower platform with 100 test instances. each instance was annotated by 3 annotators. A total of 77 annotators attempted the test. It is to be noted here that each annotator did not annotate an equal number of instances. The number of annotations by each annotator ranged from 135 judgments to 10 judgments. At the end of these experiments, the inter-annotator agreement was reached above 72 %. Fersini et al. [35] conducted the annotation process in a two-step. In the first step, tweets were labeled by two annotators, In case of difference in annotation, the third mature annotator takes the final call. Volunteers of the crowdflower labeled remaining tweets with the majority vote in the second step. Tulkens et al. [118] have used a similar approach as [35] used.

Wiegand et al. [127] adopted an interesting approach to calculate the inter-annotator agreement. 300 tweets were sampled from the dataset and annotated in parallel. the inter-annotator agreement was around $\kappa = 0.66$. Zampieri et al.[132] have used crowdsourcing platform Figure-Eight [4] for the data annotation. Authors have accepted annotation with 100% agreement by the two experience annotator while in case of disagreement, more annotations are carried out until more than 66% agreement reached.

Most of the datasets were annotated on the crowdsourcing platform. Barring a few cases [99], the inter-annotator agreement κ is more than 66% for most of the dataset. The value of κ reduced substantially from coarse-grained classification to fine-grained classification[99].

2.2.4 Text Representation

In this subsection, we will study features that are used for text representation by various supervised learning methods for the detection of hateful content. Identification of the right features helps the algorithm to predict the label correctly. Henceforth, we will use term text representation and features interchangeably in the rest of the thesis.

Text representation is about numerically representing documents so that they can be feed as an input to the classifier. This numerical representation is in the form of the vectors which together form matrices. Essentially, There are four types of text representation schemes :(i) Bag-of-words(BoW) (ii) Distributed Word representation or word embedding (iii)Sentence embedding (iv) Contextual pre-trained language model.

[4]https://www.figure-eight.com/

Bag-of-Word Model for Text Representation

The Bag-of-Words (BoW) is the simple technique to represent the document or social media posts in the vector form and also a popular feature extraction method from the text. Word count, word occurrence, or TF/IDF weight of each word n-gram or character n can be used as a feature. The dimension of the vector is equal to the size of the vocabulary of the text corpus or dataset, which results in a very high dimensional sparse document vector. It is the conventional method used for the text representation to perform various NLP downstream tasks such as text classification, clustering. However, the BoW methods ignore the word order, which may lead to loss of the context.

N-Gram This feature can be used at two levels: word level (word n-gram) and character level (Character n-gram). Kowk et al. [54] and Dinakar et al. [33] have used word unigram with TF-IDF weight. Davidson et al.[29] considered word bigram and trigram features weighted by its TF/IDF in addition to the unigram feature. Malmasi et al. [71, 70] used character bigram to character 8-gram, and word unigram, bigram, and trigram. They have achieved the best result in character 4-gram among all features. Waseem et al.[125] claimed that character n-gram outperforms word n-gram by at least 5 basis points. Badjatiya et al. [8] and Agarwal et al. [1] have used character n-gram for the text representation. Word Skip-gram features are used to estimate longer distance dependencies between words in the sentences. Such dependencies are difficult to capture using word-level bigrams alone. Malmasi et al. [71] have used 1-, 2- and 3-skip word bigrams for the Hate speech classification.

Word and Sentence Embedding

Word Embedding is the text representation technique, based on distributed word representation, to map the word in the low dimensional space so that semantically similar words have similar representation [76]. Major word embedding techniques, such as Word2vec learn word embedding using a shallow neural network. The fastText [15], an extension of Word2vec, consider the morphological structure of the word. Paragraph Vector is an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts, such as sentences, paragraphs, and documents [55].

Djuric et al. [34] used Doc2Vec [55] model to learn the sentence/paragraph embedding. Majumder et al. [69] used pre-trained fastText vector for the text representation. [8] had used glove [91] embedding for the text representation.[39] had performed an experiment with 3 different features: random Word vector, word vector generated using Word2vec and character n-gram. Authors have concluded that Word2vec is better to text representation than a random vector.

Contextual Pre-trained Language Model

In 2018, Transfer learning achieved an important breakthrough in the area of NLP. With advent of ElMo [92], ULMFiT [49], BERT [31], transfer learning got the boost in the NLP. Rother et al. [101] used the Universal Language Model for Fine-tuning for Text Classification (ULMFiT,) a transfer Learning model trained on German Wikipedia corpus, in their participation in GermanEval 2018. The NULI team has secured the first rank in level-1 binary classification OffensEval 2019 [134]. Authors have used BERT for the text representation. The second rank team NLPR @SRPOL [134] used OpenAI GPT, ELMo. Team vradivchev_anikolov [134] secured third rank using BERT representation.

External Lexical Resource

Many Instances of the Hate speech messages might contain negative, insults, profane, or swear words. The presence of these words can be considered as a feature and fed into the classifier. Liu et al.[66] and dadvar et al. [28] used Online Hate speech related dictionary [5] to collect profane words. [33] used the Ortony lexicon of words and a set of profane words. Every Hate Speech related message may not contain profanity but might have negative utterance. The Ortony lexicon contains such words denoting a negative connotation. Burnap et al.[16] have used target specific(LGBT, handicapped) hate speech word [6] [7] [8]. Spertus et al.[112] used lexicon consist of good adjectives and good verbs. Razavi et al. [95] created a weighted dictionary of word and phrase according to their degree of abusiveness. common hate-related terms are available online [9]

[5]www.noswearing.com/dictionary
[6]https://en.wikipedia.org/wiki/List_ of_ethnic_slurs
[7]https://en.wikipedia.org/wiki/List_of_LGBT_slang_terms
[8]https://en.wikipedia.org/wiki/List_ of_disability-related_terms_with_negative_connotations
[9]http://rsdb.org, http://hatebase.org

Template or Regular Expression

The Hate speech detection phrase was first coined by [123] to the community. They have created templates or patterns to detect a particular hate-related phrase. An example of such a pattern is T-1:go, T+0:back T+1:to". This is an example of a template for a two-word window on the word "back".

Sentiment

The solution proposed for Hate speech detection is quite influenced by the work done by the sentiment analysis community. Any posts with negative polarity have a high probability to get classified into hate-speech category than a post with positive polarity. Gitari et al. [41] used additional classifiers which separate negative blog from the positive blog. Blog with negative polarity was fed to the hate-speech classifier.

Linguistic feature

The part-of-speech tag of the token contains much important information about the structure of the sentences. Xu et al. [128] used a POS tag with an n-gram token. However, the PoS feature failed to improve classifier performance substantially. Davidson et al. [29] have also used Part-of-speech (PoS) tag in a feature vector.

Social Media specific feature

Social media posts have many implicit features such as hashtags, user mentions, retweets, and embedded URLs along with the text. Davidson et al. [29] have exploited these social media-specific features to build the feature vector. In addition to this, lexical features of the tweet, like no of character, no of words syllables are also useful for the hate speech classification. Nobata et al. [88] used the average length of the word, number of periods, number of punctuation, quotes, repeated punctuation, and question marks.

2.2.5 Classifier

Most of the approaches available in the literature for Hate speech detection are based on supervised learning. One of the earliest work in the flame detection [112], used C4.5, a tree-based classifier for the classification. Dinakar et al.[33] used Naive Bayes (NB), Support Vector Machine (SVM), JRip, and J48 for the classification. According to their conclusion, JRip delivers the best accuracy while SVM gives the best κ across all classifiers. Most of the work found in the literature [128, 28, 16, 118, 71] used SVM for the classification. Table 2.5 describe the details of the classifier used by the various authors. Table 2.6 and 2.7 present the approaches used by various top teams at different forums.

As mention by Schmidt et al. [107], Hate Speech detection and Sentiment analysis are closely related. International Workshop on Semantic Evaluation 2013 (SemEval-2013) [48] was the first forum that developed standard tweet dataset for the benchmarking of the various sentiment analysis system. We believe that Sentiment Analysis is one of the subfield under the umbrella of Hate Speech. Deep learning and word embedding had shown its footprints in SemEval-2015 [98]. Team UNITN [108] was the second team in the message polarity task. Authors designed a convolution neural network for the sentiment classification. They have used an unsupervised neural language model to initialize word embeddings that is further tuned by deep learning model on a distantly supervised corpus [108]. In fourth edition SemEval-2016 [87],Team SwissCheese [30] was the first ranked team with F_1 score around 63.3 %. Authors have used 2-layer convolution neural networks whose predictions are combined using a random forest classifier. SemEval-2017 [97] was the fifth edition, Team DataStories [14] was the top-ranked team with AvgRec= 68.1 and F_1 around=67.7 %. They use Long Short-Term Memory (LSTM) networks augmented with two kinds of attention mechanisms, on top of word embedding pre-trained on a big collection of Twitter messages without using any hand-crafted features.

From 2010 to 2017, SVM, Naive Bayes, and Logistic Regression are the popular classifiers found in the different literature. As the Sentiment Analysis community greatly influences the Hate-speech detection community, various authors started using methods based on deep learning and word embedding. The deep neural network learns abstract features from the text as opposed to the previous method where handcrafted features are fed to the classifier. In other words, there is a paradigm shift from feature engineering to automatic feature detection. The most popular deep neural network architectures are Long Short-Term Memory network (LSTM) and Convolutional Neural Network (CNN) reported in the literature. Nobata et al. [88] used the skip-gram model for the abusive message classification. Gamback et al. [39] used the CNN model with Word2vec embedding for the Hate speech classification. Badjatiya et al. [8] used LSTM/CNN with fastText and glove embedding for the classification. Zhang et al. [136] set up the experiment with a dataset created by [29, 125] using deep neural network based on CNN and LSTM. Authors have used SVM to create the baseline results. However, CNN-LSTM based neural network outperforms baseline results by only 1 % to 3% on the various datasets. Table 2.5 shows a summary of different well-known approaches used for the Hate speech classification.

To encourage the research in the Hate speech area, Researchers are reshaping the Hate Speech detection problem at a very fine-grained level from the coarse-grained classification at various forums like TRAC, SemEval, IberEval, GermanEval. In 2018, Transfer learning achieved an important breakthrough in the area of NLP. With advent of ElMo [92], ULMfit [49], BERT [31], transfer learning got the boost in the NLP. Team ULMFiT [101] used the Universal Language Model for Fine-tuning for Text Classification (ULMFiT,) a transfer Learning model trained on German Wikipedia corpus, in their GermanEval 2018. However, they have reported results much lower than traditional classifiers. Team TUWienKBS [83], top team in GermanEval-2018 [127] binary classification task, have used ensemble classifier of Logistic Regression and Random Forest with hand-crafted features which outperform second rank team uhhLT [126] who have used transfer learning model trained on a large corpus. However, in fine-grained or multi-class classification latter has performed better than the former. Table 2.6 shows the results of top teams at the various forum on Hate speech classification tasks.

At TRAC Forum [51]; Team Saroyehun [6] was top team in Facebook English Dataset. They have done classification by employing LSTM with vocabulary augmentation using other hate speech datasets, which was not done by the third rank team DA-LD-Hildesheim [69] with a similar approach. DA-LD-Hildesheim [69], the top team in Twitter Hindi Dataset, used a deep neural network based on CNN with fastText pre-trained vectors. Team EBSI-LIA-UNAM [7] was the second-best team in English Dataset. They have used the ensemble classifier of SVM, Naive Bayes, and Passive Aggressive. Team na14 [106], a top team in the Facebook Hindi Dataset of TRAC, used an ensemble of Logistic Regression and SVM. However, performance lower in the Twitter Hindi dataset than[69]

Table 2.5: Methods, Features and Classifier for Hate speech Detection

Authors	Features	Classifier	Results
Spertus et al.[112]	Rule based	Decision Tree(C4.5)	NA
Dinakar et al. [33]	POS,lexicon, TF/IDF n-gram	NB,SVM,J48	Accuracy 80.20 and κ at 0.79
Warner et al.[123]	Template based	SVM	F1-measure at 0.63
Xu et al. [128]-	unigrams,bigrams, POS tags	NB, SVM(linear and RBF) and Logistic Regression	Accuracy 0.81
Dadvar et al.[28]	Content-based, user-based features	SVM	F-measure around 0.64
Kwok et al.[54]	unigram	Naive Bayes	Accuracy around 76%
Burnap et al.[16]	N-gram, typed dependencies	Bayesian Logistic Regression,Decision Tree,Random Forest, SVM, Ensemble	F1-measure at 0.77
Tulkens et al.[118]	3-level Dictionary using Word2vec	SVM	F1-score at 0.46.
Nobata et al.[88]	linguistic, Pos, Word embedding	Skip-gram model	F1-score at 0.805
Davidson et al. [29]	N-gram, POS tag, Sentiment score	NB,SVM(Linear), Logistic Regression, Decision Tree	F1 score around 0.90
Mubarak et al.[85]	N-gram		Arbiac tweets, F1-score around 0.60
Gamback et al. [39]	character N-gram, Word2vec embedding	CNN	F1-score around 0.78
Badjatiya et al.[8]	fastText, Glove	LSTM, CNN	F1-score at 0.93
Malmasi et al.[71]	Word N-gram, skip-gram, character N-gram	SVM, Ensemble	F1-score at 0.798

where the system is based on the CNN model. [35] have introduced Automatic Misogyny Identification at IberEval 2018 in English and Spanish languages at different classification granularity. All the top teams at AMI have used SVM with various features. [90] used the SVM classifier with the lexicon of the abusive and sexist word. It is worth to note that that the top team in English Dataset in multi-class classification performs poor on the Spanish dataset.

Top team at OffensEval [134] NULI used BERT for the classifications. The second top team NLPR @SRPOL [134] used ensembles of Random Forest, OpenAI GPT, Universal encoder, the Transformer, ELMo, and combined embeddings from fast-Text and custom ones. The third rank team vradivchev_anikolov [134] has used the soft voting classifier of CNN, RNN, and BERT.

In the last decade, the notion of Hate speech has been formally defined by various organizations, including social media websites such as Facebook, Twitter, etc., European Union. Hate speech detection task become more fine-grained [127], [35],[132] from the simple coarse-grained binary classification. With the improvement in computing power using GPU, deep neural models have a great potential to outperform traditional classifiers with hand-crafted features. As we look at the results reported in [127] [51] [136], Deep neural network outperforms SVM by just 1 to 3% at the cost of expensive computing power (GPU). However, this is not always true on many datasets [127].

2.2.6 Evaluation Metrics

Accuracy, precision, recall, and F1-score are the common and standard metrics to measure the performance of the classification methods. The F1-score is the weighted average of precision and recall. F1-score is the better metric than accuracy in case of unequal distribution of class-labels in the dataset. Some of the initial work [33, 128, 54] used accuracy to report the results. Warner et al. [123] have used the F1 score to report the results.

F1-score has many variants like weighted F1, Macro-F1, micro-F1. In the case of multi-class classification, In most occasion, distribution of class labels are uneven. Weighted F1-score calculates the F1 score for each class independently, but when it adds them together uses a weight that depends on the number of true labels of each class. therefore it's bias the majority class. Kumar et al.[51] used weighted f1 score c for the evaluation.

Table 2.6: Methods and features used by top ranked team at TRAC and GermanEval 2018

Authors	Features	Classifier	Results
Aroyehun et al.[6] at TRAC 2018 3-class	fastText pre-trained vector	LSTM	Weighted F1-score 0.6424 FB, 0.5920 Twitter
Arroyo et al. [7] at TRAC 2018	character N-grams, TF-IDF	Ensemble of SVM,NB,Passive Aggressive	Weighted F1-score 0.6315 FB, 0.5715 Twitter
Majumder et al. [69] at TRAC 2018	fastText pre-trained vector	LSTM,CNN	Weighted F1-score 0.6177,0.5519 English FB and Twitter. 0.6080 Hindi FB and 0.4992 Twitter
Samghabadi et al. [106] at TRAC 2018	Unigram, Character N-gram, Word Embedding	Ensemble of SVM Logistic Regression	Weighted F1-score 0.5921 English FB and 0.5663 Twitter. 0.6451 Hindi FB and 0.4853 Twitter
Montani et al.[83] at GermEval	Word N-gram, Character N-gram, word embedding	meta-classifier Random Forest, Logistic Regression	F1-score binary-0.7671 multi-class 0.5142
Wiedemann et al. [126] at GermEval	fastText pre-trained vector	BLSTM+CNN and Transfer Learning	F1-score binary-0.751 multi-class 0.5271
Von et al.[121] at GermEval	fusion of fastText pre-trained vector	CNN+GRU	F1-score binary-0.7552 multi-class 0.4088
Rother et al.[101] at GermEval		ULMFiT	F1-score binary-0.7100 multi-class 0.4088

Table 2.7: Methods and features used by top ranked team at AMI, OffenEval

Authors	Features	Classifier	Results
Pamungkas et al.[90] at AMI IberEval 2018	lexicons of abusive words, sexist slurs and hate words	SVM(Radial) for English SVM Linear for Spanish	Accuracy English/Spanish 0.9132/0.815 multi-class 0.3698/0.446
Frenda et al.[38] at AMI IberEval 2018	character N-gram,sentiment Lexicon	SVM,ensemble	Accuracy English/Spanish binary-0.8705/0.8135 multi-class macro F1-score 0.4424/0.441
Canos et al.[17]at AMI IberEval 2018	unigram weighted by TF-IDF	SVM	Accuracy binary-English/Spanish 0.8147 / 0.7493 multi-class macro F1-score English/Spanish 0.4328/0.3262
NULI team [134] OffenEval 2019	NA	BERT	Macro-F1 Level-1 0.8286, level-2 0.7159, level-3 0.5598
NLPR@SRPOL team [134] OffenEval 2019	fastText embedding	ensembles of Random Forest, OpenAI GPT, Universal encoder, the Transformer, ELMo	Macro-F1 Level-1 0.80, level-2 0.69, level-3 0.63
vradivchev_anikolov team [134] OffenEval 2019	Glove embedding	CNN, RNN and BERT	Macro-F1 Level-1 0.8153, level-2 0.6674., level-3 0.6597

The micro F-score uses the global number of True Positive (TP), False Negative (FN), False Positive (FP) and calculates the F1 directly. So it does not favor any class. Finally, 'macro' calculates the F1 separated by class but not using weights for the aggregation, which results in a bigger penalization when the model does not perform well with the minority classes. The choice of the variant of F1-measure depends on the objective of the tasks and distribution of label in the dataset. Hate Speech related classification problems suffer from class imbalance. Therefore, macro F1 is the natural choice for the evaluation. Many evaluation forum [132, 127, 35] have use macro F1-score for the evaluation.

2.2.7 Challenges

The notion of Hate speech is debatable and varies across different communities and countries. [99] describe Hate speech as a vague concept and advocate for the better definition and guideline for the Hate Speech data annotation. They have reported intercoder-agreement from 0.18 to 0.29 for the fine-grained Hate speech classification, which is substantially lower than established Krippendorff recommendation of a minimum of κ= 0.80, or 0.66. Lower agreement across human annotator indicates the complexity of Hate speech detection problem, which further makes it difficult for the classification algorithm. On many occasion, offensive content or profanity are co-occurred with Hate speech. [29, 71] present the problem for separation hate speech from the offensive languages or profanity. Wiegand et al. [127] insists that annotation should be by a person having expertise in the culture and social culture.

Recently, informal Languages in social media changes very quickly. Young social media user often uses a lot of abbreviation, social media-specific slang which might represent hate on social media. People make a lot of spelling mistakes, tried to hide the offensive content by using a different form of words like f*ck, Fuckkkk. As we look at the Hate speech detection method's result on languages other than English are substantially lower than English. One of the reasons for the low score is the lack of proper lexical resources for such languages(Hindi). On many occasions, the social media user writes a native language in Roman script. Kumar et al. [52] have included the Hindi post written in Roman script.

2.3 Impoliteness/Hate Visualization

The reported work on Hate visualization is minimal in the literature. However, few works filter the content from Facebook. In [32], authors built Facebook Inspector (FBI) to filter the malicious contents in real-time using the Random Forest classifier. We have tried to download the plugin from the link [10] [11] given by the author in the paper[12], but the plugin is not available in the repository. [53] built a prototype for the Hate speech visualization. Some of the interface link [13] are available.

2.4 Discussion

In this chapter, we have tried to summarize the research carried out in the area of Microblog Summarization, Hate Speech, and related concepts like aggression detection and offensive content, factual post detection. In this first section of this chapter, we summarize work done in the area of Microblog summarization system built over Twitter. We have presented numerous similarities measures used by various authors. The volume and velocity of the data in Microblog is the biggest challenge for the summarization system.

In the second section of the chapter, we have attempted to provide a structural survey in the area of Hate speech detection and related concept like Cyberbullying, Trolling, Offensive speech. Our objective is to summarize the different problems formulated under the ambit of Hate speech detection. We present that a single level coarse-grained binary classification of Hate speech detection becomes a multi-level fine-grained multi-class classification task [132, 127, 35]. We discussed various Data collection/sampling and Data annotation methods. We have noted that annotation is the most complex task for the Hate speech dataset as the same has been reported by many authors [99]. All the approaches for Hate speech detection discussed in this chapter are based on supervised learning. We found that classification is moving from hand-crafted features to abstractive automatic feature extraction. Character N-grams, word embeddings, and lexicons of offensive words

[10]https://chrome.google.com/webstore/detail/facebook- inspector/jlhjfkmldnokgkhbhgbnmiejokohmlfc

[11]https://addons.mozilla.org/en-US/
refox/addon/fbi- facebook-inspector/

[12]accessed 06/02/2019

[13]https://www.csc2.ncsu.edu/faculty/healey/tweet_viz/

are popular features. We found that classifier based on deep neural model outperforms traditional classifier by 2 to 5 % However, this is not true for each dataset, especially in Hindi and Spanish. On different classifiers, It is tough to find the superior model, while many deep learning-based approaches produce good scores, traditional supervised classifiers may produce similar scores. Transfer Learning also mark its footprint in Hate speech classification problem [126],[101]. However, Results on the Transfer learning model are not consistent for binary and multi-class classification [127]. Results reported in Evaluation metrics are migrating from Accuracy to F1-score to Macro F1-score. Multilingual and Code-mixed Languages are the biggest challenges for the classifier.

CHAPTER 3

Microblog Summarization

Twitter provides a unique multilingual real-time Microblogging platform where users across the different geographic locations post their personal opinion or report any real-time event-related sports, politics, disasters, etc. which unfold over a period of time. Overall, Twitter contains current and vital information across different domains. Large volume and redundant data are the general characteristics of the Twitter feed. However, on many occasions, Twitter was the first media where the event gets reported first. Summarizing such events in tens of tweets might help the user to get informed about the latest development of her interesting topic.

In this chapter, we present our approaches to summarize Microblog from three perspectives/scenarios.

- **Email Digest**: To create a digest, which can be delivered in the form of an email to the user and summarizes the event update on users' interesting topics during the day in tens of tweets. In this scenario, Timeliness is not important but relevant, and novel tweets are expected in the digest.

- **Real Time Push Notification**: In this case, the system delivers real-time push notification on the users' mobile phone, whenever the system detects new update for the topic for which user is interested keeping updated. This scenario is the real-time case of email digest.

- **Summarizing Microblog during Disaster Event**: This is a domain-specific summarization case where the system is required to summarize informative tweets posted during the emergency such as an earthquake, flood.

Table 3.1: Sample digest or Tweet Summary from Twitter

Twitter Sample Stream	After Tweet Filtering	Email Digest
$T_1 ... T_n$	Brazil's president is gathering support from lawmakers as congress considers a corruption probe against him https://t.co/vUSYMztwPy Brazil's president appears safe in vote despite opposition and low rating https://t.co/zYnrtJtoOl Brazil's Temer seen defeating corruption charges in Congress https://t.co/cH0ZIflUH8	Brazil's Temer seen defeating corruption charges in Congress https://t.co/cH0ZIflUH8
$T_1 ... T_n$	Brazil House will decide today if Pres. Temer should be investigated for taking bribes from meat-packing giant JBS: https://t.co/AylXKWiaoo Brazil's president faces congressional vote on his future https://t.co/AiKiNy9w5u Brazil congress to vote on whether to remove president Brazil News https://t.co/vOLiPppSwn	Brazil House will decide today if Pres. Temer should be investigated for taking bribes from meat-packing giant JBS: https://t.co/AylXKWiaoo

3.1 Email Digest

Twitters' registered user can freely access roughly 1% sample of all tweets using the Twitter streaming API. Due to multiple users are reporting the event; on many occasion, the volume of tweets for the trending topics are massive and redundant. It would be very useful if one can summarize the event in a few tens of tweets that describe the whole incident or in other words, create a few tweets digest from thousands of tweets. Table 3.1 shows our system-generated email digest from the topic related to "Brazil president corruption". Figure 3.1 shows the proposed system at a high level.

Some of the topics discussed in social media for a very long time and might get di-

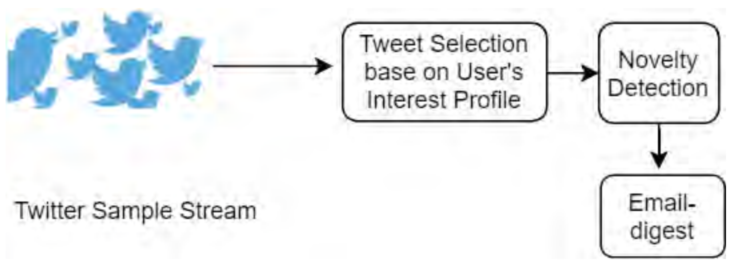


Figure 3.1: Sample Summary Generation from Microblog

verted into other sub-topic. These phenomena called topic drifting. Our system can generate tweet summary for the topic like Refugee crisis in Europe, which is being discussed every day on Twitter for many years or some of the temporal topic like Thanksgiving Day or Independence Day.

To model users' information need, is a very critical task for any summarization system. Limited length of the tweet or data sparseness is the biggest challenge for retrieval of the relevant tweet. In this thesis, we have considered TREC Microblog 2015 [62], TREC RTS 2016 [64] and TREC RTS 2017 [63] datasets for experiments. In these datasets, user information need is modeled using interest profiles similar to topics in ad-hoc retrieval. Interest profiles are consist of three fields: title having 3-4 words, sentence-long description, and paragraph-long narrative depicting specific information need. Henceforth, interest profile, and topic will be used interchangeably in the rest of the chapter.

Summarization System should include relevant and novel tweets in summary for a given interest profile. If there is no relevant tweet for a particular topic on a specific day, then this day is called a silent day for that topic and the system should not include any tweet for that topic [117]. If the system correctly identifies such a silent day, then it should be rewarded with the highest score. If the system includes tweets in summary for the topic on a silent day, it receives score 0 [117] and some metric also penalizes system with respect to proportional to the volume of non-relevant tweets on a silent day [63].

As stated earlier, the primary challenge of the Twitter-based summarization system is to filter relevant tweets against user's information need, in another way, one can say that it is a Tweet filtering problem or Tweet selection problem. Relevance score between tweets and interest profiles is calculated using a language model with JM-smoothing, Dirichlet smoothing, and Okapi BM25 model. Choosing the right smoothing parameter is critical for retrieval as tweets are sparse. The smoothing parameter controls the weight of the probability of a term in the tweet and the corpus collection. We have empirically identified the optimal value for the smoothing parameter λ and μ.

There are two kinds of text in the tweet: tweets' text and external URL text embedded with the tweet. If any word of tweet overlaps with the given interest profiles' query word, the tweet is included in the rank list with some score. We have to set a relevance threshold T_r for each topic to select top relevant tweets. To detect a silent day for a given interest profile, we have also set a silent day threshold T_s. These thresholds are estimated using the previous year dataset. We have estimated these thresholds for TREC RTS 2017 dataset using predictive models that are trained on TREC RTS 2016 Dataset and for TREC 2016 dataset; we have used labeled data of the TREC MB 2015 dataset. These predictive models are based on linear regression, support vector regression. Experiment results show that our estimation techniques predicted these thresholds reasonably well with 95% accuracy.

3.1.1 Research Objectives

In this chapter, experiments are performed on the benchmark dataset with the following objectives

1. Comparing ranking algorithms for the tweet ranking
2. Determine optimal smoothing technique and smoothing parameter for language Model.
3. We propose a threshold estimation technique to estimate the silent day threshold T_s and relevance threshold T_r.

3.1.2 Dataset

Standard benchmark datasets are essential for the reliability of the results of an Information Retrieval system. To evaluate our system on standard benchmark datasets, experiments are performed on TREC RTS 2017, TREC RTS 2016 dataset [64] and TREC 2015 dataset [62]. User information need is articulated as interest profiles.

TREC Dataset

User information need is represented by the Interest profiles. Interest Profiles are given in triplets consist of title, description, and narrative. Table 3.2 shows sample Interest profile.

Table 3.2: Sample Interest Profile TREC RTS 2016

topid	MB229
Title	legalizing medical marijuana
Description:	Find information on U.S. states considering or having legalized medical marijuana
Narrative	"The user is a journalist working on a story about the use of medical marijuana in the U.S. As part of the research for the story, she wants to find information on which states have legalized medical marijuana or are considering doing so. Legislative bills, votes, and proposals for legalizing it are all relevant, as are legal or official public safety issues concerning legalized medical marijuana. However, general public opinions on why it should be legal or not are not relevant.

Table 3.3 describes statistics of all three datasets. There are 11 Interest Profiles that are common between 2015 and 2016 datasets. In TREC RTS 2017, brand new Interest profiles were developed and did not overlap with previous datasets.

Relevance judgments are based on pooling. A common pool had been constructed based on tweets submitted by all participant teams participating in the TREC RTS challenge. First, tweets submitted by the systems were assessed for relevance by the human

Table 3.3: Dataset Statistics

Dataset Detail	TREC RTS 2017	TREC RTS 2016	TREC MB 2015
Number of Tweets	9 Mn	13 Mn	44 Mn
# Interest Profiles for evaluation	97	56	51
Size of Qrels	91000	67525	94066
Number of positive Qrel	6000	3339	8233
Common Interest profiles	0	11 with 2015	11 with 2016
Time Interval	29-11-17 to 05-08-17	02-08-16 to 11-08-16	20-07-15 to 29-7-15

assessors. Tweets were judged as non-relevant, relevant, or highly relevant. Semantically similar tweets are clustered into the same group that communicates identical information.

3.1.3 Formal Problem Statement

Events might get reported by multiple users on Twitter from different parts of the world. The first task of the system is to select top tweets, which are the best candidates to describe the event from the candidate set. Before we add tweets into the summary, we need to ensure novelty against the tweets, which are already added into summary or digest. Therefore, Tweet selection is the primary task for the summarization system. We define the summarization problem formally in the following way.

From the given set of interest profiles Q =$\{Q_1, Q_2, ..Q_m\}$, and Tweets T = $\{t_1, t_2, .., t_n\}$ from the dataset, System should generate a topic-wise summary S = $\{SQ_1, SQ_2....SQ_n\}$. Where SQ_i is the set of tweets which are relevant and novel for the i^{th} topic.

$$SQ_i = \{t_1, t_2, .., t_n\} \; where \; t_i \in T \tag{3.1}$$

Tweets, which are the potential candidates of summary for any interest profile, must satisfy the following constraints.

- Length of the day-wise summary of the Interest profile is $\leq$ 100 tweets
- Sim(t_i ,t_j) $\leq T_n \; \forall \; t_i, t_j \in S_i$ (T_n = Novelty threshold)

3.1.4 Proposed Method

In this section, we will describe our approach in detail. Various threshold estimation techniques are developed to decide the relevance of the tweet with respect to the interest profile. These methods are the major contribution of this work. Figure 3.2 shows flowchart for our system.

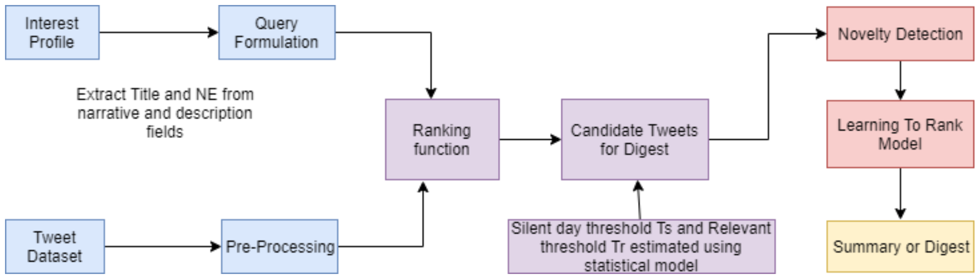

Interest Profile
Query Formulation
Extract Title and NE from narrative and description fields
Ranking function
Candidate Tweets for Digest
Novelty Detection
Learning To Rank Model
Summary or Digest
Tweet Dataset
Pre-Processing
Silent day threshold Ts and Relevant threshold Tr estimated using statistical model

Figure 3.2: System Architecture of Summarization System

Query formulation from Interest Profile

Interest profiles are given in ad-hoc retrieval style as shown in table 3.2, having 2-3 words title, sentence-long description, and paragraph-long narrative which describes information need in detail. To formulate the query, we take all the token from the title field and only consider named entity from the description and narrative field. To convert the topic into the query, we have first removed stop-words. Since tweets are short in length, the named entity plays a significant role in the retrieval of relevant tweets. We run Stanford NE tagger [1] on Interest profiles to retrieve all name entity from narrative and description fields. Table 3.2 shows the structure of the sample Interest Profile and the query generated from the interest profile is: legalizing medical marijuana US.

Tweet Pre-Processing

Tweets are very noisy, and might contains user mentions, Hashtags, Emojis, and URLs. Before indexing, Hashtag symbol # and User mentions are dropped from the tweets. Non-English tweets were dropped using the tweet's language attribute. Non-ASCII characters and stop-words are removed from tweet text. Retweets and tweets less than five tokens are dropped from further processing [79, 82, 111, 81]. Tweets token are stemmed, and the external URL is expanded and added in original tweet content. Lucene PorterStemmer [2] is used for stemming.

[1]https://nlp.stanford.edu/ner/
[2]https://lucene.apache.org/

Summarization

Microblog, like Twitter, poses a stiff challenge for any standard retrieval model due to its massive volume and data sparseness. In 2017, an event like Miss Universe 2017, was the most tweeted hour of the year [3] where Twitter users posted millions of tweets from the different parts of the world. Many of these tweets were redundant and non-relevant. In other words, a lot of these tweets did not communicate substantial information or any new update regarding the event. Summarizing such events in tens of tweets might be enough to report the entire event. Ideally, there are three primary tasks of the summarization system :

- Tweet ranking
- Threshold estimation for relevance between tweets and interest profiles
- Novelty detection or cluster formation from the relevant tweets.

Tweet Ranking: To calculate relevance score between Interest profiles and tweets, we have considered three retrieval models namely: query likelihood model with JM smoothing, Dirichlet smoothing [23] and Okapi BM25 ranking model to rank tweets based upon relevance score. We have empirically set smoothing parameter λ for JM-smoothing and μ for Dirichlet Smoothing using a grid search on TREC RTS 2016 dataset.

We studied different smoothing techniques for the language model for tweet ranking. As stated earlier, Tweets are very sparse in nature, and smoothing is one way to combat the data sparsity issue. Using MLE (maximum likelihood estimate) unseen term gets zero probability. Smoothing is a technique that assigns some non-zero probability to terms that were unseen in the tweet. Due to Smoothing, the probability mass is divided over more terms; hence, the probability distribution becomes more smooth. In this study, one of the objectives is to compare smoothing techniques for the language model which maximizes evaluation metrics.

Jelinek-mercer smoothing: This technique is the linear interpolation between the foreground and the background model [23]. λ is the parameter which controls the weight

[3]https://www.adweek.com/digital/twitter-year-in-review-2017/

of the two model.

$$P(W|T) = (1-\lambda).\frac{c(w,T)}{|T|} + \lambda.P(w|c) \tag{3.2}$$

Dirichlet Smoothing: A unigram language model is a multinomial, for which the conjugate prior is the Dirichlet distribution [23]. This is defined as follows, parameterized with μ:

$$P(W|T) = \frac{c(w,T) + \mu P(w|C)}{|T| + \mu} \tag{3.3}$$

Threshold Estimation techniques: Ideally, the summarization system includes relevant tweets in the summary or digest. The ranking algorithm would return the similarity score in terms of a numeric number between topic and tweets if the single token of the tweet matched against the query formulated from the interest profile — more the similarity higher the score. Therefore, we need to choose a threshold that can minimize false-positive judgments in this case. From the empirical analysis, it has been found that by and large relevant tweets score higher than non-relevant tweets but not always. After doing a careful analysis on the datasets, we found that non-relevant tweets have scored more than relevant tweets on many occasions. Furthermore, there are many silent days for interest profiles. We also saw that on an eventful day, some of the tweets score quite high due to some event and there are many low scoring relevant tweets that had posted to give more updates regarding the main event.

Interest Profiles can have two types of days: silent day and eventful day [117]. On an eventful day, Interest Profiles have some relevant and novel tweets. On the contrary, if there is no relevant and novel tweet for the given interest profile on a particular day, the day is considered as a silent day for that interest profile. On a silent day, the system should get rewarded if it does not include any tweet in summary for that interest profile or penalize otherwise. Detecting a silent day for a profile is a critical task for the summarization system.

We have set up two thresholds: the relevance threshold T_r and the silent day threshold T_s . For a given day, in the rank list of given Interest profiles, if all the tweets score less than given silent threshold T_s, we will consider this day as a silent day. Else, we will

Table 3.4: Feature engineering for Threshold Estimation Model

Sr no	Features	Description (Day-wise, Interest profile-wise)
1	Tweet_ max_score	Maximum score of the tweet in the rank list
2	Tweet_ stdv_score	Standard deviation of the tweets in rank list
3	Tweet_ mean_ score	Mean score of the tweets in the rank list.
4	Tweet_median_score	Median score of the tweets in the rank list
5	Tweet_min_score	minimum score of the tweet in the rank list

consider all the tweets score greater than T_r where $T_s \geq T_r$.

The major challenge is how well we estimate these thresholds. The proposed threshold estimation method is the major contribution of this work. All the previous work [114, 59, 9] used an empirical way to fix the threshold. Thresholds for the TREC RTS 2016 Interest profiles' predicted using TREC MB 2015 dataset, and TREC RTS 2017 Interest profiles' thresholds predicted using TREC RTS 2016 dataset. We have performed experiments on the TREC MB 2015 dataset and TREC 2016 with the range of thresholds. We applied topic wise grid search on results and figured out topic-wise silent threshold T_s and relevance threshold T_r, which produced the best results. We have built threshold predictive models using linear regression and support vector regression. We process the day-wise rank-list of each interest profile and considered features as shown in table 3.4, to build a linear regression model and a support vector regression model. Algorithm 1 contains pseudo-code for our Microblog summarization method.

Algorithm 1 Tweet Summarization Algorithm

```
1:  procedure TweetSummary($Ranklist, model$)   ▷ Tweet Ranklist of Interest Profile
2:      $T_s \leftarrow model(Ranklist)$   ▷ Estimate Silent day threshold $T_s$ from model
3:      $T_r \leftarrow model(Ranklist)$   ▷ Estimate Relevance threshold $T_r$ from model
4:      $Tw_1..Tw_n \leftarrow Ranklist$   ▷ similarity score of the each tweet to Interest Profile
5:      $Tw_{max} \leftarrow \text{argmax}(Tw_1,..,Tw_n)$   ▷ Select tweet with maximum score from the ranklist
6:      if $Tw_{max} \geq T_s$ then
7:          $Candidates \leftarrow \{Tw_i \mid \forall Tw_i, Tw_i \geq T_r\}$   ▷ select all tweet from Ranklist $Tw_i \geq T_r$
8:          for each $Tw_i \in Candidates$ do
9:              $Tw_j \leftarrow SQ_i$   ▷ Tweets already in summary
10:             if $similarity(Tw_i, Tw_j) < T_n, \forall Tw_j \in SQ_i$ then   ▷ $T_n$ Novelty Threshold
11:                 add $Tw_i$ in to day summary of $SQ_i$
12:     else
13:         $SQ_i \leftarrow \phi$   ▷ Silent day no tweet added to summary
```

Table 3.5: Feature engineering for Learning to Rank Model

Sr no	Features	Description
1	LM_ JM_ score(Q,T)	Language model score of tweet against query using Jelinek-Mercer smoothing.
2	LM_Diri_score(Q,T)	Language model score of tweet against query using Dirichlet smoothing
3	BM25 score(Q,T)	BM25 score of tweet against(Query,Tweet)
4	Count(Q,T)	No of common token between tweet and Query
5	Ext_URL	No of External URL embedded in Tweet

Learning to Rank Method Learning to rank [67, 57] is the application of the machine learning techniques for training the model in a ranking task. Learning to rank had proved its usefulness for many applications in the area of Information Retrieval, Natural Language Processing. Document summarization, document retrieval, collaborative filtering, keyphrase extraction are the widespread applications of learning to rank [58].

Since the tweet summarization problem is perceived as a tweet selection problem, learning to rank strategy is used to combat the ranking problem. Learning to rank is a supervised learning technique, so we have used TREC RTS 2016 [64] as training data. We have implemented learning to rank using a point-wise and pair-wise approach. We have considered features listed in table 3.5 to train our neural network using TREC RTS 2016 dataset which implements learning to rank point-wise approach. Model parameters are as follows: learning rate= 0.01,No of epoch = 100, no of hidden layer = 3. Pair-wise learning to rank approach implemented using SVM ranker [4]. In the result section, we will discuss the improvement in the results by employing the Learning to Rank strategy.

Novelty Detection Tweets, in summary, should be novel with respect to all. Before we add any relevant tweet for the interest profile summary, its similarity has to be checked with all tweets which are already in summary. We have used Jaccard similarity to check the similarity between two tweets.

$$Jaccard(t_i, t_j) = \frac{|t_i \cap t_j)|}{|t_i \cup t_j|} \tag{3.4}$$

where t_i is set of tweets which are already part of interest profile summary and t_j is

[4]https://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

current relevant tweets. We have empirically set novelty threshold $T_n = 0.4$ using a grid search on TREC RTS 2016 dataset. If the novelty score of $t_j < 0.4$ with respect to all tweets, in summary, t_j will be added into the summary of that interest profile.

3.1.5 Results

In this section, we will discuss the evaluation metrics used for the experiment and report results obtained from the datasets. We will discuss the training and test dataset used in the experiments. Finally, we propose optimal smoothing parameters for the language model, which we have identified empirically.

Evaluation metrics

In this study, experiments are performed on standard benchmark datasets like TREC MB 2015, TREC RTS 2016, and TREC RTS 2017 to evaluate our system performance. Normal discounted Cumulative gain, nDCG@10 is computed for each day for each interest profile and is averaged across them [64].

MAP (Mean Average Precision) and nDCG (Normalized Discounted Cumulative Gain) are the two most popular ranking metrics. The main difference between the two is that MAP assumes binary relevance (tweet is relevant or not), while nDCG allows relevance scores in the range of real numbers. The relevance measure of the tweet is graded relevance. Tweets were assessed on a three-way scale of "not relevant", "relevant", and "highly relevant". So nDCG as a standard evaluation metric is more sensible than the MAP.

Three variants of nDCG, namely: nDCG-1, nDCG-0, nDCG-p were used as evaluation metrics. In nDCG-1, on a silent day, the system rewarded with the highest score 1 if it does not include any tweet in summary for the particular interest profile and gets 0 otherwise. However, in nDCG-0 for a silent day, the system receives gain zero irrespective of the number of tweets included in the interest profile summary [117]. In nDCG-p, on a silent day, if the system includes a tweet in summary, a penalty will be incurred based upon the volume of the tweet on a silent day.

Table 3.6: Results on TREC RTS 2016 dataset using Threshold estimation Technique and Grid Search

Ranking functions	Using LR		SVR		via Grid Search	
	nDCG-1	nDCG-0	nDCG-1	nDCG-0	nDCG-1	nDCG-0
LM with JM smoothing	0.293	0.0662	0.2967	0.0628	0.3317	0.0978
LM with Dirichlet smoothing	0.2607	0.0964	0.3016	0.0891	0.3544	0.1101
Okapi BM25	0.2483	0.0929	0.2886	0.0886	0.3524	0.1131

Training and Test Dataset

We have used labeled data of TREC MB 2015 [62] to train our model to estimate silent day threshold T_s and relevance threshold T_r for TREC RTS 2016 dataset. Similarly, labeled data of TREC RTS 2016 [64] is used to train model to estimate threshold T_s and T_r for TREC RTS 2017 dataset [63].

Results on TREC RTS 2016 Dataset

Table 3.6 shows the results of the language model with different smoothing techniques and standard ranking algorithm Okapi BM25. It shows the results computed using different threshold estimation techniques based on linear regression and support vector regression. Table 3.6 also shows result where Silent day threshold T_s and T_r and smoothing parameter λ and μ are computed using grid search. We anticipated these parameters identification problem as a hyperparameter optimization problem. nDCG-1 and nDCG-0 are the standard evaluation metrics for the TREC RTS 2016 dataset.

Table 3.7 shows our results comparison with top TREC RTS 2016 team [59] [114]. nDCG-1 was marginally better but nDCG-0 is substantially better than [59, 114]. It is important to note that 11 interest profiles are common between above datasets. Table 3.8 display the reported nDCG-0 and nDCG-1 in the literature on TREC RTS 2016 dataset.

Results on TREC RTS 2017 Dataset

Table 3.9 shows results on TREC RTS 2017 dataset. nDCG-p was the primary metric that replaces nDCG-0. Again labeled data of TREC RTS 2016 is used to train our model to

Table 3.7: Result Comparison with TREC RTS 2016 top team

Metrics	Results Via Grid Search	Result using Threshold Estimation Technique	COMP2016	Qatar Research	Blank run
nDCG-1	**0.3544**	**0.3016**	0.2898	0.2621	0.2339
nDCG-0	**0.1101**	**0.0891**	0.0684	0.030	0

Table 3.8: Reported nDCG in literature on TREC RTS 2016 Dataset

Team	nDCG-1	nDCG-0
COMP2016 [59]	0.2898	0.0684
NUDTSNA [64]	0.2708	0.0529
QU [114]	0.2621	0.0300
IRIT [21]	0.2481	0.0321
WaterlooLin [64]	0.2352	0.0299
PKUICST [64]	0.2348	0.0151
Empty[64]	0.2339	0.0000
prna [64]	0.2334	0.0352
ISIKol [64]	0.2213	0.0196
udel [64]	0.2151	0.0008
IRLAB [79]	0.1972	0.0169

estimate silent day threshold T_s and relevance threshold T_r for TREC RTS 2017 dataset. It is worth to mention that there is no overlap of Interest profile with previous datasets.

Table 3.10 shows our results comparison with top TREC RTS 2017 team. Table 3.11 present result on Learning to Rank strategies.

Table 3.12 display the reported nDCG-p and nDCG-1 in the literature on TREC RTS 2017 dataset.

Table 3.9: Results on TREC RTS 2017 dataset using Threshold estimation Technique and grid search

Ranking functions	using LR		using SVR		via Grid Search	
	nDCG-p	nDCG-1	nDCG-p	nDCG-1	nDCG-p	nDCG-1
LM with JM smoothing	0.2570	0.2435	0.2545	0.2431	**0.3531**	**0.3386**
LM with Dirichlet smoothing	0.2557	0.2479	0.2524	0.2416	**0.3790**	**0.3570**
Okapi BM25	0.2158	0.2130	0.2295	0.2262	0.3482	0.3446

Table 3.10: Result Comparison with TREC RTS 2017 Top Team

Metrics	Via Grid Search	Using Threshold Estimation Technique	Median of all submitted run	blank run	PKUICST
nDCG-p	**0.3790**	0.2557	0.2194	0.1765	**0.3483**
nDCG-1	**0.3570**	0.2479	0.1865	0.1765	**0.3003**
nDCG-0	**0.1933**	0.1847	NA	0	0.1688

Table 3.11: Results on Learning Rank strategies on TREC RTS 2017 dataset

Metrics	Learning to Rank point-wise	Learning to Rank pair-wise
nDCG-p	0.3062	0.2562
nDCG-1	0.2148	0.2483
nDCG-0	0.1530	0.1852

Table 3.12: Reported nDCG in literature on TREC RTS 2017 Dataset

Team	nDCG-p	nDCG-1
HLJIT [45]	0.3656	0.2910
PKUICST [63]	0.3483	0.3003
udel_fang [63]	0.2933	0.2775
udel [104]	0.2808	0.2329
PRNA [56]	0.2752	0.2400
NOVAsearch [42]	0.2710	0.2587
advanse [84]	0.2669	0.2289
ICTNET [122]	0.2185	0.1527
IRIT [22]	0.2142	0.1833
umd-hcil [63]	0.1863	0.1747
BJUT [74]	0.1796	0.1456
Emptyrun [63]	0.1765	0.1765
ISIKol [63]	0.1725	0.1725
ST [63]	0.1551	0.0741
SOIC [40]	0.1442	0.1442

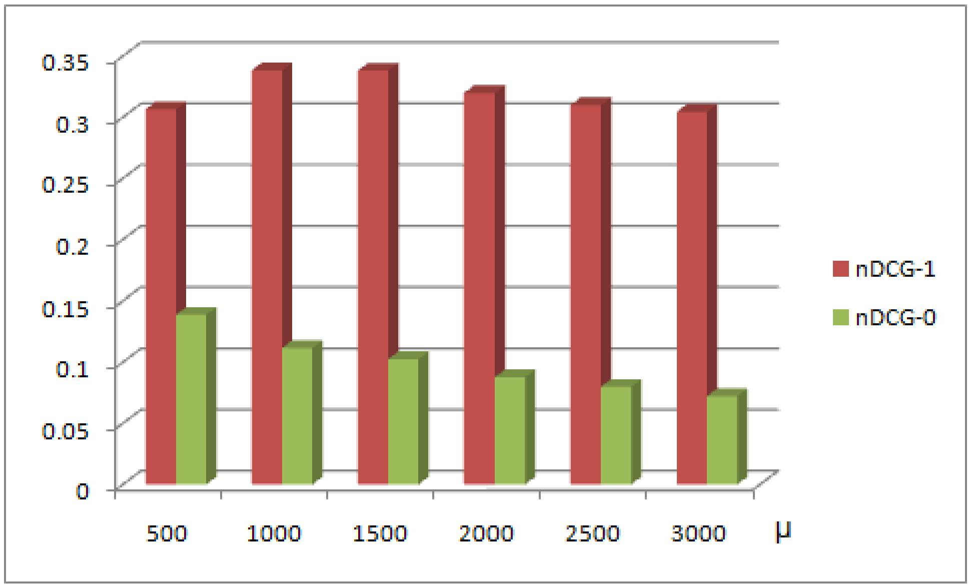


Figure 3.3: Effect of μ on the nDCG on TREC RTS 2016 dataset

Optimal Smoothing Parameters for Language Model

One of the objectives of this work is an identification of optimal smoothing parameter value for the smoothing techniques. As discussed in the previous section, we have set up two thresholds: T_s, silent day threshold T_r relevance threshold. We anticipated these parameters identification problem as the hyperparameter optimization problem. We have set smoothing parameter λ for JM smoothing and μ for Dirichlet smoothing using grid search. Figure 3.3 and figure 3.4 show the effect of these smoothing parameters on the evaluation metrics. We conclude that λ=0.1 for JM smoothing and $\mu = 1000$ for Dirichlet smoothing is the optimal value which maximizes overall evaluation metrics nDCG-p, nDCG-1, and nDCG-0.

3.1.6 Analysis on Results

In this sub-section, we analyze the results reported in the previous sub-section 3.1.5. After careful analysis of the evaluation metrics nDCG-0, nDCG-1,nDCG-p, we have concluded the followings.

- nDCG-0 neither penalizes the system if it includes tweet in the topic summary on the silent day of the topic nor rewards system if the system remains correctly identify the silent day. On an eventful day, nDCG-0 shows the quality of the topic summary

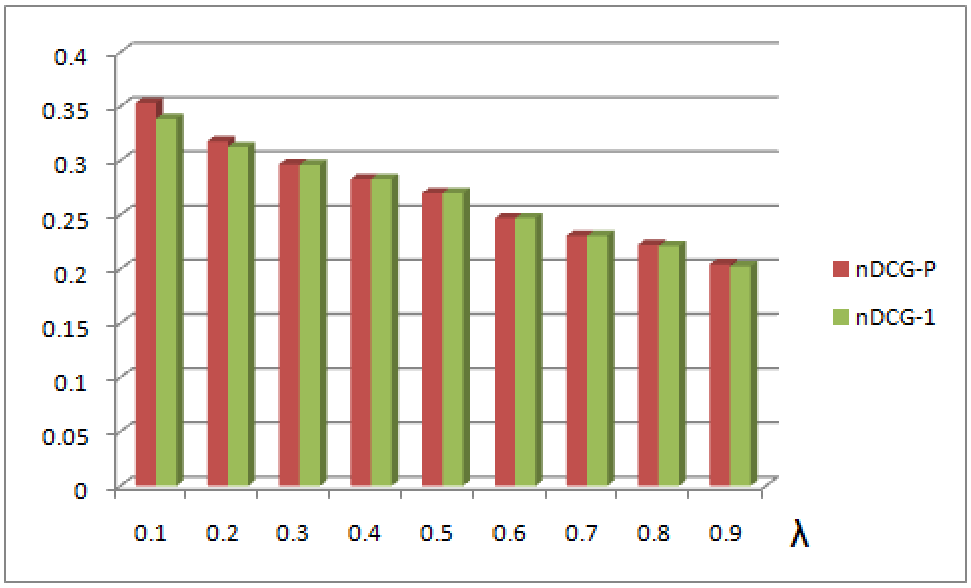


Figure 3.4: Effect of λ on nDCG on TREC RTS 2017 dataset

generated by the system. In other words, nDCG-0 shows actual relevant, and novel tweets are part of the daily digest.

- nDCG-1 includes a gain of 1 if the system does not include any tweet on the topic's silent day otherwise 0. on an eventful day it adds the same gain as nDCG-0. It reflects the gain on a silent day plus the gain on an eventful day(essentially nDCG-0).

- On a silent day of the topic, nDCG-p will penalize the system proportional to tweets volume in summary.

During the analysis of the results on the TREC RTS 2016 dataset [64], we found that empty run, i.e., blank file with zero tweets scored $nDCG-1 = 0.2339$ and $nDCG-0 = 0$ which is substantially more than the median score of all the teams participated in TREC RTS 2016 [64]. Therefore, we argue that $nDCG-1$ is not a very accurate measure for evaluating the summarization system. Table 3.6 and Table 3.7 shows our results on TREC RTS 2016 dataset. There are 56 Interest profiles evaluated for 10 days. So, Total no of days is 560. Out of 560 days, 131 days were silent days; almost 23%. Top team COMP2016 team [59] receive score nDCG-1 = 0.2898 and nDCG-0=0.0684. So it shows that 76% score of nDCG-1 obtained by the system is by remaining silent. The objective of the summarization system is to include tweets in summary. Hence, a good nDCG-0 score reflects the quality of the digest or summary. We report our best result with $ndcg - 1 = 0.3544$ in the language model with Dirichlet smoothing for $\mu = 1000$ without any sort of query expansion. Performance of Language model and standard retrieval algorithm like BM25 are moreover less inline. Our grid search results and result using threshold estimation techniques outperform Comp2016 (rank 1 in TREC RTS 2016 track) team [59] whose $nDCG - 1 = 0.2898$ and $nDCG - 0 = 0.0684$. Improvement in nDCG-0 shows the quality of the Interest profile summary. We found that by effectively choosing parameter λ and μ, we can outperform the result obtained by [114]. The factor behind this outperformance is selection of smoothing parameters $\lambda = 0.1$ and $\mu = 1000$ while [114] have set $\lambda = 0.7$ $\mu = 2000$.

IN TREC RTS 2017, track organizers have accepted arguments of [96] in which, authors claim that nDCG-0 is the flawed metric. They have dropped evaluation metric nDCG-0 and introduced nDCG-p, which is lenient than nDCG-1. nDCG-p penalize if the system includes tweets in the summary or digest on a silent day for a given topic in proportional to the volume of non-relevant tweets. So on an eventful day, both metrics give the same value On the contrary, if the system does not detect the silent day nDCG-1 receive 0 but nDCG-p receive score proportional to the volume of the tweet in summary. Table 3.9 shows that empty run or blank file with zero tweet scores nDCG-1 and nDCG-P equal to 0.1765. There are 97 Interest profiles evaluated for 8 days. So, the total no of days is 776. Out of 776 days, 137 days was a silent day, which is almost 18 percent. Table 3.10 shows our result computed using grid search and our threshold estimation model based on linear regression and support vector regression trained on labeled TREC RTS 2016 dataset. Our results using the threshold estimation technique are well above than the median of the results on TREC RTS 2017. In fact, our system achieves the best result in nDCG-0 metric compare to team PKUICST [131] and HLJIT [46] which are the top team in TREC RTS 2017 [63]. To further improvise the results, different learning to rank strategies is employed. Table 3.11 shows results of both these strategies. Point-wise learning to rank improve the score of nDCG-p but other metric nDCG-1 and nDCG-p decrease marginally. In pair-wise learning to rank, marginal improvement seen in all the metrics.

3.2 Real Time Push Notification from the Social Media

Consider a scenario, where the user wants to remain to get updated about the latest development about her interesting topic on a real-time basis. The system delivers real-time push notification on the users' mobile phone, whenever the system detects new update from Microblog against to topic. One can access or download a sample (approximately 1%) from the public tweets using Twitter streaming API. Figure 3.5 shows a schematic diagram of the real-time push notification system.

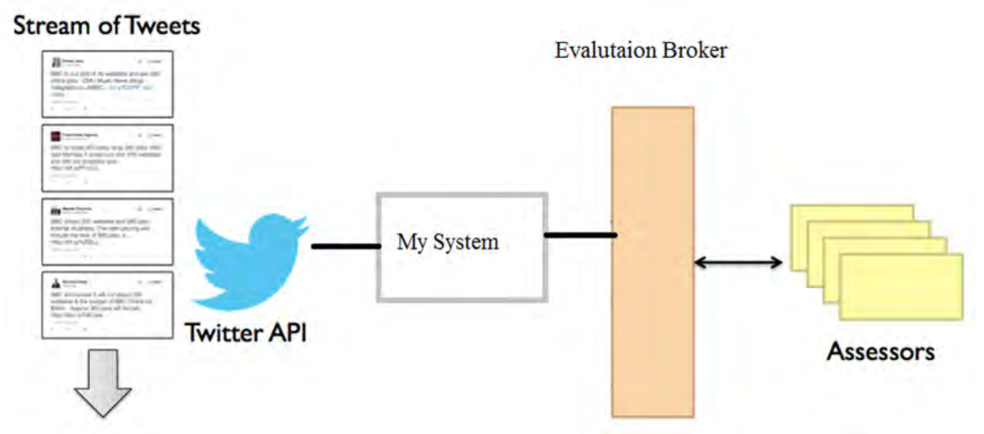


Figure 3.5: Real Time push notification: workflow

Push notification from Microblog is the real-time version of the email digest scenario where timeliness or latency is important, i.e., how fast the system can deliver relevant tweets on users' mobile phones. Query formulation, tweet, pre-processing, similarity measure, summarization method remain the same except for the evaluation scheme and metrics.

3.2.1 Evaluation Schemes

There are two independent evaluation method for the real-time push notification : (i) Live User-in-the-Loop Assessments, (ii) post hoc batch evaluation. [93] proposed a new framework for the live assessment of the push notification.

Live User-in-the-Loop Assessments

In this evaluation method, As soon as the participating system pushes the tweet to the evaluation broker, tweets are sent to the human assessor over the phone via push notification. The assessors have a choice either submit relevance judgment immediately or put the tweet in the waiting queue. [93] claimed that their proposed evaluation method has many advantages over traditional post hoc batch evaluation as it captures the live feedback from the human assessors. The push notification system can exploit this feedback to tweak the parameter to make a more reliable system.

Post hoc batch Evaluation

This is a traditional method for evaluation where tweets are manually assessed by the human in offline batch mode using a common shared pool.

3.2.2 Evaluation Metrics

In this sub-section, We present the metrics used in the evaluation of experiments. All metrics are computed for each day for each interest profile and averaged. Standard IR measures, like precision and recall, are used for the live assessment. [117] has proposed a new set of evaluation metrics for the real-time summarization case in post hoc batch evaluation.

Expected Gain (EG)

[64] has defined Expected Gain (EG) as follows:

$$EG(t) = (1/N) \sum G(t) \tag{3.5}$$

Where N represents the total tweet pushed by the system, and G(t) is the gain associated with each tweet. The gain G(T) is 0 non-relevant tweet, for the relevant and novel tweet and 1 for the highly relevant and novel tweet. EG is analogous to precision.

Normalized Cumulative Gain (nCG)

[64] have proposed Normalized Cumulative Gain (nCG) is defined as follows:

$$nCG(t) = (1/Z)\sum G(t) \tag{3.6}$$

where Z is the maximum possible gain system can achieve. nCG is analogus to Recall.

Gain Minus Pain (GMP)

Gain Minus Pain (GMP)[64] is defined as follows:

$$GMP = \alpha * G - (1-\alpha) * P \tag{3.7}$$

Where G is the gain as describe previously, while p (pain) refers to the number of non-relevant tweets that are pushed by the system. α controls the weight of Gain and Pain

Summarization System should include relevant and novel tweets in summary for a given interest profile. If there is no relevant tweet for a particular topic on a specific day, then this day is called a silent day for that topic and the system should not include any tweet for that topic [117]. If the system correctly identifies such a silent day, then it should be rewarded with the highest score i.e., 1. If the system includes tweets in summary for the topic on a silent day, it receives score 0 [117] and some metric also penalizes system with respect to proportional to the volume of non-relevant tweets on a silent day [63].

There are two variants of EG and nCG. EG-1 and nCG-1 reward the system with the highest score of 1 if the system does not include tweets on a silent day. The EG-0 and nCG-0 do not reward the system for recognizing silent days. It gives 0 scores on a silent day.[63] proposed a slightly different approach to handle the silent day. Authors have introduced a new metric variant EG-p and nCG-p which replace EG-0 and nCG-0. p stands for proportional. If the system pushes zero tweets on a silent day, the EG-p receive score 1; same as EG-1. EG-p penalizes the system in proportion to the volume of the non-relevant tweet on a silent day. i.e., if system pushes 5 tweets on a silent day, EG-p around 0.5. but EG-1 is zero.

3.2.3 Results

The push notification system has continuously monitored the Twitter stream for 10 days. We had to face some glitches related to electricity, internet. Table 3.13 shows our system [79] results on TREC RTS 2016 data in Live User-in-the-Loop Assessments in evaluation mode. Our team ranked 25 among all the 44 teams across the world [64]. Tweets were judged by more than one assessor. Many of the tweets remain un-judged, as shown in table 3.13.

Table 3.13: Results on TREC RTS 2016 Dataset-Live Assessment

Relevant	Redundant	Non_Relevant	Unjudged	Total_length
105	10	259	1721	2083

Table 3.14 shows our system [79] result on TREC RTS 2016 data using post hoc batch assessments.

Table 3.14: Results on TREC RTS 2016 Dataset - Post Hoc batch evaluation

EG1	EG0	nCG1	nCG0	GMP.33	GMP.5	GMP.66	mean latency
0.1708	0.0440	0.1546	0.0278	-0.7448	-0.5397	-0.3467	176709.4

Table 3.15 shows our system results on TREC RTS 2017 dataset. Our Team ranked 21 among all the 42 teams across the world. We have submitted two runs for this track.

Table 3.15: Results on TREC RTS 2017 Dataset-Live Assessment

Run-id	R	D	N	U	l	P	Utility
irlab1	565	122	935	57	900	0.424	-248
irlab2	640	197	2015	231	1798	0.2935	-1178

R: Relevant, D:duplicate,N:non-relevant,U:unjudged,l:total pushed tweet, p:precision , Utility = N-R-D

In 2017, the track organizer dropped some of the evaluation metrics such as EG-0 and nCG-0 and introduced some of the new evaluation metrics like EG-p and nCG-P. Table 3.16 shows our system result on TREC RTS 2017 data in post hoc batch assessments mode. Our system ranked 26 among all the 42 teams across the world [63].

Table 3.16: Results on TREC RTS 2017 - Post Hoc batch evaluation

run-id	EG-p	EG-1	nCG-p	nCG-1	GMP.33	GMP.5	GMP.66	mean
irlab1	0.2065	0.1774	0.1929	0.1638	-0.1156	-0.0696	-0.0263	72250
irlab2	0.1998	0.1617	0.1932	0.1551	-0.5084	-0.3634	-0.2269	71463

3.3 Summarization during Disaster Event

Many incidents in the past have proved that social media is the first medium through which news related to a disaster like earthquakes reach to the people. Recently, many earthquake incidents have been reported first on Twitter and then on any other media[105]. Twitter can be effectively accessed by an NGO/Government agency to assess the ground reality of the disaster area to assist in their rescue operations. The motivation behind this work is to explore IR methodologies that can be used to extract important information from social media during emergency events. The dataset is divided into two parts. Tweets posted during the first day of the Italy earthquake are included in the first part. The second part contains tweets posted during the second day of the Italy earthquake, were provided. Topics are articulated in the TREC style for which we have to extract and summarize relevant tweets. Total 52469 tweet-ids in level-1 and 19751 tweet-ids in level-2 along with 5 topics in the TREC format. for example "What resources are available" was the first topic in the dataset. The system has to retrieve and summarize tweets containing any resource-related information.

This is the domain-specific version of the summarization system discussed in the previous section. Topics in the dataset are very general. To covert the topic into a query, we have first removed stopwords. We run Stanford POS tagger [5] on topics. All keywords with the noun and verb labels are extracted and added to the query. We have used lexical database WordNet[6] for query expansion. For each term in a query, we have extracted the top 2 synonyms from WordNet and added to the query. We have set equal term weight for the original term and the expanded term. Tweet pre-processing, similarity measure, summarization method remain the same except for the evaluation scheme and metrics.

[5]http://nlp.stanford.edu:8080/parser/

[6]https://wordnet.princeton.edu/

Table 3.17: Results on SMERP 2017 Dataset level 1: Tweet Retrieval

Our system	bpref	precision@20	recall@1000	MAP
irlab2	0.3171	0.2250	**0.3171**	0.0417
irlab1	0.3074	0.2125	0.3015	0.0391
DCU-top team	0.6170	0.4125	0.1794	0.0517

Table 3.18: Results on SMERP 2017 Dataset level 2: Tweet Retrieval

Our system	bpref	precision@20	recall@1000	MAP
irlab2	0.2869	**0.3750**	**0.2869**	**0.0635**
irlab1	0.2869	0.2875	0.2869	0.0571
DCU-top team	0.7767	0.2125	0.2378	0.0600

3.3.1 Experiment Results on Retrieval

Relevant tweet retrieval is the primary task of the summarization system. Tweet retrieval task aims to retrieve top relevant tweets with respect to each of the specified topics with high precision and high recall. Table 3.17 and Table 3.18 shows results on SMERP level 1 dataset and level 2 dataset along with peer comparison [111, 13].

Standard TREC metrics like bpref, precision@20, recall@1000, and MAP are used to evaluate the results. Table 3.17 and Table 3.18 show our result in both levels dataset. In level 1 dataset, we have achieved higher Recall@1000 compared to top team dcu_ADAPT_run2. However, our bpref was substantially lower than dcu_ADAPT_run2. In the second level dataset, we have achieved Precision@20, Recall@1000, and MAP better than dcu_ADAPT_run2, but we have reported Bpref substantially lower. We will investigate poor Bpref in the future.

3.3.2 Experiment Results on Summarization

The main objective of the system is to summarize the Italy earthquake event using Twitter posts retrieved in the previous part across the different topics. Topics essentially represent different information needs, required in a disaster-related event like an earthquake. We have anticipated Text summarization as a clustering problem. Our approach is based on extractive summarization. Our method remains the same as discussed in the previous section. Table 3.19 and Table 3.20 present summarization results on both dataset along with peer comparision [13, 111].

Table 3.19: Results on SMERP 2017 Dataset level 1: Text Summarization

Our system	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-SU4
irlab2	0.3309	0.1543	0.3085	0.1055
IIEST top team	0.5109	0.2824	0.4885	0.2329

Table 3.20: Results on SMERP 2017 Dataset level 2: Text Summarization

Our system	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-SU4
irlab2	0.3515	0.1297	0.3254	0.1194
IIEST top team	0.5540	0.2436	0.5142	0.2864

3.4 Conclusion

In this chapter, we present a Microblog summarization system built over Twitter and evaluated on standard benchmark datasets. Our results show that the language model with Dirichlet smoothing marginally outperforms standard ranking algorithm like Okapi BM25 by selecting the optimal value of parameter μ. Therefore, we conclude that Dirichlet smoothing is the ideal smoothing technique for language model in case of Microblog retrieval and optimal value for its smoothing parameter is $\mu = 1000$. In the case of JM smoothing, the optimal value of its smoothing parameter is $\lambda = 0.1$. Our proposed threshold estimation techniques to estimate silent day threshold T_s and relevance threshold T_r perform reasonably well with 95% accuracy on average. However, if we compare results computed using estimated thresholds with results calculated using thresholds determined via grid search, results with estimated thresholds are substantially lower than results with grid search thresholds. Nevertheless, our results with estimated thresholds are better than the top team of TREC RTS 2016 [64] and substantially outperform one of the metric nDCG-0 and median of all the metrics of TREC RTS 2017 [63]. It is worth to note that in TREC RTS 2016 [64] dataset, 11 Interest profiles are common with TREC MB 2015 [62] dataset. In TREC RTS 2017 dataset, a set of brand new Interest profiles was developed. Therefore our threshold estimation techniques perform well on [62] dataset and reasonably well for 2017 datasets. In the future, we will try to investigate this underperformance. Furthermore, In post-processing, pair-wise learning to rank strategy marginally improvise performance while point-wise learning strategy drops many non-relevant tweets from the summary along with some of the relevant tweets during post-processing.

CHAPTER 4

Analysis On Microblog based Summarization System

Sensing Microblog from retrieval and summarization become the challenging area for the Information retrieval community. In this chapter, we present the comprehensive failure analysis performed on our proposed summarization system from various perspectives.

Microblog become popular social media to disseminate or broadcast the real-world event or opinion about the event of any nature. As of 2019, Twitter has 330 million active users across the world[1]. With this large user-base, Twitter is an interesting data source for real-time information. On many occasions, it has been observed that Twitter was the first media to break the event. Thousands of users across the world geography interact on the same topic or interest profiles on Twitter with diverse views.

4.1 General Challenges

We will start our analysis with the major general challenges for Microblog summarization.

- Since Twitter imposes a limitation on the length of a tweet, it becomes challenging for the retrieval system to retrieve tweets without the proper context. So tweet sparseness is a critical issue for the retrieval system.
- Many topics, the volume of the tweet is substantial. Most of the tweets are redundant and noisy.
- On Twitter, Some of the topics are being discussed for a longer period. They also

[1]https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/

diverted into many subtopics (e.g., demonetization in India, Refugee in Europe). It is challenging to track topic drifting for an event. To track topic drifting, one has to update the query vector by expanding or shrinking the query term.

- Tweets often include abbreviation (e.g. Lol, India written as ind), smiley, special characters, misspelling (tomorrow is written like 2moro). Tweet normalization is the biggest issue for microblog processing.
- On many occasion, it has been found that native language tweets are in transliterated romanized English

4.2 Interest Profiles

As discussed in the previous chapter, User potential information need is articulated in Interest Profile. Interest Profiles are consist of 3-4 word title, sentence-long description, and paragraph length narrative explaining detailed information need [64]. The crucial part is how we generate a query from triplet as shown in table 4.2. Tan et al. [116] reported that the title keyword plays a critical role in retrieval; our experiments also support these findings. The objective of the summarization system is to identify all the clusters (or events related to interest profile) formed across the given period for all the interest profiles and should not include any tweet if the given day is silent for any interest profile. Performance of Summarization system depends upon 2 tasks : (i) Tweet ranking (ii) Novelty detection across relevant tweets.

In this section, we list the different issues related to Microblog retrieval and summarization.

4.2.1 Spatial Restriction on Interest Profile

During post-hoc analysis, It has been observed that Interest profiles have different characteristics. Some of the interest profiles have a spatial restriction. For example, bus Service to NYC, gay marriage laws in Europe, job training for high school graduates US, Zika in Ecuador, Real estate in London, Interest rate in United States.

Table 4.1: Sample Interest Profile : RTS 225

{topid : "RTS225" Title : "Real estate in London" Description : "Prices and other issues regarding real estate in London." Narrative : "Grenfell Tower fire in West London has revealed many problems with housing in London, which are not just high prices. The user is interested about the fire security in newly built houses in London, and real estate prices in general, as she is planning to buy a flat in London " }.

Table 4.1 shows a sample interest profile [2] where the information required from the particular London city. Our system has returned many false positive for this profile. The following are examples of such tweets.

T_1*: "Real estate prices in Killara are knackered due to colourful sick cnts."*

T_2*: "Home prices in LA County likely to rise 40 % more in next four years!"*

It is very difficult to retrieve the following relevant tweets because it's similarity score with interest profile are very low.

T_3*: "Prime central London sales rebound after stamp duty changes last year."*

However, we are able to retrieve the following tweets and added into the summary.

T_4*: "How house prices in East London have DOUBLED since 2012... three times faster than the UK average."*

4.2.2 Vague or general Interest Profile

Generalized Interest profiles have many silent days; user information is either generalized or too vague. E.g., emerging music styles, adult summer camp, hidden icons in movies and television, exercise for seniors. Our system performs poor for vague Interest profiles. Following tweet's similarity score very less against profile RTS76 - exercise for seniors.

T_5*: "Exercise can slow down the physiological aging clock"*

While the following tweet has scored high but pivotal term senior was missing. We need to take care of the pivotal term from the Interest profile.

T_6*: "This exercise tests your balance and activates muscles throughout your body without you even noticing #workout #wlsa"*

[2]http://trecrts.github.io/TREC2017-RTS-guidelines.html

4.2.3 Modeling User Information need from the Interest profile

Some of the Interest profiles are very general, but the information need is from a particular perspective. e.g. interest profile shown in table 4.2 : RTS154 world best beach. As we look at this interest profile, it is tough to model user information need to retrieve relevant tweets. It is nearly impossible for our system to retrieve tweets that describe individual opinion on World best beaches, not promotional tweets. Our system performs very poorly for such interest profiles. Table 4.2 display such Interest profile.

Table 4.2: Sample Interest Profile: RTS154

{ topid : "RTS154" Title : "best beaches" Description : "What do people think are the world's best beaches" Narrative : "The user is looking for opinions regarding the best beaches in the world. The user is only interested in individual traveler's opinions, i.e., he is not interested in promotional tweets by tourism boards, travel companies, or writers of books, websites, and such" }.

The following tweet judged as non-relevant because it posted with promotional perspective

T_7: "The 10 Most Affordable Beaches In The World #travel #familytravel"

T_8: "RT : Beach? Runway? Both! #isleofbarra #airport connecting the #island with the world. #unique #scotland"

However, Some of the retrieved tweet found relevant.

T_9: "Probably the best beach in the world #ottelandet #högakusten #Övik."

Nevertheless, our system performs reasonably well for Interest profile as given in table 4.3 which has specific information needs.

Our System perform best for such interest profile where User information need is very specific. Following tweet is the part of summary for that Interest Profile.

T_{10}: "RT : Philippine: Western Media is Distorting Reality, People and Army Unite to Battle "ISIS" https://t.co/II9nYNZ2lB".

4.2.4 Unavailability of Named Entity in the Interest Profile

During post hoc analysis, we found that our system performs very poor in some of the interest profiles where a Named entity is not available; like RTS37 (Sea World), MB265

Table 4.3: Sample Interest Profile:RTS 120

```
{ topid : "RTS120"
Title : "Philippines Marawi ISIS"
Description : "I am interested in news on the battle between the Philippines
army and ISIS in the city of Marawi"
Narrative : "Since May 23, ISIS fighters have held the Philippines city of
Marawi, and Philippines armed forces have been trying to take back the city.
I am interested in any news on the progress, or lack of progress, of this effort,
as well as any involvement of US forces in this operation".}
```

Table 4.4: Sample Interest Profile : RTS82

```
{"topid" : "RTS82",
"title" : "best hot dogs"
"description" : "Who makes the best hot dogs? How do people like their hot
dogs to be prepared or served?" "narrative : "The user is a food writer and is
preparing a column on hot dogs. She would like to track people's opinions
regarding the best hot dogs as well as the way they prefer them to be prepared
or served." }
```

(cruise ship mishaps), MB365 (cellphone tracking), RTS82 (Best hot dogs) where system detected some of the silent days and obtained score in the nDCG-1 metric but did not score in the nDCG-0 metric. Named entity plays a very crucial role in relevant tweet retrieval. Some of the titles of Interest profiles do not include NE so, we extracted NE from the narrative field and included in the query.

The following two tweets have a similar score but the first one is relevant and the second one is not relevant.

T_{11}*: "The Best Kosher Hot Dog in the World Is in Highland Park, Illinois https://t.co/VcJ3ndl9Nr [Tablet]"*

T_{12}*: "He said he looked like an old hot dog with ketchup and mustard on it"*

4.2.5 Named Entity Linking

Interest Profiles sometime contain a very generalize/complex named entity. E.g., RTS 147 favorite summer vacations in US. The matching tweet contains a named entity Florida (Why do people vacation to Florida in the summer then complain of the on and off rain and humidity? Like surprise ur in the tropics). Another similar example, E.g. legalizing Medical Marijuana US and matching tweet contains a Named Entity Florida (Florida

Table 4.5: Sample Interest Profile: RTS147

{"topid" : "RTS147", "title" : "favorite summer vacations", "description" : "What were people's favorite summer vacation experiences in the U.S.?" "narrative" : "The user is planning his own vacation later this summer where he will travel in the U.S. He and is looking for reports of other people's favorite U.S.-based summer vacation experiences." }

Medical Association to oppose medical marijuana ballot amendment in Florida) [78]. Due to the NE linking problem, relevant tweet score very less against the interest profile. Table 4.5 show sample interest profile.

The following relevant tweet have a very low similarity score against interest profile. Two NE: Florida and US must be linked.

T_{13} *:"florida is literally in the middle of a tropical storm and i just thought it was a regular daily storm. i love the florida summers"*

The following tweet was judged as non-relevant by the human assessor

T_{14}*: "RT : Are you surprised to hear Montauk was named this summer's most expensive beach destination in the United States?"*

4.2.6 Named Entity Normalization

Named Entity normalization (NEN) is the task to identify the Named Entity written in the sentences. NEN is an important task in the area of Information Retrieval and NLP. Due to the limitation in length of the tweet, Microblog user often writes named entity in the abbreviated form. E.g. DEA(Drug Enforcement Agency). Though interest profile contains a term like a drug enforcement agency, the system is not able to retrieve tweets with the above normalize Named entity.

4.2.7 Clustering Issues

Microblog summarization problem exhibits the characteristic of multiple document summarization problem. Each tweet and embedded external URL is considered as one document. On Twitter, many users report the same event with different facts and personal bias. our novelty detection algorithm fails to cluster all following tweets in the same cluster.

Table 4.6: Sample Interest Profile :RTS 153

{"topid" : "RTS153", "title" : "South Beach Florida", "description" : "Find opinions regarding vacationing in South Beach, Florida in the summer.", "narrative" : "The user is planning a summer vacation to Florida, and is considering going to South Beach. She wants to see other's opinions, pro and con, of vacationing in South Beach in the summer.", }

T_{15} *: "Brazil's CORRUPT CONGRESS decides not to put Michel Temer to trial! Fora Temer!"*

T_{16} *: "Brazil President Michel Temer wins Congressional votes to block graft charge."*

T_{17} *: "Brazilian Lawmakers Reject Bribery Prosecution of President Michel Temer"*

another example.

T_{18} *: "Woman Is Eaten Alive By A Tiger At A Safari Park"*

T_{19} *: "Woman attacked by a tiger when she gets out of her car in a safari"*

T_{20} *: "Horror at Beijing Safari World as tigers attack women who exited car, killing one, injuring another."*

4.2.8 Sentiment or Subjectivity in the Interest profile

Some of the Interest profile like, RTS153: South Beach Florida includes sentiment and opinion or recommendation. Some of the tweets which are matching but do not include sentiment perspective are considered as non-relevant. In the future, one can keep a hidden feature like sentiment to increase the score of the low score relevant tweet.

The following tweets does not clearly mention pros and cons.

T_{21} *: "No beaches compare to south Florida beaches"*

T_{22} *: "BEACHING Summer Vacation on Florida Beaches Outdoors Allie SPA Bo Hudson"*

Nevertheless, system manage to retrieve following tweet with specific sentiment.

T_{23} *: "South Florida drying out after downpours cause major flooding in Miami Beach"*

4.2.9 Inclusion of Conditional event in Interest Profile

For the Interest profile like cancer and depression, our system performs very badly. Here the user is looking for the tweet regarding patients suffering from depression after diagnosed with cancer. It is very difficult to judge the co-occurrence of both events in the tweet. The following tweets are very difficult to retrieve and similarity scores are very low.

T_{24} *:"RT Awesome podcast from cancer survivor how exercise is key to living well becancerwell ht"*
T_{25} *:"RT Were there any books that helped get you through cancer treatment BookLoversDay 8Rdpzwm"*

4.2.10 Hash-tag Identification

Hash-tag can be one of the features for relevant tweet identification. Suitable hashtag identification will increase the score of the relevant tweet, e.g., the keyword is the sea world, and the hashtag is #seaworld or self-driving car the relevant hashtag is #selfdrivingcar.

T_{26} *:"RT : Will you need a driving licence in the age of #SelfDriving cars?*
T_{27} *:"Autonomous driving in a city? We're '95% of the way there' #autonomous #cars*
T_{28} *:"RT : The entangling alliances of the self-driving car world, visualized #selfdrivingcars"*

4.2.11 Effect of Query Expansion

It has been observed that our system performs very poorly in terms of evaluation metric nDCG-1 and nDCG-0 for the Interest profiles without NE in majority cases. We also hypotheses that query expansion might work positively for these interest profiles. Our result shows that query expansion for such a topic improvises the result nDCG-1 and nDCG-0. One can do query expansion based upon interest profiles on a case by case basis.

4.3 Conclusion

In this chapter, We have performed a failure analysis on the Microblog based summarization system developed in the last chapter. The system performs reasonably well where users' information need very specific and unambiguous while achieving very poor on niche topic such as emerging music style. TREC RTS organizers have developed a range of interest profiles. Interest profile includes sensitive and hot topics like North Korea missile test and low profile topic such as leafleting. If we look at the evaluation metrics, they are biased towards precision. Appendix A contains a summary created by our system and also gives information about the dataset used in the experiment. Overall Summarizing the Twitter stream is a tough task.

CHAPTER 5
Tracking Impoliteness across Social Web

Sensing social media, such as Facebook, Twitter by smart autonomous application empowers user community with real-time information which unfolds across the different part of the world. Social media provide an easy and anonymous platform for common people to voice their opinion or view on various entities like celebrities, politicians, products, stock markets, etc. . or any social movement. Sometime, such opinions might be aggressive and propagate hate in the online platform.

In this chapter, online Hate Speech and related concepts, such as aggression, cyberbullying, offensive contents are subsumed under the impoliteness. Henceforth, in the rest of the thesis impoliteness will refer to such terms unless specified otherwise. In this chapter, we will explore the various forms of impoliteness, such as user aggression, offensive content, and factual-posts identification.

Research Questions In this chapter, exhaustive experiments are performed on the benchmark dataset to answer the following questions

- RQ1: Which is the best text representation scheme to model text from the social web for the text classifiers?
- RQ2: Does a pre-trained contextual language model based on transfer learning better than pre-trained word embedding based on shallow transfer learning on Social media data?
- RQ3: Does the size of embedding matter?

5.1 User Aggression Detection

With the unprecedented increase in the user-base of social media and its availability on smartphones, incidents like Hate speech, trolling, cyberbullying, and aggressive posts are increasing exponentially. A smart autonomous system is required which enables surveillance on the social media platform and detects such incidents. Some of the researchers look at these posts from the different aspects of aggression [52] to filter the contents. Some of the posts contain words that might be qualified as either highly or overtly aggressive or have hidden aggression. Sometimes posts do not have any aggression. Based on these, posts or comments are categorized into three classes, namely: ‘Overtly Aggressive‘, ‘Covertly Aggressive‘ and ‘Non-aggressive‘ [52]. Henceforth, in the rest of the thesis, we will denote these classes by these abbreviations: OAG, CAG, NAG respectively.

The text representation in numerical form plays a pivotal role in any NLP downstream task. Bag-of-Word(BoW) is the oldest and simple technique to represent the document or post into a fixed-length vector. The BoW techniques generate very sparse and high dimensional space vectors. Text representation using distributed word/sentence representation or word embedding is gain rapid momentum recently. In this paper, one of the objectives is to find the best text representation scheme to model social web content for the traditional classifiers and deep neural models. Various Text representation scheme based on BoW, word embedding, sentence embedding, and pre-trained language model are studied empirically. We have reported result on popular word embedding technique like Word2vec, Glove and fastText, sentence embedding technique such as Doc2Vec [55], smooth-inverse frequency(SIF) [5], InferSent [24], p-means [102],Skip-Thought vectors [50],Universal Sentence Encoder (USE) [18] on traditional classifier such as, Multinomial Naive Bayes (MNB), Logistic Regression(LR), K-Nearest Neighbors (KNN), Support Vector Classifier (SVC), Decision Tree (DT),Stochastic Gradient Descent(SGD), Random forest (RF), Ridge, AdaBoost, Perceptron, deep neural models based on LSTM, CNN and Bidirectional LSTM.

Transfer Learning is well-practiced in the area of computer vision. However, in the NLP, transfer learning has limited application in the form of a pre-trained word vector, which is used to initialize the weights of the embedding layer of the deep neural network.

Contextual pre-trained language model such as ELMO [92], ULMFiT [49], and BERT [31] claimed substantial improvement in the performance of various NLP downstream tasks like sentiment analysis, question/answering, textual entailment. The main idea behind these methods is to train a language model on the large corpus and fine-tuned on the task-specific dataset. In this paper, we have evaluated the performance of these methods for aggression classification.

In this chapter, we present extensive benchmarking of text representation schemes on the TRAC dataset. Our results reveal that fastText with pre-trained vectors along with CNN outperform traditional classifiers based on BoW Model. Paragraph vector or Doc2Vec [55] performs very poor on the TRAC dataset and turns out to be the worst text representation scheme among all. We also found that models based on the deep neural nets are more robust than the traditional classifier model when tested on a lexically different dataset than the training dataset. i.e., deep neural model substantially outperforms traditional classifiers on Twitter test Dataset while trained on Facebook Dataset in this evaluation.

To validate our claims, statistical significance tests are performed on a weighted F_1-score of the classifier for each text representing scheme. Statistical inference is used to check evidence to support or reject these claims. Non-parametric test, such as Wilcoxon signed-rank and was carried out by comparing the weighted F_1 score all the text representation scheme with the fastText pre-trained vector. In the majority of the cases, p-values are less than 0.05.

5.1.1 TRAC Dataset

Experiments are performed on standard benchmarked datasets to evaluate the performance of different text representation scheme. For user aggression detection problem, Trolling, Aggression and Cyberbullying (TRAC) dataset [52] is considered, which contains posts in English and code-mixed Hindi. The dataset consists of 15,001 aggression-annotated Facebook posts and comments each in Hindi (Romanized and Devanagari script) and English for training and validation [52]. Table 5.1 shows a detail description of the training and validation Dataset. Table 5.2 gives details of the test data corpus, which also contains a dataset from Twitter.

Table 5.1: Training Dataset statistics

	English Corpus		Hindi Corpus	
	# Training	# Validation	# Training	# Validation
NAG	5052	1233	2275	538
CAG	4240	1057	4869	1246
OAG	2708	711	4856	1217
Total	12000	3001	12000	3001

Table 5.2: Test Dataset Statistics

Test Dataset	# of posts
Facebook English Corpus	916
Twitter English Corpus	1257
Facebook mixed script Hindi Corpus	970
Twitter mixed script Hindi Corpus	1194

5.1.2 Text Representation Schemes

Text representation is about numerically representing document so that they can be feed as an input to the classifier. This numerical representation is in the form of the vectors that together form a matrix. The main objective of this paper is an identification of the best text representation scheme to model social media text. There are four types of text representation schemes :(i) Bag-of-words(BoW) (ii) word embedding (iii) sentence embedding (iv) Contextual Pre-trained Language model. BoW with count vector, TF/IDF weighting, word embedding techniques(Word2Vec, Glove, fastText) and sentence embedding techniques, such as Doc2Vec [55], smooth-inverse frequency(SIF) [5], InferSent [24], p-means [102],Skip-Thought vectors [50],Universal Sentence Encoder (USE) [18] are studied empirically with a set of classifiers.

Bag-of-Word Model for Text Representation

The Bag-of-words(BoW) is the simple technique to represent the document or social media posts in the vector form and also a very common feature extraction method from the text. Word count or TF/IDF weight of each n-gram word used as a feature. The dimension of the vector is equal to the size of the vocabulary of the text corpus or dataset, which results in a very high dimensional sparse document vector. It is the conventional method used for the text representation to perform various NLP tasks like text classification, clustering. However, the BoW methods ignore the word order, which may lead to loss of the context.

Word Embedding for Text representation

Word embedding is the text representation technique to represent the word in the low dimensional space so that semantically similar words have similar representation. Major word embedding techniques like Word2Vec learn word embedding using a shallow neural network. The fastText, extension of Word2vec, consider the morphological structure of the word.

Word2Vec: Word2Vec [75] is the unsupervised and predictive neural word embedding technique that learns word representation in the nth-dimensional vector space. It maps words into vectors of real numbers. Word2Vec is a two-layer shallow neural net that takes text corpus as an input and output a set of vectors.

Mikolov et al.[75] proposed two novel model architectures: Skip-gram and CBOW(Continuous bag of words) are proposed for computing continuous vector representations of words from datasets. The main objective behind the training of the Word2vec model is to learn the weights of the hidden layer. These weights are the feature vector for each word in the vocabulary. If the size of the vocabulary is V, and the dimension of the vectors is N, and then the size of the weight matrix is V*N. Word2vec CBOW model takes context words as an input predicts target word. Word2Vec skip-gram model predicts context words for the given target word. The objective of the skip-gram model is to maximize the average log probability of the

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_j) \tag{5.1}$$

Mikolov et al. [77] proposed two methods to train the Word2Vec model: (i) hierarchical softmax (ii) negative sampling. Hierarchical softmax is an approximation of the full softmax which needs log2(W) Instead of evaluating W nodes. The negative sampling method is an alternative to the hierarchical softmax. Authors proposed subsampling of frequent words like the, a, is to reduce the number of the training example. The model is trained with the negative sample, which updates only a small percentage of models'weights. Authors claimed that subsampling and negative sampling reduce the computation cost and also produce quality word vectors. The negative sample size is 5-20 for the small and 2-5 for large datasets.

Glove: GloVe stands for Global vector for [Word Representation] [91] is an unsupervised method for learning word embedding. A co-occurrence word matrix is constructed from the text corpus for the training and is reduced in low dimensional space, which explains the variance of high dimensional data and provides word vector for each word. Glove focus on the ratio of co-occurrence probability for a similar word. The Glove is a count-based model, while Word2vec is a predictive model. Therefore, Glove has the advantage to use global statistics of the word for the prediction. The glove model is very fast to train and scalable on a huge corpus. Even with a small corpus, Glove provides good performance.

fastText: fastText [15] is the neural word embedding technique that learns distributed low dimensional word embedding. Word2vec, Glove consider each word as a single unit and ignore the morphological structure of the word. Therefore, they are not able to generate word embedding for the unseen or out of vocabulary word during the training. The fastText overcome this limitation of Word2vec and Glove by considering each word as n-gram of characters. A word vector for a word is computed from the sum of the n-gram characters. The range of N is typically 3 to 6. Since the user on social media often makes a spelling error, typos, fastText will be more effective than the rest of the two.

Generation of Tweet Vector from word vectors: Word2Vec, Glove, fastText maps each to word to the corresponding vector of the real numbers. Deep neural models such as LSTM, CNN takes each word vector as an input, while this is not possible for the traditional classifier or feed-forward neural network where tweet has to be represented by a fixed-length of the vector. Tweet and Facebook posts contain a sentence or more than one sentence. Therefore, the tweet vector has to formed using the word vectors. Three simple techniques are used to generate a tweet vector. (i) using the arithmetic mean of word vectors of tweet's terms (ii) addition of word vectors (iii)using the arithmetic mean of word vector weighted by tf-idf score of each term.

$$tweet - vector = \frac{1}{n} \sum_{i=1}^{n} (v_i) * TF - IDF(t_i) \tag{5.2}$$

Where n is the total number of words. v_i is the word vector of the t_i term. Empirically, averaging of word vector yield better result than addition and averaging out by weighted tf-idf.

Sentence Embedding for Text Representation

Word embedding techniques provide better embedding for the words and improved many downstream NLP tasks. However, a text representation of a sentence, paragraph, or document by a fixed-length vector is still a challenging task. In the following paragraph, we will discuss the major sentence embedding technique.

Doc2vec: Doc2Vec or paragraph vector is an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts, such as sentences, paragraphs, and documents [55]. Henceforth, we will refer paragraph and Doc2vec interchangeably. The paragraph vector represents each document by a dense vector that is trained to predict words in the document. Authors believe that the Paragraph vector has the potential to overcome the weaknesses of bag-of-words models and claimed that paragraph vectors outperform bag-of-words models as well as other techniques for text representations. The Doc2vec model has two architecture, namely : (i) DM: This is the Doc2Vec model analogous to the CBOW model in Word2vec. The paragraph vectors are obtained by training a neural network on the task of inferring a center word based on context words and a context paragraph. (ii) DBOW: This Doc2Vec model is analogous to the skip-gram model in Word2Vec. The paragraph vectors are obtained by training a neural network on the task of predicting a probability distribution of words in a paragraph given a randomly-sampled word from the paragraph.

In Doc2vec, each paragraph is mapped to a unique vector, every word is also mapped to a unique vector, represented by a column in matrix W. The paragraph vector and word vectors are averaged or concatenated to predict the next word in a context.

Smooth Inverse frequency (SIF) and p-mean: SIF [5] was one of the first major unsupervised sentence embedding techniques that show improvement over the averaging of the word embeddings which is used for the common baseline. SIF takes the weighted average of the word embeddings in the sentence. Each word vector is weighted by a/(a + p(w)), where a is a parameter that is typically set to 0.001 and p(w) is the estimated frequency of the word in a reference corpus. SIF computes the principal component of the resulting embeddings for a set of sentences. It then subtracts from these sentences embeddings their projections on their first principal component. [102] proposed a sentence embedding technique, called p-mean, which extends the average word embedding concept to power mean word embedding. The author claimed substantial improvements over SIF[5].

Skip-thoughts: Skip-thoughts[50] is the unsupervised technique that learns fixed-length representations of sentences without labeled data using the encoder-decoder model. The model aims to predict the previous and next sentences given the current sentence. The representation or output of the encoder is used for any downstream NLP task such as machine translation, text classification. Skip-Thought considers the order of the word in which they appear in the sentence that will lead to giving different embedding to sentences. for example, US defeats China and China defeats US.

Infersent: To address the limitation of unsupervised sentence embedding technique, such as Skip-Thought [24] proposed supervised technique which is trained using Stanford Natural language inference dataset to learn the embedding for the small piece of text such as sentence or paragraph analogous to the computer vision extract feature from the Imagenet and use in another set of the task. The dataset contains 570k manually created English sentence pairs and annotated in one of the three classes- entailment, contradiction and neutral. Natural language inference aims at finding a directional relationship between the two sentences. The authors tested 7 different architectures for the sentence encoder and the best results are achieved with a bi-directional LSTM (BiLSTM) encoder. So The idea is to learn sentence embedding for the NLI task and transfer these embedding to the other downstream task such as text classification.

Universal sentence Encoder: Recently Google introduces one of the popular sentence encoding modules which captures the semantic better than by simply averaging word vector to generate the sentence vector. Google proposed two architecture. The first model trained with transformer encoder and second trained with Deep Averaging Network (DAN). The major difference between the two models lies in terms of accuracy and computational resource requirement. While the one with a Transformer encoder has higher accuracy, it is computationally more intensive. The one with DNA encoding is computationally less expensive and with little lower accuracy.

Transfer Learning

Transfer Learning in NLP is not as matured as compare to in Computer Vision. Transfer learning is a method in which a model is trained on a large corpus for a particular task and use this pre-trained model for a similar task. There are two ways to use transfer learning in NLP (i) use of pre-trained word embedding to initialize the first layer of the neural network model, which can be termed as a shallow representation. (ii) use the full model and fine-tune for the task-specific in supervised learning way.

Context-free Pre-trained Vectors: Word embedding gives a similar vector for a word in the vocabulary. Therefore, they do not consider the context in which word appears in the tweet. e.g., the word bank has the same vector representation for two semantically different contexts: bank account and river bank. Word2Vec, Glove, and fastText provide pre-trained word vector trained on the large corpus. Google Word2vec pre-trained model has word vectors for 3 million words with size 300 and trained on Google news. Glove pre-trained model available with different embed size and trained on the common crawl, Twitter. We have used the Glove pre-trained model with a vocabulary size of 2.2 million and trained on the common crawl. fastText pre-trained models are available in 157 languages. We have used fastText pre-trained vector for English and Hindi language trained on the common crawl and Wikipedia.

Contextual Pre-trained Language Model: Recently, transfer learning in NLP done in a new way; the First language model is trained on large text corpus in an unsupervised way and fine-tuned on task-specific labeled data. [92] argued that word representation depends upon the context. So each word has a different word vector depending upon the position of the word in the sentence. Essentially, each word has a dynamic word vector to the context as opposed to the traditional word embedding techniques which always give the same word vector ignoring the context. Embedding from Language Models (ELMos) use the language model for the word embedding. [49] proposes a Universal Language Model Fine-Tuning for Text Classification (ULMFiT) which is a bi-LSTM model that is trained on general language modeling (LM) task and then fine-tuned on text classification.

ELMo: ELMo [92], embedding from the language model, is the attempt to empower machine to learn the semantics of the sentence. Unlike popular word embedding technique, such as Word2Vec, Glove, fastText which gives same word vector for the word thereby ignoring the context of the word, the ElMo attempt solve the problem of polysemy of the word by generating different embedding of the same word depending upon the context in which word appear. ELMo generates word vectors using the bi-directional LSTM model. vectors are computed on top of a two-layer bidirectional language model. Higher-level layers capture context-dependent aspects of word embeddings while lower-level layers capture model aspects of syntax. ELMO uses character embeddings to build word embeddings to overcome the OOV word.

ULMFiT: Universal Language Model Fine-tuning for Text Classification (ULMFit) [49] is a transfer learning method, inspired from the success of ImageNet from the computer vision, aims to capitalize the benefits of using a pre-trained model on text classification. The core idea behind this method is to train a language model with a large corpus and fine-tune the language model with a task-specific dataset. ULMFIT uses AWD-LSTM architecture for its language model. Authors propose slanted triangular learning rates (STLR) in which, the learning rate first increases linearly and then decays linearly. The authors used different learning rates at each layer. The lower layer has a lower learning rate than the top layer. In the final phase, the model is fine-tuned for the text classification task.

BERT: Devlin et al. [31] proposed the Bidirectional Encoder Representations from Transformers (BERT), which has achieved significant improvement in various NLP tasks.

Table 5.3: Embed Size of word and sentence embedding schemes

Text Representation scheme	Embed Size
Word2vec Skip-gram	300
Glove	300
fastText	300
pre-trined Word2Vec	300
pre-trined Glove Common crawl	300
pre-trined FastText Common crawl	300
doc2vec-dmc	300
doc2vec-dbow	300
InferSent	4096
SIF	300
USE	512
Skip-thought	4800
p-mean	3600
BERT	512
ELMO	1024

The biggest challenge of the NLP task is the unavailability of the labeled data. The main objective behind BERT is to address this issue. BERT is based on a language model, trained on a large corpus, considers the previous and next tokens into account when predicting the next word as opposed to traditional language models which only consider the previous n tokens and predict the next one. Therefore, BERT exhibits a contextual representation of word as opposed to Word2Vec, which gives context-free representation. We have fine-tuned pre-trained BERT representations to the text classification task. BERT is based on transformer architecture for encoding the text and performs better in case of small training datasets.

Many variants of the BERT pre-trained model are available. We have used the Uncase BERT base model with 12 layers and 110M parameters to classify the aggression on the English dataset. While for the Hindi dataset, a multilingual version of BERT is used with a 110 M parameter. The hyperparameters are set as follows: Sequence length is 128. The model is trained for 3 epochs with a batch size of 32, and the learning rate is set to 0.00002.

Table 5.3 shows the embed size of each text representation scheme discussed over here.

5.1.3 Problem statement

Fundamentally, aggression detection is a text classification problem. Formally, the task of text Classification is stated as follows. Given a set of social media feed T and a set of classes, we need to compute a function of the form:

$$C = f(T, \Omega) \tag{5.3}$$

where f is the multi-class classifier that is computed using training data, T is the numeric representation of the text of the dataset, Ω is the set of parameters of the classifier and C is the pre-define class-labels.

5.1.4 Model Architectures and Hyperparameters

In this subsection, we will discuss the architectures and hyperparameters of the deep neural models used for the classification. The model learns abstract features from the input texts instead of hand-crafted features which used to encode text into features vector.

Bidirectional LSTM

The first deep neural model is based on the bidirectional LSTM include embedding layer with embed size 300, which converts each word from the post into a fixed-length vector. Short posts are padded with zero values. Subsequent layers include a bidirectional LSTM layer with 50 memory units followed by a one-dimensional global max-pooling layer, a hidden layer with size 50 and an output layer with softmax activations. ReLU activation function is used for the hidden layer activation. A drop out layer is added between the last two layers to counter the overfitting with parameter 0.1. Hyperparameters are as follows: Sequence length is fixed at 1073 words; maximum length of posts in the dataset. No of features is equal to half of the total vocabulary size. Models are trained for 10 epochs with batch size 128. Adam optimization algorithm is used to update network weights.

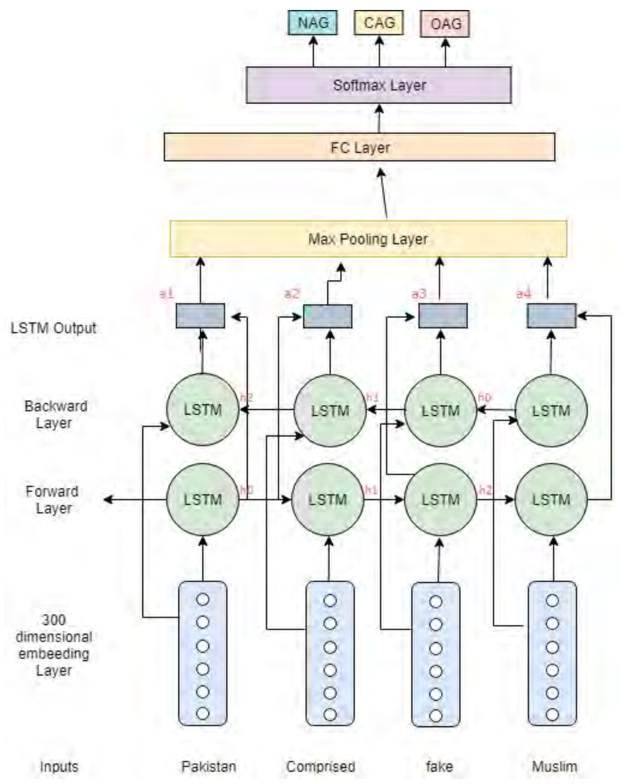


Figure 5.1: Bidirectional LSTM Architecture.

Single LSTM with higher dropout

This model is based on the Long Short Term Memory, a type of recurrent neural network with a higher dropout. This model is having one embedding layer, one LSMT layer with a size 64 memory unit, and one fully connected hidden layer with Relu activation and size 256 and an output layer with softmax activation. Hyperparameters are the same as discussed in the previous model. A dropout layer is added between the hidden layer and an output layer with a drop out rate 0.2 to address the overfitting issue.

CNN Model

This model includes one embedding layer whose weights are initialized with a fastText pre-trained vectors with embed size is 300, followed by a one-dimensional convolution layer with 100 filters of height 2 and stride 1 to target bigrams. In addition to this, the Global Max Pooling layer added to fetch the maximum value from the filters which are feed to the fully connected hidden layer with size 256, followed by the output layer. ReLU and softmax activation function are used for the hidden layer and output layer, respectively.

.

CNN model with Multiple Convolution layer

This model includes an embedding layer with embed size 300. Three one dimensional convolution layers with size 100 and different filters with height 2,3,4 to target bigrams, trigrams, and four-grams features, followed by max-pooling layer which concatenate max pooled result from each of one-dimensional convolution layer. The final two layers include a fully connected hidden layer with a size 250 and an output layer with ReLu and softmax activation. A Drop out layer is added between the last two layers with rate 0.2. Hyperparameters are the same as discussed in the first model. This model is similar to proposed by [135].

Traditional Classifiers

We have reported results on traditional classifiers such as Multinomial Naive Bayes (MNB), Logistic Regression(LR), K-Nearest neighbors KNN, Support Vector Classifier (SVC), feed-forward neural network, Decision Tree (DT), Stochastic Gradient Descent(SGD), Random forest (RF), Ridge, AdaBoost, Perceptron using text representation scheme such as Bow, word embedding, and sentence embedding, We have done little pre-processing on the dataset. Hashtag symbol # and User mentions are dropped from the text. Non-ASCII characters and stop-words are removed from the text

5.1.5 Experiment Results

In this section, we present the results of the classifiers on the TRAC dataset [52] with different text representation schemes. Tweets are very noisy and contain user mentions, Hashtags, Emojis, and URLs. We do not perform any text pre-processing on the text in experiments with deep neural models

Evaluation Metrics

Precision, recall, and F1-score are the standard metrics that are used to evaluate classifier performance. Precision is the ratio of correctly predicted positive posts to the total predicted positive posts, while recall is the ratio of correctly predicted positive posts to all posts in the actual class. Precision considers the false positive while recall considers false negative. F1-measure is the trade-off metric between precision and recall. F1-score is the harmonic mean of precision and recall. The harmonic mean is the reciprocal of the arithmetic mean of the reciprocals.

$$precision = \frac{TP}{TP + FP} \tag{5.4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5.5}$$

$$F1 - score = \frac{2 * Precison * Recall}{Pricision + Recall} \tag{5.6}$$

There are many variants of F1-score such as weighted F1, macro F1, and micro F1. In the TRAC dataset [52], the distribution of class labels is uneven, not the highly imbalanced. Therefore, a weighted F1-score is used to measure classifier performance instead of macro F1.

Results On TRAC Dataset

A set of classifiers (16) performance, based on traditional and deep neural models, are evaluated on 4 datasets (2 English+2 Hindi) with 17 text representation schemes (9 in the case of Hindi Dataset). Classifiers' results based on LSTM and CNN on BoW text representation schemes are not possible due to the higher dimensionality. Bernoulli classifier is used instead of the Naive Bayes Classifier in case of text representation schemes other than BoW. Since word vectors might have negative weights, it is impossible to calculate probabilities with negative weights. Skip-gram variant of Word2Vec and fastText is used in this experiment instead of continuous bag-of-word.

Count Vector scheme: Count-vector is the basic text representation scheme based on Bag-of-Word(BoW). Count of the word or term frequency is used to weight the term. Table 5.4 and 5.5 show results on the TRAC English and the Hindi dataset respectively.

TF/IDF Scheme: TF/IDF is an alternative to the count-vector scheme. The acronym stands for "Term Frequency-Inverse Document Frequency" which are the components of

Table 5.4: Results on TRAC English Dataset : Count-Vector scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.57	0.56	0.5676	0.68	0.51	0.5571	0.51	0.50	**0.5102**
LR	0.55	0.563	0.5578	0.68	0.56	0.5953	0.50	0.50	0.4849
KNN	0.40	0.42	0.3824	0.56	0.54	0.5466	0.39	0.41	0.3539
SVC	0.52	0.53	0.5255	0.67	0.54	0.5801	0.47	0.47	0.4642
DT	0.48	0.48	0.4813	0.61	0.49	0.5269	0.43	0.44	0.4229
SGD	0.54	0.54	0.5367	0.65	0.53	0.5706	0.48	0.48	0.4682
RF	0.51	0.51	0.4929	0.62	0.53	0.5621	0.44	0.45	0.4199
Ridge	0.55	0.56	0.5456	0.66	0.57	0.6009	0.50	0.50	0.4703
AdaB	0.51	0.50	0.4700	0.64	0.62	**0.6210**	0.45	0.44	0.3343
Perc	0.52	0.51	0.5136	0.65	0.50	0.5387	0.49	0.49	0.4930
Ensemble	0.59	0.59	**0.58**	0.69	0.53	0.58	0.51	0.51	0.495
ANN	0.57	0.56	0.57	0.65	0.53	0.5703	0.51	0.50	0.4912

the resulting scores assigned to each word. Term Frequency summarizes how often a given word appears within a document. Inverse document frequency penalizes words that appear a lot across the documents and assign more weight to the rare term.

Results on Word2vec: Table 5.8 and Table 5.9 shows results on TRAC English and Hindi corpus using Word2vec skip-gram model. Results show that Word2vec is more effective with deep neural net compare to traditional classifiers.

Results on Glove: Table 5.10 and 5.11 show result on English and Hindi dataset respectively.

Results on fastText: fastText [15] is an open-source, free, lightweight library that allows users to learn low dimensional word embedding. Results on English and Hindi corpus and are shown in Table 5.12 and 5.13.

Results on Doc2Vec: Doc2Vec or Paragraph Vector is an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts, such as sentences, paragraphs, and documents [55]. Authors believe that Paragraph vector has the potential to overcome the weaknesses of bag-of-words models and claimed that Paragraph Vectors outperform bag-of-words models as well as other techniques for text representations.

Table 5.5: Results on TRAC Hindi corpus: Count Vector scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.53	0.53	0.5202	0.56	0.56	0.5535	0.30	0.30	0.2970
LR	0.59	0.59	**0.5873**	0.60	0.59	**0.5855**	0.38	0.38	0.3787
KNN	0.48	0.36	0.3538	0.43	0.34	0.3340	0.32	0.36	0.2527
SVC	0.57	0.57	0.5693	0.56	0.56	0.5556	0.38	0.38	0.3781
DT	0.52	0.52	0.5155	0.54	0.53	0.5307	0.37	0.38	0.3685
SGD	0.56	0.55	0.5543	0.56	0.56	0.5533	0.41	0.41	0.3996
RF	0.56	0.56	0.5568	0.56	0.55	0.5473	0.35	0.38	0.3585
Ridge	0.58	0.58	0.5748	0.59	0.58	0.5780	0.41	0.37	0.3616
AdaB	0.62	0.54	0.4982	0.67	0.57	0.5373	0.59	0.34	0.1886
Perc	0.54	0.51	0.5197	0.53	0.52	0.5213	0.39	0.40	0.3931
Ensemble	0.58	0.58	0.58	0.58	0.57	0.57	0.49	0.44	**0.4400**
ANN	0.55	0.54	0.55	0.57	0.56	0.5703	0.0.44	0.43	0.43

Table 5.6: Results on TRAC English dataset : TF/IDF scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.59	0.56	0.5237	0.67	0.53	0.5596	0.51	0.50	0.4528
LR	0.58	0.58	**0.5700**	0.68	0.57	**0.6046**	0.52	0.52	0.4890
KNN	0.39	0.41	0.3436	0.52	0.58	0.5428	0.35	0.37	0.2891
SVC	0.56	0.56	0.5595	0.68	0.55	0.5902	0.49	0.49	0.4853
DT	0.48	0.48	0.4773	0.59	0.47	0.5055	0.42	0.43	0.4111
SGD	0.56	0.57	0.5620	0.69	0.56	0.5938	0.51	0.52	0.5020
RF	0.50	0.51	0.4884	0.61	0.54	0.5582	0.41	0.44	0.3917
Ridge	0.57	0.57	0.5650	0.68	0.56	0.5999	0.51	0.51	0.5003
AdaB	0.51	0.51	0.4744	0.63	0.61	0.6141	0.50	0.46	0.3696
Perce.	0.52	0.52	0.5198	0.66	0.51	0.5491	0.48	0.48	0.4778
ANN	0.57	0.57	0.5600	0.68	0.49	0.5350	0.52	0.52	**0.5164**
Ensemble	0.59	0.57	0.5500	0.68	0.55	0.5900	0.52	0.51	0.4842

Table 5.7: Results on TRAC Hindi dataset : TF/IDF scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.61	0.58	0.5654	0.66	0.61	0.6031	0.36	0.32	0.2902
LR	0.63	0.61	**0.6058**	0.66	0.62	**0.6134**	0.38	0.37	0.3724
KNN	0.47	0.24	0.1903	0.43	0.24	0.1721	0.31	0.37	0.2553
SVC	0.59	0.59	0.5858	0.60	0.59	0.5862	0.39	0.39	0.3886
DT	0.51	0.51	0.5100	0.50	0.50	0.5025	0.39	0.41	0.3936
SGD	0.62	0.60	0.5945	0.63	0.60	0.5922	0.41	0.40	0.3993
RF	0.57	0.56	0.5604	0.57	0.55	0.5473	0.37	0.38	0.3737
Ridge	0.59	0.59	0.5865	0.60	0.59	0.5850	0.40	0.39	0.3872
AdaB	0.61	0.53	0.4939	0.67	0.56	0.5233	0.60	0.34	0.1903
Perce.	0.55	0.55	0.5480	0.56	0.56	0.5598	0.39	0.39	0.3868
ANN	0.60	0.60	0.6000	0.68	0.49	0.5350	0.48	0.46	0.44
Ensemble	0.60	0.59	0.5900	0.64	0.61	0.6087	0.49	0.47	**0.4600**

Table 5.8: Results on TRAC English Corpus : Word2vec scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.51	0.50	0.4999	0.63	0.45	0.4870	0.55	0.56	**0.5551**
LR	0.51	0.52	0.5016	0.63	0.53	0.5675	0.36	0.36	0.3457
KNN	0.47	0.48	0.4691	0.60	0.46	0.5061	0.39	0.40	0.3843
SVC	0.50	0.51	0.4980	0.60	0.50	0.5369	0.33	0.31	0.3078
DC	0.43	0.43	0.4331	0.56	0.40	0.4468	0.39	0.39	0.3884
SGD	0.43	0.43	0.4282	0.57	0.42	0.4647	0.45	0.45	0.4512
RF	0.47	0.48	0.4674	0.60	0.48	0.5199	0.44	0.45	0.4333
Ridge	0.51	0.52	0.4955	0.59	0.50	0.5347	0.36	0.34	0.3352
AdaB	0.49	0.50	0.4908	0.64	0.51	0.5491	0.48	0.47	0.4485
Perce.	0.44	0.45	0.4406	0.61	0.48	0.5230	0.35	0.36	0.3521
ANN	0.52	0.52	0.5240	0.68	0.49	0.5350	0.52	0.52	0.5164
Ensemble	0.51	0.51	0.4934	0.64	0.51	0.5558	0.49	0.49	0.4500
LSTM	0.58	0.57	**0.5735**	0.70	0.52	0.5649	0.55	0.55	0.5385
BLSTM	0.53	0.54	0.5333	0.67	0.54	0.5759	0.53	0.53	0.5314
CNN	0.56	0.56	0.5633	0.68	0.51	0.5515	0.50	0.50	0.5012
NCNN	0.57	0.57	0.5610	0.69	0.55	**0.5919**	0.53	0.52	0.5120

Table 5.9: Results on TRAC Hindi Corpus: Word2vec scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.44	0.42	0.3263	0.40	0.41	0.3001	0.32	0.36	0.3215
LR	0.59	0.58	0.5800	0.60	0.58	**0.5779**	0.29	0.29	0.2819
KNN	0.49	0.49	0.4937	0.50	0.50	0.4998	0.39	0.37	0.3704
SVC	0.55	0.55	0.5425	0.48	0.49	0.4806	0.29	0.28	0.2821
DC	0.46	0.45	0.4556	0.46	0.46	0.4629	0.37	0.35	0.3572
SGD	0.44	0.41	0.4172	0.42	0.39	0.3912	0.30	0.29	0.2822
RF	0.52	0.51	0.5109	0.54	0.54	0.5374	0.34	0.33	0.3286
Ridge	0.58	0.57	0.5659	0.55	0.54	0.5293	0.29	0.28	0.2811
AdaB	0.54	0.54	0.5392	0.54	0.54	0.5342	0.34	0.33	0.3256
Perce.	0.46	0.38	0.3917	0.48	0.42	0.4232	0.29	0.29	0.2802
ANN	0.57	0.56	0.5525	0.57	0.55	0.5455	0.32	0.32	0.3163
LSTM	0.61	0.60	0.6017	0.70	0.52	0.5649	0.40	0.41	**0.3840**
BLSTM	0.58	0.56	0.5546	0.67	0.54	0.5759	0.32	0.36	0.2846
CNN	0.62	0.61	**0.6106**	0.68	0.51	0.5515	0.36	0.38	0.3323
NCNN	0.63	0.61	0.6017	0.69	0.55	0.5919	0.37	0.38	0.3338

Table 5.10: Results on TRAC English Corpus: Glove scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.38	0.42	0.3712	0.49	0.37	0.3873	0.38	0.46	0.3936
LR	0.47	0.47	0.4380	0.59	0.52	0.5358	0.45	0.44	0.3959
KNN	0.37	0.39	0.3662	0.57	0.49	0.5130	0.37	0.40	0.3607
SVC	0.47	0.48	0.4484	0.60	0.47	0.5037	0.36	0.38	0.3627
DC	0.39	0.39	0.3922	0.51	0.36	0.4067	0.37	0.37	0.3673
SGD	0.42	0.44	0.4218	0.56	0.33	0.3571	0.45	0.47	0.4182
RF	0.38	0.40	0.3845	0.53	0.45	0.4752	0.37	0.40	0.3634
Ridge	0.48	0.48	0.4415	0.59	0.51	0.5336	0.42	0.43	0.3877
AdaB	0.40	0.44	0.3893	0.53	0.49	0.4932	0.35	0.42	0.3552
Perce.	0.43	0.44	0.3844	0.65	0.38	0.4020	0.48	0.47	0.3938
LSTM	0.59	0.57	0.5619	0.70	0.50	0.5454	0.54	0.54	**0.5156**
BLSTM	0.49	0.49	0.4598	0.48	0.51	0.4760	0.46	0.46	0.3860
CNN	0.58	0.57	**0.5737**	0.68	0.49	0.5365	0.54	0.56	0.5377
NCNN	0.56	0.55	0.5485	0.68	0.50	**0.5488**	0.51	0.50	0.4984

Table 5.11: Results on TRAC Hindi dataset: Glove Scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.43	0.40	0.361	0.47	0.42	0.372	0.30	0.29	0.273
LR	0.55	0.51	0.497	0.48	0.47	0.464	0.30	0.30	0.279
KNN	0.47	0.47	0.462	0.44	0.43	0.425	0.35	0.34	0.334
SVC	0.52	0.51	0.495	0.38	0.38	0.373	0.27	0.28	0.261
DC	0.43	0.40	0.406	0.40	0.39	0.388	0.34	0.33	0.326
SGD	0.44	0.41	0.414	0.41	0.39	0.393	0.38	0.34	0.287
RF	0.46	0.44	0.443	0.45	0.44	0.440	0.39	0.36	0.344
Ridge	0.54	0.51	0.497	0.38	0.39	0.381	0.24	0.26	0.242
AdaB	0.47	0.45	0.452	0.52	0.48	0.479	0.40	0.37	0.362
Perce.	0.39	0.38	0.380	0.37	0.36	0.364	0.40	0.35	0.329
LSTM	0.60	0.59	0.590	0.62	0.59	**0.590**	0.39	0.40	0.376
BLSTM	0.54	0.53	0.530	0.57	0.54	0.527	0.35	0.36	0.318
CNN	0.56	0.56	0.557	0.59	0.57	0.566	0.35	0.36	0.317
NCNN	0.61	0.60	**0.596**	0.61	0.58	0.573	0.39	0.41	**0.380**

Table 5.12: Results on TRAC English dataset: fastText scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.51	0.51	0.5070	0.64	0.46	0.5035	0.55	0.56	**0.5495**
LR	0.51	0.52	0.5067	0.65	0.49	0.5400	0.40	0.40	0.3871
KNN	0.48	0.49	0.4840	0.61	0.47	0.5113	0.41	0.42	0.3997
SVC	0.51	0.52	0.5104	0.61	0.47	0.5137	0.29	0.29	0.2858
DC	0.45	0.45	0.4503	0.59	0.46	0.5002	0.39	0.40	0.3948
SGD	0.41	0.43	0.4171	0.58	0.48	0.5167	0.40	0.39	0.3838
RF	0.47	0.48	0.4705	0.63	0.52	0.5513	0.43	0.43	0.4069
Ridge	0.52	0.52	0.4995	0.63	0.48	0.5225	0.32	0.33	0.3180
AdaB	0.51	0.51	0.5074	0.67	0.52	0.5644	0.45	0.44	0.4215
Perce.	0.42	0.43	0.4259	0.57	0.44	0.4848	0.34	0.34	0.3340
ANN	0.54	0.54	0.5343	0.67	0.50	0.5380	0.46	0.47	0.4532
Ensemble	0.50	0.51	0.4912	0.64	0.52	0.5617	0.51	0.49	0.4471
LSTM	0.59	0.57	**0.5693**	0.70	0.46	0.5062	0.54	0.54	0.5335
BLSTM	0.56	0.55	0.5478	0.69	0.52	**0.5641**	0.52	0.51	0.4985
CNN	0.58	0.57	0.5586	0.70	0.52	0.5638	0.52	0.51	0.4849
NCNN	0.58	0.56	0.5608	0.69	0.44	0.4849	0.54	0.51	0.5179

Table 5.13: Results on TRAC Hindi dataset: fastText scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.39	0.41	0.3098	0.23	0.41	0.2959	0.35	0.37	0.3359
LR	0.58	0.57	0.5682	0.57	0.55	0.5457	0.32	0.32	0.3184
KNN	0.51	0.51	0.5118	0.52	0.51	0.5106	0.34	0.34	0.3381
SVC	0.57	0.57	0.5631	0.55	0.52	0.5186	0.31	0.31	0.3087
DC	0.48	0.48	0.4774	0.45	0.44	0.4392	0.35	0.35	0.3475
SGD	0.42	0.42	0.4144	0.36	0.37	0.3670	0.29	0.28	0.2739
RF	0.53	0.53	0.5262	0.51	0.50	0.5047	0.34	0.35	0.3449
Ridge	0.58	0.57	0.5612	0.55	0.52	0.5092	0.34	0.33	0.3346
AdaB	0.52	0.51	0.5110	0.54	0.54	0.5336	0.35	0.34	0.3261
Perce.	0.39	0.39	0.3890	0.38	0.37	0.3763	0.29	0.28	0.2787
ANN	0.60	0.57	0.5664	0.62	0.59	0.5842	0.26	0.25	0.2399
Ensemble	0.57	0.57	0.5649	0.64	0.52	0.5617	0.36	0.35	0.3426
LSTM	0.63	0.61	0.6091	0.65	0.61	**0.6021**	0.37	0.39	**0.3667**
BLSTM	0.59	0.57	0.5582	0.64	0.60	0.5770	0.31	0.35	0.3005
CNN	0.61	0.60	0.6008	0.64	0.61	0.5950	0.32	0.35	0.2669
NCNN	0.63	0.61	**0.6101**	0.65	0.61	0.5912	0.37	0.38	0.3494

Table 5.14 and table 5.15 show results of Doc2Vec-dm and Doc2VEC-dbow models on TRAC English corpus. Similarly, table 5.16 and 5.17 show results of Doc2Vec-dm and Doc2VEC-dbow models on TRAC Hindi corpus respectively. From the table, one can observe that results paragraph model are substantially lower than Bag-of-Word model.

From the DOC2Vec results, we conclude that Doc2vec dbow model marginally perform better than dmc.

Results on Pre-trained Embedding Vector: Word embedding represents a word into a vector of real numbers in nth dimensional space. It is features that words that have similar meaning can be made to correspond to close vector and obtain meaningful results. Word embedding is used for various NLP applications such as part-of-speech tagging, information retrieval, question answering, etc.

Many Natural Language Processing applications nowadays rely on pre-trained word representations estimated from large text corpora such as news collections, Wikipedia, and Web Crawl. Google's Word2vec pre-trained model includes word vectors for a vocabulary of 3 million words and phrases that they trained on roughly 100 billion words from a Google News dataset. Word embedding size is 300 features. Table 5.18 shows result on

Table 5.14: Results on TRAC English dataset : Doc2Vec-dm scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.38	0.38	**0.3827**	0.55	0.42	0.4585	0.32	0.33	0.3254
LR	0.39	0.42	0.3752	0.54	0.52	0.5266	0.32	0.35	0.3041
KNN	0.35	0.39	0.3467	0.53	0.50	0.5114	0.35	0.36	0.3225
SVC	0.39	0.41	0.3768	0.56	0.53	0.5388	0.31	0.35	0.3019
DC	0.33	0.33	0.3338	0.52	0.38	0.4198	0.33	0.34	0.3326
SGD	0.37	0.39	0.3697	0.53	0.50	0.5060	0.33	0.33	0.3251
RF	0.33	0.34	0.3358	0.52	0.38	0.4230	0.33	0.33	0.3301
Ridge	0.39	0.42	0.3735	0.55	0.53	**0.5385**	0.31	0.35	0.2994
AdaB	0.36	0.38	0.3611	0.53	0.43	0.4689	0.33	0.34	**0.3288**
Perce.	0.37	0.36	0.3617	0.53	0.34	0.3800	0.30	0.31	0.3015
ANN	0.39	0.41	0.3844	0.53	0.48	0.4980	0.32	0.33	0.3230

Table 5.15: Results on TRAC English dataset : Doc2Vec-Dbow scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.43	0.42	0.4267	0.60	0.42	0.4634	0.36	0.35	**0.3536**
LR	0.44	0.45	0.4383	0.58	0.48	**0.5139**	0.33	0.34	0.3274
KNN	0.40	0.42	0.3964	0.54	0.49	0.5095	0.33	0.35	0.3191
SVC	0.44	0.45	**0.4403**	0.58	0.47	0.5033	0.33	0.34	0.3274
DC	0.33	0.33	0.3338	0.52	0.38	0.4198	0.33	0.34	0.3326
SGD	0.40	0.42	0.3988	0.58	0.32	0.3521	0.34	0.34	0.3350
RF	0.34	0.34	0.3359	0.52	0.38	0.4210	0.33	0.33	0.3293
Ridge	0.44	0.45	0.4389	0.58	0.47	0.5083	0.33	0.34	0.3243
AdaB	0.41	0.42	0.4124	0.58	0.44	0.4852	0.32	0.33	0.3223
Perce.	0.41	0.37	0.3664	0.60	0.30	0.3253	0.32	0.30	0.2990
ANN	0.43	0.43	0.4163	0.59	0.40	0.4401	0.35	0.34	0.3281

Table 5.16: Results on TRAC Hindi dataset : Doc2Vec-dmc scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.37	0.32	0.2962	0.42	0.37	0.3459	0.34	0.35	**0.3270**
LR	0.41	0.42	0.3892	0.45	0.42	0.3894	0.39	0.30	0.2438
KNN	0.39	0.39	0.3855	0.38	0.38	0.3768	0.33	0.32	0.3051
SVC	0.42	0.44	**0.4064**	0.43	0.42	0.3879	0.39	0.32	0.2580
DC	0.38	0.38	0.3789	0.35	0.35	0.3485	0.33	0.31	0.2988
SGD	0.38	0.41	0.3712	0.47	0.44	0.3331	0.35	0.31	0.2605
RF	0.38	0.38	0.3785	0.35	0.35	0.3512	0.33	0.31	0.2981
Ridge	0.43	0.44	0.4022	0.45	0.42	0.3866	0.40	0.32	0.2549
AdaB	0.41	0.43	0.4029	0.38	0.40	0.3751	0.35	0.30	0.2614
Perce.	0.39	0.38	0.2894	0.32	0.35	0.2661	0.37	0.33	0.2616
ANN	0.42	0.43	0.3933	0.47	0.44	**0.4091**	0.34	0.30	0.2419

Table 5.17: Results on TRAC Hindi dataset : Doc2Vec-dbow scheme

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.47	0.46	0.4631	0.48	0.48	**0.4736**	0.35	0.35	**0.3205**
LR	0.48	0.49	0.4738	0.45	0.45	0.4380	0.33	0.32	0.2833
KNN	0.45	0.43	0.4264	0.41	0.41	0.4038	0.34	0.34	0.3299
SVC	0.47	0.48	0.4704	0.44	0.45	0.4344	0.34	0.32	0.2905
DC	0.38	0.38	0.3789	0.35	0.35	0.3485	0.33	0.31	0.2988
SGD	0.43	0.42	0.4135	0.41	0.42	0.4134	0.36	0.32	0.2588
RF	0.38	0.38	0.3792	0.34	0.35	0.3477	0.33	0.31	0.2988
Ridge	0.47	0.48	0.4696	0.44	0.44	0.4292	0.34	0.32	0.2875
AdaB	0.46	0.47	0.4563	0.44	0.43	0.4214	0.34	0.33	0.2933
Perce.	0.41	0.41	0.3507	0.40	0.38	0.3282	0.34	0.32	0.2752
ANN	0.48	0.48	**0.4751**	0.45	0.45	0.4440	0.36	0.34	0.3132

Google Word2vec pre-trained model. Glove pre-trained vector is created on Common Crawl, having 840B tokens, 2.2M length of vocabulary is 2.2 M. size of Each word vector is 300. Table 5.19 show results obtained via GLOVE pre-trained Model. Pre-trained word vectors for 157 languages [76], trained on Common Crawl and Wikipedia using fastText are publicly available. These models were trained using CBOW with position-weights, in dimension 300, with character n-grams of length 5, a window of size 5 and 10 negatives. Table 5.20 and table 5.21 shows results on English and Hindi dataset respectively.

Results on Sentence Embedding Schemes: Table 5.22 and 5.23 show results of various sentence embedding methods such as Infersent, SIF, USE, Skip-thoughts, p-mean, BERT, and ELMO. It has been worth to note that Google's universal sentence encoder produces a better-weighted F1-score among all across both the datasets.

Results on Contextual Pre-trained Language Model: Table 5.24 shows results of various transfer learning methods such as ELMo, ULMFit, and BERT. It has been worth to note that BERT produce better F1-score than ElMo, and ULMFiT.

Table 5.25 show result comparison with the peers[51] on TRAC dataset [52]. One can observe that the weighted F_1 score on the test data is better than the validation dataset. Posts in the datasets are related to different events. Without any given context, our model gives a weighted F_1 score around 0.6407 for the English Facebook dataset and 0.6081 for Hindi Facebook Dataset. The Hindi and English social media test datasets were created

Table 5.18: Results on TRAC English dataset: Google pre-trained Word2Vec model

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.50	0.50	0.4996	0.67	0.49	0.5342	0.41	0.42	0.4152
LR	0.52	0.53	0.5151	0.67	0.54	0.5799	0.45	0.44	0.4197
KNN	0.47	0.47	0.4722	0.62	0.46	0.4981	0.34	0.35	0.3405
SVC	0.52	0.53	0.5208	0.67	0.55	0.5832	0.47	0.46	0.4446
DC	0.43	0.43	0.4282	0.56	0.43	0.4700	0.36	0.37	0.3640
SGD	0.42	0.43	0.4234	0.64	0.46	0.5019	0.37	0.37	0.3692
RF	0.45	0.46	0.4481	0.60	0.51	0.5402	0.35	0.36	0.3394
Ridge	0.53	0.53	0.5095	0.67	0.55	**0.5829**	0.45	0.43	0.4092
AdaB	0.48	0.49	0.4791	0.64	0.54	0.5713	0.45	0.45	0.4241
Perce.	0.47	0.47	0.4683	0.64	0.47	0.5114	0.42	0.43	0.4224
ANN	0.51	0.51	0.5034	0.53	0.51	0.5025	0.40	0.38	0.3728
LSTM	0.60	0.57	0.5695	0.73	0.45	0.4979	0.56	0.55	**0.5537**
BLSTM	0.58	0.57	0.5768	0.68	0.51	0.5501	0.54	0.55	0.5359
CNN	0.59	0.57	0.5780	0.69	0.52	0.4749	0.52	0.53	0.5226
NCNN	0.59	0.58	**0.5795**	0.71	0.47	0.5169	0.56	0.55	0.5384

Table 5.19: Results on TRAC English dataset: Glove pre-trained model

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.52	0.51	0.5114	0.67	0.50	0.5373	0.46	0.47	0.4527
LR	0.54	0.55	0.5364	0.69	0.57	0.6050	0.48	0.49	0.4527
KNN	0.47	0.47	0.4664	0.61	0.47	0.5103	0.41	0.40	0.3959
SVC	0.54	0.54	0.5345	0.67	0.53	0.5678	0.48	0.49	0.4581
DC	0.43	0.43	0.4285	0.56	0.41	0.4515	0.39	0.40	0.3949
SGD	0.48	0.48	0.4806	0.64	0.51	0.5521	0.39	0.39	0.3793
RF	0.48	0.49	0.4726	0.59	0.50	0.5338	0.40	0.40	0.3716
Ridge	0.54	0.55	0.5317	0.68	0.56	0.5952	0.49	0.49	0.4530
AdaB	0.52	0.53	0.5206	0.65	0.54	0.5781	0.44	0.44	0.4261
Perce.	0.48	0.49	0.4850	0.61	0.48	0.5201	0.41	0.42	0.4118
ANN	0.53	0.51	0.5056	0.58	0.56	0.5498	0.42	0.38	0.3722
LSTM	0.60	0.57	0.5695	0.73	0.45	0.4979	0.56	0.55	0.5518
BLSTM	0.59	0.59	0.5893	0.71	0.57	**0.6062**	0.55	0.56	0.5466
CNN	0.60	0.59	**0.5941**	0.70	0.50	0.5405	0.57	0.57	**0.5667**
NCNN	0.57	0.58	0.5672	0.66	0.56	0.5883	0.51	0.53	0.5067

Table 5.20: Results on TRAC English dataset: pre-trained fastText model

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.52	0.52	0.5206	0.67	0.51	0.5519	0.42	0.44	0.4276
LR	0.56	0.56	0.5478	0.69	0.57	0.6045	0.46	0.47	0.4441
KNN	0.48	0.48	0.4756	0.60	0.43	0.4819	0.40	0.40	0.3912
SVC	0.55	0.55	0.5433	0.70	0.58	0.6120	0.45	0.45	0.4350
DC	0.44	0.44	0.4373	0.58	0.45	0.4900	0.36	0.37	0.3632
SGD	0.46	0.47	0.4625	0.62	0.50	0.5360	0.39	0.39	0.3852
RF	0.46	0.48	0.4609	0.61	0.52	0.5505	0.38	0.40	0.3687
Ridge	0.55	0.54	0.5315	0.70	0.58	0.6140	0.48	0.47	0.4461
AdaB	0.51	0.52	0.5075	0.66	0.56	0.5907	0.42	0.43	0.4033
Perce.	0.47	0.48	0.4686	0.64	0.53	0.5660	0.41	0.41	0.4049
ANN	0.56	0.56	0.5575	0.70	0.53	0.5722	0.52	0.49	0.4842
Ensemble	0.51	0.51	0.4934	0.64	0.51	0.5558	0.49	0.49	0.4500
LSTM	0.59	0.58	**0.5900**	0.69	0.58	**0.6178**	0.57	0.58	**0.5541**
BLSTM	0.59	0.58	0.5800	0.69	0.56	0.6000	0.55	0.56	0.5423
CNN	0.58	0.59	0.5800	0.71	0.61	**0.6407**	0.56	0.59	**0.5520**
NCNN	0.60	0.58	0.5800	0.72	0.51	0.5600	0.55	0.56	0.5407

Table 5.21: Results on TRAC Hindi dataset: fastText pre-trained model

Classifiers	Validation Dataset			FB Test Dataset			Twitter Test Dataset		
	P	R	F1	P	R	F1	P	R	F1
NB	0.42	0.40	0.3348	0.42	0.40	0.3176	0.29	0.30	0.2897
LR	0.55	0.55	0.5448	0.57	0.55	0.5518	0.40	0.36	0.3524
KNN	0.47	0.47	0.4696	0.50	0.49	0.4909	0.35	0.33	0.2917
SVC	0.53	0.53	0.5304	0.57	0.54	0.5442	0.39	0.36	0.3472
DC	0.44	0.44	0.4383	0.43	0.43	0.4288	0.38	0.35	0.3473
SGD	0.47	0.47	0.4622	0.48	0.48	0.4746	0.34	0.32	0.3163
RF	0.49	0.48	0.4834	0.48	0.48	0.4788	0.37	0.35	0.3288
Ridge	0.55	0.54	0.5413	0.59	0.55	0.5544	0.39	0.35	0.3361
AdaB	0.50	0.50	0.5024	0.50	0.49	0.4913	0.44	0.36	0.3441
Perce.	0.45	0.45	0.4539	0.49	0.49	0.4873	0.40	0.39	0.3835
ANN	0.54	0.54	0.5345	0.56	0.52	0.5190	0.37	0.36	0.3593
LSTM	0.65	0.61	0.6042	0.62	0.60	0.5916	0.53	0.47	0.4600
BLSTM	0.58	0.57	0.5695	0.63	0.60	0.5900	0.51	0.47	0.4600
CNN	0.63	0.62	**0.6244**	0.63	0.61	**0.6081**	0.52	0.50	**0.4992**
NCNN	0.63	0.62	0.6200	0.65	0.61	0.5965	0.51	0.47	0.4600

Table 5.22: Weighted F1 score on TRAC Facebook English Test Dataset using Sentence Embedding methods

Classifiers	InferSent	SIF	USE	Skip-thought	p-mean	BERT	ELMo
NB	0.477	**0.563**	0.530	0.535	0.412	0.504	0.438
LR	**0.616**	0.610	0.605	0.603	0.541	0.586	0.591
KNN	0.544	0.491	**0.548**	0.435	0.514	0.520	0.493
SVC	0.578	0.600	0.605	0.584	0.442	0.575	0.574
DT	0.513	0.478	0.497	0.484	0.480	0.472	0.473
SGD	0.570	**0.645**	0.519	0.617	0.622	0.441	0.421
RF	**0.580**	0.544	0.551	0.514	0.575	0.535	0.546
Ridge	0.601	0.593	0.606	0.608	0.543	0.584	0.586
AdaB	0.594	0.552	0.585	0.592	0.571	0.565	0.565
Perce.	0.398	0.460	0.447	0.559	0.404	0.435	0.516
ANN	**0.609**	0.588	0.594	0.590	0.561	0.590	0.596
Ensemble	**0.616**	0.610	0.605	0.603	0.541	0.586	0.591
LSTM	0.560	0.536	0.561	0.561	0.547	0.483	0.491
BLSTM	0.555	0.509	0.561	0.561	0.561	0.514	0.524
CNN	0.517	0.536	0.561	0.559	0.561	0.561	0.515
NCNN	0.505	0.528	0.561	0.559	0.541	0.501	0.530

Table 5.23: Weighted F1 score on TRAC Twitter English Test Dataset using Sentence Embedding methods

Classifiers	InferSent	SIF	USE	Skip-thought	p-mean	BERT	ELMO
NB	0.5138	0.5391	0.4915	0.4491	0.5228	0.5259	0.3472
LR	0.5180	0.4940	**0.5485**	0.4515	0.4470	0.4946	0.4513
KNN	0.4465	0.4241	**0.4580**	0.3659	0.4418	0.4471	0.3913
SVC	0.4923	0.4976	**0.5402**	0.4606	0.4229	0.4977	0.4265
DT	0.4135	0.3750	**0.4238**	0.3828	0.4005	0.3774	0.3906
SGD	0.4713	0.4893	0.4809	0.3997	0.2064	0.3513	0.3835
RF	0.4009	0.4012	**0.4507**	0.3620	0.3791	0.4024	0.3805
Ridge	0.5043	0.4793	**0.5351**	0.4727	0.4311	0.4975	0.4238
AdaB	0.5180	0.4818	**0.5281**	0.4595	0.5104	0.4484	0.4312
Perce.	0.3721	0.4005	0.4439	0.4285	0.3973	0.4266	0.3960
ANN	0.5183	0.5214	**0.5482**	0.4952	0.2150	0.5045	0.4724
Ensemble	0.5180	0.4940	**0.5485**	0.4515	0.4470	0.4946	0.4513
LSTM	0.2133	0.3417	0.2133	0.2133	0.3774	0.3163	0.2843
BLSTM	0.2360	0.3861	0.2133	0.2133	0.2133	0.3616	0.3639
CNN	0.3078	0.2265	0.2133	0.2133	0.2133	0.2133	0.3918
NCNN	0.3078	0.2318	0.2133	0.2133	0.2991	0.3550	0.3876

Table 5.24: F1 score on TRAC English Test Dataset using Transfer Learning methods

Transfer learning Model	English	
	Facebook Dataset	Twitter Dataset
ELMO	0.3699	0.3854
ULMFiT	0.4725	0.4664
BERT	**0.6184**	**0.5683**

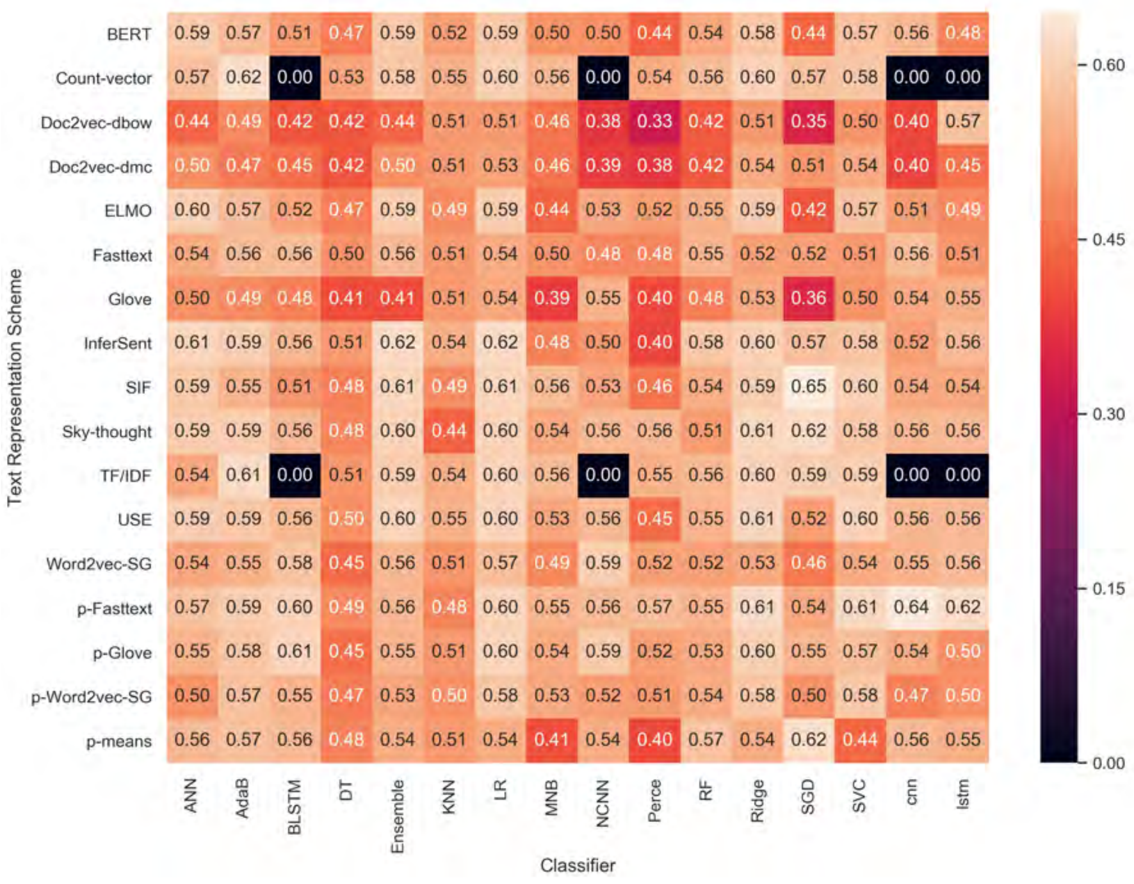
Table 5.25: weighted F1-score TRAC Test Dataset: comparison with peers

System	English Dataset		Hindi Dataset	
	Facebook Dataset	Twitter Dataset	Facebook Dataset	Twitter Dataset
Our system result	**0.6407**	0.5541	0.6081	**0.4992**
DA-LD-hildesheim [69]	0.6178	0.552	0.6081	**0.4992**
saroyehun [6]	**0.6425**	0.5920	NA	NA
EBSI-LIA-UNAM [7]	0.6315	0.5715	NA	NA
TakeLab [51]	0.5920	0.5651	NA	NA
taraka rama [51]	0.6008	0.5656	**0.6420**	0.40
vista.ue [51]	0.5812	**0.6008**	0.5951	0.4829
na14 [51]	0.5920	0.5663	0.6450	0.4853

from Twitter with 3 or 4 domains like #JNUShutdown, #Cricket2015, #demonetization. We have trained models on Facebook comments or posts and tested on Twitter posts. It is worth to note that there are lexical differences between Twitter posts and Facebook posts. Twitter posts are 140 characters long, and the majority contain user mentions, external URL, and hashtags while most of Facebook posts are longer than Twitter posts does not have Hashtags or user mentions in the text. However, our model gives a weighted F1 score around 0.5520 for the English Twitter dataset and for the code-mixed Hindi Twitter dataset, our model gives F1 weighted around 0.4992 in model based on Convolution Neural network.

5.1.6 Result Analysis

In this section, we will present the comprehensive result analysis and try to answer the research questions which were framed before the experiments were performed. As we look at the heatmaps in figure 5.2 5.3, Overall, LSTM and CNN with pre-trained fastText word embedding marginally outperform (around 2 % to 4%) traditional and ensemble of



	ANN	AdaB	BLSTM	DT	Ensemble	KNN	LR	MNB	NCNN	Perce	RF	Ridge	SGD	SVC	cnn	lstm
BERT	0.59	0.57	0.51	0.47	0.59	0.52	0.59	0.50	0.50	0.44	0.54	0.58	0.44	0.57	0.56	0.48
Count-vector	0.57	0.62	0.00	0.53	0.58	0.55	0.60	0.56	0.00	0.54	0.56	0.60	0.57	0.58	0.00	0.00
Doc2vec-dbow	0.44	0.49	0.42	0.42	0.44	0.51	0.51	0.46	0.38	0.33	0.42	0.51	0.35	0.50	0.40	0.57
Doc2vec-dmc	0.50	0.47	0.45	0.42	0.50	0.51	0.53	0.46	0.39	0.38	0.42	0.54	0.51	0.54	0.40	0.45
ELMO	0.60	0.57	0.52	0.47	0.59	0.49	0.59	0.44	0.53	0.52	0.55	0.59	0.42	0.57	0.51	0.49
Fasttext	0.54	0.56	0.56	0.50	0.56	0.51	0.54	0.50	0.48	0.48	0.55	0.52	0.52	0.51	0.56	0.51
Glove	0.50	0.49	0.48	0.41	0.41	0.51	0.54	0.39	0.55	0.40	0.48	0.53	0.36	0.50	0.54	0.55
InferSent	0.61	0.59	0.56	0.51	0.62	0.54	0.62	0.48	0.50	0.40	0.58	0.60	0.57	0.58	0.52	0.56
SIF	0.59	0.55	0.51	0.48	0.61	0.49	0.61	0.56	0.53	0.46	0.54	0.59	0.65	0.60	0.54	0.54
Sky-thought	0.59	0.59	0.56	0.48	0.60	0.44	0.60	0.54	0.56	0.56	0.51	0.61	0.62	0.58	0.56	0.56
TF/IDF	0.54	0.61	0.00	0.51	0.59	0.54	0.60	0.56	0.00	0.55	0.56	0.60	0.59	0.59	0.00	0.00
USE	0.59	0.59	0.56	0.50	0.60	0.55	0.60	0.53	0.56	0.45	0.55	0.61	0.52	0.60	0.56	0.56
Word2vec-SG	0.54	0.55	0.58	0.45	0.56	0.51	0.57	0.49	0.59	0.52	0.52	0.53	0.46	0.54	0.55	0.56
p-Fasttext	0.57	0.59	0.60	0.49	0.56	0.48	0.60	0.55	0.56	0.57	0.55	0.61	0.54	0.61	0.64	0.62
p-Glove	0.55	0.58	0.61	0.45	0.55	0.51	0.60	0.54	0.59	0.52	0.53	0.60	0.55	0.57	0.54	0.50
p-Word2vec-SG	0.50	0.57	0.55	0.47	0.53	0.50	0.58	0.53	0.52	0.51	0.54	0.58	0.50	0.58	0.47	0.50
p-means	0.56	0.57	0.56	0.48	0.54	0.51	0.54	0.41	0.54	0.40	0.57	0.54	0.62	0.44	0.56	0.55

Figure 5.2: Heatmap on English Facebook Test Dataset Results.

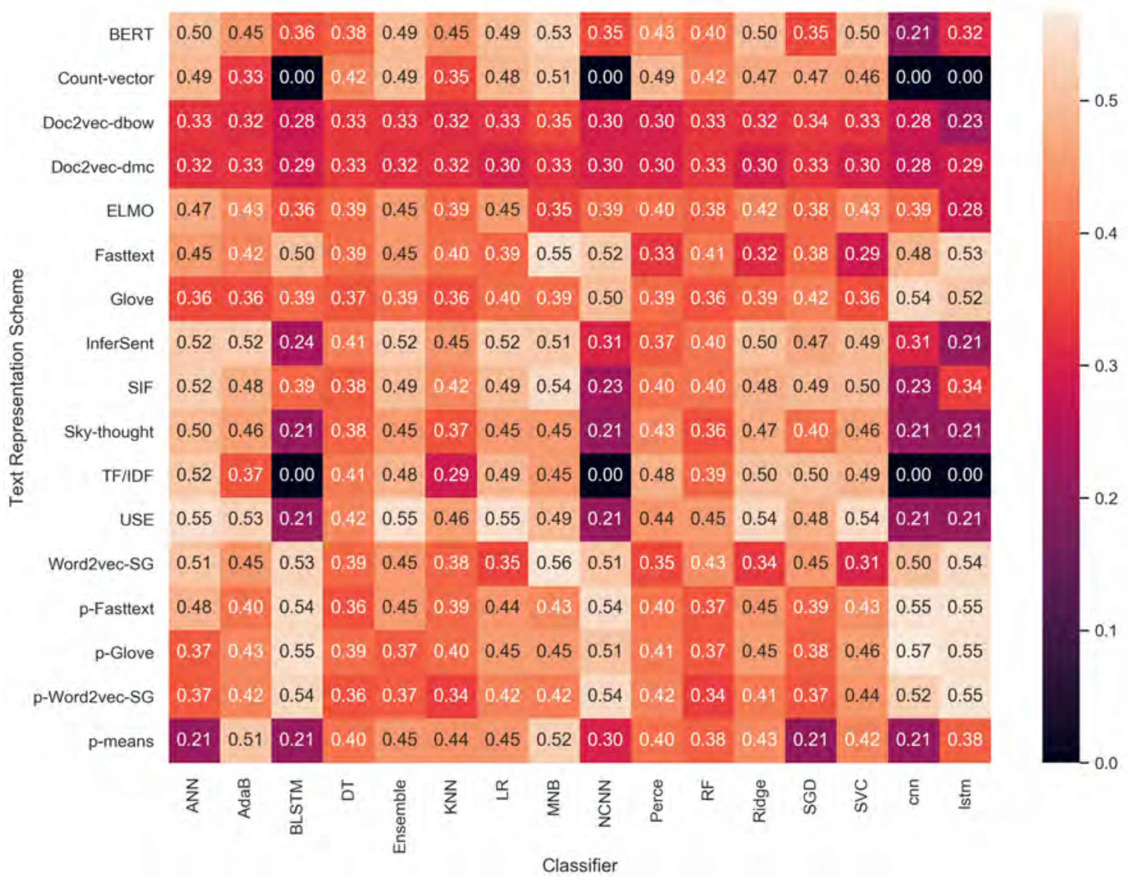


Figure 5.3: Heatmap on English Twitter Test Dataset Results.

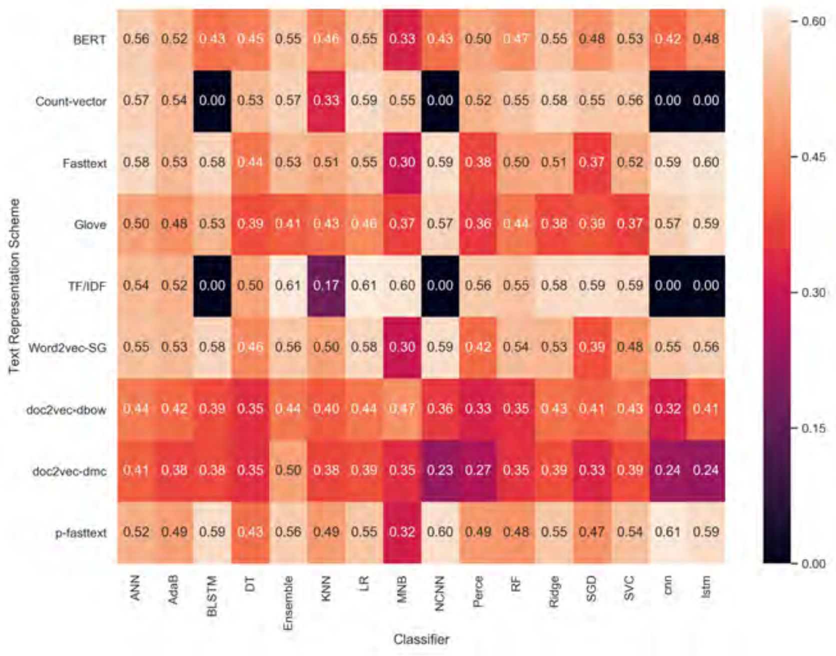


Figure 5.4: Heatmap on TRAC Hindi Facebook Test Dataset Results.

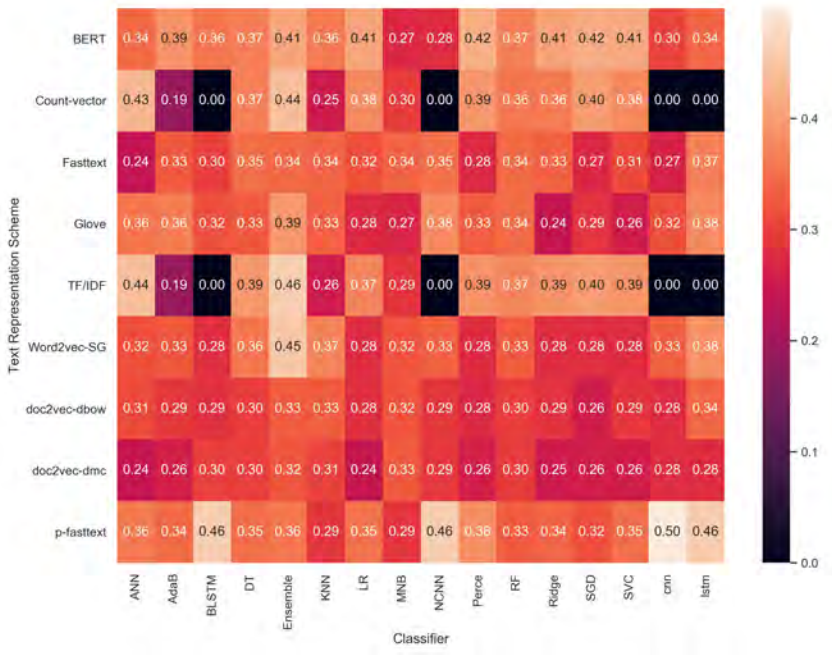


Figure 5.5: Heatmap on TRAC Hindi Twitter Test Dataset Results.

Table 5.26: Results Comparison with CNN model and Logistic Regression TRAC Facebook English Test Dataset

Class	CNN model			Logistic Regression			#Posts
	P	R	Weighted F1	P	R	Weighted F1	
NAG	0.86	0.64	0.73	0.83	0.60	0.70	630
CAG	0.28	0.46	0.35	0.23	0.54	0.32	142
OAG	0.42	0.61	0.50	0.46	0.39	0.42	144
overall	0.70	0.61	0.64	0.68	0.56	0.60	916

Table 5.27: sample post for the CAG class.

Text	Gold Label	CNN label	LR label
Mauni singh trying very hard to convince himself what is written in script... body language says it all	CAG	CAG	NAG
Indian govt is all Abt giving money to Bangladesh on the terms of Bangladesh ll give that all projects to amabani n adani for thier benefits lol who cares Abt soldiers or India they r just puppets of thier owners feku or pappu	CAG	CAG	NAG
When asked to speak in Parliament ran away. Speaks only in TV,radio or in election rally. Can we expect Another crying drama after Demonetisation disaster ? #cryBaby"	CAG	CAG	NAG

classifier with respect to weighted F_1- score on Facebook English corpus and substantially outperforms on Twitter English corpus. By and large similar results are observed on code-mixed Hindi corpus as shown in figure 5.4, and 5.5.

Similarly, Googles' universal sentence encoder (USE) perform better across all traditional classifier than the rest of other schemes for both the English dataset.

Table 5.26 presents the detailed comparative results of two classifiers: CNN model with fastText pre-trained vector and the logistic regression with TF/IDF weighting on TRAC Facebook English dataset. The CNN model classifies Facebook posts better than logistic regression at the individual class level and overall. It has been quite evident that posts belong to CAG class are hard to classify and [70] reported that the same observation. Table 5.27 show posts that are miss-classified by logistic regression however, the CNN model correctly classified them into the CAG class.

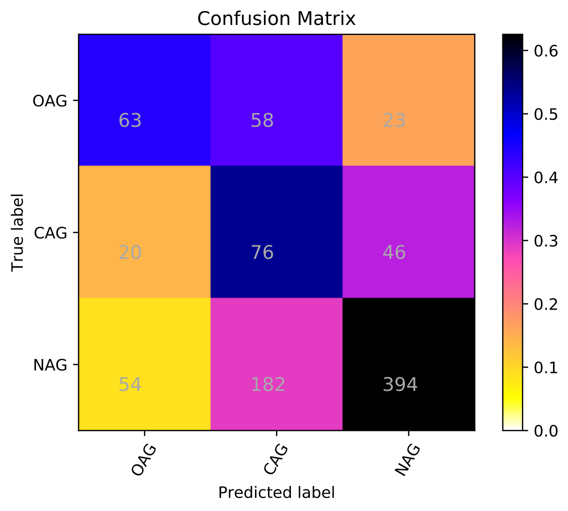


Figure 5.6: Confusion Matrix: LSTM model on Facebook English Dataset

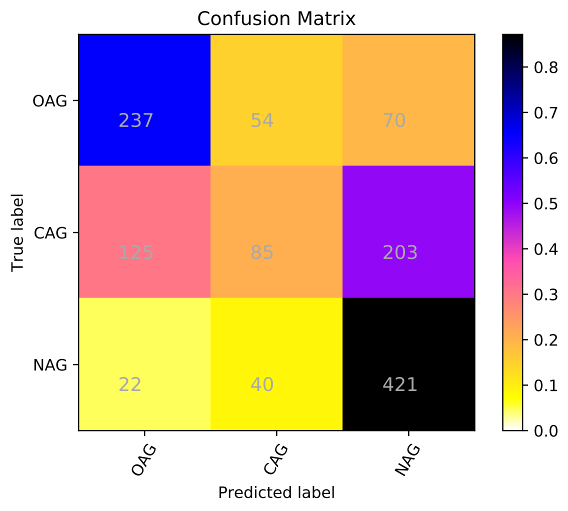


Figure 5.7: Confusion Matrix for:CNN model on Twitter English dataset

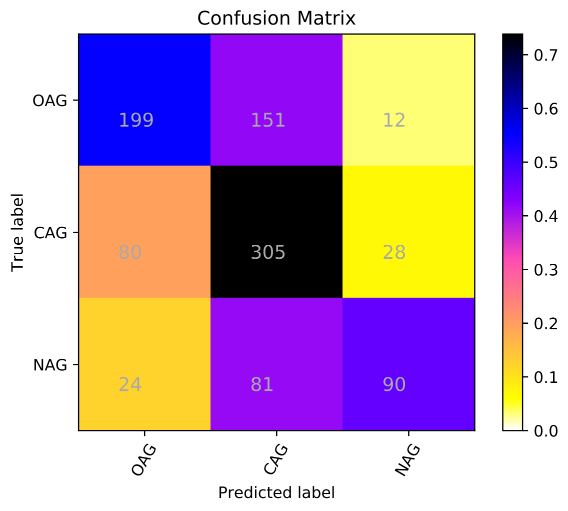


Figure 5.8: Confusion Matrix:CNN model On Facebook Hindi dataset

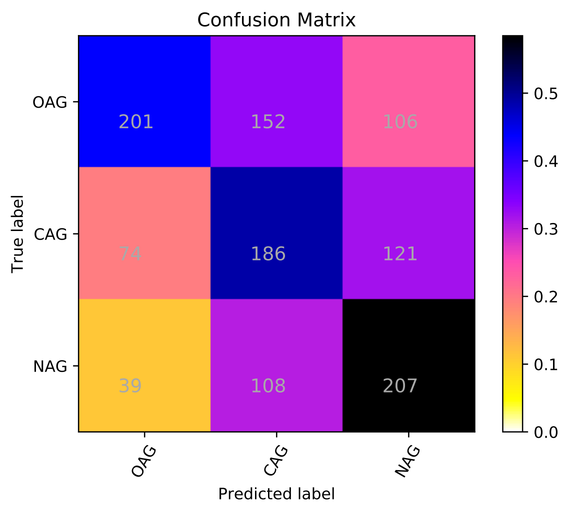


Figure 5.9: Confusion Matrix : CNN Model Twitter Hindi dataset

Significance Test

The statistical significance test theoretically reaffirms claims drawn from the experiment outcome. parametric tests, such as Student t-test and non-parametric, such as Wilcoxon signed-rank test can be considered to check the significance of the results. The parametric test assumes that data is drawn from Gaussian distribution. The results of the parametric test might be misleading if the data are not drawn from the Gaussian distribution. Statistical normality test such as Shapiro-Wilk is performed to ensure that data drawn from the normal distribution. Table 5.28 shows the Shapiro-Wilk normality test. The F1-score of the many text representation schemes are not normally distributed. Student t-test can not be performed where data of any text representation scheme is failed in the normality test. Therefore Non-parametric statistical test, Wilcoxon signed-rank test is performed by comparing the weighted F_1 score of each classifier for each text representation scheme with a fastText pre-trained vector scheme and universal sentence encoder scheme.

Table 5.29 and 5.30 summarizes the p-values in 3 groups of Wilcoxon signed-rank test on English and Hindi Dataset respectively. $<$ symbol denotes that p-values of test is less than significance level $\alpha = 0.01$. $\ll$ symbols used to describe that p-values of the results lie between 0.01 to 0.05. $\sim$ symbol used to describe that p-values are more than 0.05. By and large, our results are statistically significant. We have selected a text representation scheme numerically not statistically.

In the following sub-section, we will try to answer all the research questions framed

Table 5.28: Shapiro-Wilk Normality test result on TRAC Dataset

Text Representation schemes	FB English	Tw English	FB Hindi	Tw Hindi
Count Vector	Y*	N+	N	Y
TF/IDF	Y	N	N	Y
Word2vec	Y	Y	N	N
Glove	N	N	Y	Y
fastText	Y	Y	N	Y
P-Word2vec	Y	N	-	-
P-Glove	Y	N	-	-
P-fastText	Y	Y	Y	N
Doc2vec-dmc	Y	Y	Y	Y
Doc2vec-dbow	Y	N	Y	Y
InferSent	Y	N	-	-
SIF	Y	N	-	-
USE	Y	N	-	-
Skip-thoughts	N	N	-	-
p-mean	N	N	-	-
BERT	Y	Y	Y	Y
ELMO	Y	Y	-	-

* Y: positive outcome of Shapiro-Wilk Normality test

+N: Shapiro-Wilk Normality test fail to ensure Normality significant difference with $\alpha = 0.05$

Table 5.29: p-values of Significance test on F_1-score on TRAC English Dataset

Text Rep. scheme	Facebook English		Twitter English	
	p-fastText	USE	p-fastText	USE
Count Vector	$\ll^{**}$	$\ll$	$<$	$<^{*}$
TF/IDF	$\ll$	$\ll$	$<$	$<$
Word2vec	$\ll$	$\ll$	$\sim^{+}$	$\sim$
Glove	$<$	$<$	$<$	$\sim$
fastText	$<$	$\ll$	$\sim$	$\sim$
p-Word2vec	$<$	$<$	$<<$	$\sim$
p-Glove	$\ll$	$\sim$	$\sim$	$\sim$
p-fastText	$NA^{\#}$	$\sim$	NA	$\sim$
doc2vec-dmc	$<$	$<$	$<$	$<$
doc2vec-dbow	$<$	$<$	$<$	$<$
InferSent	$\sim$	$\sim$	$\sim$	$\sim$
SIF	$\sim$	$\sim$	$\sim$	$\sim$
USE	$\sim$	NA	$\sim$	NA
Skip-Thought	$\sim$	$\sim$	$\sim$	$\ll$
p-mean	$\ll$	$\ll$		$\sim$
BERT	$<$	$<$	$\sim$	$\sim$
ELMO	$<$	$<$	$\sim$	$\sim$

* $<$ denotes that p-value is less than 0.01.**
$\ll$ denotes that p-value lies between 0.01 and 0.05.
$^{+}$ $\sim$ denotes that p-value is greater than 0.05. $^{\#}$ NA means test can not be performed with same scheme.

Table 5.30: p-values of Significance test on F_1-score on TRAC Hindi Test Dataset

Text Rep. scheme	Facebook Hindi	Code-mixed	Twitter Hindi	Code-mixed
Count Vector	$<$		$<$	
TF/IDF	$<$		$<$	
Word2vec	$\sim$		$\ll$	
Glove	$<$		$\ll$	
fastText	$\sim$		$\ll$	
doc2vec-dmc	$<$		$<$	
doc2vec-dbow	$<$		$<$	
BERT	$\sim$		$\sim$	

:* $<$ denotes that p-value is less than 0.01.
** $\ll$ denotes that p-value lies between 0.01 and 0.05.$^{+}$ $\sim$ denotes that p-value is greater than 0.05.

during the experiments were planned.

RQ1: Which is the best text representation scheme to model the text from social web for the text classifier?

Text representation is the primary task to address any NLP downstream task such as Question/answering, text classification, etc. As discusses in section 5.1.2, there are four types of text representing schemes: Bag-of-Words(BoW), word embedding, sentence embedding, and pre-trained contextual language model.

Results clearly indicate that models with fastText pre-trained vector outperform Glove and Word2Vec pre-trained vector on Facebook test dataset as well as the Twitter test dataset. The main reason behind the outperformance of fastText over Glove and Word2vec is that The fastText considers each word as N-gram characters. A word vector for a word is computed from the sum of the n-gram characters. Glove and Word2vec consider each word as a single unit and provide a word vector for each word. Since Facebook users make a lot of mistakes in spelling, typos, fastText is more convenient than Glove [69]. from Figure 5.2 and 5.3 shows that BoW is still effective text representation scheme for the traditional classifier which takes hand-crafted feature and n-grams as inputs. Logistic Regression and Support Vector performs better than other classifiers in English as well as Hindi Dataset. Adaboost performs better than LR and SVC on Facebook English Dataset but substantially underperforms them on the rest of the three datasets. Our participation [69] in TRAC competition and FIRE Information Retrieval from Microblogs during Disasters [12] track performed well and secured top position.

RQ2:Transfer Learning Model vs. Pre-trained Word Embedding Model

Transfer learning is focused on storing knowledge gained while solving one problem and applying it to a different but related problem. On many occasions, NLP researchers face the problem of unavailability of sufficient labeled data to train the model. The new transfer learning method like ELMO [92], Universal language model fine-tuning for Text Classification (ULMFiT) [49], BERT, etc. attracts interest among NLP Researchers. These models are trained on large text corpus with language model objective and fine-tuned on the task-specific corpus. We have used these transfer learning model on TRAC English dataset [**?**] and results are presented in table 5.24. one can observe that results except are substantially lower than the results reported in figure 5.2 and 5.3 where pre-trained word vectors are used to initialize the first layer of deep neural model and rest of the network is trained from scratch achieves better results than transfer learning model. [49] termed the use of pre-trained vector as shallow representation.

In these experiments, we trained different classifier models on Facebook posts. Heatmaps in Figure 5.3 5.5 shows the results on TRAC Twitter dataset. There are lexical differences between Facebook and Twitter posts. From the results shown in figure 5.2 5.3 5.4, and 5.5, one can conclude that weighted F_1 score of traditional classifier substantially lower on Twitter dataset as compare to Facebook Dataset. While deep learning models reasonably perform better than traditional classifiers on the Twitter Dataset. Thus, deep learning models are more robust than traditional classifiers across the diverse datasets.

RQ3: Does the size of embedding matter?

Sentence embedding schemes provide embed size ranging from 300 to 4800 as shown in Table 5.3. As we look at figure 5.2 and 5.3, Googles' Universal encoder which generates sentence vector of the size 512 performs better than other sentences embedding schemes, such as Infersent, Skip-thoughts, and p-means that use sentence vector of size 4096,4800 and 3600. Hence one can conclude that high dimensional vector does not improve the performance. In fact, the high dimensional vector adversely impacts the performance of the classifiers.

5.2 Offensive Content Detection

NLP researchers are developing innovative systems based on the input of the text data. The power of predictions has moved from simple text classification tasks too much more advanced labeling of the content. The task-related to hate, aggression, abusive or offensive speech currently attracts research more to algorithms making decisions which can also be ambiguous for humans. Researchers working in the area of domain-specific sentiment analysis move to the problem of domain-specific or open domain hate or offensive speech detection. They are reshaping the hate speech problem into a new notion like

Table 5.31: sample post for the each class at each level OLID dataset.

Tweet	Level-1	Level-2	Level-3
@USER @USER Sharyl Attkisson is NOT the MSM! She is a truth-teller.	NOT	NULL	NULL
@USER still waiting for mine fat	OFF	UNT	NULL
@USER She is f*cking delusional	OFF	TIN	IND
Such delusional liberals..! So twisted with hate you can't even acknowledge the U.S. Airforce..!!	OFF	TIN	OTH
@USER EVERYTHING is an issue with Muslims..go live in another country like Iran..	OFF	TIN	GRP

abusive, aggressive, or offensive speech. Such categorization of social media posts, help law-enforcement agencies with the surveillance of social media. The simple binary classification task problem turning into a much more fine-grained classification problem which not only filters the offensive content but also predicts the target and target type.

5.2.1 Problem Formulation

In this research task, an offensive language identification problem is considered as a 3-level classification task [133]. In the first level, systems are required to classify tweets into two classes, namely: Offensive (OFF) and Non-offensive (NOT). In the second level, offensive tweets are further needed to be categorized into two labels, namely: targeted (TIN)-post which contain threat/insult to the targeted entity and untargeted (UNT), respectively. In the third level, the target of insults and threats are further classified to Individual (IND), Group (GRP), or Other (OTH) classes. Table 5.31 shows the sample post of each class at each level.

5.2.2 OLID Dataset

OLID stands for Offensive Language Identification. Table 5.32 presents the statistic about the dataset. One can observe that classes in the dataset, particularly for the level-2 and level 3, is highly imbalanced. Tweets are sampled using keywords that often occur in offensive language. These keywords include: she is, 'to:BreitBartNews', 'you are'. [132] have used crowdsourcing platform Figure Eight [1] for the Data annotation. Authors have ac-

[1] https://www.figure-eight.com/

cepted annotation with 100 % agreement by the two experience annotator while in case of disagreement, more annotations are carried out until more than 66 % agreement reached.

Table 5.32: OLID Dataset statistics

Details	# Tweets in Train Dataset	# Tweets in Test Dataset
Level-1: Offensive Content Identification	13240	860
Offensive posts	4440	240
Non-offensive posts	8800	620
Level-2 Target Identification	4440	240
Targeted (TIN) posts	3876	213
Non-Targeted (UNT) posts	524	27
Level 3: Target Categoztion	3876	213
Individual	2407	100
Group	1074	78
Other	395	35

Our approach for this fine-grained classification problem is based on distributed word representation and deep learning. fastText pre-trained word embedding [76] is used to initialize the embedding layer or first layer of the model and fine-tuned for the classification task. The rest of the model is still needed to be trained from scratch. In [49], the author termed this technique as a shallow representation against the hierarchical representation.

5.2.3 Methodology

Since the social media data suffers from the data sparsity problem, classifiers based on the BoW features might not be appropriate as compared to distributed word representation. Our previous work [69] also supported this intuition. Empirical evidence [69] suggests that a pre-trained vector trained on a huge corpus provides better word embedding than embedding generated from a limited training corpus. Some authors [49] termed it as a shallow-transfer learning approach. In this method, the first layer or embedding layer of the deep neural network is initialized with pre-trained vectors, and the rest of the network is trained from scratch. Since fastText generates word embedding for a word that is unseen during the training by using the subword or n-gram of the word, it is the better choice than Word2vec and Glove. As discussed in the previous section, there is a substantial class imbalance in level-2 and level 3 classification. To address this issue, class weights are incorporated into the cost function of the classifier which gives higher weight to minority class and lower weights to the majority class. Four deep learning-based models: Bidirectional LSTM, Single LSTM, CNN, and stacked CNN are designed for the classification.

Pre-processing and Word embedding

Tweets in the dataset are partially pre-processed. User mention, URL is replaced with standard tags. We did not perform any sort of further pre-processing or stemming on the texts. fastText pre-trained word vectors with dimension 300 are used to initialize the embedding layer. This model is trained on 600B tokens of commonly crawled corpus.

Model Architecture and Hyperparameters

In this sub-section, we briefly describe our models used for the classification. The first model is based on the Bidirectional LSTM model includes the embedding layer with 300 dimensions, Bidirectional LSTM layer with 50 memory units followed by one-dimensional global max pooling and dense layer with softmax/sigmoid activations. Hyperparameters are as follows: Sequence length is fixed at 30. The number of features is equal to half of the total vocabulary size in each task. Models are trained for 10 epochs. Adam optimization algorithm is used to update network weights.

The second model is based on LSTM. The model includes an embedding layer with 300 dimensions, LSTM layer with 64 memory units, followed by two dense layers with softmax/sigmoid activations. A dropout layer is added to the hidden layer to counter the overfitting. Hyperparameters of the model are the same as the first model.

The rest of the two models are based on Convolution Neural Network, includes embedding layer with embed size of 300, followed by a one-dimensional convolution layer with 100 filters of height 2 and stride 1 to target bigrams. In addition to this, the Global Max Pooling layer is added. The pooling layer fetches the maximum value from the filters, which are fed to the dense layer. There are 256 nodes in the hidden layer without any dropout. The last model is the same as the previous CNN model except three one-dimensional convolution layer are stacked together. Different one-dimensional filters with height 2,3,4 to target bigrams, trigrams, and four-grams features. After convolution layers and max pool layer, model concatenate max pooled results from each of one-dimensional convolution layers, then build one output layer on top of them. [69]. Hyperparameters of

Table 5.33: Cross Validation results of Level 1,2,and 3 classification

Model	**Accuracy**	**F1 (macro)**
Level-1 Binary Classification		
Bidirectional LSTM	0.79	0.75
CNN	0.7915	**0.76**
LSTM-balanced	0.7708	0.74
Level-2 Binary Classification		
CNN	0.8875	**0.60**
LSTM-balanced	0.867	0.60
CNN-balanced	0.8943	0.55
level-3 Multi-class Classification		
CNN-balanced	0.695	**0.5231**
BLSTM-balanced	0.6959	0.5223
Stacked CNN	0.6920	0.5074

the model are the same as the first model.

Attention Model The attention mechanism first used in the machine translation area [10]. Attention is a mechanism that was developed to improve the performance of the Encoder-Decoder RNN on machine translation. The main objective of the attention mechanism is to pay more attention to the important word, which is decisive for the categorization of the post into a specific label. The attention layer is created inside the Bidirectional LSTM model.

5.2.4 Results

In this section, we report the results obtained by the model discussed in the previous section. We have randomly split the dataset into 80% training and 20% validation data. Table 5.33 present cross-validation results. As discussed earlier, there is a major class imbalance, especially in level 2 and level 3 datasets. Therefore, macro F1-score is used as a primary metric to measure classifier performance.

Table 5.34, 5.35, 5.36 present results of classification at level-1,2, and 3 respectively. By and large, results on test dataset are better than cross-validation. Results are comparable with the top team and substantially outperforms all the random baselines.

Table 5.34: Results for Level-1 binary classification OLID dataset

Model	Accuracy	micro-F1	macro-F1	weighted-F1
BLSTM	0.8395	0.8395	**0.7833**	**0.8321**
CNN	0.8337	0.8337	0.7799	0.828
LSTM	0.8046	0.8046	0.75	0.8062
BLSTM-attention	0.8267	0.8267	0.7664	0.8188
Bert-light	0.8325	0.8325	**0.7875**	**0.8307**
LR	0.7941	0.7941	0.6658	0.7573
SVC	0.7976	0.7976	0.7195	0.7849
NB	0.7709	0.7709	0.5947	0.7128
Peer comparison				
NULI	0.8628	-	0.8286	-
NLPR@SRPOL	-	-	0.80	-
vradivchev_anikolov	0.8547	-	0.8153	-
Random Baseline	07209		0.4189	

Table 5.35: Results on Level-2 binary classification on OLID dataset

Model	Accuracy	micro-F1	macro-F1	weighted-F1
BLSTM	0.8875	0.8875	0.5344	0.8486
CNN	0.9042	0.9042	**0.6456**	**0.8802**
lstm	0.825	0.825	0.5471	0.822
CNN-bal	0.8916	0.8916	0.6454	0.8744
bert-light	0.8917	0.8917	**0.6991**	**0.8856**
BLSTM-attention	0.8833	0.8833	0.5796	0.8565
LR	0.8875	0.8875	0.4702	0.8346
SVC	0.8875	0.8875	0.6046	0.8638
NB	0.8875	0.8875	0.4702	0.8346
Peer comparison				
NULI	0.8958	-	0.7159	-
NLPR@SRPOL	-	-	0.69	-
vradivchev_anikolov	0.8208	-	0.6674	-
Random Baseline	0.8875		0.4702	

Table 5.36: Results on Level-3 multi-class Classification on OLID dataset

Model	Accuracy	micro-F1	macro-F1	weighted-F1
CNN	0.6854	0.6854	**0.5532**	0.6495
CNN-bal	0.6666	0.6666	0.5245	0.6274
BLSTM-bal	0.6619	0.6619	0.4829	0.6104
lstm bal	0.615	0.615	0.4589	0.5709
NCNN-bal	0.6619	0.6619	0.4971	0.6154
bert-light	0.7042	0.7042	**0.5679**	0.6695
BLSTM-attention	0.6854	0.6854	0.4928	0.6227
LR	0.6525	0.6525	0.4635	0.5873
SVC	0.6384	0.6384	0.46	0.5829
NB	0.5633	0.5633	0.3671	0.4746
Peer comparison				
NULI	0.6948	-	0.5598	-
NLPR@SRPOL	-	-	0.63	-
vradivchev_anikolov	0.7277	-	0.6597	-
Random Baseline	0.4695	-	0.2130	-

5.2.5 Analysis on results

In this sub-section, we analyze the results reported in the previous sub-section. Table 5.34, 5.35, and 5.36 shows classification results on traditional classifier, deep learning models, and recently proposed BERT transformer [31] for multi-level offensive text classification.Figure 5.10, 5.11, and 5.12 show the confusion metrics of the classification. It has been evident that deep learning models and BERT substantially outperform traditional classifiers in terms of macro f1 score. The deep learning model and BERT report reasonable performance. Macro f1 and accuracy score around 78.75% and 84% in the level-1 binary classification. Fig 5.10 shows the confusion matrix for the binary classification. One can observe that many offensive posts are miss-classified as a non-offensive post while models perform better to predict non-offensive posts. In the second level binary classification, the deep learning model performs the worst in UNT class(offensive post without target). Figure 5.11 shows confusion metrics for this classification. Most of the untargeted posts are miss-classified into targeted posts.

The reason behind this underperformance is a few numbers of training examples and a high class imbalance between TIN and UNT class. However, BERT performs substantially better than deep learning models based on CNN and BLSTM. We have tried to handle class imbalance using class weights, but it turns out to be ineffective. A similar

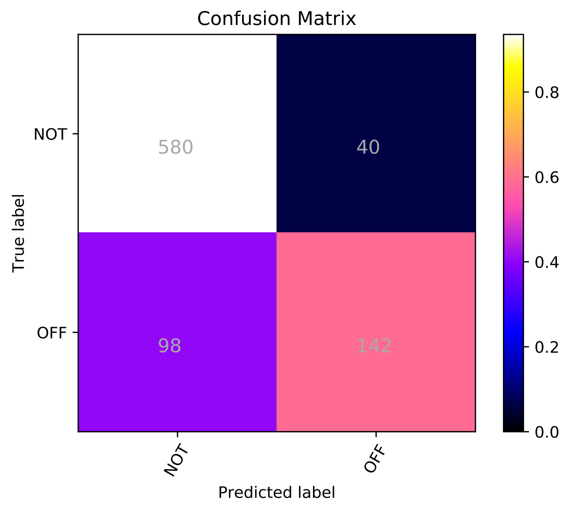


Figure 5.10: Confusion Matrix for Level 1 binary Classification:BLSTM

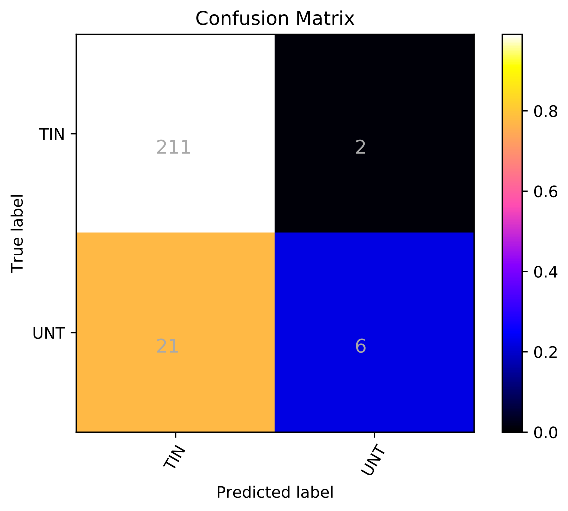


Figure 5.11: Confusion Matrix for Level 2 Binary Classification: CNN classifier

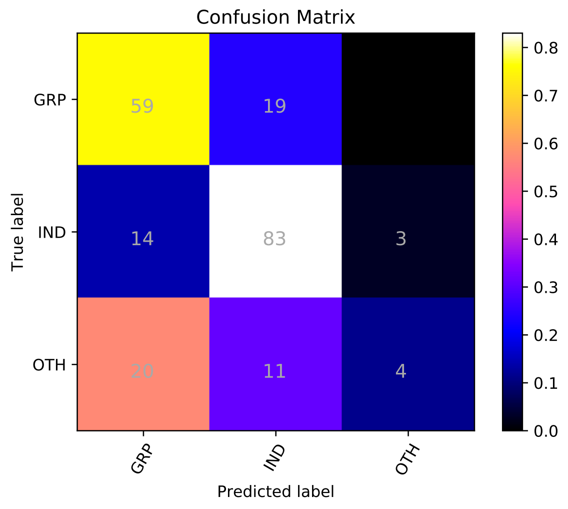


Figure 5.12: Confusion Matrix for Level 3 Multiclass Classification::CNN -balanced

case happened in the level-3 multi-class classification; we have set class weights in the cost function of the model. Unfortunately, it did not work. BERT has marginally outperformed the CNN model in terms of macro f1-score: 0.5679 vs. 0.5532. Figure 5.12 shows confusion metrics for this multi-class classification. Model incorrectly predict majority posts belongs other(OTH) class. From the results, one can conclude that BERT is a better classifier on classification tasks where the dataset contains highly imbalance class-label distribution in the training dataset. For example, the NULI team has secured the first rank in level-1 binary classification but performs poor in the level-3 multi-class classification and reported macro F1 score 0.5598 vs. 0.8286 reported in the level-1 binary classification. NLPR @SRPOL [134] used ensembles of Random Forest, OpenAI GPT, Universal encoder, the Transformer, ELMo, and combined embeddings from fast-Text and custom ones. They have reported better results than us in level-1 and level-3 classification. Team vradivchev_anikolov [134] has used the soft voting classifier of CNN, RNN and BERT base.

5.3 Factual Post/Tweet Detection from Social Media

Twitter, as one of the most used media sources for quickly spreading news, covers a range of necessary information, particularly in emergency cases. However, these situations are inevitably come along with tweets expressing the sympathy of people, rumors, and claims that are not very clearly referenced. Operating the disaster relief properly and rapidly the information processing and trustworthiness of the tweets are required to be examined. The experiment is set up on the FIRE IRMiDis dataset [12]. We have reported the best results available in the literature. Table 5.37 shows the example of the posts belonging to these classes.

During the emergency like earthquake or floods, Microblog plays a very important role as an anonymous communication medium. The various entity like, volunteers, NGOs involved in relief operation always look for real-time information which contains facts instead of prayer, and condolence messages. In more technical terms, these agencies are looking for factual information from Microblog instead of subjective information. In addition to this, the system should generate rank-list of the tweets based upon the worthiness

Table 5.37: sample post for the each class.

	Post text	Class Label
1	ACAN and NCASA Earthquake Relief program yesterday at Kathmandu District: Tents, Food supplies and Sanitary items... http://t.co/aaQu4Pwqib	Factual posts
2	Prey for victims I prey to Allah no more #earthquake plz , don't kill humans whom you created to live the way they love plz be Merciful !"	Non-factual Post

Table 5.38: FIRE IRMiDis Dataset statistics

Particulars	# tweets	Remark
Number of Tweets	50000+	
Labelled Tweets	83	only tweets belong to Factual class
Classes	2	

of facts. We considered this problem as a binary classification problem plus a pure IR Ranking problem. Two classes are labeled as a factual and non-factual post.

5.3.1 FIRE IRMiDis Dataset

Forum for Information Retrieval Evaluation, have introduced Microblog track since 2016 as Information Retrieval from Microblogs during Disasters (IRMiDis). IRMiDis track [12] of FIRE [2] is organized with the objective to extract factual or fact-checkable tweets during the disaster. The IRMiDis dataset [3], sampled from the tweet posted during the Nepal earthquake 2015 [12], is considered for the experiment. The total no of tweets in the dataset is more than 50000. Table 5.38 shows a detail statistics of FIRE IRMiDis Dataset. As we look at the table, There are only 83 tweets is annotated as a factual post. None of the tweets annotated for the non-factual class.

5.3.2 Preparation of Training Data

Due to the unavailability of adequate training data, The first task is to prepare training data to train the deep neural model. We randomly choose 100 tweets from the dataset and labeled as a non-factual tweet and 83 factual tweets present in the dataset labeled as

[2] http://fire.irsi.res.in/fire/2018/home
[3] https://sites.google.com/site/irmidisfire2018/

factual-post. We have trained our CNN model on these training data and tested the model on the remaining 50000 tweets. At this stage, we are not interested in the classification, but we have sorted all the tweets based upon the predicted probability of the factual-post class and selected the top 2000 tweets. We have randomly selected tweets and gave relevance judgment based upon the availability of factual information in the first 1000 tweets and manually extracted 300 tweets and labeled them as non-factual tweets to minimize the false positives. the rest of 1700 tweets are marked as factual tweets. We selected the last 1700 tweets out of 50000 with the least probability of the class factual and labeled them as non-factual tweets. So our training corpus has 1783 factual and 2000 non-factual tweets.

We agree that the deep neural model needs huge data for the training for better prediction. However, The FIRE IRMiDis [12] research task was pure IR ranking task, and the dataset does not have sufficient labeled data. Only 83 tweets marked as a factual class out of 50000+ tweets. Dataset does not contain tweets from the non-factual class. We have anticipated this research task as a text classification problem. We have weakly labeled 2000 tweets as a non-factual and 1783 as a factual using our method discussed previously. Table 5.39 shows our approach produces better results than peers in most of the metrics. Of course, more labeled data may improve the performance, but one has to consider the cost associated with the annotation.

5.3.3 Proposed Approach

We have used word embedding to represent the text instead of bags-of-words. fastText [76] pre-trained vector with 300 dimensions is used to initialize the weight matrix of the embedding layer of the network. We trained the CNN model on this training corpus with 10-fold cross-validation. The model gives validation accuracy around 94%. Finally, we ran the model on the entire corpus and sorted the tweet based upon the predicted probability of the factual class. Essentially, this approach termed as weakly-supervised classification.

5.3.4 Results

As discussed previously, This task involves classification plus ranking task. Table 5.39 shows our system results on IRMiDis dataset [12] along with peers. nDCG overall is the

Table 5.39: Results Comparison with rest of team on FIRE 2018 IRMiDis Dataset.

System	p@100	R@1000	MAP@100	MAP	nDCG @100	nDCG
Our System	0.4	**0.2002**	0.0129	**0.1471**	0.4021	**0.7492**
MIDAS-semiauto	**0.9600**	0.1148	**0.0740**	0.1345	**0.6007**	0.6899
MIDAS-1	0.8800	0.1292	0.0581	0.1329	0.5649	0.6835
FAST_NU_Run2	0.7000	0.0885	0.0396	0.0801	0.5723	0.6676
UEM_DataMining	0.6800	0.1427	0.0378	0.1178	0.5332	0.6396
iitbhu_irlab2	0.3900	0.0447	0.0144	0.0401	0.3272	0.6200

primary metric used for the evaluation. Our system substantially outperforms the rest of the team in the most of the metrics which justifies our claim established on TRAC dataset [52]

5.4 Summary

In this chapter, the multilingual Social media stream is studied with aggression, fact, and offensive content perspective. Exhaustive experiments are performed to benchmark the text representation scheme on machine learning classifiers and deep neural nets. The performance of various work reported in the literature is too far from perfect. for example, [51] reported best weighted F1 score around 0.64 in the multi-class classification problem. while [134] reported macro F1-score around 0.82 in the binary classification and 0.72 for the multi-class classification. Weighted F1-score is used in case of minimal class imbalance across the class labels while in case of highly class-imbalance, macro f1 is the ideal metric to measure the classifier performance.

CHAPTER 6

Impoliteness Visualization

There is a substantial amount of rising in Hate speech-related incidents in online platforms. The enormous data volume makes it hard to capture such cases and either moderate or delete them or consequences by authorities. This chapter presents an approach to visualize online aggression, a special case of hate speech, over social media. We have designed a user interface based on web browser plugin over Facebook and Twitter to visualize the aggressive comments posted by the user on political leaders or celebrities' timeline. This plugin interface might help to the security agency to keep a tab on the social media stream. It also provides citizens with a tool that is typically available only for large enterprises and thus alleviates the technological imbalance. Moreover, the system might be helpful to the research community to prepare weakly labeled training data in a few minutes using comments posted by users on celebrity's Facebook, Twitter timeline. We have reported the results on a newly created dataset of user comments posted during on Facebook/Twitter Live.

Our Contribution In this chapter, we focus on different ways to visualize aggression live on Facebook and Twitter. We have deployed our deep learning model over the internet, which filters the aggressive content from the social media and web browser plugin raised appropriate flag based upon the type of aggression predicted by the model. We have developed three interfaces for the visualization (i) Plugin: which run inside the Chrome browser and raised the flag as soon as it model detect aggressive contents (ii) WEB UI: interactive, standalone interface where user can input the text and check the aggression level predicted by the model. (iii) Personalized Social media dashboards.

6.1 Introduction

The large fraction of Hate speech and other offensive and objectionable content online poses a considerable challenge to societies. Offensive language such as insulting, hurtful, derogatory, or obscene content directed from one person to another person and open for others undermines objective discussions. Such type of language can be more increasingly found on the social web and can lead to the radicalization of debates. The debate on hate speech falls within the broader topic content moderation In 2018, the discussion on content moderation duties for companies within the country, e.g., Germany had led to intense public debate. Public opinion-forming requires rational-critical discourse with an exchange of arguments [44]. Objectionable content can pose a threat to opinion-forming and ultimately to democracy. At the same time, open societies need to find an adequate way to react to such content without imposing rigid censorship regimes. As a consequence, many platforms of social media websites monitor user posts. Content moderation is also a legal requirement in some countries.

This leads to a pressing demand for methods to automatically identify suspicious posts. Online communities, social media enterprises, and technology companies have been investing heavily in technology and processes to identify the offensive language to prevent abusive behavior in social media. Popular social media like Facebook and Twitter have deployed their Hate speech detection tools to track offensive or hate-related content. Facebook announced an AI-based tool called Rosetta4, which enables the company to block text, images, and videos which do not adhere to their hate speech policy. However, such tools are not accessible to the common user or law enforcement agency. On many occasions, such tools have failed to detect hate speech related content. Some examples of highly aggressive tweets are listed in table 6.1.

Table 6.1 shows the example for the comments posted in the social media and written in either an overtly aggressive manner or covertly aggressive/sarcastic way. Sometimes posts are written in Hindi language using Roman script instead of Devanagari script. Therefore, a multilingual text content offers severe challenges to the hate speech detection tool. The amount of hate speech or and offensive content, in general, is a research problem of central attention within the natural language processing community. The research community

Table 6.1: sample post for the each class in TRAC.

	Post text	Class Label	Language
1	shut up you bloody actor, f*ck yourself!! why do you took Modi's biography which is useless for the whole country, better f*ck yourself for acting in this film	OAG	English
2	Hamare modi ji kisi Aladdin se kam hai ke?	CAG	Code-mixed Hindi

is reshaping the Hate Speech problem into a very fine-grained issue, a problem like aggression, abusive, or offensive text. These fine-grained problems are also ambiguous for humans.

The availability of social media on different platforms leads to an exponential rise in Hate speech or offensive content. Volume and velocity of the data is the biggest challenge in the social media stream. In many countries, Hate speech is a punishable offense under the various sections. The job of security agencies become more challenging to keep track of the massive quantity of social streams. In this chapter, we have tried to address issues related to aggression visualization, and deployment on live social media streams like Facebook and Twitter. The following are the objectives of this work.

- Report on a new labeling interface in a web browser to which visualizes aggressive content on Facebook and Twitter.
- Can be proposed interface is effective for social web surveillance?
- To explore the various applications of this interface.

6.2 Purposes of the User Interface

The main objective or purpose behind the development of the tool is to visualize the aggressive or offensive contents posted by the social media user. In this section, we will present the various motivational factors behind the development of aggression visualization interface.

6.2.1 Existing Benchmarks

Facebook and Twitter have deployed their Hate speech detection tools to track aggressive or hate-related content internally. However, on many occasions, such tools are miserably failed to detect such contents. It is nearly impossible for the common user or law-enforcement agencies to access these tools. Multilingual content and massive volumes of the social media stream poses serious challenges to the Hate speech detection tools.

6.2.2 Supporting safety during browsing

Social Media has tremendous power to viral any sensitive information across the billions of people within a short period. Sometime, this sensitive information might include aggressive and potentially harmful content that is disseminated over the web. Social media become the platform where bullying in the physical world is can be termed as cyberbullying. Incidents of trolling and cyberbullying badly affect the normal life of common people. [47] reported that victim such incidents feel extreme physical or mental suffering, and that will lead to the deactivation of their social media account. In rare cases, such incidents forced the victim to commit suicide. Our plugin interfaces as shown in figure 6.1 change the color of the contents from black to red or yellow if the model detects aggressive content in the comments. As soon as comments load in the user timeline, the plugin cautions the user before she read such content. Overtly aggressive (OAG) comments are rendered in red color while covertly/sarcastic comments are rendered in yellow color.

6.2.3 Protect victims of Aggression

Social media gives freedom of speech and anonymity to its users. However, often, social media users exploit this liberty to spread abuses and hate through posts or comments. On many occasions, these user-generated contents are offensive or actively aggressive. Such contents are written in a way that might defame or insult individuals or groups of people without actually using any explicit hate-related or abusive words. Genuine social media users may become victim of such abusive or hateful comments. Our classifier model can classify the sarcastic comments into a covertly aggressive (CAG) category. Plugin change the color of the text of such covertly aggressive comment to yellow, which enables the user to delete content or block the user without reading the content.

6.2.4 Research the best ways to Visualize Aggressive Contents

To the best of our knowledge, we did not come across any interface or tool which enables a user to visualize online hate or textual aggression present in the text. This is our novel approach to empower the common user by visualizing hate over the internet. The user needs to add our plugin inside the browser extension.

6.2.5 Sovereignty for individuals: Knowing the classification for own input

Social media user often gives an opinion on the controversial topic discussed across the thousands of people. Some of the examples of such controversial topics are #Brexit, #Refugeecrisis, #tradewar. The difference between freedom of speech and hate speech is highly debatable in society and needs more constructive discussion. Our plugin and WEB UI will help the user to ensure that their critical comments do not fall into Hate Speech or aggressive content category. In this way, Our plugin or WEB UI helps the user to uphold its sovereignty on social media.

6.3 Approaches for the Classification

Most of the approaches available in the literature for Hate speech detection are based on supervised learning. Initially, we set up classification experiments with popular machine learning classifiers like SVM and Logistic Regression to create the baseline results. After that, experiments are performed using a deep neural model based on CNN, bidirectional LSTM with attention, and recently proposed much-hyped Bidirectional Encoder Representations from Transformers or BERT transformer.

6.3.1 Traditional Classifiers

Logistic regression and support vector machines are popular classifiers used in the various text classification work. From our previous work [80], SVC and Logistic Regression are found to be the best classifiers among the all available classification algorithm. Our feature vector includes Word unigram with TF-IDF weight, Char 5-gram, length of the post, and no of special characters. Table 6.2 6.3, and 6.4 show results on the Logistic Regression and SVC. Results from these classifiers are used to create the baseline results.

6.3.2 Deep Neural Model based on CNN

Deep neural models with distributed word representation are the popular approach for the text classification. We did not perform any Facebook or Twitter-specific text pre-processing or stemming on the text. Since the social media data suffer from the data sparsity problem, classifiers based on the BoW features might not be appropriate as compared to distribution word representation. fastText pre-trained word vectors with dimension 300 are used to initialize the embedding layer. This model trained on 600B tokens of common crawl corpus. The classifier is based on Convolution Neural Network (CNN) includes embedding layer with embed size 300, followed by a one-dimensional convolution layer with 100 filters of height 2 and stride 1 to target bigrams. Besides, a global Max Pooling layer added. The pooling layer fetches the maximum value from the filters and feeds to the dense layer. There are 256 nodes in the hidden layer without any dropout. The model is trained for the 5 epochs.

6.3.3 Deep Neural Model with attention

[10] proposed attention mechanisms in machine translation using neural machine translation. [130] used the attention mechanism for the text classification. Authors claimed that some of the words in the sentence or posts are responsible for determining the correct label of the post. Bag-of-Words(BoW) is the traditional text representation techniques to model the text numerically. TF-IDF and Count-vector are the most popular text representation technique based on BoW. It can capture the important word but the lost context or ignore the sequential structure of the text, while the deep neural model considers word order but failed to give a higher preference to the important word. The attention mechanism addresses both these issues.

Our attention-based model consists of 2 bidirectional layers as a forward layer and backward layer. The model includes embedding layer with embed size 300, convert each word from the post into a fixed-length vector. Short posts are padded with zero values. Subsequent layers include 2 Bidirectional LSTM layer with 128 and 64 memory units respectively, followed by attention layer and dense layer with size 64 units, and an output layer with softmax activations. ReLU activation function is used for the hidden layer activation. Hyperparameters are as follows: Sequence length is fixed at 1073 words; maximum length of posts in the dataset. No of features is equal to half of the total vocabulary size. Models are trained for 10 epoch with batch size 128. Adam optimization algorithm is used to update network weights. The attention mechanism uses an average of the encoded state's output by the LSTM. But all of the encoded states of the LSTM are equally valuable. Thus, we are using a weighted sum of these encoded states to make our prediction. Figure 6.1 shows our model architecture with an attention mechanism.

6.3.4 BERT

BERT stands for Bidirectional Encoder Representations from Transformers [31], achieved significant improvement in various NLP tasks. The biggest challenge of the NLP task is the unavailability of the labeled data. The main objective behind BERT is to address this issue. BERT is based on a language model, trained on a large corpus, considers the previous and next tokens into account when predicting the next word as opposed to traditional language models which only consider the previous n tokens and predict the next one. Therefore, BERT exhibits a contextual representation of word as opposed to Word2Vec, which gives context-free representation. We have fine-tuned pre-trained BERT representations to the text classification task. BERT is based on transformer architecture for encoding the text and performs better in case of small training datasets.

Many variants of the BERT pre-trained model are available. We have used the Uncase BERT base model with 12 layers and 110M parameters to classify the aggression on the English dataset. While for the Hindi dataset, the Multilingual version of BERT is used with a 110 M parameter. The hyperparameters are set as follows: Sequence length is 128. The model is trained for 3 epochs with a batch size of 32, and the learning rate is set to 0.00002.

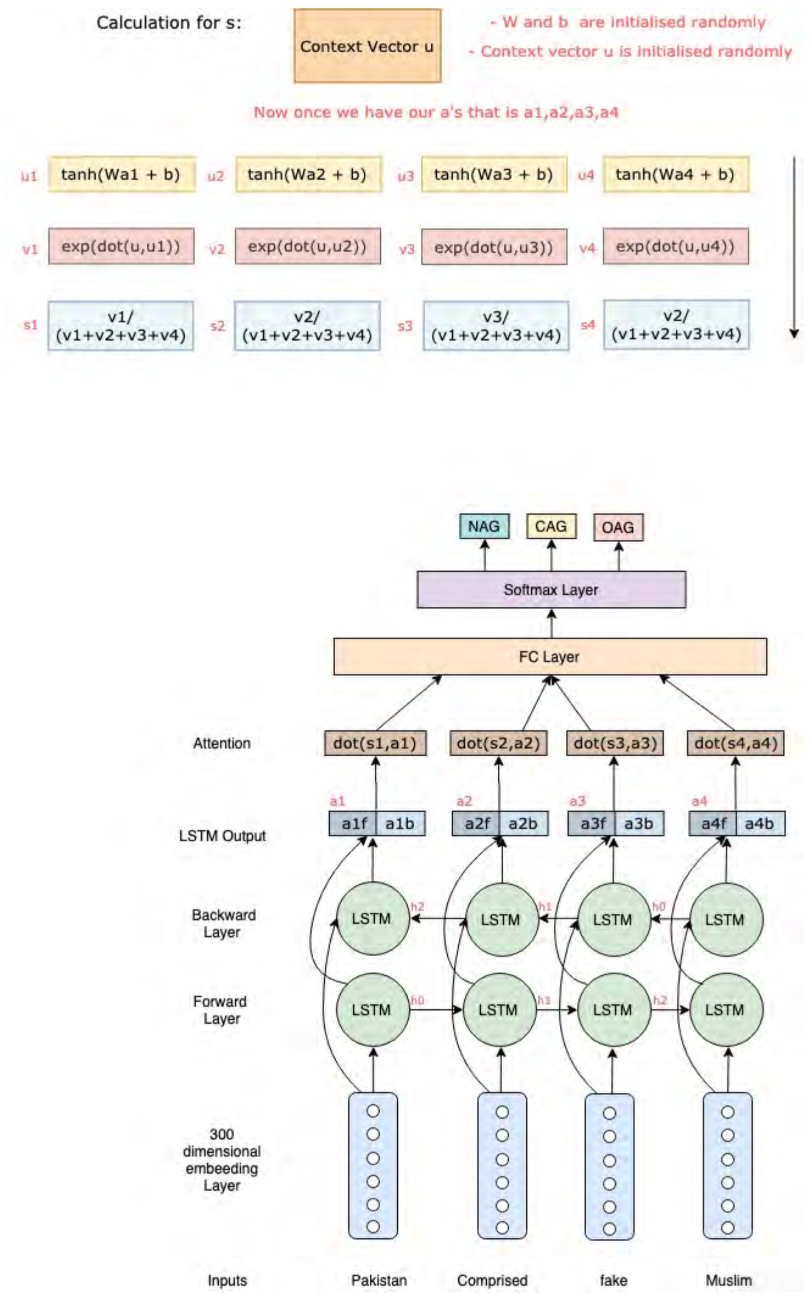


Figure 6.1: Model Architecture with attention mechanism

Table 6.2: Results on TRAC English Dataset

Classifier	Facebook English			Twitter English		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
CNN with fastText	0.6996	0.6091	**0.6407**	0.5664	0.5911	0.5520
LSTM with attention	0.7059	0.5098	0.5476	0.5766	0.0.5839	**0.5782**
BERT	0.7156	0.5840	0.6184	0.5623	0.5807	0.5683
Logistic Regression	0.6811	0.5710	0.6046	0.5232	0.5179	0.4890
SVC	0.6795	0.5524	0.5902	0.4899	0.4924	0.4853

Table 6.3: Results on TRAC Hindi Dataset

Classifier	Facebook Hindi			Twitter Hindi		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
CNN with fastText	0.6261	0.6124	0.6081	0.5206	0.4975	**0.4992**
LSTM with attention	0.5904	0.5651	0.5690	0.4861	0.4623	0.4639
BERT	0.6474	0.6216	**0.6182**	0.5101	0.3852	0.3353
Logistic Regression	0.6607	0.6185	**0.6133**	0.3779	0.3731	0.3723
SVC	0.5998	0.5886	0.5861	0.3942	0.3874	0.3886

6.4 Evaluation and Results

We have evaluated our classifier models on two different social media dataset : (i) Trolling Aggression and Cyberbullying (TRAC) dataset (ii) Comments which are posted on political leaders' Facebook pages during Facebook Live and Twitter broadcast. Dataset is created using our proposed web browser plugin.

Table 6.2 and 6.3 presents results on TRAC dataset [52] on various classifier discussed in the previous section. model based on BERT has produced good weighted F1-score but not better than our deep learning model.

The deep neural model based on the CNN model (Team name DA-LD-Hildesheim, [69]) has produced the best results for TRAC Twitter Hindi Dataset. Based on the results reported in table 6.2 and 6.3, we have deployed the CNN model on a server to classify aggressive posts.

6.4.1 Dataset from Facebook/Twitter Live

Facebook Live and Twitter broadcast are the popular features that enable celebrities to broadcast their speeches delivered during public meetings to their followers across Facebook. Fans, followers, and critics might post their views during the live broadcast. India,

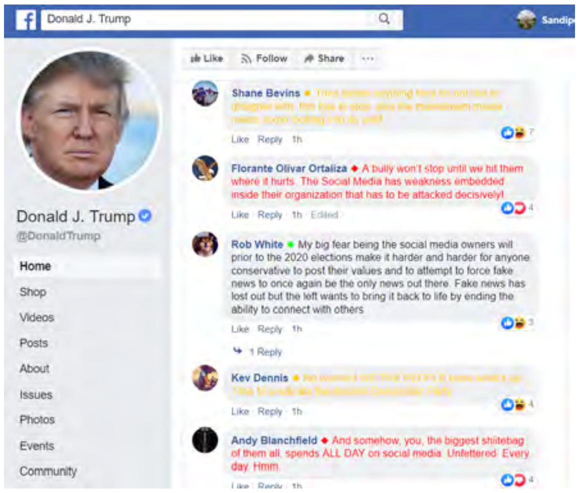
Table 6.4: Results on Facebook/Twitter Live dataset

Dataset	Precision	Recall	weighted F1	macro F1
Facebook	0.7941	0.7344	0.7549	0.64
Twitter	0.8116	.7386	.7593	0.66
Total	0.8024	0.7361	0.7555	0.66

the biggest democracy of the world, had undergone for the general election, from April 2019 to May 2019. We have identified the top four Facebook pages of politicians having the highest followers. We have deployed our Facebook plugin and extracted comments from their timeline. During the annotation process, we found that out of 652 majority comments are non-aggressive. Therefore, we have visited some of the popular world-wide social media profiles such as president Trump, etc. extract the comments posted in their timelines. Table 6.4 shows results on the Facebook/Twitter dataset. The weighted F1-score is around 0.75 while macro f1-score is at 0.66 Out of 652 comments, 464 were annotated as non-aggressive, 100 were from the OAG category, and 88 annotated as CAG. Our model performs better for posts belonging to the NAG category.

6.5 Cyber Watchdog : Interface for Aggression Visualization on Social Media

This chapter focused on the real-time visualization of aggressive social media posts. We have developed a web browser plugin that runs inside the browser and read and send the real-time comments posted on any user's Facebook timeline to the server. The server contains a classifier based on the CNN model and categorize comment into NAG, CAG, and OAG categories and revert the labels to the plugin. Plugin rendered the comment according to the aggression predicted by the model deployed on the server. Figure 6.2 shows the comment rendered by the plugin posted on president Donald trump's Facebook page. Similarly, figure 6.3 shows the Indian Prime minister's Narendra Modi Facebook Page. Overtly aggressive (OAG) comments are rendered in red color, covertly aggressive comments are displayed in yellow color, and non-aggressive comments are flagged by green diamond by the web browser plugin and the color of the comment's text remains unchanged.



Shane Bevins ● They delete anything they do not like or disagree with. this has to stop. also the mainstream media needs some looking into as well!
Like · Reply · 1h 7

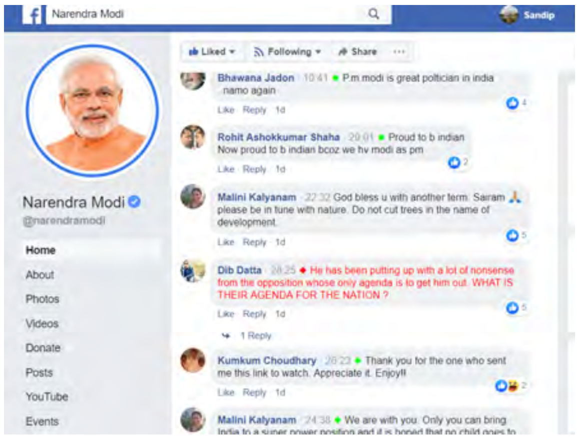
Florante Olivar Ortaliza ◆ A bully won't stop until we hit them where it hurts. The Social Media has weakness embedded inside their organization that has to be attacked decisively!
Like · Reply · 1h · Edited 4

Rob White ● My big fear being the social media owners will prior to the 2020 elections make it harder and harder for anyone conservative to post their values and to attempt to force fake news to once again be the only news out there. Fake news has lost out but the left wants to bring it back to life by ending the ability to connect with others
Like · Reply · 1h 3
1 Reply

Kev Dennis ◆ No worries! WE THE PEOPLE know what's up. Time to eradicate the devilish Democratic Party
Like · Reply · 1h 4

Andy Blanchfield ◆ And somehow, you, the biggest shitebag of them all, spends ALL DAY on social media. Unfettered. Every day. Hmm.
Like · Reply · 1h 4

Figure 6.2: Web Browser Plug-in on President Donald Trump Facebook Page



Bhawana Jadon 10:41 ● P.m modi is great poltician in india .namo again
Like · Reply · 1d 4

Rohit Ashokkumar Shaha 20:01 ● Proud to b indian Now proud to b indian bcoz we hv modi as pm
Like · Reply · 1d 2

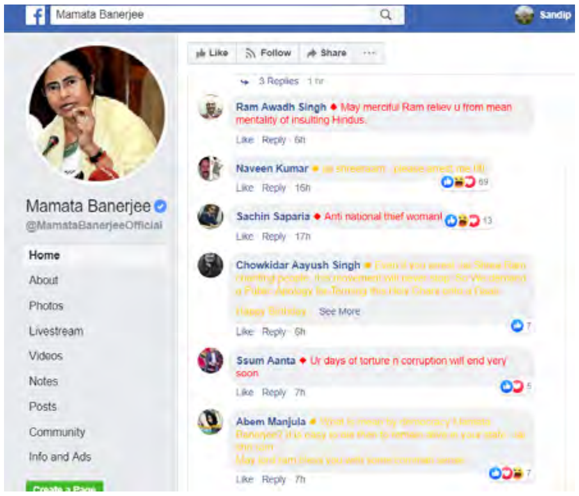
Malini Kalyanam 22:32 God bless u with another term. Sairam 🙏 please be in tune with nature. Do not cut trees in the name of development.
Like · Reply · 1d 5

Dib Datta 20:25 ◆ He has been putting up with a lot of nonsense from the opposition whose only agenda is to get him out. WHAT IS THEIR AGENDA FOR THE NATION ?
Like · Reply · 1d 5
1 Reply

Kumkum Choudhary 20:23 ● Thank you for the one who sent me this link to watch. Appreciate it. Enjoy!!
Like · Reply · 1d 2

Malini Kalyanam 24:38 ● We are with you. Only you can bring India to a super power position and it is hoped that no child goes to

Figure 6.3: Web Browser Plug-in on Narendra Modi Facebook Page



3 Replies · 1 hr

Ram Awadh Singh ◆ May merciful Ram reliev u from mean mentality of insulting Hindus.
Like · Reply · 6h

Naveen Kumar ● jai shreeram . please arrest me !!!
Like · Reply · 16h 69

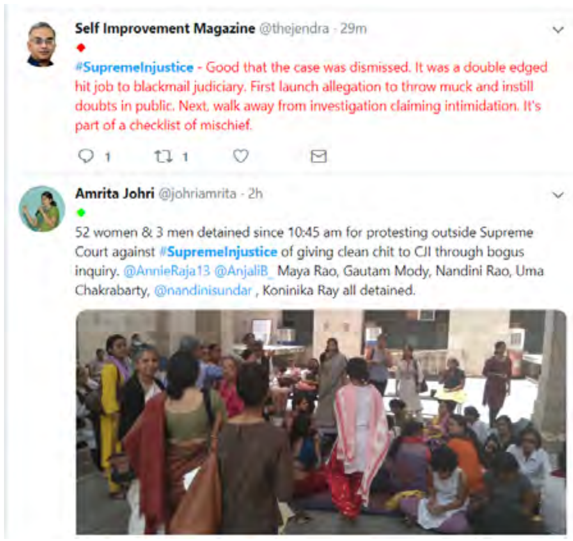
Sachin Saparia ◆ Anti national thief woman! 13
Like · Reply · 17h

Chowkidar Aayush Singh ● Even if you arrest Jai Shree Ram chanting people, this movement will never stop. So We demand a Public Apology for Terming this [illegible]
Happy Birthday · See More
Like · Reply · 6h 7

Ssum Aanta ◆ Ur days of torture n corruption will end very soon
Like · Reply · 7h 5

Abem Manjula ● What is mean by democracy? Mamata Banerjee? it is easy to die than to remain alive in your state. Jai shri ram
May lord ram bless you with some common sense.
Like · Reply · 7h 7

Figure 6.4: Web Browser Plugin on Mamta Banerjee Facebook Page


Self Improvement Magazine @thejendra · 29m

●

#SupremeInjustice - Good that the case was dismissed. It was a double edged hit job to blackmail judiciary. First launch allegation to throw muck and instill doubts in public. Next, walk away from investigation claiming intimidation. It's part of a checklist of mischief.

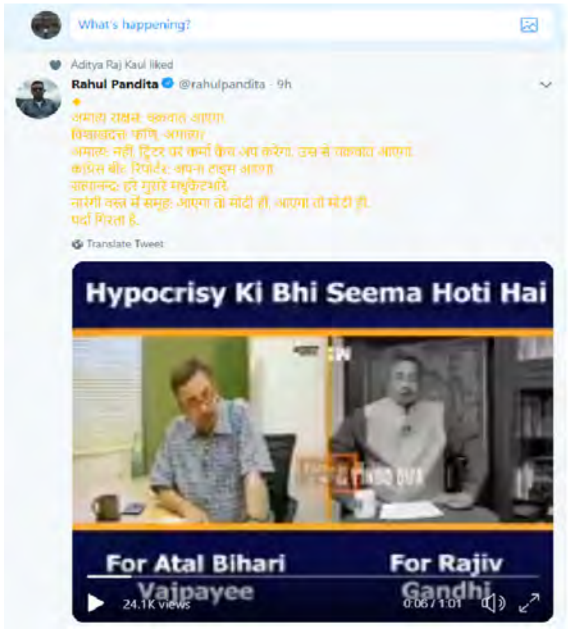
1 1

Amrita Johri @johriamrita · 2h

●

52 women & 3 men detained since 10:45 am for protesting outside Supreme Court against #SupremeInjustice of giving clean chit to CJI through bogus inquiry. @AnnieRaja13 @AnjaliB_ Maya Rao, Gautam Mody, Nandini Rao, Uma Chakrabarty, @nandinisundar , Koninika Ray all detained.

Figure 6.5: Web browser plugin for Twitter



Aditya Raj Kaul liked

Rahul Pandita @rahulpandita · 9h

●

अमात्य राक्षसः चक्रवात आएगा.
विशाखदत्त: फणि, अमात्य?
अमात्यः नहीं, ट्विटर पर कर्मा फीच अप करेगा, उस से चक्रवात आएगा.
कांग्रेस बीट रिपोर्टर: अपना टाइम आएगा.
सत्यानन्द: हरे मुरारे मधुकैटभारे.
नारंगी वस्त्र में समूह: आएगा तो मोदी ही, आएगा तो मोदी ही.
पर्दा गिरता है.

Figure 6.6: Web browser plugin-2 for Twitter

6.5.1 User Interface Architecture

In this sub-section, we will describe our user interface deployed over Facebook, Twitter, and standalone Interactive Web UI. Figure 6.7 shows the detailed architecture of the system and User Interface. We will describe each of the component of the UI in the following sub-sections.

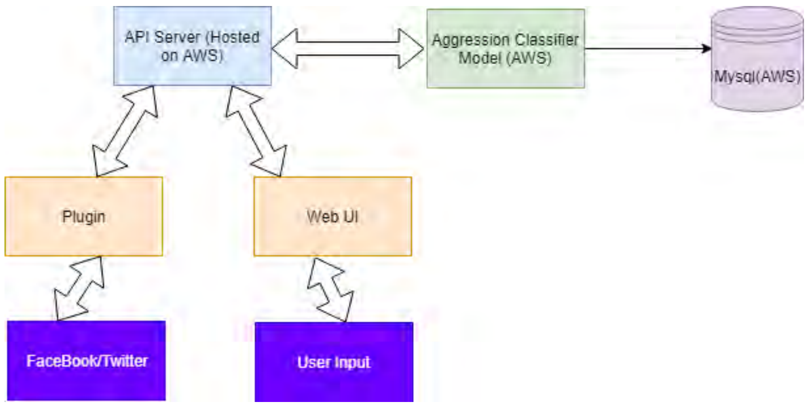


Figure 6.7: User Interface Architecture

Web Browser Plugin

A plugin is a program component that runs over another program like a web browser to add the specific functionality to it. As soon as the Facebook post is loaded into the web browser, a plugin read the comments and send it to the API server. The API server returns the label of the comment predicted by the aggression classifier model. The green diamond is used for NAG Label, the red color is used for OAG Label, and the yellow color is used to annotate comments belong to the CAG class. Figure 6.2 shows a screenshot of the Facebook page rendered by the plugin. Similarly, Figure 6.5 shows a screenshot of the Twitter page rendered by the plugin. We have recorded the screen while the plugin was annotating comments on Facebook. The sample videos are uploaded on Youtube and links for the Facebook plugin is [1] and for the Twitter plugin is [2].

Instruction for the Plugin Download We have published our web browser plugins for Facebook and Twitter for the community. Facebook plugin can be downloaded from this

[1]https://www.youtube.com/watch?v=AbTnDtGEcYw
[2]https://www.youtube.com/watch?v=j3JGtNw4EhA

[3] URL and Twitter plug-in can be downloaded from this [4] URL. Plugins are tested on Google Chrome browser version id 73.0.3683.86 under Windows 10, Ubuntu and MAC operating system. It has been observed that firewall might block the plugin call to the server, and in such cases, the plugin might not work within the browser. After adding the plugin in the browser, users can visit e.g. Twitter link [5] and Facebook link [6] for a quick view. The Twitter link must be accessed without a login or Twitter ID. Plugin classifies Twitter posts and comments made by the user for the respective post. However, to use the Facebook plugin, one must have a Facebook ID. Plugin annotates the comments made by the user for the respective post.

API Server

The API server is responsible for the communication. It handles request and response mechanism across the various clients. The API server aggregate comments received from the multiple clients and pass the text to the classifier model. The API server receives the predicted label from the model and sends it back to the client. The API server is deployed on an Amazon AWS cloud.

Aggression Classifier Model

A deep neural model based on the Convolutional neural network, discussed in the previous section, is deployed in the Amazon AWS cloud. The model classifies social media comment text into the three classes, namely: NAG, OAG, and CAG and returned these labels to the API server. All the comments, along with predicted labels, are stored in the MySQL database to prepare the weakly labeled training dataset.

6.5.2 Web UI

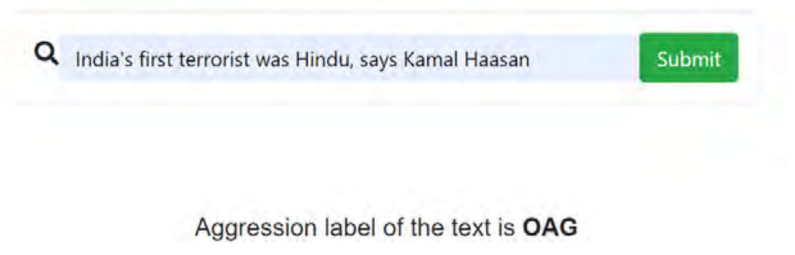
In addition to the web browser plug-in, we developed an interactive, standalone web user interface (UI) where user can input the text, and the predicted label will be displayed on

[3]https://chrome.google.com/webstore/detail/toxic-comments-identifier/ijboffkkobjfdlmkgancjhfdamaiodab?hl=en-GB&gl=IN&authuser=0

[4]https://chrome.google.com/webstore/detail/toxic-comments-identifier/pbaclanlopikmijoedoaendgacalmfpe?hl=en-GB&gl=IN&authuser=0

[5]https://twitter.com/realDonaldTrump

[6]https://www.facebook.com/DonaldTrump/


India's first terrorist was Hindu, says Kamal Haasan

Aggression label of the text is **OAG**

Figure 6.8: Web UI

the web UI. The Web UI can be accessed through the following link [7]. Figure 6.8 shows a snapshot of the WebUI.

6.5.3 Social Media Dashboard for Hate Visualization

We have developed PoCs (proof of concept) for personalized standalone dashboard for Twitter and Instagram. Demo of such PoCs can be found from this YouTube link [8] [9]

6.6 Applications

Facebook and Twitter are very popular across the different age groups of users. They give them a real-time platform to disseminate their opinion about the real-time event. Many celebrities regularly use these social media to connect with their followers. But at the same time, they become a victim of trolling and hate speech. In the following sub-section, we will discuss the potential application of our proposed UI. .

6.6.1 Social Media Surveillance

Most of the countries have enforced strong laws and installed security agencies against the crime-related Hate speech and offensive content. On many occasions, a person who had committed such crimes might be getting away from these security agencies if the victim does not report it to them. Using our plugins, the security establishments can keep a tab on a particular set of users' timelines or trending hashtags to filter the aggressive posts.
Post1: If you have nothing to hide, release the transcripts you corrupt b*sterd!

[7]http://3.16.1.236:8000/
[8]https://www.youtube.com/watch?v=2xwDj9CvWus
[9]https://www.youtube.com/watch?v=HJY0EOluzUk

Post2: Trump is worried about Joe ...that's why he's abusing his power of the Oval Office. Trump has been a lying cheating cr*ok his entire life.

The above tweets are highly abusive and offensive. Twitters' Hate speech detector failed to capture such offensive posts while our plugin quickly classifies them into overtly or covertly aggressive posts. Law Establishment agency uses this label as a piece of prima-facie evidence to launch the initial investigation.

6.6.2 Creation of Weakly Labeled Data

The creation of the labeled data for any classification task is expensive and time-consuming. As we look at figure 6.7, the API server stored each comment along with the aggression label into MySQL database send by the plugins. Our plugin, deployed on Facebook/Twitter, will help to create weakly annotated multilingual training data for any classification task in a crowdsourcing way. Thus, one can save time and money for the data annotation with some errors.

6.6.3 Empowering Citizens

Politicians or celebrities often use Facebook live or Twitter broadcasts to connect with their fans and followers. However, on many occasions, Some of the users post an abusive, offensive, sarcastic, or covertly aggressive comments on their posts. The same can happen to any user who writes about a controversial topic. Most users of social media will not like abusive or offensive content to be visible on their timeline when their profile is accessed. Our proposed plugin helps them to identify such comments by raising the appropriate flag, which enables them to delete or hide such abusive comments from their timeline using personalized Social media dashboards [10] [11]

Using our plugin, ordinary users also have a tool which enables them to predict the offensiveness of text. Typically, such advanced AI tools are only available to large companies like Google or Facebook who develop such sophisticated technology and apply it for content moderation. Our tool gives every citizen the power to use the same kind of

[10]https://www.youtube.com/watch?v=2xwDj9CvWus
[11]https://www.youtube.com/watch?v=HJY0EOluzUk

technology. They can use it to check text which they want to enter into a social media system to predict the probability of it being criticized and moderated by the companies. Also, further tools can be developed based on the technology described, e.g., a system calculating an offensiveness profile for a particular user. Thus, the plugin alleviates technological imbalance and give citizens more sovereignty regarding the digital world.

6.7 Summary

In this chapter, we have presented user interfaces based on web browser plugins to visualize aggressive content expressed either implicitly or explicitly. The plugins are developed for the two most popular media: Facebook and Twitter. The work shows which kind of problems are moving into the center of attention for research in machine learning and natural language processing.

CHAPTER 7

Conclusions and Future works

This thesis attempts to explore social media streams such as Facebook and Twitter from summarization and impoliteness perspectives. Summarizing Microblog is a non-trivial task. We have considered the 3 cases for the Microblog summarization :(i) Email Digest (ii)Real-time push notification (iii)Summarizing Microblog during the disaster event. Tweet selection and novelty across the tweets are the primary tasks. We have used term interest profile to encapsulate the user information need or topic of its interest. We have done a comprehensive analysis of our summarization system. User information need or information requirement becomes very precise than ever. The restriction on the length of the tweet is the most severe challenge. We have used various similarity measures from keyword matching to the probabilistic language model. However, none of the measures proved to be effective. The absent of a context in the tweet is the primary reason for the under-performance.

In the second part of the thesis, we examine social media streams from the various form of impoliteness such as hate, aggression, offensive content. We have focused on text representation of the social media texts. Range of text representation schemes from Bow, word embedding, sentence embedding, pre-trained word embeddings, to the contextual pre-trained language models studied and benchmarked on various classifiers.

In the last part of the thesis, we tried to explore various ways to visualize impoliteness live on Facebook and Twitter. We have been attempting to build an interface which monitors social media stream and filter the aggressive or offensive content.

7.1 Microblog Summarization

In the first part of the thesis, We have developed a Twitter-based summarization system to track the event for the longer duration(e.g.10 days). The system is evaluated on TREC RTS 2016 and 2017 dataset. Since the primary evaluation metrics such as nDCG-1@10 and nDCG-0@10, EG-1, EG-0 are biased towards precision rather than recall, query expansion may include non-relevant tweets in summary. It may improve recall but precision decreases substantially and produced an adverse effect on the overall results. The language model with Dirichlet smoothing is the better similarity measure for Microblog Retrieval. We have empirically identified the optimal value of the smoothing parameter $\mu = 1000$. Our approach to the Microblog summarization problem is the fusion of supervised and unsupervised methods. [68] have focused on detecting the silent day for the interest profile. On the contrary to their approach, we have built prediction models that predict silent day thresholds. Furthermore, we have also used the relevance threshold to determine the tweet to be part of the candidate list. Our techniques to estimate silent day threshold T_s and relevance threshold T_r perform reasonably well with 95% accuracy on average. However, if we compare results computed using estimated thresholds with results calculated using thresholds determined via grid search, results with estimated thresholds are substantially lower than results with grid search thresholds. Nevertheless, our results with estimated thresholds are better than the top team of TREC RTS 2016 [64] and substantially outperforms one of the metric nDCG-0 and median of all the metrics of TREC RTS 2017 [63].

We have also presented a comprehensive failure analysis and discuss many issues that adversely affect the performance of the summarization system. Similarity score (between interest profile and tweets) should include some of the latest features like sentiment while ranking the tweet. NE normalization, NE linking might improve the performance. In the future, we will try to address these issues.

7.2 Impoliteness Detection

In the second part of the thesis, various form of impoliteness such as Online Hate speech, aggression, offensive content is explored on the Microblog. These problems are fall under the fundamental problem of text classification.

From the results on the TRAC dataset [52], [12], we found that deep Neural model with fastText pre-trained word embedding is the better choice than traditional classifier and transfer learning model. CNN model is better than LSTM and Bidirectional LSTM. However, on SemEval 2019 Offeneval dataset [132], Contextual pre-trained language model such as BERT has performed better than our deep learning model. Distribution of class labels and the size of the training dataset is a critical issue for the classification system. BERT handle these issues better than deep neural models and traditional classifiers. The attention mechanism does not make any improvement in the result. In fact, performance is degraded compared to the deep neural model results.

Using deep learning models, there is a great potential to solve some of these problems, yet still, the performance is far from perfect. Model transfer between problems and the application of derived knowledge are areas directions for future work. In the future, we want to work on sentence embedding or post-embedding technique. We will investigate the underperformance of BERT, ELMo, and ULMFit on the TRAC dataset.

7.3 Impoliteness Visualization

In the last part of the thesis, we focus on the visualization part. We have developed web browser plugins to visualize aggressive content expressed either implicitly or explicitly on Facebook and Twitter. In the future, we want to develop a separate personalized interface that aggregates all feed from the different social media for a particular celebrity. Our proposed interface will have the following visualization features.: Sentiment visualization, Hate visualization, Automatic identification, and blocking of a hatemonger.

References

[1] S. Agarwal and A. Sureka. Characterizing linguistic attributes for automatic classification of intent based racist/radicalized posts on tumblr micro-blogging website. *arXiv preprint arXiv:1701.04931*, 2017.

[2] C. Akimushkin, D. R. Amancio, and O. N. Oliveira Jr. Text authorship identified using the dynamics of word co-occurrence networks. *PloS one*, 12(1):e0170527, 2017.

[3] D. R. Amancio. Comparing the topological properties of real and artificially generated scientific manuscripts. *Scientometrics*, 105(3):1763–1779, 2015.

[4] R. Angelova and G. Weikum. Graph-based text classification: learn from your neighbors. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 485–492. ACM, 2006.

[5] S. Arora, Y. Liang, and T. Ma. A simple but tough-to-beat baseline for sentence embeddings. 2017.

[6] S. T. Aroyehun and A. Gelbukh. Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 90–97, 2018.

[7] I. Arroyo-Fernández, D. Forest, J.-M. Torres-Moreno, M. Carrasco-Ruiz, T. Legeleux, and K. Joannette. Cyberbullying detection task: the ebsi-lia-unam system (elu) at coling'18 trac-1. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 140–149, 2018.

[8] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760. International World Wide Web Conferences Steering Committee, 2017.

[9] M. Bagdouri and D. W. Oard. Clip at trec 2015: Microblog and liveqa. In *TREC*, 2015.

[10] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[11] V. Basile, C. Bosco, E. Fersini, D. Nozza, V. Patti, F. Rangel, P. Rosso, and M. Sanguinetti. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019). Association for Computational Linguistics", location="Minneapolis, Minnesota*, 2019.

[12] M. Basu, S. Ghosh, and K. Ghosh. Overview of the fire 2018 track: Information retrieval from microblogs during disasters (irmidis). In *Proceedings of the 10th annual meeting of the Forum for Information Retrieval Evaluation*, pages 1–5. ACM, 2018.

[13] M. Basu, A. Roy, K. Ghosh, S. Bandyopadhyay, and S. Ghosh. Microblog retrieval in a disaster situation: A new test collection for evaluation. In *SMERP@ ECIR*, pages 22–31, 2017.

[14] C. Baziotis, N. Pelekis, and C. Doulkeridis. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, 2017.

[15] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[16] P. Burnap and M. L. Williams. Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy & Internet*, 7(2):223–242, 2015.

[17] J. S. Canós. Misogyny identification through svm at ibereval 2018. In *3rd Workshop on Evaluation of Human Language Technologies for Iberian Languages, IberEval 2018*, volume 2150. CEUR-WS, 2018.

[18] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[19] D. Chakrabarti and K. Punera. Event summarization using tweets. *ICWSM*, 11:66–73, 2011.

[20] A. Chellal and M. Boughanem. Optimization framework model for retrospective tweet summarization. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, pages 704–711. ACM, 2018.

[21] A. Chellal, M. Boughanem, and B. Dousset. Multi-criterion real time tweet summarization based upon adaptive threshold. In *Web Intelligence (WI), 2016 IEEE/WIC/ACM International Conference on*, pages 264–271. IEEE, 2016.

[22] A. Chellal, M. Boughanem, G. Hubert, J. G. Moreno, K. Pinel-Sauvagnat, and Y. Pitarch. Irit at trec real-time summarization 2017. In *TREC*, 2017.

[23] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–394, 1999.

[24] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.

[25] M. Conover, J. Ratkiewicz, M. R. Francisco, B. Gonçalves, F. Menczer, and A. Flammini. Political polarization on twitter. *Icwsm*, 133:89–96, 2011.

[26] M. D. Conover, B. Gonçalves, J. Ratkiewicz, A. Flammini, and F. Menczer. Predicting the political alignment of twitter users. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*, pages 192–199. IEEE, 2011.

[27] J. Culpeper. *Impoliteness: Using language to cause offence*, volume 28. Cambridge University Press, 2011.

[28] M. Dadvar, D. Trieschnigg, R. Ordelman, and F. de Jong. Improving cyberbullying detection with user context. In *Advances in Information Retrieval*, pages 693–696. Springer, 2013.

[29] T. Davidson, D. Warmsley, M. Macy, and I. Weber. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of ICWSM*, 2017.

[30] J. Deriu, M. Gonzenbach, F. Uzdilli, A. Lucchi, V. D. Luca, and M. Jaggi. Swisscheese at semeval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision. In *Proceedings of the 10th international workshop on semantic evaluation*, pages 1124–1128, 2016.

[31] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[32] P. Dewan and P. Kumaraguru. Facebook inspector (fbi): Towards automatic real-time detection of malicious content on facebook. *Social Network Analysis and Mining*, 7(1):15, 2017.

[33] K. Dinakar, R. Reichart, and H. Lieberman. Modeling the detection of textual cyberbullying. In *The Social Mobile Web*, pages 11–17, 2011.

[34] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati. Hate speech detection with comment embeddings. In *Proceedings of the 24th International Conference on World Wide Web Companion*, pages 29–30. International World Wide Web Conferences Steering Committee, 2015.

[35] E. Fersini, P. Rosso, and M. Anzovino. Overview of the task on automatic misogyny identification at ibereval 2018. 2018.

[36] D. Fišer, T. Erjavec, and N. Ljubešić. Legal Framework, Dataset and Annotation Schema for Socially Unacceptable On-line Discourse Practices in Slovene. In *Proceedings of the Workshop Workshop on Abusive Language Online (ALW)*, Vancouver, Canada, 2017.

[37] P. Fortuna and S. Nunes. A Survey on Automatic Detection of Hate Speech in Text. *ACM Computing Surveys (CSUR)*, 51(4):85, 2018.

[38] S. Frenda, G. Bilal, et al. Exploration of misogyny in spanish and english tweets. In *Third Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018)*, volume 2150, pages 260–267. Ceur Workshop Proceedings, 2018.

[39] B. Gambäck and U. K. Sikdar. Using Convolutional Neural Networks to Classify Hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90, 2017.

[40] Z. Gao and J. Wolohan. Fast nlp-based pattern matching in real time tweet recommendation. In *TREC*, 2017.

[41] N. D. Gitari, Z. Zuping, H. Damien, and J. Long. A lexicon-based approach for hate speech detection. *International Journal of Multimedia and Ubiquitous Engineering*, 10(4):215–230, 2015.

[42] G. Gonçalves, F. Martins, and J. Magalhães. Novasearch at trec 2017 real-time summarization track. In *TREC*, 2017.

[43] G. Gonçalves, F. Martins, and J. Magalhães. Analysis of subtopic discovery algorithms for real-time information summarization. In *Companion of the The Web Conference 2018 on The Web Conference 2018*, pages 1855–1856. International World Wide Web Conferences Steering Committee, 2018.

[44] J. Habermas et al. Vorstudien und ergänzungen zur theorie des kommunikativen handelns. 1984.

[45] Z. Han, S. Li, L. Kong, L. Tian, and H. Qi. Hljit at trec 2017 real-time summarization. In *TREC*, 2017.

[46] Z. Han, S. Li, L. Kong, L. Tian, and H. Qi. Hljit at trec 2017 real-time summarization. In *Proceedings of the 26th Text REtrieval Conference*, 2017.

[47] S. Hinduja and J. W. Patchin. Bullying, cyberbullying, and suicide. *Archives of suicide research*, 14(3):206–221, 2010.

[48] J. Hltcoe. Semeval-2013 task 2: Sentiment analysis in twitter. *Atlanta, Georgia, USA*, 312, 2013.

[49] J. Howard and S. Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

[50] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015.

[51] R. Kumar, A. K. Ojha, S. Malmasi, and M. Zampieri. Benchmarking Aggression Identification in Social Media. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbulling (TRAC)*, Santa Fe, USA, 2018.

[52] R. Kumar, A. N. Reganti, A. Bhatia, and T. Maheshwari. Aggression-annotated Corpus of Hindi-English Code-mixed Data. In *Proceedings of the 11th Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan, 2018.

[53] N. Kurniasih, L. A. Abdillah, I. K. Sudarsana, I. Yogantara, I. Astawa, R. F. Nanuru, A. Miagina, J. O. Sabarua, M. Jamil, J. Tandisalla, et al. Prototype application hate speech detection website using string matching and searching algorithm. *International Journal of Engineering & Technology*, 7(2.5):62–64, 2018.

[54] I. Kwok and Y. Wang. Locate the hate: Detecting Tweets Against Blacks. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[55] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196, 2014.

[56] K. Lee, A. Qadir, Y. Ling, J. Liu, S. A. Hasan, V. V. Datla, A. Prakash, and O. Farri. Recognizing tweet relevance with profile-specific and profile-independent supervised models. In *TREC*, 2017.

[57] H. Li. Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies*, 4(1):1–113, 2011.

[58] H. Li. A short introduction to learning to rank. *IEICE TRANSACTIONS on Information and Systems*, 94(10):1854–1862, 2011.

[59] H. Li, D. Tan, and W. Luo. Polyu at trec 2016 real-time summarization. In *TREC*, 2016.

[60] Q. Li. A cross-cultural comparison of adolescents' experience related to cyberbullying. *Educational Research*, 50(3):223–234, 2008.

[61] Y. H. Li and A. K. Jain. Classification of text documents. *The Computer Journal*, 41(8):537–546, 1998.

[62] J. Lin, M.Efron, G. Y.Wang, and R. andT.Sakai. "overview of the trec 2015 microblog track," in text retrieval conference, trec, gaithersburg, usa, november 17-20, 2015. In *Proceedings of the 24th Text REtrieval Conference, TREC*, volume 15, 2015.

[63] J. Lin, S. Mohammed, R. Sequiera, L. Tan, N. Ghelani, M. Abualsaud, R. McCreadie, D. Milajevs, and E. Voorhees. Overview of the trec 2017 real-time summarization track (notebook draft). In *Pre-Proceedings of the 26th Text REtrieval Conference, TREC*, 2017.

[64] J. Lin, A. Roegiest, L. Tan, M. Richard, E. Voorhees, and F. Diaz. Overview of the trec 2016 real-time summarization track. In *Proceedings of the 25th Text REtrieval Conference, TREC*, volume 16, 2016.

[65] B. Liu et al. Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2(2010):627–666, 2010.

[66] S. Liu and T. Forss. New classification models for detecting hate and violence web content. In *2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K)*, volume 1, pages 487–495. IEEE, 2015.

[67] T.-Y. Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009.

[68] K. Lu and H. Fang. Silent day detection on microblog data. In *International Conference on Applications of Natural Language to Information Systems*, pages 443–455. Springer, 2018.

[69] P. Majumder, T. Mandl, et al. Filtering aggression from the multilingual social media feed. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 199–207, 2018.

[70] S. Malmasi and M. Zampieri. Detecting Hate Speech in Social Media. In *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP)*, pages 467–472, 2017.

[71] S. Malmasi and M. Zampieri. Challenges in Discriminating Profanity from Hate Speech. *Journal of Experimental & Theoretical Artificial Intelligence*, 30:1–16, 2018.

[72] D. Maynard and A. Funk. Automatic detection of political opinions in tweets. In *Extended Semantic Web Conference*, pages 88–99. Springer, 2011.

[73] P. Meladianos, C. Xypolopoulos, G. Nikolentzos, and M. Vazirgiannis. An optimization approach for sub-event detection and summarization in twitter. In *European Conference on Information Retrieval*, pages 481–493. Springer, 2018.

[74] Q. Meng, K. Wang, and Z. Yang. Bjut at trec 2017: Real-time summarization track. In *TREC*, 2017.

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

[76] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

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

[78] S. Modha. Microblog processing: A study. In *FIRE (Working Notes)*, pages 164–167, 2017.

[79] S. Modha, K. Agrawal, D. Verma, P. Majumder, and C. Mandalia. Daiict at trec rts 2016: Live push notification and email digest. In *TREC*, 2016.

[80] S. Modha and P. Majumder. An empirical evaluation of text representation schemes on multilingual social web to filter the textual aggression. *arXiv preprint arXiv:1904.08770*, 2019.

[81] S. Modha, C. Mandalia, K. Agrawal, D. Verma, and P. Majumder. Real time information extraction from microblog. In *FIRE (Working Notes)*, pages 79–80, 2016.

[82] S. Modha, R. Singla, and C. Soni. Summarizing disaster related event from microblog. In *SMERP@ ECIR*, pages 109–115, 2017.

[83] J. P. Montani. Tuwienkbs at germeval 2018: German abusive tweet detection. In *14th Conference on Natural Language Processing KONVENS 2018*, page 45, 2018.

[84] B. Moulahi, L. B. Jabeur, R. Abbes, and L. Tamine. Summarizing tweet in real-time by filtering quality, relevant and non redundant tweets. In *TREC*, 2017.

[85] H. Mubarak, K. Darwish, and W. Magdy. Abusive language detection on arabic social media. In *Proceedings of the First Workshop on Abusive Language Online*, pages 52–56, 2017.

[86] H. Mubarak, D. Kareem, and M. Walid. Abusive Language Detection on Arabic Social Media. In *Proceedings of the Workshop on Abusive Language Online (ALW)*, Vancouver, Canada, 2017.

[87] P. Nakov, A. Ritter, S. Rosenthal, F. Sebastiani, and V. Stoyanov. Semeval-2016 task 4: Sentiment analysis in twitter. In *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)*, pages 1–18, 2016.

[88] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang. Abusive Language Detection in Online User Content. In *Proceedings of the 25th International Conference on World Wide Web*, pages 145–153. International World Wide Web Conferences Steering Committee, 2016.

[89] J. T. Nockleby. Hate speech. *Encyclopedia of the American constitution*, 3(2):1277–1279, 2000.

[90] E. W. Pamungkas, A. T. Cignarella, V. Basile, V. Patti, et al. 14-exlab@ unito for ami at ibereval2018: Exploiting lexical knowledge for detecting misogyny in english and spanish tweets. In *3rd Workshop on Evaluation of Human Language Technologies for Iberian Languages, IberEval 2018*, volume 2150, pages 234–241. CEUR-WS, 2018.

[91] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[92] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[93] X. Qian, J. Lin, and A. Roegiest. Interleaved evaluation for retrospective summarization and prospective notification on document streams. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 175–184. ACM, 2016.

[94] D. R. Radev, E. Hovy, and K. McKeown. Introduction to the special issue on summarization. *Computational linguistics*, 28(4):399–408, 2002.

[95] A. H. Razavi, D. Inkpen, S. Uritsky, and S. Matwin. Offensive language detection using multi-level classification. In *Canadian Conference on Artificial Intelligence*, pages 16–27. Springer, 2010.

[96] A. Roegiest, L. Tan, and J. Lin. Online in-situ interleaved evaluation of real-time push notification systems. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 415–424. ACM, 2017.

[97] S. Rosenthal, N. Farra, and P. Nakov. Semeval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, 2017.

[98] S. Rosenthal, P. Nakov, S. Kiritchenko, S. Mohammad, A. Ritter, and V. Stoyanov. Semeval-2015 task 10: Sentiment analysis in twitter. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 451–463, 2015.

[99] B. Ross, M. Rist, G. Carbonell, B. Cabrera, N. Kurowsky, and M. Wojatzki. Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis. In *Proceedings of the Workshop on Natural Language Processing for Computer-Mediated Communication (NLP4CMC)*, Bochum, Germany, 2016.

[100] B. Ross, M. Rist, G. Carbonell, B. Cabrera, N. Kurowsky, and M. Wojatzki. Measuring the reliability of hate speech annotations: The case of the european refugee crisis. *arXiv preprint arXiv:1701.08118*, 2017.

[101] K. Rother, M. Allee, and A. Rettberg. Ulmfit at germeval-2018: A deep neural language model for the classification of hate speech in german tweets. In *14th Conference on Natural Language Processing KONVENS 2018*, page 113, 2018.

[102] A. Rücklé, S. Eger, M. Peyrard, and I. Gurevych. Concatenated power mean word embeddings as universal cross-lingual sentence representations. *arXiv preprint arXiv:1803.01400*, 2018.

[103] K. Rudra, S. Ghosh, N. Ganguly, P. Goyal, and S. Ghosh. Extracting situational information from microblogs during disaster events: a classification-summarization approach. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 583–592. ACM, 2015.

[104] K. Sabhnani and B. Carterette. University of delaware at trec 2017 real-time summarization track. In *Proceedings of the 26th Text REtrieval Conference*, 2017.

[105] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM, 2010.

[106] N. S. Samghabadi, D. Mave, S. Kar, and T. Solorio. Ritual-uh at trac 2018 shared task: Aggression identification. *arXiv preprint arXiv:1807.11712*, 2018.

[107] A. Schmidt and M. Wiegand. A Survey on Hate Speech Detection Using Natural Language Processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media. Association for Computational Linguistics*, pages 1–10, Valencia, Spain, 2017.

[108] A. Severyn and A. Moschitti. Unitn: Training deep convolutional neural network for twitter sentiment classification. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 464–469, 2015.

[109] B. Sharifi, M.-A. Hutton, and J. K. Kalita. Experiments in microblog summarization. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 49–56. IEEE, 2010.

[110] L. Silva, M. Mondal, D. Correa, F. Benevenuto, and I. Weber. Analyzing the targets of hate in online social media. In *Tenth International AAAI Conference on Web and Social Media*, 2016.

[111] R. Singla, S. Modha, P. Majumder, and C. Mandalia. Information extraction from microblog for disaster related event. In *SMERP@ ECIR*, pages 85–92, 2017.

[112] E. Spertus. Smokey: Automatic recognition of hostile messages. In *Aaai/iaai*, pages 1058–1065, 1997.

[113] H.-P. Su, C.-J. Huang, H.-T. Chang, and C.-J. Lin. Rephrasing Profanity in Chinese Text. In *Proceedings of the Workshop Workshop on Abusive Language Online (ALW)*, Vancouver, Canada, 2017.

[114] R. Suwaileh, M. Hasanain, and T. Elsayed. Light-weight, conservative, yet effective: Scalable real-time tweet summarization. In *TREC*, 2016.

[115] H. Tan, Z. Lu, and W. Li. Neural network based reinforcement learning for real-time pushing on text stream. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 913–916. ACM, 2017.

[116] L. Tan, A. Roegiest, C. L. Clarke, and J. Lin. Simple dynamic emission strategies for microblog filtering. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 1009–1012. ACM, 2016.

[117] L. Tan, A. Roegiest, J. Lin, and C. L. Clarke. An exploration of evaluation metrics for mobile push notifications. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 741–744. ACM, 2016.

[118] S. Tulkens, L. Hilte, E. Lodewyckx, B. Verhoeven, and W. Daelemans. A Dictionary-based Approach to Racism Detection in Dutch Social Media. In *Proceedings of the Workshop Text Analytics for Cybersecurity and Online Safety (TA-COS)*, Portoroz, Slovenia, 2016.

[119] A. Tumasjan, T. O. Sprenger, P. G. Sandner, and I. M. Welpe. Predicting elections with twitter: What 140 characters reveal about political sentiment. *Icwsm*, 10(1):178–185, 2010.

[120] H. Vandebosch and K. Van Cleemput. Cyberbullying among youngsters: Profiles of bullies and victims. *New media & society*, 11(8):1349–1371, 2009.

[121] D. von Grünigen, F. Benites, P. von Däniken, M. Cieliebak, R. Grubenmann, and A. SpinningBytes. spmmmp at germeval 2018 shared task: Classification of of-

fensive content in tweets using convolutional neural networks and gated recurrent units. In *14th Conference on Natural Language Processing KONVENS 2018*, page 130, 2018.

[122] X. Wang, P. Fan, L. Cheng, G. Xing, Z. Yu, and X. Cheng. Ictnet at trec 2017 real-time summarization track. In *TREC*, 2017.

[123] W. Warner and J. Hirschberg. Detecting hate speech on the world wide web. In *Proceedings of the second workshop on language in social media*, pages 19–26. Association for Computational Linguistics, 2012.

[124] Z. Waseem, T. Davidson, D. Warmsley, and I. Weber. Understanding Abuse: A Typology of Abusive Language Detection Subtasks. In *Proceedings of the First Workshop on Abusive Langauge Online*, 2017.

[125] Z. Waseem and D. Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93, 2016.

[126] G. Wiedemann, E. Ruppert, R. Jindal, and C. Biemann. Transfer learning from lda to bilstm-cnn for offensive language detection in twitter. *arXiv preprint arXiv:1811.02906*, 2018.

[127] M. Wiegand, M. Siegel, and J. Ruppenhofer. Overview of the GermEval 2018 Shared Task on the Identification of Offensive Language. In *Proceedings of GermEval*, 2018.

[128] J.-M. Xu, K.-S. Jun, X. Zhu, and A. Bellmore. Learning from bullying traces in social media. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 656–666. Association for Computational Linguistics, 2012.

[129] M. Yang, W. Tu, Q. Qu, K. Lei, X. Chen, J. Zhu, and Y. Shen. Mares: multitask learning algorithm for web-scale real-time event summarization. *World Wide Web*, pages 1–17, 2018.

[130] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016.

[131] L. Yao, C. Lv, F. Fan, J. Yang, and D. Zhao. Pkuicst at trec 2016 real-time summarization track: Push notifications and email digest. In *TREC*, 2016.

[132] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar. Predicting the Type and Target of Offensive Posts in Social Media. In *Proceedings of NAACL*, 2019.

[133] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar. Predicting the Type and Target of Offensive Posts in Social Media. In *Proceedings of NAACL*, 2019.

[134] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *arXiv preprint arXiv:1903.08983*, 2019.

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

[136] Z. Zhang, D. Robinson, and J. Tepper. Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network. In *Lecture Notes in Computer Science*. Springer Verlag, 2018.

[137] Q. Zhao, P. Mitra, and B. Chen. Temporal and information flow based event detection from social text streams. In *AAAI*, volume 7, pages 1501–1506, 2007.

[138] Y. Zhou and W. B. Croft. Query performance prediction in web search environments. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 543–550. ACM, 2007.

CHAPTER A

Microblog Summarization : Dataset and Sample Summary

We will give meta information about the dataset and Sample summary generated by our system.

A.1 Availability of Dataset

Datasets which are used in the experiments are publicly available from the following link.

- For TREC MB 2015 : https://trec.nist.gov/data/microblog2015.html
- For TREC RTS 2016 :https://trec.nist.gov/data/rts2016.html
- For TREC RTS 2017 :https://trec.nist.gov/data/rts2017.html

A.2 Sample Summary

Table A.1 show the sample summary generated by our system.

Table A.1: Sample Summary :RTS 120 - Philippines Marawi ISIS

datw	Tweet	Relevant or Not
20170729	RT : Philippine: Western Media is Distorting Reality, People and Army Unite to Battle ISIS https://t.co/II9nYNZ2lB	Relevant
20170729	RT #MannyPacquiao visits troops in Philippine warzone of #Marawi https://t.co/lyZ87ZH5V1 https://t.co/3QHO4IBddW	Non-relevant
20170729	RT : Marawi Takeover: A Wake-Up Call to ISIS Presence in Pacific https://t.co/ey0Qlw2n13	Relevant
20170729	Bid to tackle ISIS menace in Philippines - Sky News Australia #Islam #Muslims	Relevant
20170729	RT : Live Coverage: Salute to all Police; Soldiers in #Philippines finish them all ISIS Terrorist https://t.co/RQkVQJlAKY	Non-relevant
20170729	Commentary: The challenge of rebuilding Marawi - https://t.co/QpIo4ErdYP	Relevant
20170730	RT : Manny Pacquiao Rallies Philippine Troops in Battle Against ISIS https://t.co/idiP81187H #boxing https://t.co/IUTVwbOUo8	Non-Relevant
20170730	ISIS praises jihadist attacks in Philippines, Iran https://t.co/5Y2Xj7823G"	Non-Relevant
20170730	Philippines: Only 60 militants fighting in Marawi siege https://t.co/HUCmTp6mvC #Iran	Relevant
20170730	Soldiers also battling diseases in Marawi - SOLDIERS fighting members of the Maute Group in Marawi City are als... https://t.co/NYqlaXTA9c	Relevant
20170730	US donates more counterterror weapons for Marawi troops https://t.co/zLhFoc2ifd	Relevant

CHAPTER B
HASOC

The identification of Hate Speech in Social Media has received much attention in research recently. There is a particular demand for research for languages other than English. The first edition of the HASOC (Hate Speech and Offensive Content Identification in Indo-European Languages) shared task creates resources for Hate Speech Identification in Hindi, German, and English. Three datasets were developed from Twitter, and Facebook and made available. HASOC intends to stimulate research and development for Hate Speech classification for different languages.

B.1 Introduction

There has been significant work in several languages in particular for English. However, there is a lack of research on this recent and relevant topic for most other languages. This track intends to develop data and evaluation resources for several languages. The objectives are to stimulate research for these languages and to find out the quality of hate speech detection technology in other languages. In the long run, the track aims at supporting researchers to develop robust technology which can cope with multilingual data and to develop transfer learning approaches which can exploit learning data across languages. For future editions, we envision the integration of further languages.

We have offered a track: Hate Speech and Offensive Content Identification in Indo-European Languages (HASOC)[1] in Forum for Information Retrieval Evaluation FIRE 2018. HASOC 2019 will be offered in English, German, and Hindi Languages.

[1]https://hasoc2019.github.io/

B.2 HASOC Shared Task

HASOC is inspired from two evaluation forums, OffensEval and GermanEval 2018. Our objective behind the HASOC shared task is to leverage the synergies of both forums. HASOC shared task is offered in 3 sub-tasks.

B.2.1 Sub-task A

Sub-task A focus on Hate speech and Offensive language identification offered for English, German, Hindi. Sub-task A is coarse-grained binary classification in which participating system are required to classify tweets into two class, namely: Hate and Offensive (HOF) and Non- Hate and offensive (NOT).

- (NOT) Non Hate-Offensive - This post does not contain any Hate speech, offensive content.
- (HOF) Hate and Offensive - This post contains Hate, offensive, and profane content

In our annotation, we label a post as HOF if it contains any form of non-acceptable language such as hate speech, aggression, profanity otherwise NOT

B.2.2 Sub-task B

Sub-task B is a fine-grained classification. Hate-speech and offensive posts from the sub-task A are further classified into three categories.

- Hate : - Posts under this class contain Hate speech content.
- Offen :- Posts under this class contain offensive content.
- Profane :- These posts contain profane words.

HATE SPEECH

Describing negative attributes or deficiencies to groups of individuals because they are members of a group (e.g. all poor people are stupid). Hateful comment toward groups because of race, political opinion, sexual orientation, gender, social status, health condition or similar.

OFFENSIVE

Posts which are degrading, dehumanizing,insulting an individual,threatening with violent acts are categorized into OFFENSIVE category.

PROFANITY

Unacceptable language in the absence of insults and abuse. This typically concerns the usage of swearwords (Scheiße, F*ck etc.) and cursing (Zur Hölle! Verdammt! etc.) are categorized into this category.

We expect most posts to be OTHER, some to be HATE and the other two categories to be less frequent. Dubious cases which are difficult to decide even for humans, should be left out.

B.2.3 Sub-task C

Sub-task c check the type of offense. Only posts labeled as HOF in sub-task A are included in sub-task C. The two categories in sub-task c are the following:

Targeted Insult (TIN) : Posts containing an insult/threat to an individual, group, or others

Untargeted (UNT) : Posts containing nontargeted profanity and swearing. Posts with general profanity are not targeted, but they contain non-acceptable language.

B.3 HASOC Corpus

The HASOC dataset was subsequently sampled from Twitter and partially from Facebook for all the three languages. The Twitter API gives a large number of recent tweets which resulted in an unbiased dataset. Thus the tweets were acquired using hashtags and keywords that contained offensive content. The collection was provided to participants without metadata. The size of the data corpus is shown in tables B.1 and B.2.

Table B.3 shows sample posts from our proposed HASOC corpus.

Table B.1: Collection and Class Distribution for Training Set

Lang.	NOT	HOF	HATE	OFFN	PRFN	Total
English	3591	2261	1143	667	451	5852
Hindi	2196	2469	556	676	1237	4665
German	3412	407	111	210	86	3819

Table B.2: Collection and Class Distribution for test Dataset

Lang.	NOT	HOF	HATE	OFFN	PRFN	Total
English	865	288	124	71	93	1153
Hindi	713	605	190	197	218	1318
German	714	136	41	77	18	850

Table B.3: sample post for the each class at each level.

Text	Level-1	Level-2	Level-3
@RT @Srbh1303: @ivijayup @Payal_Rohatgi First terrorist is Jinnah because of him many people killed and he is Muslim.	HOF	HATE	TIN
RT @AartiTikoo: Nathuram Godse was an assassin, a murderer, a bigot, a hate-monger. But what do you call all those Muslims, Sikhs;Hindus who massacred each other through the Partition of India? Were they all the first Muslim terrorists, the first Sikh terrorists, the first Hindu terrorists?	HOF	OFF	TIN
RT @SinghLions: Sikhs delivered 5 copies of the Holy Quran and prayer mats to a refugee camp near Mosul in Iraq. This gesture by Sikhs towards our Muslim brothers in need during Ramadan is heart-touching and an epitome of interfaith co-operation	NOT	NULL	NULL

The overlap between annotators for task A for English, Hindi, and German for a subset to tweets and posts annotated twice was 89%, 91%, and 32%, respectively. Further statistical details of the annotation process can be seen in table B.4. The effects of such disagreement need to be analyzed in the future.

Overall, 103 registrations were submitted for the track. 37 teams submitted runs and 25 teams have submitted papers. 321 runs were submitted by 37 teams in all the sub-tasks.

Table B.4: Interrater Statistics on HASOC Multilingual Datasets

Task	No. of Posts annotated twice/Total Posts	Percentage of Posts annotated twice	No. of Posts with same annotation	Interrater Agreement
English sub-task A	6246/7005	89%	4838	77.46%
English sub-task B	6246/7005	89%	4311	69.02%
English sub-task C	6246/7005	89%	4669	74.75%
Hindi sub-task A	5440/5983	91%	4281	78.69%
Hindi sub-task B	5440/5983	91%	3421	62.89%
Hindi sub-task C	5440/5983	91%	3488	64.12%
German sub-task A	1483/4669	32%	1305	88%
German sub-task B	1483/4669	32%	1283	86.51%

B.4 Publicly Available Hate Speech Datasets

- Zampieri et al.[132] Offenseval at Semeval 2019 : https://scholar.harvard.edu /malmasi/olid.
- Basile et al. [11] HateEval at Semeval 2019 : https://competitions.codalab.org/ competitions/19935.
- Kumar et al.[52] at TRAC COLING 2018:https://sites.google.com/view /trac1/home.
- Fersini et al. [35]AMI 2018: https://amievalita2018.wordpress.com
- Wiegand et al.[127] GermanEval 2018: https://projects.fzai.h-da.de/iggsa/germeval-2018/
- Davidson et al. [29] Hate Speech corpus:-https://github.com/t-davidson/ hate-speech-and-offensive-language
- Mubarak et al.[85] Hate speech corpus Arabic social media:- http:// alt.qcri.org/hmubarak/ offensive/ TweetClassification-Summary.xlsx
- Ross et al. [99] German hate speech corpus for the refugee crisis.:- https://github.com/UCSM-DUE/ IWG_hatespeech_public.
- tulkens et al. [118]Racism Detection in Dutch Social Media: https://github.com/clips/hades.

CHAPTER C

List of Publications

Journal

- Modha Sandip, Thomas Mandl, Prasenjit Majumder, and Daksh Patel .Tracking Hate in Social Media: Evaluation, Challenges, and Approaches in Special issue of Forum for Information Retrieval, Springer Nature.
- Modha Sandip, Prasenjit Majumder, Thomas Mandl,and Chintak Mandlia. Detecting and Visualizing Hate Speech in Social Media : A Cyber Watchdog for Surveillance . Expert Systems with Applications - Journal - Elsevier: Submitted
- Modha Sandip and Prasenjit Majumder. An Empirical Evaluation of Text Representation Schemes to classify Social media streams. Journal of Experimental and Theoretical Artificial Intelligence : Major revision
- Modha, Sandip and Prasenjit, Majumder and Singla, Risab "Design and Analysis of Microblog based Summarization System" in Social Network Analysis and Mining journal - submitted.

Conferences and Workshops

- Sandip Modha, Thomas Mandl, Prasenjit Majumder and Daksh Patel. 2019. Overview of the HASOC track at FIRE 2019: Hate Speech and Offensive Content Identification in Indo-European Languages. In Proceedings of the 11th annual meeting of the Forum for Information Retrieval Evaluation ACM.
- Thomas Mandl, Sandip Modha, and Daksh Patel. 2019. HASOC: A Resource for

Hate Speech Identification in Hindi, English and German. In 12th Edition of its Language Resources and Evaluation Conference-Submitted

- Modha Sandip and Prasenjit Majumder, Thomas Mandl, Daksh Patel. DA-LD-Hildesheim at SemEval-2019 Task 6: Tracking Offensive Content with Deep Learning using Shallow Representation In Proceedings of The 13th International Workshop on Semantic Evaluation (SemEval)

- Modha Sandip and Prasenjit Majumder and Thomas Mandl. Filtering Aggression from the multilingual Social Media in Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), 199–207

- Modha Sandip. Microblog Processing: A Study in Doctoral Colloquium Forum for Information Retrieval Evaluation 2017 Indian Institute of Science, Bangalore

- Modha, Sandip and Singla, Rishab and Soni, Chintak,and Prasenjit Majumder. Information Extraction from Microblog for Disaster Related Event. In SMERP@ ECIR 2017 (pp. 85-92).

- Modha, Sandip and Singla, Rishab and Soni, Chintak,and Prasenjit Majumder "Summarizing Disaster Related Event from Microblog" In SMERP@ ECIR 2017 (pp. 109-115).

- Modha, Sandip Soni, Chintak, Prasenjit Majumder, Agrawal, Krati, and Verma, Deepali, "Real Time Information Extraction from Microblog". In FIRE 2016 (pp. 79-80).

- Modha, Sandip Soni, Chintak,and Prasenjit Majumder, Agrawal, Krati, and Verma, Deepali. "DAIICT at TREC RTS 2016: Live Push Notification and Email Digest." TREC. 2016.

- Modha, Sandip "Performance Analysis of Clustering Algorithm in Sensing Microblog for Smart Cities." Proceedings of the International Congress on Information and Communication Technology. Springer, Singapore, 2016(pp. 467–475).