A DEEP LEARNING ENSEMBLE HATE SPEECH DETECTION APPROACH FOR SINHALA TWEETS

Munasinghe Imiyage Sidath Asiri Munasinghe

(209358D)

Master of Science in Data Science

Department of Computer Science and Engineering
Faculty of Engineering

University of Moratuwa
Sri Lanka

March 2022

A DEEP LEARNING ENSEMBLE HATE SPEECH DETECTION APPROACH FOR SINHALA TWEETS

Munasinghe Imiyage Sidath Asiri Munasinghe

(209358D)

Thesis/Dissertation submitted in partial fulfillment of the requirements for the degree Master of Science in Data Science

Department of Computer Science and Engineering
Faculty of Engineering

University of Moratuwa
Sri Lanka

March 2022

DECLARATION

I declare that this is my work and this dissertation does not incorporate without acknowledgment any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief, it does not contain any material previously published or written by another person except where the acknowledgment is made in the text.

Also, I hereby grant to the University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic, or another medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: Date:

The above candidate has carried out research for the Master's dissertation under my supervision.

Signature of the supervisor: Date:

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to all those who provided support to make my research on “A DEEP LEARNING ENSEMBLE HATE SPEECH DETECTION APPROACH FOR SINHALA TWEETS” successful.

First of all, I would like to express my gratitude towards my project supervisor Dr. Uthayashanker Thayasivam, Senior Lecturer, Department of Computer Science and Engineering. I am highly indebted to him for his guidance and constant supervision as well as for providing necessary information regarding the project and for his support in completing the project successfully.

I am sincerely thankful to the final year project coordinator Dr. Charith Chittaranjan, Senior Lecturer, Department of Computer Science and Engineering for the support given throughout the project time period. Further, I would like to extend my gratitude to Dr. Sapumal Ahangama for participating in evaluations and providing me very useful guidance to make research successful.

Especially I would like to thank Prof. Indika Perera, Head of Department, Department of Computer Science and Engineering for his assistance and coordination to conduct the research without any issues during the final year.

Finally, I wish to thank the academic and non-academic staff of Department of Computer Science and Engineering and everyone who supported me.

Abstract

We live in an era where social media platforms play a key role in the society. With the advancement of technology, these platforms have become more closer to people and currently, they can interact with most of the native languages including the Sinhala language. This has enabled people to express their opinions more conveniently. At the same time, it is very common to observe that people express very hateful offensive opinions on social media platforms and in certain applications it a mandatory to block this kind of content.

Several studies have been carried out on this area for the Sinhala language with traditional machine learning models and as per the results, none of them have shown promising results. Further, current approaches are far behind the latest techniques carried out in high-resource languages like English. Hence this study presents a deep learning-based approach for hate speech detection which has shown outstanding results for other languages. Three deep learning models namely LSTM, CNN and BiGRU which have proven performance in Natural Language Processing domain have been considered here. Moreover, a deep learning ensemble was constructed from these three models to evaluate whether the ensemble technique can further improve the model performance. These models were trained and tested on a newly created dataset using the Twitter API. Moreover, the model generalizability was further tested by applying it to a completely new dataset.

As per the results, it can be clearly observed that the deep learning-based approach has outperformed the traditional machine learning models. Moreover, further tests on the model generalizability reveal that this approach is more generalized and produces better predictions than the prior approaches.

Finally, this study experiments with using extra features in addition to the Tweet content such as retweet count, favourited count, etc, to evaluate whether those can be utilized to improve the performance further. As per the results obtained in this study, it can be observed that there is an impact on the performance using extra features. It is recommended to experiment further on this area in future studies.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

With the development of technology, it can be seen that the digital world is supporting different native languages other than English at present. This has enabled people to express their opinions on social media platforms in their native language. However, it is very common to observe that people express very hateful offensive comments on social media platforms. Since Social Media platforms do not have a proper solution to control this, it has become a good platform to promote their backward thoughts without being overseen and monitored. Although numerous studies have been done on abusive text detection in other languages [1] [2] [3] [4], there are very few of them have been done in Sinhala as it is a very unique language that is only used in Sri Lanka.

1.1 Hate Speech Detection Approaches

Hate speech detection approaches can be categorized mainly into three categories namely Lexicon-based approaches, Natural Language Processing (NLP) based approaches and deep learning-based approaches.

Common lexicon-based approaches are based on black lists or regular expressions. The approach was to compare a given phrase against the black list or apply a set of regular expressions and determine whether it is hateful or not based on the output. Although the approach is very simple creating and maintaining such a blacklist or regular expressions is difficult and often, they lead to false positives.

These drawbacks were addressed in machine learning based NLP approaches. Models such as Regression models, Support Vector Machines, Naïve Bayes, Decision trees belong to this category. The need of analysing data manually and creating blacklists or regular expressions is not required anymore since the machine learning models can identify those patterns by training the models over a given dataset. However, it was observed that although these models can detect the patterns,

it is very limited. Deep learning-based approaches were introduced to handle this drawback since different types of deep learning models have their own special characteristics that can be very useful in handling textual data. According to the recent studies on deep learning models, they have shown prominent results compared to the all-other prior approaches.

1.2 Challenges in Hate Speech Detection

Posting hateful opinions on social media platforms is not acceptable and automated machine learning solutions are used to avoid this kind of content. However being a very uncommon language, fewer studies [5] [6] [7] [8] [9] have been done on this area for the Sinhala language. Existing studies have not shown promising results as they are not well generalized for actual use cases as explained comprehensively in the literature review section. Moreover, as per the best knowledge, no study has focused on deep learning for the Sinhala language which has proven to be superior in this area according to literature in other languages. Current studies that have been done for the Sinhala language have focused only on the text content. However, adding user-centric extra features and features related to its context can help the model to identify patterns more accurately [10]. Another challenge in this domain is, compared to other languages there is a smaller number of tools available to apply NLP techniques to pre-process the data. Further, these existing tools are also not matured compared to the same tools in other languages as well.

1.3 Research Objectives

- Create a new labelled Sinhala hate speech dataset and publish publicly for future researches
- Experiment with recent feature engineering techniques used in high resourced languages
- Evaluate LSTM, 1D CNN, BiGRU classifiers which have shown good results in high resourced languages
- Build a deep learning ensemble
- Evaluate the constructed model against other Sinhala datasets to evaluate the test generalizability

1.4 Contributions of Research

One of the main challenges in this research area is not having sufficiently large datasets to train models. Hence in this study, a newly labelled hate speech dataset is constructed by annotating manually and made available for future studies. In addition to that, this study experiments with new features that can be useful in modelling hate speech classification for the Sinhala language rather than only depending on the textual content of speech.

This study contributes to the domain of Sinhala hate speech detection by presenting an ensemble deep learning model with more generalized and accurate predictions compared to the prior studies. Multiple deep learning techniques have been experimented here comparing their performance which provides useful directions for

future research purposes. This is the first attempt on applying deep learning techniques in this domain and future studies can refer this to construct more improved solutions.

2 LITERATURE REVIEW

Hate speech detection on social media platforms is an active study area in the research community at present due to the wide usage of these platforms. Among those, the majority of studies have been done on the English language compared to other languages as it is the dominant language in the aspect of usage in these platforms. Moreover, the availability of more tools/libraries related to natural language processing for the English language compared to other languages also encourages the development of more active researches.

In their study [11] Sean et. al have observed that with the growth of online content, the spread of hate speech is also growing. Hence in their study, they have identified and examined the key challenges that exist in automatic hate speech detection systems. In addition to that, they have identified that recent studies are not interpretable to understand how they make decisions. Thus, in their study, they have proposed an approach consisting multi-view SVM model that has more interpretable decisions while maintaining state-of-the-art performance.

As per the authors, the first challenge in hate speech detection is the definition of hate speech as there is no globally accepted proper definition. Without having a proper definition, the annotation of the dataset can vary from each annotator. Authors have analyzed various definitions of hate speech from prior studies as well as online sources and they have observed that most of such definitions are not complete as there can be scenarios that will not be captured correctly. The next challenge they have identified is getting a properly annotated dataset. In the absence of a proper definition of what is hate speech, the annotating process can be highly subjective. Moreover, they have identified that there are not many freely available datasets that recognize aggressive, hateful and abusive text. Although these are common in online platforms, they have very strict privacy policies on data usage and distribution. Another key challenge related to the datasets in this domain is the class imbalance due to most online comments are accepted while only a few of them have hateful content.

After stating key challenges, the authors have presented an overview of existing automated approaches for hate speech discovery including keyword-based approaches, source metadata-related approaches, and machine learning-based approaches. After analysing the existing approaches, they have proposed a novel approach to detect hate text using a multi-view SVM model. It contains a multiple-view stacked SVM producing a view-classifier for particular features. Then they have united the view classifiers with another Linear Support Vector Machine to construct a meta-classifier. As per results, the proposed new approach has outperformed existing systems with the advantage of enhanced model interpretability

which is a key factor of a machine learning model. Finally, the authors have stated that with the mentioned challenges in this research area, there is still a need for more researches to address both technical and practical challenges.

2.1 Approaches in Detecting Abusive Text

Although there are numerous studies have done on detecting abusive text, it is possible to classify these approaches broadly into three main categories called lexicon-based abusive text detection, machine learning-based abusive text detection, and hybrid approach by combining the mentioned approaches [9].

"A lexicon-based hate speech detection" study [12] has carried out in 2015 to identify the existence of hate speech in the web such as blogs and web forums. The objective of this study is to construct a classifier that reduces the document size by reducing objective sentences and use subjectivity and semantic attributes associated to hate speech. Subjective sentence detection has been carried out by a rule-based method to classify sentences depending on sentiment lexicon resources of Wilson et al. [13] [14] and SentiWordNet [15].

Determining whether a given sentence is subjective or not has been done by calculating the average negative or positive scores from each sentiment token in the sentence by referring to the sentiment lexicon resources. Then in the next step, the lexicon of hate speech was built with 3 diverse sets of features including negatively opinionated tokens from the subjectivity analysis, verbs that are linked to their hate speech dataset but not in the initial feature set, and hate nouns with the help of a Named Entity Recognition (NER) tool.

Finally, the created lexicon was used to create the rule-based classifier based on sentence-level valuation directed by the number and reliability of opinion words in a sentence as well as incidences of other lexical terminologies in the sentence. As a summary their model predicts the sentence to be strongly hateful if two or more words are tagged as strongly negative relative to their constructed lexicon. Or else, if only a single word seems as strongly negative, they predict the sentence as faintly hateful.

Although the study has shown promising results, this approach is not scalable for modern social media platforms due to the restriction of the scope of the constructed lexicon. It is not possible to construct an ultimate lexicon that covers all the necessary lexicons to make correct classifications and maintaining such a lexicon is an even more difficult task. Hence this approach leads to long-term poor performance making the researchers focus on machine learning-based approaches which have proven to be better.

Machine learning approaches have several advantages compared to this pre-build lexicon-based approach. The main advantage is the programmer does not need to define rules or instruct how to classify correctly. Instead, the model can gain such required knowledge while in the training phase. A supervised learning paradigm in machine learning can be used to classify abusive text conveniently with a labelled dataset for the training phase. In addition to the traditional machine learning models such as “Support Vector Machine” classifier, “Naïve Bayes”, deep “Artificial Neural Networks” (ANN) have proven promising performance in this domain which can model more advanced features compared to traditional models.

Such a machine learning study has been done by Razavi et.al [16] using a Naive Bayes model for hate speech recognition using a multi-level classification approach. It consists of three-level classifications on the training data. Due to a large number of features even after removing stop words, in the first layer, they have used a “Complement Naïve Bayes” classifier [17] for choosing the utmost discriminative features. Then the result of the first layer has passed to the second layer which consists of a Multinomial Naïve Bayes classifier [17] for adaptive learning based on the labelled data. Productions of this second level have the accumulated features taken out from the prior level feature space as the input for their final level classification task. At the final level, a rule-based classifier named “Decision Table/Naive Bayes” hybrid classifier (DTNB) [18] has been applied to the production of the second stage to make the final classification on whether the text is offensive or not. By arranging the machine learning models in a multi-layer structure, they have obtained significant results. Moreover, they have observed that the stability of the system has increased due to the special multi-level structure. The notable features of this approach are that it is not sensitive to punctuation or grammatical mistakes and it can be adapted easily with users' input with time unlike in the lexicon approaches.

Another study [19] on detecting hateful text has been conducted to bring together different kinds of literature on the domain and avoid contradictions. In addition to that, they have suggested a typology that synthesizes these different subtasks. Authors have claimed that abusive language subtasks can be classified into a 2-fold typology based on whether the abuse is focused at a precise target or the degree to which it is explicit. A specific target can be a specific individual or it can be a generalized group such as people included in a certain ethnicity or sexual orientation. It can be seen that in online platforms; abusive or offensive comments/ posts are directed to both of these types. The other type, explicit abusive language measures the unambiguous of its abusiveness. A phrase that contains racial/ homophobic slurs is very explicitly abusive. On the other hand, implicit abusive language does not

immediately imply abuse. Usually, it is hidden by ambiguous terms or sarcasm without using hateful slurs. This type of abusive text is very difficult to predict due to its ambiguous nature compared to the explicit type. Authors have mentioned that annotating text using crowd-sourcing or any other methods is straightforward only when explicit instances of abusive language are present and it would be very challenging when implicit abuse is considered as it is very subjective. In addition to that, they have mentioned that to make accurate classifications on explicit abusive language detection, the features should not be limited to the tokens, n-grams of the corpus since in implicit abusive it is expressed without using any hateful terms.

2.2 Studies Carried Out for the Sinhala Language

A handful number of studies have been done for the Sinhala language related to abusive text detection and the following Figure 1 depicts how the research area evolved gradually. Initial studies have more focused on creating the essentials resources to do NLP tasks such as stop words, NER, tokenizers and how to create a proper annotated dataset. After that, researchers have stared constructing machine learning models to solve this problem with commonly used machine learning models such as Logistic Regression, Naïve Bayes, SVM and Random Forest. However, they have noticed that these models are not performing well for certain scenarios. To overcome these issues, they have combined the machine learning models with a set to rules to correctly classify these kinds of scenarios. Another area related to Sinhala hate speech detection is Romanized Sinhala which relates to Sinhala words written in English.

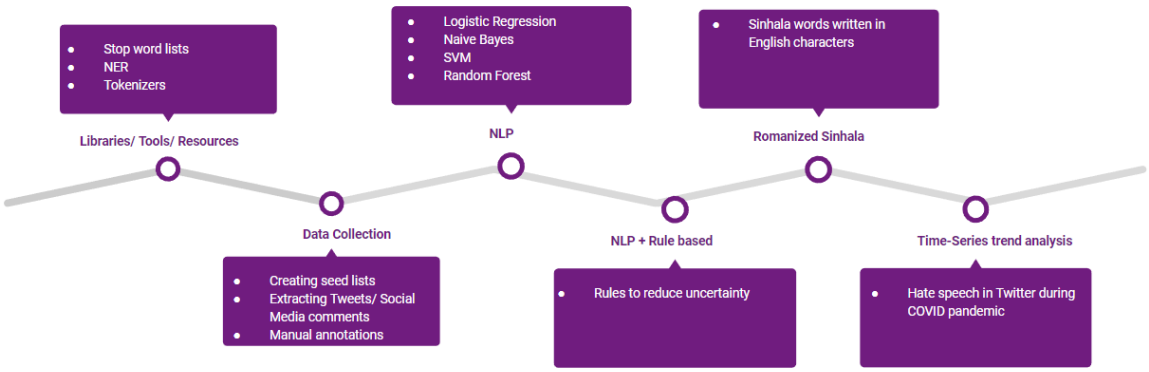


Figure 1: The flow of how the research area for Sinhala language evolved from creating the NLP related tools to building machine learning classification models

One such study [9] has been done recently in 2020 to recognize abusive Sinhala comments on social media via text mining and machine learning approaches. They have used social media platforms like Facebook, YouTube, and an online gossip website to collect user comments for the dataset. Since a labelled dataset is required for supervised machine learning, they have manually annotated data using three

annotators. In the annotated dataset, they had 1100 comments in each abusive and neutral comment class.

Authors have done some prior analyses on the dataset such as sentence/word length analysis, vocabulary analysis, and Zipf's law analysis before training the machine learning models. Analysis has revealed that the average word and sentence length is shorter in offensive texts compared with the neutral class. Similarly, the number of words and the number of unique words are lower in abusive texts. As pre-processing steps, they have applied stop words removal and stemming the tokens. According to Zipf's law analysis, they have observed that, even after spread over pre-processing procedures, the corpus follows Zipf's law.

As there is no lexicon store for Sinhala offensive speech available, in the same study they have proposed two approaches to make them. The first approach uses google bad word list [20] and converts them to Sinhala words using an online converter to construct the lexicon. The other approach uses the collected corpus of abusive comments and with the help of a seed word set occupied from an online source, they have investigated their discrepancies in the offensive corpus. Results have been cross-checked with the annotators and the correct accuracy was 99.28%.

Authors have used multiple techniques such as the "Bag of Words" model, "character n-gram" model, "word skip-gram" model and "word n-gram" model to extract features. In addition to that, they have considered feature vectorizers such as CountVectorizer and TfIdfVectorizers to prepare the features to feed into the machine learning models. Here, they have trained three machine learning models which are Naïve Bayes, SVM, and Random Forest classifier. To evaluate these models, they have used different matrices such as recall, precision, accuracy, and f-score. Results have clearly shown that when increasing the n in different feature extracting models such as n-gram, the accuracy has increased significantly for all the machine learning models. Naïve Bayes classifier which is their best performer has increased from 0.645 to 0.955. The reason for this observation is that tokens do not help to detect whether the text is abusive or not by the token itself. The other tokens that are around the token are also important. Deep learning models such as RNNs are very suitable for these kinds of applications as they consider the input data as a sequence.

A similar study [7] has done by Dulan S. Dias et.al on detecting racist Sinhala comments in social with Machine Learning Models. Racism is another key problem similar to hateful text in social media which discriminates against people based on factors such as their race, gender, skin color, etc. To address this problem, the

authors have presented a text analytics model based on a 2 class SVM to classify whether a given comment is racist or not.

To train the SVM model they have collected comments from social media platforms like Facebook and they have manually annotated them by reading carefully whether they are racist or not. While annotating, they have observed that correct classifications cannot be done by simply identifying racist words in a given sentence as those words can occur in non-racist comments as well without making a racist intention. Compared to other studies, this study has a very small corpus containing only 184 instances that have nearly 40% racist comments. In data pre-processing, they have removed numbers, stop words, duplicate characters, special characters, URLs, and email addresses from comments. To extract the features, they have used an n-gram model with size two and TF-IDF as the weighting function.

After extracting features, they have trained the SVM model with randomly selected 75% of labelled data in the dataset. They have used a linear kernel in the SVM model and have trained it for 53 iterations setting the lambda to 0.096. They have used parameter tuning methods to get these optimal values to make the most accurate results. As per the results, they have scored a good accuracy of 0.708 and precision of 1.000. However, the recall of the system is very low which is 0.364 making a low f1 score of 0.533. Since this performance is not acceptable, they have increased the data corpus assuming that this behaviour has occurred due to a lack of data for training. However, even after that, the model performance has dropped again. Hence this solution is not enough generalized to make accurate results and scalable for a large amount of data. The main reason to have a very low recall is that the model has overfitted to the training set and it does not consider the sequential order of words in the input text to understand the intention of the phrase. Only considering the tokens alone will lead to poor performance like this as it was observed that certain words that are prominent in a racist text can occur in non-racist text as well.

Hate speech detection in Social Media Articles written in Romanized Sinhala was done by Nimali Hettiarachchi et al [8] which is an area that has not been considered in prior studies. A Romanised Sinhala word is a word that is written using English characters as it is pronounced in the Sinhala language. Authors have identified that publishing hateful content is growing fast and need to tackle this problem by using machine learning and computer science-based solutions. In this study, they have compared several feature extraction approaches and multiple machine learning algorithms including numerous feature engineering techniques.

For the study, the authors have created a dataset by collecting comments posted by users on online social media platforms written in Romanized Sinhala language. Then

it has annotated manually into two classes indicating whether it is hated or not. The final dataset has included 2500 records having 1400 records in hate class and 1100 records in non-hate speech. In the pre-processing step, they have removed HTML tags, non-alphanumeric characters, special characters, stop words, and applied tokenization and stemming. In the feature extraction phase, they have used CountVectorizer and TF-IDF Vectorizer with a bag of word representation utilizing the n-gram model. To compare the performance among different machine learning models, they have considered "Multinomial Naive Bayes" Classifier, "Logistic Regression", "Random Forest" Classifier, and "Linear SVM".

To evaluate these models, they have followed a quantitative approach by measuring their performance with precision, accuracy, recall, f1 score, classification report and confusion matrix. As per the results, they have observed that the best results are scored by Multinomial Naive Bayes with TF-IDF vectorizer. However, when comparing the accuracy on train and test sets it can be seen that all the models have been overfitted. The reason for this should be due to only considering hyperparameter optimizing techniques without using any regularization technique.

Another interesting study [5] has been conducted for detecting cyberbullying Sinhala comments on social media using machine learning approaches. Authors have identified that as a result of continuous technological development, bullying which was limited to physical margins has now extended to online causing more damage. They have stated that rude words are dynamic, and the same word can have numerous meanings/explanations conferring to the context. Hence it is very difficult to correctly classify as additional information related to the context is also required. In the absence of lexical databases such as WordNet for the Sinhala language, they have used five rules to overcome the challenge. The novelty of this study is that it has focused on the Sinhala language and Cyberbullying for these languages has not been evaluated before.

As the first step, they have collected comments from social media platforms related to both cyberbullying and not cyberbullying. They have referred to a set of Sinhala bad words to extract comments from Twitter using Twitter API. However, they could able to collect a very small dataset that has only 652 records. Then they have defined five rules to add more intelligence to the model in addition to simply focusing on keywords. These rules are related to factors such as the percentage of bad words in a tweet, the combination of first-person/ second-person or third-person pronoun with a bad word, etc. To annotate the records, they have used a manual annotating step using four levels based on confidence. The final label has determined by the total number of participants labelled in each level and the given weight by each level.

In the pre-processing, they have considered records that have a word count less than 6 and a word count of more than 23 as outliers. Further, they have removed re-tweets to remove duplicates. After these steps, the dataset has further reduced to 292 records. In addition to that, they have removed emojis, punctuation, numbers, URLs, indications in the tweets, and stop words before training the models. For feature extracting, they have applied the previously mentioned five rules across all the records in the dataset and returning values have been used as new features. After extracting all the features, they have trained KNN (K-nearest Neighbour), NB (Naïve Bayes), and SVM (Support Vector Machine) algorithms by dividing the dataset to train and test sets.

As per the results, they have concluded that the SVM model with the RBF kernel has the best performance in terms of the F1 score. In addition to that, they have further reduced the dataset and tested the SVM model with cross-validation again. Still the results have shown good performance. However, although the authors claim that they are not only depending only on keywords and considering context as well for classifications, it is still challenging due to the usage of this small dataset.

A special study [6] has carried out on time-series-based trend analysis for hate speech on Twitter during the COVID'19 pandemics. Authors have identified that, during the COVID period, social media platforms have been used as a medium of information propagation more frequently. This study presents the results on how the Sinhala language and its words written in English were spread. In addition to that, the authors have discussed the trend analysis techniques that can be used in trend analysis to recognize the hate speech spread trends over the COVID pandemic period.

Data collection has done by extracting Twitter posts from a selected period written in the medium of Sinhala language. Any post that is written in English or Sinhala words written in English were removed. Then they have classified data into 2 categories namely posts by a user and replies. To label the data points as hateful, not hateful, or neutral they have followed a crowdsourcing approach by manually annotating them by 50 university students. For the labelling process, they have distributed each post among 5 students.

At first, the study was carried out by considering the seasonal effect and the life span to produce the boundaries of the experiment, by setting the maximum life span to 7 days. While carrying out the time series analysis, the authors have done a minute-based analysis. According to that analysis, the observed average life span is 3.54 days. Moreover, the "Augmented Dickey-Fuller" (ADF) test was applied and the results reveal that there is no seasonality or trend. Hence, the authors have further

investigated the experiment by applying exponential smoothing. After smoothing data, it has been identified that the subsequent comments connected to hate speech with a downwards movement. The final analysis has conducted using the "Box-Jenkins" forecasting method, which revealed a steady time series with less volatility.

In the conclusions, the authors have highlighted the importance of this study stating that it is required to recognize the pattern of hate speech in a selected period. Further, as per the results of the analysis, they have concluded that hate comments have no specific seasonality. However, hate speech during the outbreak season has decreased with time. The pattern with time series analysis has provided a compact forecast model. As the next step, authors are eager to produce novel predicting models for short-range trend analysis.

2.3 Studies carried out for the English language

The initial hate speech detection approaches for English language have been constructed based on black lists and regular expressions as per Figure 2. However, researchers have identified that these approaches are not scalable as these lists and regular expressions have to be maintained continuously. Due to that limitation, they have more focused on machine learning approaches and they have shown better results relatively. Further due to enormous data available, deep learning models have experimented and they have shown significant performance improvements when comparing with prior approaches. A lot of studies can be seen in this area where various types of deep leaning architectures have been experimented. After that, a set of studies have been carried out to investigate on the different errors that occur in hate speech detection. With these findings more studies have been conducted to address these errors by introducing new techniques. The latest studies are more focused on constructing solutions that can handle multiple languages.

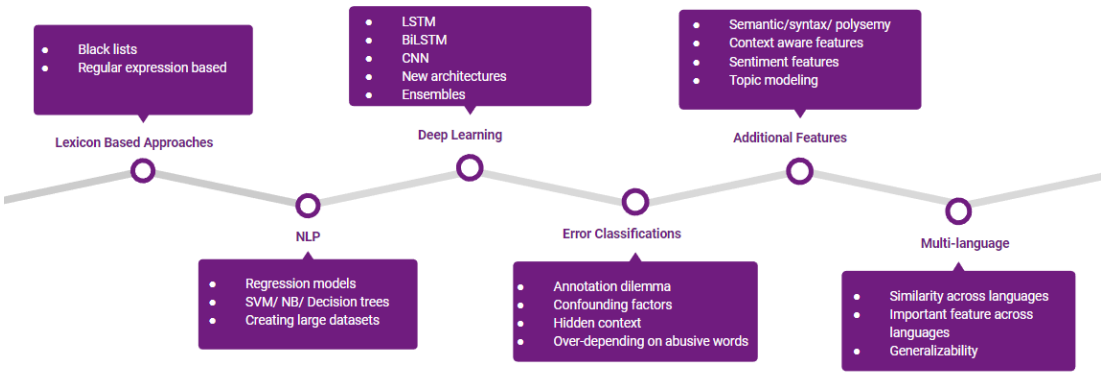


Figure 2: The flow of how the research area for English language evolved from lexicon-based approaches to deep learning approaches

Among the majority of studies in English language, an approach to analyse abusive language over time and enhance knowledge of the behavioural patterns has proposed by Chilakshi et. al [21] in 2016 with a machine learning method. They have identified that most solutions use blacklists, regular expression-based approaches which have a lot of limitations and not suitable for the long run. Hence, the authors are presenting a state-of-the-art method to build a supervised classification procedure with NLP techniques. The importance of this study is that it addresses the issue of language changing nature and users' behavioural changes over time.

Being the authors work at Yahoo, they have used comments from Yahoo Finance and News to prepare the dataset. Mainly they have used three types of datasets namely primary, temporal and WWW2015 datasets. They have used comments that get posted daily as well the comments that they receive as reported by the visitors to create the primary dataset. All these comments have been reviewed by Yahoo's in-house trained raters and split into clean and abusive categories in the finance and news domains separately. The temporal dataset has also been created similarly within the period between April 2014 and April 2015 to do their temporal experiments. Finally, they have used the WWW2015 dataset to compare their approach against prior work.

Authors have used Vowpal Wabbit's regression model with its standard set of parameters using different NLP techniques. Mainly they are using four types of features which are Linguistic, N-grams, Syntactic, and Distributional Semantics. In addition to that, they have used few pre-processing steps such as transformations which include replacing long unknown words with a specific token, normalizing numbers, swapping repetitive punctuations with the same token, etc. As semantic features, they have used grandparent of a node, parent of a node, POS of those, children of the node, etc and for the distributional semantic features, they have used 3 types of embedded-derived features. For the initial one, they have used pre-trained embeddings taken from a big corpus of news text and for the second one, they have trained one from their corpus. The third one is a comment embeddings model where every comment is linked to an exclusive vector in a matrix representing comments and every word is linked to a unique vector in a matrix on behalf of words. The main advantage of this approach is that the algorithm is no longer subtle to comment length and will not need optimizing for word weights.

When the approach is evaluated against the primary dataset, an accuracy of 0.795 has been scored for the finance domain and 0.817 for the news domain. Then they have evaluated the model with the WWW2015 dataset, to compare the proposing approach with the prior studies. The results have shown that the new approach is outperforming prior studies. As per the evaluation of the temporal dataset, the

authors have observed that having more fresh data than a larger data set is desirable to train the models. Finally, the authors have concluded that, although the work has been attentive to abuse found in English, it probably works well in other languages when they have trained with enough data.

Due to the existence of numerous approaches, a comparative study [22] on detecting abusive language on a Twitter dataset comparing different machine learning approaches has been done by Younghun Lee, Seunghyun Yoon, and Kyomin Jung. The authors have identified that the prior studied datasets in offensive text detection have been inadequate in volume to train deep learning models and get their full benefit. In this study, they have used a recently released Twitter dataset which has a sufficient number of data points and reliable data to train machine learning models more accurately. A few traditional machine learning classifiers and neural network-based models are compared in detail by introducing additional features and different model variants.

In the data pre-processing step, the text sequences are converted into Bag Of Words representations while normalizing them with the Term Frequency-Inverse Document Frequency technique. Moreover, word-level features have been captured with n-grams variating from 1 to 3, and character-level features from 3 to 8. They have used feature-based models such as Logistic Regression, Naïve Bayes, Support Vector Machine, Random Forest, Gradient Boosted Trees, and neural network-based models such as CNNs, RNNs, and their modified models.

When evaluating the models, they have applied cross-validation technique along with the weighted average of recall, precision, and f1-score for all the labels. The results reveal that the neural network-based models outperform the feature-based models. The “Logistic Regression” model has become the best model among the feature-based model having the same F1 score as the CNN model. Within the neural network models, RNNs with LTC modules have shown the best performance. Further, the authors have concluded that the ensemble of models can be used for further improvements as well. As per the results of this study, it can be seen that neural network-based deep learning models have superior performance over traditional models which should be experimented with the Sinhala language as well.

In their research [23] Thomas Davidson et al. have studied the challenge of offensive language and created an automated hate speech recognition solution. Authors have identified that the lexical detection approaches tend to have less precision as they categorize messages considering a specific set of terms related to hate speech and prior studies built on supervised machine learning have failed to classify the two categories accurately.

To create the hate speech dataset, first they have created a hate speech lexicon holding hateful phrases and words acknowledged by online users as hate speech. Then they have used these terms as keywords to query tweets from Twitter using the Twitter API. Collected data has been annotated manually by crowdsourcing into 3 categories namely neither offensive nor hate speech, offensive but not hate speech, and hate speech. They have clearly defined these three classes and shared them with the annotators to reduce the ambiguity as there is no clear definition available for these classes. The final label to a record has been assigned by the majority vote of the annotators creating a dataset of 24,802 records. In data pre-processing, they have converted all tweets to lower and have applied Porter stemmer. After that, unigram, bigram, and trigram features have constructed weighted by their TF-IDF value. To capture the syntactic structure of tweets they have used the NLTK Penn Part-of-Speech tagging technique.

In the model constructing step, they have evaluated multiple models starting from the Logistic Regression model to Naive Bayes, Decision Trees, Random Forests, and Linear SVM models. They have evaluated each model with five-fold cross-validation while using the regularization technique to avoid model overfitting. After evaluating all models, the authors have observed that the Linear Regression and SVM models have outperformed other models. However, they have observed that these models are influenced towards classifying tweets as less offensive or hateful by showing low recall and precision for that class. When investigating the reason for this behaviour, they have observed that these tweets hold terms that can be considered sexist/racist although the overall intention of the tweet does not imply such intention. To overcome this issue, the authors have suggested that the sources of training data should be considered to distinguish the classes more accurately.

To tackle this challenge Lei Gao and Ruihong and Huang have done a study [24] on detecting online hate speech using context-aware models. In this paper, the authors have provided an annotated dataset of hate speech which includes context information. Then they have proposed mainly 2 types of hate speech recognition approaches that consider context evidence. These two approaches are based on a “Logistic Regression” model with context-aware features and an ANN model with learning mechanisms for context.

As the dataset, they have used user comments on Fox news which contains 1528 total annotated comments having 435 labelled as hateful posted by 678 different users. Compared to prior datasets, the special feature of this dataset is that it contains context information for each comment including the original news article, user screen name, and the nested structure of comments in the thread. Moreover, the

corpus has creative and implicit hateful comments which require context information to make accurate classifications.

For the Logistic Regression model, they have extracted four types of features including word-level features, character-level features and two types of lexicon-derived features. After extracting these features from the comment, they have extracted the same from two sources of context texts, precisely the headline of the news article and the user's screen name who posted the comment. Moreover, they have incorporated the LIWC 2015 dictionary which supports to identify the semantic meaning of each word, and the NRC emotion lexicon to capture the emotion clues in the text. Their other approach based on the Neural Network has three parallel LSTMs which have three different inputs for target comment, new title, and the posted user's name. The three LSTM output layers have concatenated and sent through a sigmoid activation to output the classifications. When evaluating the models, they have evaluated the models individually as well as the ensemble of both to see if there is a performance improvement from the ensemble.

Results have clearly shown that adding the context-related features improves the performance of both models. Further, the performance has significantly improved by making the ensemble of models. In further analysis, authors have identified that both types of models have exceptional strengths in classifying certain types of hateful comments and when they are in an ensemble the performance gets enhanced due to that reason.

A similar context-aware approach [4] has been presented by Usman Naseem et al. on the deep learning-based solution for context-aware embedding for offensive and hate speech detection on Tweets. When developing the solution, they have considered syntax, polysemy, semantic of posts, and out of vocabulary words including sentiment knowledge as well. Their proposed deep learning context-aware embedding approach involves 2 core modules namely BiLSTM with an attention mechanism and deep hybrid contextual word illustration. Here they have used different data representations at the same time to include the context and semantic information. This has been achieved by the concatenation of word and character level representation (word-char) followed by the contextual and lexicon level representations. Then they have concatenated with their first word-char words representation to get the final deep hybrid contextual words representation. This word-char representation can capture syntax, semantics, and out of vocabulary words while contextual-level illustration captures the meaning of the same word in different contexts. In addition to the above, authors have applied the traditional pre-processing steps such as replacing emojis with their meanings, correcting spelling mistakes, etc.

Data prepared in the mentioned representation have fed into a bi-directional LSTM model as the input with an attention layer for classification. This model can identify only the words that have a higher impact on classifying hateful/ abusive content while ignoring other words. This is done via the attention layer by assigning a weight to each word through the SoftMax function and creating the final representation by taking the weighted sum of all. They have used the cross-entropy function as the loss function and other hyperparameters have been optimized by using the grid search approach. To tackle the overfitting problem, they have used the L2 regularization technique as well as dropping out connections in the network.

To evaluate the performance of this approach, they have used three benchmark datasets based on tweets to prove that the approach is not limited to a single dataset. They have done a comprehensive comparison of the proposed method with prior approaches with F1-Score to highlight the performance of the new approach. The performance of the proposed model has outperformed existing approaches for these datasets. The authors have claimed that as they are enhancing the value of tweets by eliminating noise and normalizing the unstructured and poor quality of text helps to get improved illustration and have better performance. In addition to that, the capacity of the model to capture the meaning of words in different contexts also has a large impact on superior performance as most of the prior studies do not have that capability. As the proposing approach has shown consistently good performance on all three datasets, it can be concluded that this approach is a robust solution for detecting hate and abusive language.

A study [25] to evaluate how an ensemble of deep neural networks perform in detecting abusive text for the English language has been done with an ensemble of RNN classifiers considering numerous sets of features including user-related information. The key characteristic of this study compared to other studies is that unlike in other approaches, it considers users' inclination on the way to hatred behaviour. Further, it has an architecture, which combines the output of numerous LSTM classifiers to enhance the classification capability using the behaviour of an ensemble classifier. This ensemble architecture has three LSTM models which use a majority vote that enforces to get the agreement of at least two models to make the final decision. If all models get disagree on a single output, it will use the prediction of the classifier with the highest confidence. Each classifier has four layers including the embedding layer, hidden LSTM layer with sigmoid activation units, Relu activated dense layer, and finally, the output layer with SoftMax activation units to predict the three classes Neutral, Racism, and Sexism.

Similar to other approaches, this study also depends on a labelled dataset from Twitter but a huge one containing 16k data points in three classes. After applying the

usual data pre-processing steps, they have used cross-validation with precision, recall, and F-Score matrices to evaluate the method. The results have discovered that this method has outperformed the present state-of-the-art approaches. Moreover, the authors have claimed that none of the other models have obtained superior results than this method. In addition to that, a key observation in the evaluation is that a noticeable performance improvement has been gained via an ensemble as an alternative to a single classifier.

Another such study [2] has been done using an ensemble of deep CNN models to enhance hate speech detection. According to prior studies, neural network approaches have become the state-of-the-art for NLP problems. Hence authors are expecting to use an ensemble method with neural networks for better classification accuracy. In addition to that, they have two main contributions by presenting the experimental results of their improvements and recommendations and suggestions for future studies on this research area.

To create the ensemble model, first they have taken soft-max results as well as their sum from each model. Then the average sum of softmax results has calculated by separating it according to the models considered. Using the average score of all models, the class with the maximum average has been selected to be the classification class. To evaluate whether this solution is well generalized, it is tested on two Twitter datasets. All tweets in datasets have been pre-processed in the same manner by normalizing all URLs to _URL_, mentions to _MENTION_, and numbers to _NUMBER_ and format respectively without changing the case. Due to the usage of word embeddings and CNN classifiers, the capability to handle sequential data through the chain of token embeddings into a matrix is available contrary to the traditional n-gram method which eliminates the information of position in tokens. In addition to that CNN models, can handle variable length documents.

In their CNN architecture, the convolution layer has a window of 3 tokens including 150 filters. Further, padding has been configured to the same since the length of input and the length of output of the convolution layer should be matched. For the feature reduction purpose, the output of the convolution layer has been forwarded to a max-pooling layer. Then the output of this layer has fed into a hidden dense layer with relu activated 250 units. Interestingly, they haven't used the regularization technique. However, the dropout technique has been used with a rate of 0.2 after the max-pooling layer.

When evaluating the performance of the model, they have reviewed results for multiple ensemble models by changing the batch size, number of epochs and seed parameters. Dataset has split by 85/15 ratio for train/test sets and the ensemble

solution has performed better showing an average gain of 1.97% in F1 score. After tuning the parameters optimally, they have applied cross-validation technique and compared the results with the prior studies. They have observed that the ensemble model has outperformed prior approaches. More interestingly they have higher variance compared to the ensemble model implying that the ensemble approach is more confident in its classifications.

As recent studies have focused on detecting abusive text in several different languages, an interesting study [26] has been done to identify which aspects have an impact on multilingual sceneries, focusing on the compatibility of data. In this paper, the authors have considered English and German languages to present the performance differences between languages and special NLP techniques that work better/poor in each language. The study has focused on answering few interesting questions including whether the same classifiers perform similarly across languages, what kind of features are important across languages, how oversampling helps to tackle the class imbalance problem, and how two languages behave when considering the sentiment analysis with topic features.

When collecting data for two languages, they have selected two data sets that are similar to each other without creating new datasets for each language. For the English language, they have chosen a publicly available Twitter hate speech dataset that includes 15715 records. Since it was an unlabelled dataset, it has manually annotated with a well-controlled procedure into three classes including racism, sexism, and none. For the German dataset also have chosen an existing unlabelled dataset and they have manually annotated it into two classes. Since the two datasets have a different number of classes, in the English dataset racism and sexism classes have merged to make two overall classes making the two datasets more similar for the study. In addition to that, even two datasets have a different number of records, both have similar class imbalance rates.

For the different classifiers, they have used a variety of classifiers together with XGBoost, Random Forest Classifier, SVM, and ANN based methods. In parameter optimizations, used for traditional models, they have used a grid-search-based approach and for neural networks, they have used dropout layers and batch normalization techniques. To train these models, they have used character n-grams with stemmed word n-grams and dependency parse-derived features. For stemming they have used the YASS stemming method with a modification to replace cluster members with the shortest member of the cluster instead of the cluster centroid. To extract dependency features, for the English language they have used the Tweebo parser, and for the German language, they have used the Mate parser pipeline. In addition to that, several pre-processing steps have followed including removal of

hashtags, punctuations, removal of the # sign from a hashtag in the middle of the tweet, and removal of all emojis. To tackle the class imbalance problem, they have considered four over-sampling techniques and two under-sampling techniques. Finally, before training the models they have trained two LDA topic models for each language to extract topics for each language to be used as a feature.

When evaluating the model performance, the authors have considered using precision, recall, and F1 due to the skewness in the data set. Then all these measures were calculated as the macro average of precision, recall by taking the average across the two classes. As per the results, the authors have presented several conclusions. It has been observed that for English, XGBoost provides the best performance while German SVMs have the best performance. Stemming technique in addition to the character n-grams has helped in the German language while they have created a negative impact on the English language. From the different class imbalance handling, techniques for English language sampling have improved results while for German it hasn't. This study proves that no globally accepted model will work for all languages. In addition to that, the effectiveness of NLP techniques will change from language to language based on different characteristics or nature of the language.

Another such study [27] on multilingual hate speech detection with deep learning has been done on 9 different languages (Arabic, English, German, Indonesian, Italian, Polish, Portuguese, Spanish, and French) to conduct a large-scale hate speech detection analysis. The study has presented how to build a solution that will effectively work on languages that have more resources as well as for languages that have a very little number of resources to work on. While data collection, the authors have found 16 publicly available datasets for 9 different languages. However, each dataset has different labels such as hateful, abusive, sexual, etc. To maintain consistency, they have only considered the hateful class and the normal class in each dataset by ignoring other classes.

Before training the models in a multilingual setting, multilingual word/ sentence embeddings are required. Here, the authors have used LASER for sentence embeddings and MUSE embeddings for sentences. The study mainly has four experiments including a MUSE embedding and CNN-GRU model, language translation and BERT model, LASER model, and Logistic Regression (LR) model, and finally the mBERT model. In some techniques, they have used language translations via Google Translate to get the benefit of sentiment analysis for better results.

For all the experiments, they have split the dataset into train, validation and test sets with the ratio of 70%, 10%, and 20% respectively. These splits follow the stratified

technique to make sure the distribution of classes remains the same in all sets. The authors have evaluated models in both monolingual settings and multilingual settings. A key observation under a monolingual setting is that LASER + LR has performed the best in low-resource settings for all the languages and MUSE + CNN-GRU has performed worst in all languages. Overall, the study states that there is no perfect approach for all languages. However, the Translation + BERT approach has shown promising results to be an excellent compromise. Moreover, they claim that the performance can be further improved by improving the language translation. In the multilingual setting under zero-shot evaluation, authors have observed that mBERT performs better than LASER + LR in three major languages (Arabic, German, and French). For other languages, LASER + LR performs better. Unlike in other studies, authors have focused on model interpretability as well. Here they have noticed few important facts such as LASER + LR focuses more on the hateful keywords while mBERT seems to search for some context of the hate keywords. Finally, the authors have analyzed the errors that occur frequently in the hate speech detection domain and they have classified error types into four categories.

Although there are many studies have conducted in the hate speech detection domain, an interesting study has done stating that most of the prior studies are overestimated and will not work practically in their study [28]. Authors have highlighted that the results obtained by state-of-the-art systems imply that supervised methods score nearly perfect performance but only within certain specific datasets. They have closely examined the experimental methodology applied in prior work and their generalizability to other datasets than that they have experimented with. They have found evidence that there are methodological issues in those studies, as well as dataset bias showing the overestimated result.

The authors have identified three different datasets that have been used widely in prior studies. Then they have pointed out the drawbacks of those such as having around 90% hateful comments that are published by a single person. Due to these kinds of scenarios, the final trained model will not be generalized well when deployed in actual applications. To replicate the state of art systems, they have considered two pieces of research that have source codes available. The authors have carefully analyzed the methodology of those studies and used their source codes to replicate the solution being able to reproduce the same performance metrics as presented in the original paper.

One of the methodological issues they have observed was extracting features from the whole dataset which includes both train and test data. It is expected to extract features from the trainset and not from the test set as it is supposed to be unseen data to the model. Extracting features from those datasets provides synthetic performance

to the model. Once the authors have changed the implementation to the correct way, the performance of the model has dropped significantly. It has been observed that the F1 score for each class has decreased from 96.1 to 88.1 (Neither), 94.0 to 70.2 (Racist), and 89.3 to 60.9 (Sexist), which leads to a macro-average F1 drop of 20 points (from 93.1 to 73.1). Another such methodological issue was oversampling the dataset before splitting it to train test sets. Doing this will change the natural distribution of data in the test set as well which is not accepted. Authors have mentioned the error due to this and by reverting this error, they have observed that the F1 score has dropped from 90 to 70. Finally, they have tested whether these solutions get generalized properly for other datasets in the same domain. As per results for the hateful class, the F1 score has dropped from 70 to 21.1 in the first study and 75 to 21.6 in the other study proving their poor performance.

Finally, the authors have stated that even small issues may mislead researchers into over-optimistic conclusions. Further, they recommend that the extremely high performances should be better analyzed carefully regarding its methodology, implementation, and evaluation.

In their study [10] Ziqi et al, have combined LSTM and CNN architectures to introduce a novel deep learning-based method for hate speech detection. They have identified that existing studies have issues in several ways, such as the absence of relative assessments which makes the assessment of the involvement of individual work difficult. Here the authors have evaluated the proposing approach in contrast to numerous reference points and state-of-the-art on the major collection of seven datasets.

For a given tweet in the dataset, they have applied a series of pre-processing steps to make them normalized. It includes removal of symbols, normalizing hashtags, converting to lowercase, stemming, and removing tokens with document frequency less than 5. These pre-processing steps have reduced the vocabulary size and it has fixed the sparsity in word-based feature representations to some extent. In their proposing CNN + LSTM architecture, the first layer is a 300-dimensional word embedding layer. Then to avoid the model getting overfitted to the training set, they have added a 100x300 dropout layer with a dropout rate of 0.2. The output of this layer has fed to 1D convolutional layer with 100 filters through a window size of 4 having rectified linear neuron units. The downsampling has done by max-pooling to reduce the dimension shape to 25x100 which has fed to the LSTM layer in the next step. It treats extracted feature dimensions as timesteps and outputs hundred hidden units per timestep. The global max-pooling layer follows to flatten the output space by taking the maximum value in each timestep dimension, creating 1×100 vectors.

As the final layer, a SoftMax layer has been used to predict the probability in each class using the categorical-cross-entropy as the loss function.

For the comparative analysis, they have implemented several baseline models applied in prior studies such as linear SVM to be trained on all datasets to compare the performance. To compare the results, they have reported the micro-average precision, recall, F1 over all classes in each dataset. As per the results, the authors have observed that CNN+LSTM models learn better when the pre-trained word embeddings are used. Compared to the baseline models, their best performing model achieves the highest F1 on 6 datasets out of the seven datasets they have used which proved their solution is well generalized. Moreover, even the same word embedding has been used with the baseline neural network models, they have observed that the proposed CNN+LSTM model has a better F1 score. The authors state that this observation has occurred due to the usage of the drop-out and the global max-pooling layers. As per the future directions, they suggest improving this architecture by tuning it further. In addition to that, they suggest incorporating user-centric features for better classification performance.

2.4 Studies Carried Out for Other Languages

Since posting abusive comments on social media is a critical problem, most of other languages also have started using text mining and machine learning-based approaches for those languages. A similar study [29] has been done for Hindi-English code-switched language using a CNN based approach. Apart from the solution, the authors present their novel dataset based on tweets, including tweets in Hindi-English code-switched language labelled into 3 categories namely hate-speech, abusive and non-offensive.

Before training the CNN model with the created dataset, they have applied a series of data pre-processing steps that include, removing punctuations/URLs, replacing hashtags with plain text, converting to lower case, replacing emojis, etc. The proposing approach suggests using a Ternary Trans-CNN model including transfer learning that consists of three layers of Convolutional 1D layers with a filter size of 15,12 and 10 respectively and a kernel size of 3. The final layers are dense layers with the size of 64 and 3 units activated by ReLU and SoftMax respectively. To overcome the issue of the neural networks that get overfitted quickly, a dropout technique has been used while training the model for 25 epochs with a batch size of 128.

The authors have evaluated the constructed model concerning F1 score, precision, and recall relative to the macro metrics as there is no significant class imbalance. The results have indicated that when transfer learning is introduced, the performance of

the model has increased significantly. Moreover, the authors state that by introducing techniques like gradient boosting, the neuron network can be further optimized.

A similar study [30] on the Hindi language has been done using the faxtText classification model for a Devanagari Hindi Offensive Tweets (DHOT) data corpus. The authors have shared a survey among university students and have created a list of swear words. Then they have used these words in this list as a seed words to extract tweets via the Twitter API. They assume that the tweets that contain these words should be abusive. To introduce non-abusive data, they have pulled tweets from popular and trending hashtags. After creating the corpus creation, annotating the dataset has done by three Hindi language experts. Then the annotated corpus has been sent through the primary pre-processing steps including converting to lower case, stop words removal, etc. To process the dataset, the word2vec feature modelling method has been used with the character n-gram technique. All the parameters in the fastText classification model were tuned properly such that the model performance gets optimal. The authors haven't evaluated the model formally with matrices such as precision, recall but interpreted the results of prediction. Although the authors claim that the model has good performance, the data collection approach of this approach is not suitable as it has a fixed systematic process based on the seed words they used. Hence, the model trained by this data will not generalize properly.

Another study [31] has done on hate speech detection on Hindi language but from Hindi and English code-mixed tweets using deep learning models. Compared to prior studies, all of them have focused only on the Hindi language although people use the mix of two languages in a single post as well. Authors have compared three different deep learning models using domain-specific embeddings while evaluating with a benchmark dataset of English-Hindi code-mixed tweets. They have observed that using domain-specific embeddings can improve the representation and it has led to an improvement of around 12% in F-score compared to prior statistical classifiers.

The three different models they have experimented with within this study are 1D CNN, LSTM, and BiLSTM models. In the CNN model, they have used 300 as the embedding dimension with a filter size of 64 using max-pooling while measuring the loss with binary cross-entropy loss. Both LSTM and BiLSTM have been configured with 100 recurrent units having a dropout rate of 0.2 while measuring the loss with the same binary cross-entropy loss. All three models have different characteristics that are specific to each and comparing all helps to identify the most suitable one. For the training of these models, they have created a dataset using Twitter API by searching tweets that contain Hindi cuss words. They have been able to construct a larger dataset containing 255,309 records. However, they have used the entire dataset

for training while using a different dataset created by separate research for evaluation. To construct the word embeddings, they have used the Gensim word2vec model while using Google Translate API to calculate the average Hindi proportion of the collected data.

For the evaluation purpose, they have re-implemented the prior study done on the selected dataset to compare the results of this study with theirs. Authors have observed that out of the three neural network models CNN-1D has shown the best performance with an accuracy of 82.62% and an F-score of 80.85%. This shows a performance improvement in F-score about 12% higher than the prior study that they are comparing with. Although CNN has the highest precision, BiLSTM has shown the best recall. This observation reveals that creating an ensemble of these neural networks can have an even better performance.

Arabic is another such language that is used in a considerable number of countries in the world. Hence, a predictive modeling approach [32] has been proposed to detect the offensive language in online communication for the Arabic language using a Support Vector Machine (SVM) model. YouTube is a social media platform where people can upload content and anyone can add any comments freely to those. Authors have observed that this is a platform where people post abusive content regularly to offend others. Hence, when collecting the dataset, they have used both offensive and non-offensive comments that are posted on certain YouTube channels. The collected data were annotated manually by three annotators from three different Arab countries making sure that the labeling process is not biased to a certain country as the Arabic language is used across multiple countries.

Compared to other languages, Arabic is special being a Semitic language, and phrases are organized from right to left. However, techniques such as tokenization, filtering, and normalization remain the same for Arabic same as for other languages. One of the key challenges related to Arabic is that some Arabic letters are alike phonetically leading users on social media to misspell words by using incorrect but phonetically similar letters. For the baseline model, they have trained the SVM model without including any pre-processing steps. Then they have done few experiments with pre-processing including normalizing, stopword removal, stemming, etc.

As per the results, the best performance was scored when pre-processing steps were included with stemming. However, introducing n-gram features with pre-processing has reduced the model accuracy. This behavior could occur due to certain characteristics of the language as in most other languages performance was improved by introducing n-gram features.

A similar study [3] has been done for the Arabic language but using deep neural networks such as CNNs, RNNs, and their variants with gated recurrent units (GRU). Moreover, they have evaluated BERT in the classification job by using the pre-trained BERT model and then refining the model on the hate speech classification task. In addition to the classification model due to the limited resources available to this language, this paper also describes a novel approach for creating a hate speech dataset for covering racist, Arabic, religious, and ideological hate speech.

The dataset has created using Twitter stream API by using keyword scanning and thread-based search for data for 6 months. In the keyword scanning, they have made sure that they include different religions and tribes to make an unbiased dataset. The annotation process has done by crowdsourcing by a well-defined process. The prepared dataset has abusive comments from different categories such as racist, regional, tribal, inter-religious, and ideological. In the data pre-processing step, authors have removed hashtags, stop words, punctuation marks and have replaced emojis with their corresponding descriptions. Finally, tokens are normalized and lemmatized to their base form.

In this study, four different neural network models have been evaluated namely Convolutional Neural Network (CNN), Gated Recurrent Unit (GRU), CNN + GRU, and Bidirectional Encoder Representations from Transformers (BERT) model. Their CNN model has five layers including an input embedding layer, a convolution layer, a pooling layer, a hidden dense layer, and finally the output layer for binary classification. In addition to these, they have used a dropout layer as well to reduce overfitting. GRU network has four layers including the input embedding layer, the GRU layer, the hidden dense layer, and finally the dense output layer. By combining these architectures, they have created their next architecture CNN + GRU including 6 layers consisting of an input layer (embedding layer), a convolution layer with 100 filters and a 4 sized kernel, a max-pooling layer, a GRU layer, another max-pooling layer, and finally the output layer. The special characteristic of this network is that the CNN layers act as feature extractors for the GRU layers. The final model is the BERT model which has shown dominant performance in NLP-related tasks. BERT is a pre-trained model on monolingual corpora in 104 languages including Arabic. However, in this study, the authors have re-trained it by adding a binary classification layer to classify the tweets into hate or non-hate.

For each different model, they have conducted different experiments by training them using the same pre-processed dataset. After evaluating the performance of each model on different metrics, they have observed that all the models have scored promising results. However, the CNN model has outperformed other models, with an F1-score of 0.79 and AUC of 0.89. Moreover, they have pointed out that BERT has

failed to advance over the baselines and the other evaluated models due to its pre-trained nature on different datasets.

Another such study [33] has been done in Vietnam to detect hateful text in the Vietnamese language by comparing traditional machine learning models and neural network models. The authors have identified that, although there are many approaches proposed to this problem, it still requires further research. In the study, the authors have done several experiments on four different classifiers for the same dataset. After analyzing the experimental results, they have identified the best model that is appropriate for this problem stating its advantages and disadvantages. Finally, the authors have proposed the possible research areas in this domain for more improvements.

Their dataset has 20,345 comments/posts fetched from Facebook and they have been annotated into three classes namely clean, offensive and hate. Unlike in other studies, they have clearly defined the meaning of the three classes. After annotating data, they have observed that their dataset is unbalanced as there were much more data points in the clean class compared to the other classes. For the word embedding, they have used a Vietnamese word embedding done by a separate study which has trained on Wikipedia and Common Craw using CBOW with position-weights and in a dimension of 300 with character n-grams. For the traditional models, they have used Logistic Regression (LR) model and Support Vector Machine (SVM) model while Text-CNN and GRU models are used as deep neural networks.

In the data pre-processing phase, they have removed special characters such as icons, emoji, URL links, hashtags, and digits. Moreover, they have changed words to lower-case before tokenizing to reduce the number of unique tokens. The logistic regression model has trained with a balanced class weight with the inverse of regularization strength of 1. Similarly, SVM model has trained with a linear kernel and 1 as the regularization parameter. The architecture of the Text-CNN has an embedding layer and four layers. The embedding layer consists of embedding of size 300 and 11,221 maximum features. Each convolution layer has 32 filters and they use ELU (Exponential Linear Unit) as the activation function. Finally, the output layer has a Dense-three layer including sigmoid activation function to classify the three output classes. This model has an embedding layer with an embedding size of 300 and a GRU Bidirectional RNN layer with 80 units. The final output layer is a Dense layer with 3 units activated by a sigmoid function, making predictions to the three classes. Their final model which is the GRU model includes an embedding layer with a size of 300. Further, the bidirectional RNN layer has 80 GRU units. Similar to the CNN architecture output layer is again a Dense 3 layer with a sigmoid activation function for three classes.

After evaluating the models, they have observed that among the traditional models, Support Vector Machine has scored the best performance having a 65.1% F1 score. From the deep neural networks, Text-CNN has become the best model scoring 83.04% F1 score. When comparing all four models, it can be seen that deep neural networks outperform traditional models. Moreover, after analyzing the results, authors have found that profanity words and personal pronouns have a large impact in determining offensive and hateful texts. Hence, they recommend future studies to focus on those areas to improve prediction accuracy.

A similar study [34] on the Italian language which is one of the most popular languages apart from English has done by Valentino et al. on creating hate speech detection in social media content. In this paper, the authors present a system named HSD4I PG developed by a joint team from two universities including the architecture of the system, software components, experiment results, and finally future research directions.

Their solution consists of a tokenizer for the Italian language, the FastText tool for word embedding, a feature generator that generates a vector of numeric features, and finally a trainable classifier. In addition to that, they have utilized the Ita_Twitter corpus which includes 1,234,865 tweets as the dataset. As the lexicon, they have used a publicly available Italian monolingual dictionary named *Italian Lexicon of the Hate Speech* while using *Sentix* Italian lexicon for sentiment analysis. After applying a series of preprocessing steps such as fixing misspellings, replacing digits as NUM, etc, they have trained the word embedding model by FastText using the Ita Twitter corpus. Finally, numerical features are calculated by combining the aggregated FastText features and generating 20 additional extra features such as the number of hateful tokens, number of web links, number of mentions, etc. These features have been used to train their machine learning model which is a Support Vector Machine (SVM).

As per the results, the proposing approach has obtained significant results having competitive scores compared to the other approaches. Authors have pointed out that it is important to notice whether the hate annotation is objective or subjective as certain posts in the datasets look to be difficult to annotate correctly even for a human being. Hence, they state that different people can produce different annotations for the same post.

A study [35] on presenting a new dataset and baseline evaluation on the Bengali language regarding hate speech detection to address a variety of challenges including finding an appropriate model, handling small imbalanced data sets, and the choice of

the feature analysis method. The authors present a new dataset containing 30,000 records collected by YouTube and Facebook comments that are labeled by crowdsourcing. Moreover, they have carried out baseline experiments and several deep learning models as well with Bengali word embedding such as FastText, Word2Vec, and based on this dataset to support future researches.

When creating the dataset, they have considered a variety of topics such as sports, entertainment, crime, etc to create an unbiased dataset. To extract hateful comments, they have searched YouTube videos and Facebook posts that are related to controversial events and used their comments. After extracting, they have removed all the irrelevant comments such as comments written only in English. As hate speech is a very subjective topic, when annotating the dataset, the authors have defined a few strict rules to make sure all annotators are following the same definition. With the given instructions, they have used 50 annotators to label 30,000 comments. Each comment has been annotated by 3 annotators and the majority voted label has been selected as the final label. The final dataset has included 10,000 hateful content and 20,000 non-hate content.

When pre-processing data, they have followed several steps such as removing emojis, punctuations, numerical values, non-Bengali alphabet, etc. They have used three-word embedding models which are Word2Vec, FastText, and BengFast for word embedding. To create the Word2Vec model they have used the CBoW method and to create the FastText model, they have used the skip-gram method using the 30k constructed dataset while keeping the embedding dimension to 300.

In the model construction step, they have used three major models that have shown prominent results in this domain which are Support Vector Machine (SVM), Long Short Term Memory, and Bi-directional Long Short Term Memory. In the SVM model, they have used the linear kernel setting while using 100 LSTM layers in the LSTM model with a 0.2 dropout rate to tackle the overfitting problem. Similarly in the Bi-LSTM, they have used 64 Bi-LSTM layers with the same 0.2 dropout rate. All the models have trained on the 80% training set and evaluated on the rest of the dataset by measuring the accuracy and F1 score.

Surprisingly, compared to studies done on other languages, the SVM model has outperformed all the other neural network models with a considerable gain in performance. On the error analysis, the authors have observed that the model has predicted certain content as hate speech although they are not. The reason for this behavior has occurred due to having aggressive words that are normally used in hate speech, but in these cases, they were not used with such offensive intentions. Hence

authors have concluded that the Bengali language can be very complicated in the context which will be a challenge for machine learning models to understand.

3 METHODOLOGY

This study focuses on evaluating the performance of deep learning models compared to the existing studies to observe whether it is possible to develop a more robust and accurate solution for the Sinhala language. A dataset containing Sinhala Tweets has been collected via the Twitter API as depicted in Figure 3 and annotated manually to train the models and evaluate their performance. The dataset has been cleaned and standard NLP techniques such as stop words removal/stemming/lemmatization etc have been applied using the tools and resources that are available publicly to obtain the best results. To have context awareness, in addition to the content of tweets, features such as the author description, users' status count, retweet count, favourite count, etc have been captured according to prior studies done in other languages. Multiple deep learning architectures have been trained on the prepared data. In addition to that, existing approaches also has implemented on the same dataset to compare the performance among them. As the next step, a deep learning ensemble has been implemented according to Figure 4 as mentioned in literature for other languages to evaluate how it can affect the performance. Further, the study has experimented with additional features to evaluate whether having extra features can improve performance. Finally, all these approaches have been compared with results and the best model has been applied to a completely new dataset to validate the model's generalizability.

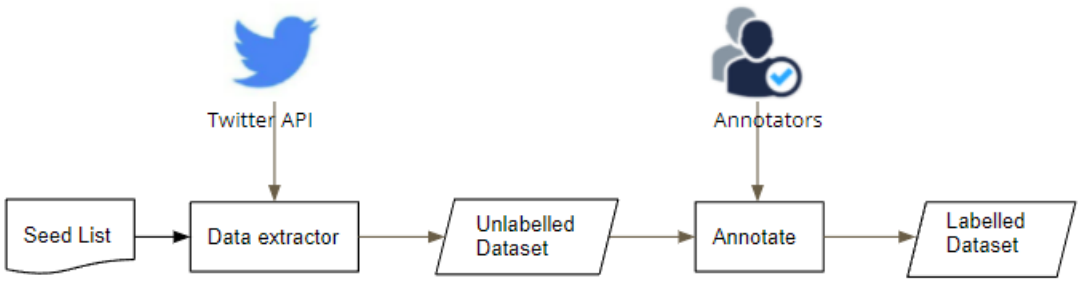


Figure 3: The overall data collection process by fetching Tweets via Tweet API

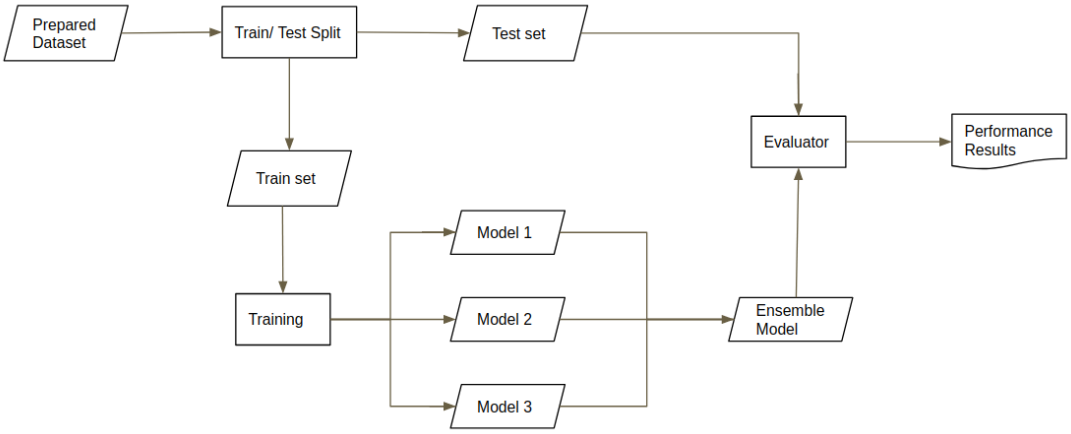


Figure 4: The ensemble model construction process with the majority vote

3.1 Data Collection

Solving this kind of supervised machine learning problem requires a labelled dataset stating whether a given record is hateful or not. However, for the Sinhala language, there are not enough publicly published datasets with sufficiently large data points. Hence in this study, a labelled dataset was created and published as a contribution to the research community by following the process depicted in Figure 3. Twitter API was used to extract tweets belonging to these two classes, as same as done in most of the prior studies. In addition to the content of the tweet, extra information such as the author's description, user's status count, retweet count, favourite count, etc was also extracted to have contextual meaning. A set of seed words list was used to obtain hateful records. As the study is focused on deep learning techniques, a larger number of records compared to existing datasets were collected to take the advantage of the characteristics of a deep learning model.

After collecting data, the annotation process was conducted by three independent annotators to have an unbiased dataset. Out of the three votes by the three annotators, the majority vote was considered as the final label of the data point.

3.2 Dataset Description

The collected dataset contains 4508 records and after dropping the null rows it was reduced to 4491 records. The dataset contains 43 columns as features collected from the Twitter API response and the following Table 4 depicts the most informative fields that are relevant to this study.

Table 1: Dataset Description

Feature	Description
Place	Place if associated with Tweet
in_reply_to_screen_name	The display name of the original Tweet's author when presented in a reply
Id	Unique id
retweeted	Retweet status
Source	Source used to post the Tweet
is_quote_status	Quoted Tweet status
reply_count	Count of replies to the Tweet
in_reply_to_user_id	If the Tweet is a reply, this field will include the author id of Tweet
possibly_sensitive	Whether the Tweet contains a link
retweet_count	Count of retweets
full_text	Tweet text
created_at	Tweet created UTC time
Label	Class label
Truncated	Truncated status
Favorited	Favourite status
favorite_count	Count of liked by Twitter users
quote_count	Quoted count
full_text_without_emoji	Full text excludong emojis
emoji_count	Number of emojis included in the Tweet text

3.3 Data Pre-processing and Preparation

Text data consists of words, sentences, and paragraphs in an unstructured manner. Due to this nature of textual data, it is very hard to work with raw data for machine learning models as well as for humans. On the other hand, data pre-processing and preparation is so important because it needs to convert the unstructured text into a numerical format such that it can be fed into a machine learning model. Hence several effective data pre-processing methods have been used in this research to build this form of data and construct machine learning or deep learning models

3.3.1 Tokenization

Tokenization splits an unstructured text into individual reduced units. In other words, it is the process of splitting text into minimal meaningful units. A token is a single entity that is the fundamental block for a sentence or paragraph. It is not possible to feed sentences or paragraphs directly into any machine learning model. Thus, it is a mandatory step before any kind of processing. There are many types of tokenization techniques available such as,

- Text into sentences tokenization
- Sentences into words tokenization

The appropriate method of tokenizing should be decided based on the application. In this study sentences into words tokenization method was used since word tokenization is a crucial part of the text to numeric data conversion. This numerical conversion of text data is essential to feed those text data into the machine learning models. The output of the sentence tokenizer is a list of words. Here "word" is considered as the most basic building block in modelling a review. When dealing with multiple languages, tokenizers that are specially optimized for a certain language should be used in order to capture the rules in a particular language. Tokenizing is the most fundamental step in Natural Language Processing. But it has a huge impact on the accuracy of the model since the constructed tokens are essential to identify the patterns in the text.

3.3.2 Removal of stop words

Stop words are the words that are used frequently in the language such as "සහ", "සමඟ", "අනේ". These words are used to infer certain details in a more explainable manner. However, these words do not contain much information. Moreover, since these words are more frequent, existence of these words will make a huge impact on the storage requirements of data. In addition to that, this will add an overhead to performance as well due to the existence of a lot of tokens. When there are too many tokens, the model will need to go through each and every word and process. Thus, in most Natural Language Processing applications, these stop words are removed. Removing these words will help the model to focus more on the other

more informative words. Moreover, this will help the model to learn fast and predict fast. However, the removal of stop words is not always recommended. Even though the stop words do not have a meaning, when it is applied to a sentence or a phrase, it has a semantic meaning. This decision of remove stop words or not will depend on the application. In this study, a pre-defined set of stop words were used due to the lack of available tools.

3.3.3 Stemming

The role of a stemmer is similar to the lemmatizer. The goal of adding a stemmer is same as lemmatizer to bring the words in different forms to the base form. The difference between the stemmer and lemmatizer is, stemmer creates the base token by stripping out few characters from the token end and it does not use the dictionary form of the word. Due to this, the result of a stemmer might not an actual word in a language. Still stemming is used in data pre-processing since it has the capability to improve the performance of machine learning models. In this study, a pre-defined mapping of words to its stemmed form is used for the stemming purpose due to the lack of available tools.

3.3.4 Data Shuffling

Some datasets have sorted data such that similar data points are closer to each other. In that case, while training the model, it will learn specially for those data points and it will not be generalized for all possible scenarios when used in real-world applications. Shuffling data helps to reduce variance and construct generalized models. Moreover, this will verify that the training/test/validation sets contain the overall distribution of the data and not a set of special cases.

3.4 Feature Engineering

Feature engineering is a very important and crucial step in any classification problem since it is not straightforward, and needs a lot of data analysis. Feature engineering intends to find out new features or attributes that can help the classification task. Here, additional features were engineered from the existing review text to understand its impact on the classification task according to the findings from the preliminary analysis phase. There were several attributes that can identify differences in offensive and non-offensive reviews. Hence, those identified significant features were used as new features in the hate review detection process.

3.4.1 Emoji Count

The emoji count feature was calculated by counting the occurrences of emojis added in a Tweet. As per Figure 5, it can be seen that the emoji count in non-offensive Tweets is slightly higher than in offensive Tweets.

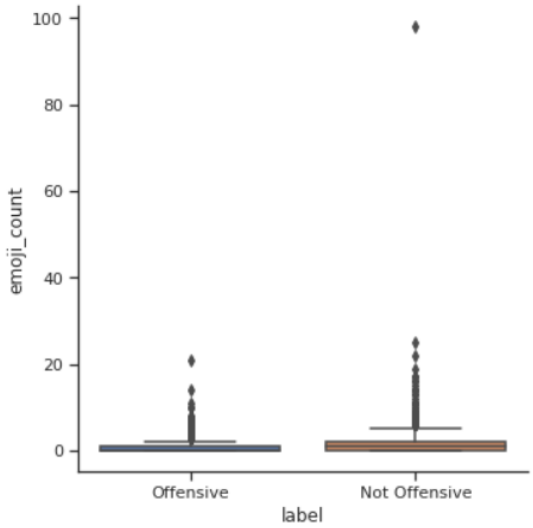


Figure 5: Emoji count vs class label revealing a higher emoji count in non-offensive Tweets

3.4.2 Tweet Length

Tweet length feature was constructed by counting the number of words in the Tweet. As per Figure 6, it can be seen that offensive Tweets usually have a shorter review compared to non-offensive Tweets.

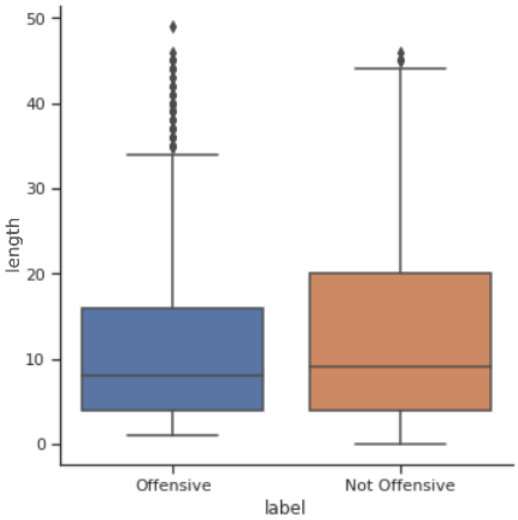


Figure 6: Tweet length showing that the offensive Tweets have a shorter review length

3.5 Exploratory Data Analysis

Before training the models from the collected dataset, an exploratory analysis was conducted to get more insights about the available dataset. Here different visualization techniques were carried out on the dataset to investigate and summarize the main characteristics of the dataset to discover patterns, check anomalies, and check assumptions.

3.5.1 Reply Count

The **reply count** attribute shows a relationship to the class label indicating that the Tweets that have a large reply count tend to not be offensive as depicted in Figure 8. Usually, Tweets which are not offensive get more attention from its audience and people tend to reply and respond to them compared to offensive Tweets which could be the reason behind this observation.

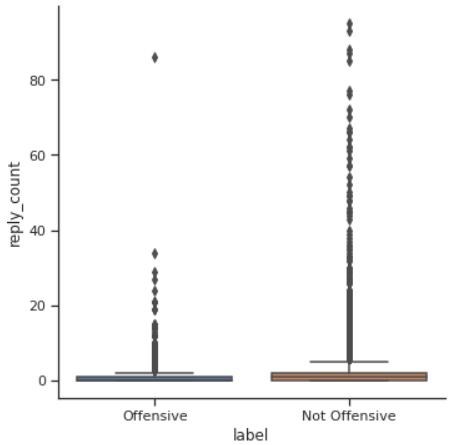


Figure 7: Reply count depicting that non-offensive Tweets have considerably large reply counts

3.5.2 Retweet Count

The same behaviour can be seen around the **retweet_count** as per Figure 9 where the larger number of retweets were there for not offensive Tweets compared to offensive Tweets since offensive content is usually rejected by society.

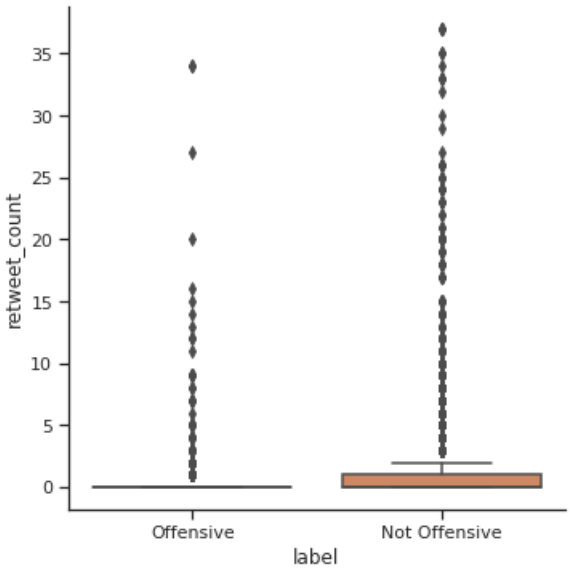


Figure 8: Retweet count showing a higher count for non-offensive Tweets

3.5.3 Possibly Sensitive Editable

This feature has defined as a boolean variable that reflects whether the Tweet contains a link or not.

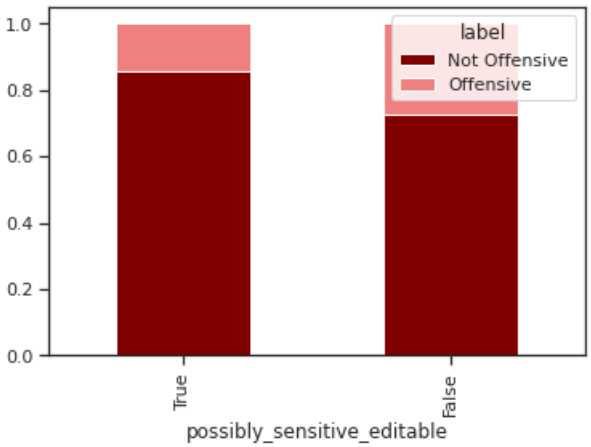


Figure 9: possibly_sensitive_editable revealing that a large proportion of offensive Tweets when the Tweet doesn't contain any links

As per Figure 10, it can be seen that, either the feature value is true or false, majority is not offensive. This observation happens since the dataset has a majority of not offensive class. However, it can be seen that, there is a large proportion of offensive Tweets when the Tweet doesn't contain any links.

3.5.4 Is Quote Status

is_quote_status feature Indicates whether the Tweet is a Quoted Tweet or not as a boolean variable. However, this variable does not show a significant relationship to the class variable as per the following Figure 11, since both proportions are almost similar for both possible values.

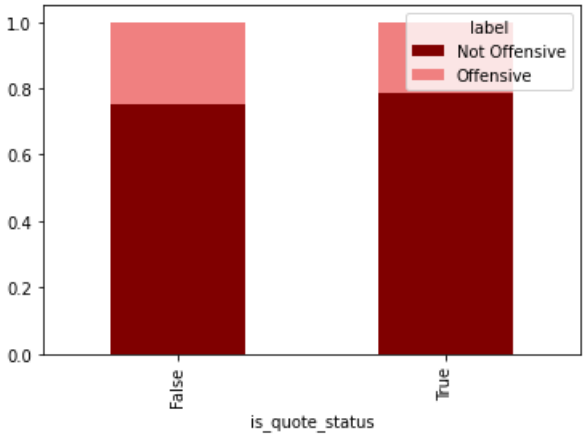


Figure 10: is_quote_status depicting that it doesn't show a significant relation to the class label

3.5.5 Favourite Count

Favourite count feature reflects how many times the Tweet was marked as a favourite. As per the Figure 12, it can be seen that, when the favourite count is high, it tends to be not offensive. Moreover, the majority of the favourite count for offensive Tweets is zero.

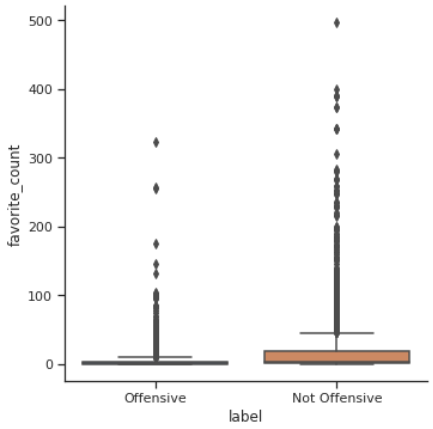


Figure 11: favourite_count showing that the majority of non-offensive Tweets have a higher favourite count

3.6 Model Construction

As this study is mainly focused on evaluating the performance of deep learning in the context of hate speech in the Sinhala language, several deep learning models that have shown proven performance in other languages were used here. This includes a CNN, LSTM, and a BiGRU model. The ensemble of deep learning models is constructed by taking the majority vote of these models. In addition to that, the traditional machine learning models such as “Naïve Bayes” and SVM models have been implemented by training on the same dataset to compare the performance.

3.6.1 Convolution Neural Network (CNN)

Convolutional Neural Networks have shown breakthrough results in image processing tasks initially. In the same manner, 1D CNNs have shown prominent performance in NLP tasks including text classifications. A Convolutional Neural Network usually involves two operations known as **convolution** and **pooling**. These convolution and pooling layers are applied one after other to create the deep learning model followed by a fully connected layer at the end as per Figure 13.

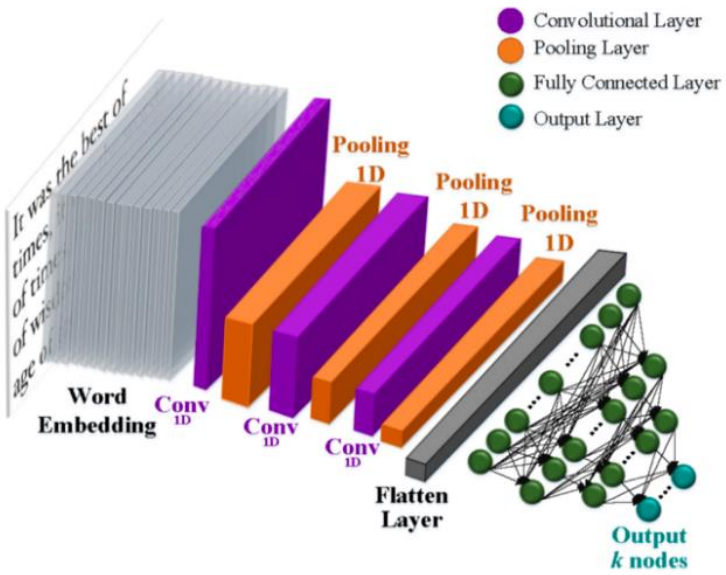


Figure 12: Architecture of a CNN having embedding, convolution, pooling, flatten and output layers

Convolution Layer

The convolution layer is to convert the input to extract features and differentiate them correctly. This can be accomplished by convolving the image with a kernel. A kernel is specific to extract certain features. It is possible to apply several kernels to the same input to extract multiple features. Typically, an activation function applies to the convoluted values to grow the non-linearity. This is the special feature of deep learning models which is the ability to identify features by the model itself.

Pooling layer

Pooling layers help to reduce the sizes of the feature maps when it is going through the network. Hence, it decreases the number of constraints to learn by the network and the computation effort needed to execute by the network. The pooling layer recaps the features available in a region of the feature map constructed by a convolution layer. Hence, further operations are executed on summarised features instead of precisely positioned features constructed by the convolution layer. This produces the model to identify more complex features and more robust to disparities in the position of the features in the input. There are different types of pooling types such as max pooling, average pooling, and global pooling. These series of convolution and pooling layers help to identify the features and the dense layers at the end of the network help for the predictions later.

In this study, the CNN modal was constructed with an embedding layer, one convolutional layer, one max pooling layer, a flatten layer followed by two dense layers. The embedding layer was configured with a 17532-vocabulary size, 60 length of input sequence, and a 60 dimension of the dense embedding. The one-dimensional convolution layer was configured with kernel size of 3 and 32 filters activated by Relu activation function. Max pooling layer was configured with a pool size of 2 with two strides. After that, the flatten layer was added to flatten the multi-dimensional input tensors into a single dimension before feeding into the dense layers. The immediate dense layer contained 250 units activated by Relu function and the final dense layer contained 2 units activated by the sigmoid function to make classifications for two classes. When training the model, Adam optimizer was used with a learning rate of 0.001 and a decay of 0.0001 while the loss was measured with binary cross entropy.

3.6.2 Long Short-Term Memory (LSTM)

LSTM is a type of Recurrent Neural Network in Deep Learning that has been precisely advanced for the use of handling sequential prediction applications such as text classification, weather forecasting, stock market prediction, product recommendation, etc. The special characteristic of LSTM compared to other deep learning models is that it has a special type of neuron called memory cells. These

cells contain weights and gates. There are 3 gates inside of every cell which are the input gate, the output gate, and the forget gate as depicted in Figure 14. These weights and gates control which information should be kept in memory and which information should be dropped off.

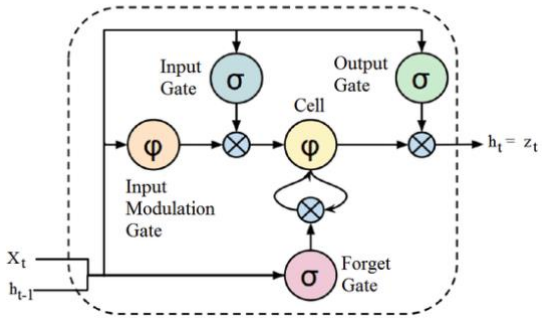


Figure 13: Structure of an LSTM unit having input, input modulation, output, and forget gates

The structure of the models can be identified as a series of these cells connected in a serial manner which is very similar to the Hidden Markov Model (HMM) as shown in Figure 15.

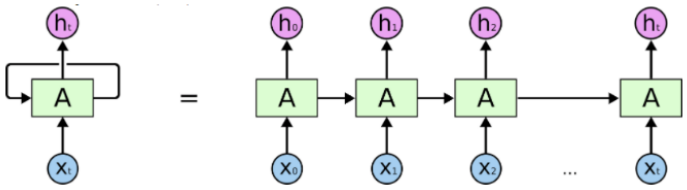


Figure 14: Structure of a set of LSTM units connected in a serial manner to handle sequential data

One of the main benefits of LSTM is the tactlessness to gap length. RNN and HMM depend on the hidden state before sequence/emission. If it is required to predict the sequence afterward at 1,000 intervals instead of 10, the model forgets the starting point by then while LSTM can remember due to its special cell type. This property allows the LSTM model to perform more accurately with sequential data.

In this study, a simple LSTM architecture was constructed with an embedding layer, LSTM layer followed by Dense layers. The embedding layer was configured as same as in CNN with a 17532-vocabulary size, 60 length of input sequence and a 60 dimension of the dense embedding. LSTM layer was configured with 600 units activated by tanh function while sigmoid was used for recurrent activation. The final dense layers have 250 and 2 units respectively activated by Relu and Softmax. The same Adam optimizer with 0.001 learning rate with 0.0001 decay was used while using categorical cross-entropy to calculate the loss.

3.6.3 Bidirectional Gated Recurrent Unit (BiGRU)

A Bidirectional GRU is a sequence processing model that contains two Gated Recurrent Units (GRUs). A GRU is a kind of RNN similar to LSTM, but only has two gates called an update gate and a reset gate and remarkably lacks an output gate. BiGRU is a bidirectional recurrent neural network with only the input and forget gates.

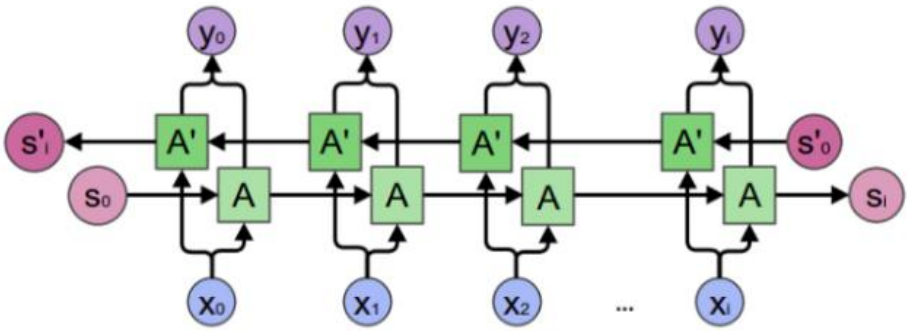


Figure 15: General structure of a RNN having an update gate and a reset gate

Figure 16 depicts the general structure of a recurrent neural network. Here, when every A and A' replaced with a gated recurrent unit yields a bidirectional GRU. In a Bi-GRU neural network of each layer, the forward layer computes the output of the hidden layer at each time from forward to backward and the backward layer computes similarly in the other direction. This allows for the use of data from both prior time steps and latter time steps to produce predictions on the present state.

The BiGRU model constructed in this study has an embedding layer, a GRU layer, a one-dimensional layer followed by flattened and dense layers. The embedding layer has a 17532-vocabulary size, 60 length of input sequence, and a 60 dimension of dense embedding. The GRU layer has 256 units and the return sequences parameter was enabled to make the model bi-directional. The max-pooling layer was configured with a pool size of two with padding enabled with the stride of two. The dense layers were configured similar to CNN and LSTM with 250 and 2 units with Relu and Softmax activation functions. Models were trained with Adam optimizer while the loss was measured with categorical cross-entropy.

3.6.4 Ensemble of Deep Learning Models

Present, deep learning neural network models with multilayer processing architecture show better results in contrast to the traditional classification models. However, since neural network models are nonlinear, they can be lead to have a high variance producing unexpected predictions. Due to this nature, they can learn complex nonlinear associations easily in the data but at the same time, they become sensitive to initial conditions such as initial random weights and the nature of the training dataset.

Ensemble learning is a technique that combines multiple individual machine learning models to obtain a better performance in terms of generalizability. Merging the predictions from multiple neural networks introduces a bias that reduces the variance of a single trained model. In addition to decreasing the variance in the prediction, the ensemble can also produce better robust predictions compared to any single best model. Deep ensemble learning models bring the benefits of both the deep learning models along with the ensemble learning such that the ultimate model has improved generalization performance.

4 EVALUATION

Model evaluation is one of the key steps in the process of building a prediction model. Model evaluation helps to quantify the capability of the model to perform well with unobserved examples. Further, it measures how precisely the model can accomplish for unseen data. In this study, all the constructed models were tested carefully to confirm that the model is suitably fitted to the training dataset without overfitting or underfitting. Models can be evaluated by comparing to the ground truth data and associating with the model predictions. The simplest approach is to split the dataset into two subcategories called as a training and testing dataset with 80% and 20% portion of the original dataset. 80% portion will be used as the training set while the other unseen portion is used for testing.

4.1 K-Fold Cross-Validation

Train/test split evaluating method cannot detect the problem of overfitting in machine learning. Due to that, the trained model can lead to be less precise on unseen data. The cross-validation evaluation technique can be used to detect whether a trained model has overfitted or not. In this study 10-fold cross validation technique was used to evaluate the model. Here the data gets divided into ten subsets such that each time, one of the subsets gets utilized for testing while the other nine subsets get to construct the training set.

4.2 Accuracy

Accuracy defined in the below equation is the ratio between correct prediction and total predictions which the model has acceptably classified. This is a decent measure for the evaluating majority of the binary classification models for datasets that have equally distributed labels. Since this dataset was not skewed, accuracy can be treated as a good measure of the model performance.

$$\text{Accuracy} = \frac{\sum TP + \sum TN}{\sum TP + \sum TN + \sum FP + \sum FN}$$

4.3 Precision

The precision of a model is the ability to identify only the relevant data points which are expressed by the following equation.

$$\text{Precision} = \frac{\sum TP}{\sum TP + \sum FP}$$

Basically, this will measure among the predictions that the model predicted as true, how many of them are correct. If the model precision is low, even the model predicts that a review is hateful, in most cases it could be wrong. In a properly learned model, its precision should be high.

4.4 Recall

Recall refers to the percentage of total relevant results correctly classified by the model which is expressed by the below equation.

$$\text{Recall} = \frac{\sum TP}{\sum TP + \sum FN}$$

Basically, this will cover among the all-positive data in the dataset, how many of them were able to predict as positive by the model correctly. If the recall is low, even though there are hateful tweets in the dataset, the model might not predict them as hateful correctly. Thus, in a well-trained model, the recall should be higher.

4.5 F-Score

Precision and recall measures, two aspects of a model regarding its performance. In an ideal model, both of these measurements must be high. However, there is a trade-off between precision and recall when training a model.

As an example, when a model has a high recall, the number of predictions that it will classify as positive will get higher. Although getting a higher recall is good, this is affecting the precision of the model badly. When the positive predictions get high, the chance of getting a false positive gets higher as well. Ultimately this high recall will cause a lower precision.

Finding the best values for precision and recall depends on the application of the model. Applications like cancer prediction, the precision must be high. Otherwise, people who do not have cancer will be informed that they are infected by cancer.

But since there is a trade-off between these two measurements, it is difficult to come up with a proper model when there are two different values. Thus, a new measurement called as the F-score was calculated using the values of precision and recall according to the following equation.

$$\text{F - Score} = \frac{2\ X\ Precision\ X\ Recall}{Precision + Recall}$$

4.6 Receiver Operating Characteristic (ROC)

ROC curve can be used for evaluating and visualizing the performance of binary classification problems. It is formed by plotting the true positive rate against the false-positive rate at numerous threshold settings as shown in Figure 17. The true-positive rate is also recognized as recall or sensitivity whereas the false-positive rate is known as the probability of false alarm.

$$\text{TPR} = \frac{TP}{P} = \frac{TP}{TP + FN} = 1 - FNR$$

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} = 1 - TNR$$

TPR gives how many truthful positive outcomes arise among all positive samples available throughout the test. FPR gives how many incorrect positive outcomes arise among all negative samples available throughout the test. Each prediction result of a confusion matrix characterizes one point in the ROC space.

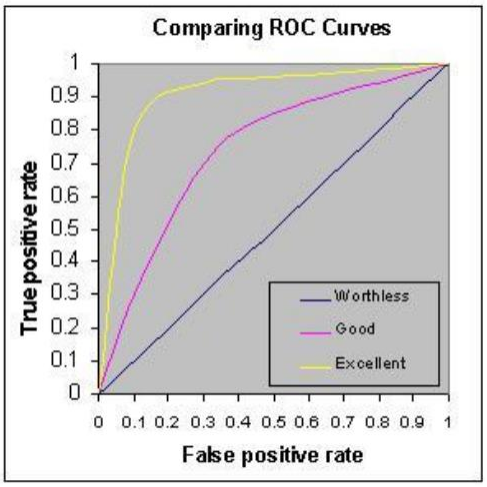


Figure 16: ROC Curve showing the behaviour of curves for different scenarios

4.7 Area Under the Curve (AUC)

AUC measures the complete two-dimensional area below the entire ROC curve as per Figure 18. It quantifies the excellence of the model's prediction capability regardless of what classification threshold is chosen.

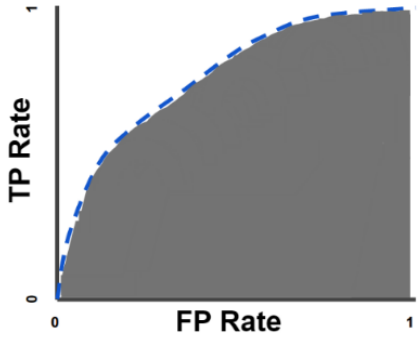


Figure 17: AUC showing the TP rate vs FP rate quantifying the performance of the model by the area.

5 RESULTS

This section presents the results obtained for the proposing approach including the performance for individual deep learning models and for the ensemble of models. In order to compare the performance, the traditional machine learning models that have been used in literature have been used on the same dataset. Finally, to test the generalizability of the model, it was evaluated on a completely new dataset.

5.1 Performance by Deep Leaning Models

The following Table 2 summarizes the performance of individual deep learning models as well as the ensemble of deep learning models in terms of precision, recall, f-score, accuracy, and the AUC matrices with 10-fold cross-validation. As per the results, it can be seen that all models have scored more than 90% in all matrices which implies that the models have trained properly. Since these matrices were calculated with the cross-validation technique, it is possible to conclude that the models are not overfitted as well. The ensemble model does not show a significant performance improvement here compared to the individual models. However, the overall accuracy has improved slightly.

Table 2: Performance metrices for deep learning models

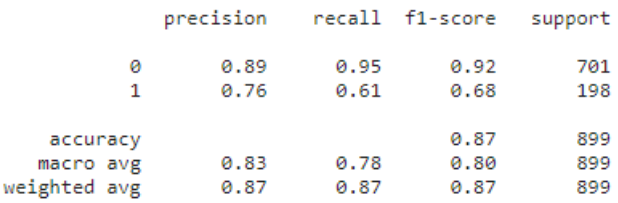
Model	Precision	Recall	F-Score	Accuracy	AUC
CNN	0.901	0.912	0.906	0.911	0.962
LSTM	0.919	0.921	0.920	0.924	0.967
BiGRU	0.928	0.936	0.932	0.946	0.973
Ensemble of Deep Learning Models	0.930	0.901	0.915	0.941	0.970

5.2 Performance by Traditional Machine Learning Models

The performance of traditional machine learning models was taken by evaluating Logistic Regression, SVM, Random Forest Classifier, and Gradient Boosting Classifier on the same dataset trained and tested by the same cross-validation technique. The following Table 3 depicts the performance of those models and it can be clearly seen that the recall of all these models is low compared to the precision which leads to a poor f-score. As per the classification report in Figure 19, it can be seen that the model can classify non-offensive Tweets better and offensive Tweets poorly which leads to an overall poor recall.

Table 3: Performance metrices for traditional machine learning models

Model	Precision	Recall	F-Score	Accuracy	AUC
Logistic Regression	0.884	0.514	0.776	0.861	0.916
SVM	0.836	0.599	0.808	0.872	0.909
Random Forest Classifier	0.778	0.633	0.805	0.865	0.911
Gradient Boosting Classifier	0.760	0.596	0.786	0.854	0.874



	precision	recall	f1-score	support
0	0.89	0.95	0.92	701
1	0.76	0.61	0.68	198
accuracy			0.87	899
macro avg	0.83	0.78	0.80	899
weighted avg	0.87	0.87	0.87	899

Figure 18: Classification report for Logistic Regression model by train-test split

5.3 Performance on the Separate Dataset

The following Table 4 reveals the performance of the same set of deep learning models in the same configuration on the completely new dataset. As per the results, it can be seen that all the models have shown a decent performance. Here the ensemble of deep learning models has shown slightly superior performance compared to other models. Moreover, according to the classification reports in Figure 20, it can be clearly seen that the trained model has the ability to predict both classes correctly.

Table 4: Performance metrices for deep learning models on a new dataset

Model	Precision	Recall	F-Score	Accuracy	AUC
CNN	0.848	0.851	0.849	0.861	0.922
LSTM	0.831	0.838	0.834	0.847	0.906
BiGRU	0.851	0.860	0.855	0.864	0.911
Ensemble of Deep Learning Models	0.857	0.859	0.858	0.867	0.912

	precision	recall	f1-score	support
0	0.87	0.89	0.88	346
1	0.86	0.84	0.85	289
accuracy			0.86	635
macro avg	0.86	0.86	0.86	635
weighted avg	0.86	0.86	0.86	635

Fold 1

	precision	recall	f1-score	support
0	0.89	0.88	0.89	346
1	0.86	0.88	0.87	289
accuracy			0.88	635
macro avg	0.88	0.88	0.88	635
weighted avg	0.88	0.88	0.88	635

Fold 2

	precision	recall	f1-score	support
0	0.85	0.88	0.87	346
1	0.86	0.82	0.84	289
accuracy			0.86	635
macro avg	0.86	0.85	0.85	635
weighted avg	0.86	0.86	0.85	635

Fold 3

	precision	recall	f1-score	support
0	0.88	0.83	0.85	345
1	0.81	0.86	0.84	289
accuracy			0.85	634
macro avg	0.84	0.85	0.84	634
weighted avg	0.85	0.85	0.85	634

Fold 4

Figure 19: Classification report per fold in Ensemble of deep learning models

5.4 Performance with Extra Features

Since the literature states that adding extra features helps to improve the model accuracy, in this study following five features were selected as per the analysis done in the exploratory data analysis.

- Emoji count
- Tweet length
- Favourite count
- Reply count
- Retweet count

Model was trained and tested by adding features one by one creating different feature sets and evaluated the performance of the model in each step.

- set 1 – emoji count
- set 2 – emoji count + tweet length
- set 3 – emoji count + tweet length + favourite count
- set 4 – emoji count + tweet length + favourite count + reply count
- set 5 – emoji count + tweet length + favourite count + reply count + retweet count

Following Figure 21 depict the performance for each feature set and it can be seen that set 2 which includes the emoji count and tweet length has helped the model to obtain the maximum performance.

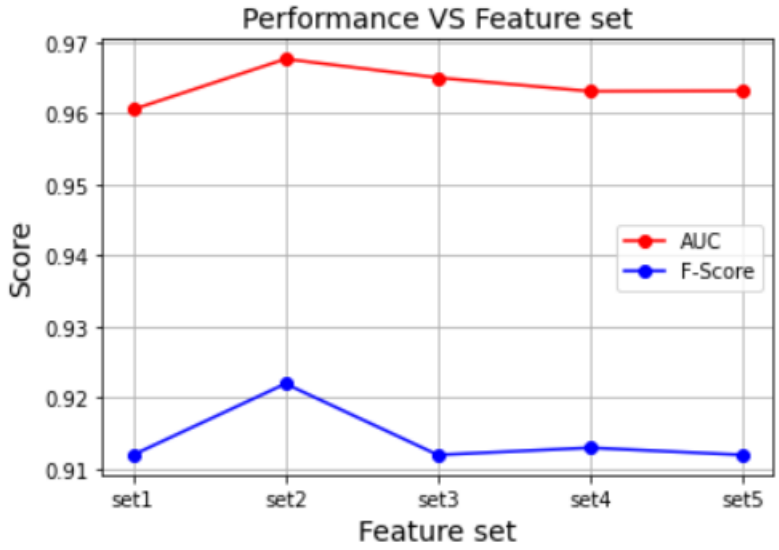


Figure 20:Performance vs feature set with showing highest performance with set 2

When comparing this performance, with the same LSTM model trained without extra features, it can be seen that there is a slight performance gain due to the extra features in terms of precision, recall f score, and accuracy.

6 DISCUSSION

As per the above results, it can be clearly seen that all Deep Learning based approaches have outperformed all the traditional machine learning models significantly. Among the deep learning models, BiGRU model has a slightly superior performance compared to others. The ensemble of deep learning models constructed out of these deep learning models showed a performance increase but not significantly. The reason for this behaviour could be that all the individual deep learning models perform well for this dataset which would not take the advantage of the ensembling technique.

When evaluating the performance of these models, the cross-validation technique was used to check whether the models are not overfitted. Moreover, when the deep learning models were applied to a completely new dataset, it showed an acceptable performance proving that this approach is well generalized.

The experimentation with extra features also showed a slight performance gain. Those features have contributed to improving the precision, recall, f score, and accuracy of the model. In this study, it is not possible to see a huge performance gain due to extra features as the models have been able to classify them correctly even only with the Tweet content. Hence it is recommended to conduct future studies focusing on this area considering datasets that cannot be classified easily only with Tweet body.

7 CONCLUSION

This study presents a deep learning-based approach for hate speech detection in the Sinhala language to solve several problems found in prior studies. In the absence of a sufficiently large dataset to experiment, this study presents a completely new dataset to the research community to further experiment in this area. The results obtained in this study showed that all the deep learning models as well as the constructed deep learning ensemble outperform prior approaches scoring over 90% for all the performance matrices. Moreover, the experimentation using extra features for hate speech detection also has shown promising results. Most importantly, the proposed solution has proven that it is well generalized as it has shown good results for a completely new dataset as well.

8 REFERENCES

[1] P. Badjatiya, S. Gupta, M. Gupta and V. Varma, “Deep Learning for Hate Speech Detection in Tweets,” in *Proceedings of ACM WWW'17 Companion*, Perth, 2017.

[2] S. Zimmerman, C. Fox and U. Kruschwitz, “Improving Hate Speech Detection with Deep Learning Ensembles,” in *Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan, 2018.

[3] R. Alshalan and H. Al-Khalifa, “A Deep Learning Approach for Automatic Hate Speech Detection in the Saudi Twittersphere,” *Applied Sciences*, vol. 10, 2020.

[4] U. Naseem, I. Razzak and I. A. Hameed, “Deep Context-Aware Embedding for Abusive and Hate Speech detection on Twitter,” 2019.

[5] I. Amali and S. Jayalal, “Classification of Cyberbullying Sinhala Language Comments on Social Media,” in *MERcon 2020*, Moratuwa, 2020.

[6] H. Caldera, G. Meedin and I. Perera, “Time Series Based Trend Analysis for Hate Speech in Twitter During COVID 19 Pandemic,” in *20th International Conference on Advances in ICT for Emerging Regions (ICTer)*, Colombo, 2020.

[7] D. S. Dias, M. D. Welikala and N. G. J. Dias, “Identifying Racist Social Media Comments in Sinhala Language Using Text Analytics Models with Machine Learning,” in *2018 International Conference on Advances in ICT for Emerging Regions (ICTer)*, 2018.

[8] N. Hettiarachchi, R. Weerasinghe and R. Pushpanda, “Detecting Hate Speech in Social Media Articles in Romanized Sinhala,” in *20th International Conference on Advances in ICT for Emerging Regions (ICTer)*, 2020.

[9] H. Sandaruwan, S. Lorensuhewa and M. Kalyani, “Sinhala Hate Speech Detection in Social Media using Text Mining and Machine learning,” in *19th International Conference on Advances in ICT for Emerging Regions (ICTer)*, 2019.

[10] Z. Zhang, D. Robinson and J. Tepper, “Hate Speech Detection Using a Convolution-LSTM Based Deep Neural Network,” 2017.

[11] S. MacAvaney, H.-R. Yao, E. Yang,, K. Russell, N. Goharian and O. Frieder, “Hate speech detection: Challenges and solutions,” in *PLOS ONE*, 2019.

[12] Z. Zuping, N. D. Gitari, H. Damien and J. Long, “A Lexicon-based Approach for Hate Speech Detection,” in *International Journal of Multimedia and Ubiquitous Engineering*, 2015.

[13] R. E and W. J, “Learning extraction patterns for subjective expressions,” in *Empirical Methods in Natural Language Processing (EMNLP).*, 2003.

[14] W. J and R. E, “Creating subjective and objective sentence classifiers from

unannotated texts," in *6th International Conference On Intelligent Text Processing and Computational Linguistics*, Mexico, 2005.

[15] E. A and S. F, "SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining," in *5th International Conference on Language Resources and Evaluation*, Genoa, 2006.

[16] A. H. Razavi, D. Inkpen, S. Uritsky and S. Matwin, "Offensive Language Detection Using Multi-level," in *Advances in Artificial Intelligence*, 2010.

[17] I. Witten, E. Frank and J. Gray, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, 2008.

[18] M. Hall and E. Frank, "Combining Naive Bayes and Decision Tables," in *FLAIRS*, 2008.

[19] Z. Waseem, T. Davidson, D. Warmsley and I. Weber, "Understanding Abuse: A Typology of Abusive Language Detection Subtasks," in *Association for Computational Linguistics*, Vancouver, BC, Canada, 2017.

[20] "Google Bad Words List," [Online]. Available: https://www.freewebheaders.com/full-list-of-bad-words-banned-by-google/.

[21] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad and Y. Chang, "Abusive Language Detection in Online User Content," in *International World Wide Web Conference Committee*, 2016.

[22] Y. Lee, S. Yoon and K. Jung, Comparative Studies of Detecting Abusive Language on Twitter, Belgium, 2018.

[23] T. Davidson, D. Warmsley, M. Macy and I. Weber, "Automated Hate Speech Detection and the Problem of Offensive Language," in *Proceedings of the International AAAI Conference on Web and Social Media*, 2017.

[24] L. Gao and R. Huang, "Detecting Online Hate Speech Using Context Aware Models," 2018.

[25] G. K. Pitsilis, H. Ramampiaro and H. Langseth, "Effective hate-speech detection in Twitter data using recurrent neural networks," in *Applied Intelligence*, 2018.

[26] K. Steimel,, D. Dakota,, Y. Chen, and S. Kubler, "Investigating Multilingual Abusive Language Detection: A Cautionary Tale," in *International Conference on Recent Advances in Natural Language Processing*, Varna, Bulgaria, 2019.

[27] S. S. Aluru, B. Mathew, P. Saha1, and A. Mukherjee, "Deep Learning Models for Multilingual Hate Speech Detection," 2020.

[28] A. Arango, J. Pérez and B. Poblete, "Hate Speech Detection is Not as Easy as You May Think: A Closer Look at Model Validation," in *Association for Computing Machinery*, New York, 2019.

[29] P. Mathur, R. R. Shah, R. Sawhney and D. Mahata, "Detecting Offensive

Tweets in Hindi-English Code-Switched Language," in *Sixth International Workshop on Natural Language Processing for Social Media*, Melbourne,, 2008.

[30] V. K. Jha, H. P, V. P. N, V. Vijayana and P. P, "DHOT-Repository and Classification of Offensive Tweets in the Hindi Language," *Procedia Computer Science*, vol. 171, pp. 2324-2333, 2020.

[31] S. Kamble and A. Joshi, "Hate Speech Detection from Code-mixed Hindi-English Tweets Using Deep Learning Models," 2018.

[32] A. Alakrot, L. Murray and N. S. Nikolov, "Towards Accurate Detection of Offensive Language in Online," in *4th International Conference on Arabic Computational Linguistics*, Dubai, 2018.

[33] S. T. Luu, H. P. Nguyen, K. V. Nguyen and N. L.-T. Nguyen, "Comparison Between Traditional Machine Learning Models And Neural Network Models For Vietnamese Hate Speech Detection," in *International Conference on Computing and Communication Technologies (RIVF)*, Vietnam, 2020.

[34] V. Santucci, S. Spina, A. Milani, G. Biondi and G. D. Bari, "Detecting Hate Speech for Italian Language in Social Media," in *EVALITA*, Torino, Italy, 2018.

[35] N. Romim, M. Ahmed, H. Talukder and M. S. Islam, "Hate Speech detection in the Bengali language: A dataset and its baseline evaluation," 2020.

[36] "Tools and resources of Natural Language Processing Center at University of Moratuwa," University of Moratuwa, 2020. [Online]. Available: https://uom.lk/nlp/tools. [Accessed 27 02 2021].