FAKE NEWS DETECTION FOR AMHARIC LANGUAGE USING DEEP LEARNING

BY

ERMIAS NIGATU HAILEMICHAEL

OFFICE OF GRADUATE STUDIES

ADAMA SCIENCE AND TECHNOLOGY UNIVERSITY

JANUARY 2021

ADAMA, ETHIOPIA

Fake News Detection for Amharic Language Using Deep Learning

By: Ermias Nigatu Hailemichael

Advisor: Mesfin Abebe (PhD.)

A Thesis Submitted to the Department of Computer science and Engineering,

School of Electrical Engineering and Computing

Presented in Partial Fulfillment for the Degree of Masters of Science in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

January 2021

Adama, Ethiopia

Approval of Board of Examiners

We, the undersigned, members of the Board of Examiners of the final open defense by Ermias Nigatu have read and evaluated his/her thesis entitled "Fake News Detection for Amharic Language Using Deep Learning" and examined the candidate. This is, therefore, to certify that the thesis has been accepted in partial fulfillment of the requirement of the Degree of Masters of Science in Computer Science and Engineering.

______________________	______________________	______________________
Supervisor /Advisor	Signature	Date
______________________	______________________	______________________
Chairperson	Signature	Date
______________________	______________________	______________________
Internal Examiner	Signature	Date
______________________	______________________	______________________
External Examiner	Signature	Date

Declaration

I hereby declare that this MSc thesis is my original work and has not been presented as a partial degree requirement for a degree in any other university and that all sources of materials used for the thesis have been duly acknowledged.

Name: Ermias Nigatu

Signature: ________________

This MSc thesis has been submitted for examination with my approval as a thesis advisor.

Name: Mesfine Abebe (PhD.)

Signature: ________________

Date of Submission: ________________

ACKNOWLEDGMENT

First, I thank the almighty GOD for his blessing me with good health and wellbeing which made me successful in my all academic pursuits.

Then, I would like to express my profound gratitude to Dr. Mesfine Abebe, whose expertise was invaluable in giving the right direction, useful comments, and engagement from the beginning till the end. He continuously encourages me and always has been ready and enthusiastic to assist in any way he could throughout the process.

Next, I would like to thank Amanuel Sbhat, Agence France-Presse's fack-checking journalist, for inviting me to his workplace, the continuous and fruitful communication we had. He made me knowledgeable in the area of misinformation and fake news.

Next, I would like to thank journalist Mubarek Akmel and journalist Rediwan Sefa for their patience and supportiveness in the processes of annotating the dataset that I used in my research. It will not be easy if they were not kind to me.

I would also like to thank all my colleagues and classmates for that they shared their knowledge with me. Their advice and encouragement are vital to keeping me on track.

Also, I would like to thank my beloved and missed mother, whom I lost amid my writing, for wishing me a bright future even in her worst days. And for my dad, for being light in my doomsday. Furthermore, I could not have completed this paper without the support of my friends, relatives, and neighbors of mine, I thank them all very much!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ACRONYMS AND ABBREVIATIONS

ACC: Accuracy

AFP: Agence France-Presse

API: Application Programming Interface

BERT: Bidirectional Encoder Representation from Transformers

Bi-GRU: Bidirectional Gated Recurrent Unit

Bi-LSTM Bidirectional Long Short-Term Memory Network

BOW: Bag of Word

CBOW: Continuous Bag-of-Words

CNN: Convolutional Neural Networks

CNVnets: Convolutional Neural Networks

COVID-19: Coronavirus Disease 2019

CUDA: Compute Unified Device Architecture

GloVe: Global Vectors for Word Representation

GPU: Graphics Processing Unit

GRUs: Gated Recurrent Units

IDF: Inverse Document Frequency

LSTMs: Long Short-Term Memory Networks

MIL: Media and Information Literacy

NLP: Natural Language Processing

OOV: Out-of-Vocabulary

PoS: Part of Speech

RNN: Recurrent Neural Networks

TF: Term Frequency

TPU:	Tensor Processing Unit
UNESCO:	United Nations Educational, Scientific and Cultural Organization
WSGI:	Web Server Gateway Interface

ABSTRACT

The new media age we are living in has enabled many of us to be connected through the internet, which has been changing the way updates arrive at hand and from whom we get those pieces of information. Due to the usage of such convenient technology for ill intent, fake news dissemination has become a great problem to societies all over the globe. Especially the phenomenon is more common in the time of election and chaos like we have been experiencing in the COVID-19 pandemic. Despite the issue that fake news and misinformation are not constricted *to language and culture, most of the attempts to automate the detection of fake news are concerned with a specific group of languages, especially the English language. In this study, we used a newly collected and annotated dataset of 12000 news to build an automated fake news detection system for one of a low-resourced language; Amharic, using deep learning algorithms. The research employed several experiments to determine the best performing deep learning architecture among the ones used in the field of Natural Language Processing and got the Bidirectional Gated Recurrent Unit(Bi-GRU) and Convolutional Neural Network(CNN) to outperform over the other recurrent and attention-based models. The CNN model surpasses all other models with an accuracy of 93.92% and an f1-score of 94%. The effect of Morphological normalization on the Amharic fake news detection was also accessed over the best two performing models and the experiment revealed that applying normalization has an unfavorable effect on the classification performance that reduces the f1-measure of both models from 94% to 92%. In addition to that, different combinations of CNN hyperparameters are tested. Even though significant improvement in terms of performance was not seen from the tuning process, an important correlation between the performance of the model and hyperparameters was seen. Besides our contribution in the evaluation of these deep learning models to one of the morphologically rich languages, we expect the newly proposed dataset will be a reason for more research and findings regards to the detection and prevention of Amharic fake news soon.*

Keywords*: Fake news, Amharic, Deep learning, Low Resource Language, Natural Language Processing*

CHAPTER ONE

1. INTRODUCTION

1.1 Background of the Study

A great many people in Ethiopia are nowadays seen stuck to their communication apparatuses to be updated on public affairs in their country and on other global disclosures; immersing themselves into the internet. Social media platforms like Facebook are prominent in such a situation: besides their usage for trading, entertainment, education, and maintaining acquaintanceships. Considering the ease and choices to get information right away, masses have been turning to social networks[1], YouTube channels, and instant messaging apps alternative to the more traditional and corporate media outlets such as newspapers, radios, and televisions.

As a result, media have started to react to the change, and traditional media began to prioritize its online presence and started to use the new distribution channels, mainly social networks which also opened the door for deceivers. By using the advantage of a broader audience and reliable information provided by prestige media, misinformation generators have been working hard to get more reputation in the mind of their followers as it may help them to monetize their content through splashy headlines and captions aimed to be shared virally. Moreover, people with some direct political agenda or who gain economic incentives for their dissemination of misinformation from somebody they are compelled to or from the advertisement platforms, such as Google AdSense are more likely engaged in such activity.

However, this kind of trend eventually can lead to deadly conditions. These days most information that people access is usually unproven and frequently assumed as real, where the problem arises; and leading the country to chaos. Especially, this problem has reached its peak since some significant political changes have been seen from two years ago to the emergence of the COVID-19 pandemic. Fake news has been putting groups in extremes of their ethnic, political, and religious identity, which in turn promoting instability in almost all over the country's regions. And here is a mad thing, a falsehood spread significantly more rapidly, deeper, and broadly than the truth[3]. Moreover, as there is no strong and Non-Partisan Fact-Checking organization, the problem is even becoming worse.

More Recently, Facebook claimed to expand its fact-checking program to Ethiopia[2] as part of its ongoing work in helping the legitimacy and quality of information people find on the platform. The other relatively successful Fact-checking organization is AFP Fact Check, a global fact-checking organization based in France. The organization has dedicated journalists in different countries, but as they are using English as a medium, they may not address the majority of the people. Besides mentioned above, there are also self-motivated journalists like Elias Mesert who are trying to provide fake news alerts to newsrooms and their broader audience through social media and website using the country's working language Amharic; a bridge language and the most widely spoken one in Ethiopia.

From the government side, under the Computer Crime proclamation, the Ministers Council approved a bill drafted by the attorney-general to combat hate propaganda and misinformation being circulating, in November 2019[3]. Although it is believed the regulation is important to minimize the severity of the problems, some also believe it could undermine free speech and digital rights by relating to the government's experience of formulating oppressive laws, internet censorship, and complete Internet outages. Because news literacy is not provided for the citizens at an early stage and the essence of strong professional journalism is not entertained well, enforcing the law by itself may not be effective as it was expected. So to unleash such difficulties, it is believed that technology could be in support of society by identifying misinformation, reducing financial gain for those who do it for money, and improving online accountability.

But, Even though the problem is obvious and causing great distractions, nothing practical is being done from either the government office or academician side in terms of detecting Amharic fake news technologically. Moreover, as there is no annotated dataset, it has been making it difficult and untouchable. So, the main objective of this study is to prepare an Amharic fake news dataset and use various deep learning algorithms to detect fake news computationally.

1.2 Motivation of the Study

As we can clearly notice, the internet has been becoming a weapon for fake news generators aiming at a political or economic gain[4] that had already shown its havoc on individuals, groups, and even the government.

Furthermore, fake news has turned into a serious problem with increased media exposure and its apparent negative impact on society. Generally, fact-checking remains a wearisome task that the user has to report fake stories to the social network service providers for it to be manually reviewed and then suppressed or entirely removed from the platform; which is difficult to handle by humans as users communicate in many different languages.

As the use of social media increased disproportionately relative to the media literates, it is creating a fertile environment for the misuse of such platforms. Hence, we were inspired to study fake news detection for the Amharic language on social media as detecting online fake news is one of the important tasks to tackle the actual problem in the first place. Besides that, we also want to be part of the development in improving detection systems and as a contributor to publicly available fake news datasets for future researches.

1.3 Statement of the Problem

Detecting fake news has never been an easy task. It can't be free from error soon as social media consists of a large amount of user-generated content written in an intentional and genuine-looking language to mislead readers [5].

Misleading and fake content spreading on social media will not only affect the society who are online frequently, but its consequence will be more harmful when the information shared across the media start to cause evident problems to the public at large.

After deep learning has shown its success on tasks related to image classification and object recognition, the abundance of natural language data and advancement in representing such data makes deep learning ideal for texts and speech processing which makes it now a state-of-the-art in many NLP tasks[6].

Fake news detection based on the content of the news is merely a text classification problem and different scholars have been trying, especially for the English language, to detect fake news using various machine learning techniques and different forms such as written, visual, or combination of the two mentioned[5], [7], [8]. But as far as our knowledge, the study of Fake news detection for the Amharic language is not conducted yet; indeed which is a reason to have no dataset to work with it.

The purpose of this study is to classify Amharic written social media stories as fake or real using different deep neural networks by building a new annotated dataset from the Facebook social media platforms.

1.4 Research Questions

This research work intends to answer the following three research questions.

RQ1: To what extent we can achieve fake news detection using only its content regardless of the social media engagement data?

RQ2: Which deep learning algorithm will perform best for Amharic Fake News detection?

RQ3: How can the overall performance of Amharic Fake News detection be enhanced?

1.5 Objectives of the Study

1.5.1 General Objective

The main objective of the study is to explore and assess different deep learning architectures for automatic Amharic fake news detection.

1.5.2 Specific Objectives

- ✓ To scrap and collect public Facebook posts or "news" and build a dataset.
- ✓ To formulate the criteria for labeling the dataset as fake and real.
- ✓ To train different deep learning algorithms with the newly collected dataset.
- ✓ To recommend the best detection model by evaluating the performance of different deep learning architectures.
- ✓ To tune the best-performing model for further improvement.
- ✓ To deploy the best model on a web server.

1.6 Scope and Limitations

1.6.1 Scope of the Study of the Study

The study focuses on introducing a fake news detection model for the Amharic language using a deep neural network. The study aimed to train the dataset on recurrent, attention, and convolutional based neural networks; and to use k-fold cross-validation for model performance evaluation with different metrics. As there is no available dataset for the language considered in this study, building a new dataset is required. The dataset is built by collecting Amharic news feeds from public pages of the Facebook social media platform.

1.6.2 Limitation of the Study

The limitation of this research arises from two major problems. The first one is related to the creation and annotation of large datasets; which include the perplexity and biases regarding the dataset annotators and intra-class imbalance among different types of fake news. Here, as we do have a limited number of journalists and an insufficient amount of already fact-checked stories, the number of clickbait type fake news is larger than the others. The other issue is concerning the computational resource which prohibited us from carrying out more GPU intensive processes like automatic hyperparameters tunings, as we got a low-end GPU.

1.7 Significance of the Study

Fake news campaigns have been rising unprecedently over recent years. Even though there are efforts from the government side through the formulation of rules and regulations, tackling the problem with technology has been not give attention. This research work will have a benefit from those who use social media as a source of information, media corporations, government to social media companies. Although this research is not meant to fully automate the fact-checking process by journalists, it could help them in spotting suspicious contents early before dissemination let say if it was integrated with browsers as an extension. Furthermore, this research could be used as an initial point to explore the issue better, and also the dataset we collected could be used as a great resource for anyone curious about the problem.

1.8 Organization of the Study

In this section, we provide a glimpse of the topics we covered in our thesis report. The report comprises six chapters.

Chapter one, where this section belongs, is an introductory chapter that focuses on the background of the study, the motivation behind the study, statement of problems with research questions to be raised and addressed, the scope and limitation of the study, objectives to be reached and application of the study.

Chapter two presents definitions and theoretical backgrounds related to fake news, an overview of social media literacy and its context in Ethiopia, literature related to fake new detection mechanisms, feature extraction methods, deep learning models used, and works related to our study.

The third chapter goes into how the Amharic fake news dataset is collected, prepared, and annotated. Then it discusses how the data is preprocessed, features are extracted and the models used in the study. In addition to that, it also points out the validation mechanisms used, metrics considered, and the prototyping strategy planned.

The fourth chapter is all about the tools we use in data preparation, modeling, and prototyping of the proposed system. It also includes the implementation regarding the preparation of the embedding matrix and actual model implementations.

Chapter five presents the result gotten from the experimentation of our dataset with different deep learning models and also presents the result of the top-performing model with and without normalizing the data. Besides that, this chapter discusses and analyses the major results obtained from the experimentation.

Finally, Chapter six concludes the report by spotting future research directions and recommendations.

CHAPTER TWO

2. LITERATURE REVIEW AND RELATED WORKS

2.1 Fake News

The term “Fake News” is very controversial and widely open for misinterpretations. Cambridge Dictionary defines it as *"false stories that appear to be news, spread on the internet, or using other media, usually created to influence political views or as a joke"* [9]. Another similar definition of how the term has been used scholarly found that fake news is referred to as various forms of fictitious content, from political satires and news parodies to state propaganda and false advert [10].

Definitions around the issue vary along two main dimensions: first, the extent of facticity and, second, the actual intent to deceive. Fake news is also well known for its attempt to impersonate the traditional news format than the other forms of disinformation. Thus, fake news refers to a specific type of disinformation: It is false, it is intended to deceive people, and it does so by trying to look like real news[11]. Here, one controversy that arises is that sometimes political actors use the term to describe news coverage that is unsympathetic to their administration and performance, even when the news reports are accurate.

PolitiFact, one of the foremost widely known fact-checking organizations, define Fake News as fabricated and skillfully manipulated stuff that appears legit journalistic reports, which are pervasively disseminated online to large audiences, potentially makes the audience believe the fabrications and spread it to the word.

First Draft, an international not-for-profit organization that addresses issues relating to trust and truth in the information age, developed a categorization of information and online content into seven different types that could be interpreted as "fake news" [12]:

- ✓ ***Satire or parody***: Comedic and sarcastic content that doesn’t have an intention to cause harm, even though it has the potential to be misconceived by the users.

- ✓ ***False connection***: When unrelated things are made to seem coherent and timely, while headlines, visuals, or captions don't support the content.
- ✓ ***Misleading content***: The deceptive use of information to frame a point of issue or a person.
- ✓ ***False context***: When content generated previously is claimed as it is happening now with false contextual information.
- ✓ ***Imposter content***: When real sources are impersonated.
- ✓ ***Manipulated content***: When genuine content or pictures is doctored to deceive.
- ✓ ***Fabricated content***: A deliberately designed and fabricated content to cause harm.

2.2 Social Media Literacy

In this information age, where information providers are eager to satisfy their customers with user-defined or custom contents, information flow, and search results: finding accurate, unbiased information online has become increasingly difficult. As a result, such environments require literacy skills which makes citizens skeptical observers and aware of the devastation that can be caused by misinformation[13]

As of UNESCO's definition of Media and Information Literacy (MIL), Social Media Literacy can be defined as *"a set of proficiency that allows peoples to access, retrieve, understand, evaluate and utilize, create and publish media content in all formats, using numerous tools, in a critical, ethical, and productive way to participate in personal, professional, and societal activities. This means that a 'media and information literate person must be a responsible information seeker, knowledge creator, and innovator rather than just being a consumer of information and media content only"*.

2.2.1 Social Media Literacy in Ethiopia

A considerable number of people in Ethiopia are smartphone users, which most of them are inclined to the most popular social media platforms, looking for information and communication with others. Hence, many users are expanding their information sources from more conventional media to social networks and forums.

More recently internet users have been increasing exponentially, especially using social media platforms like Facebook becomes very prominent in the younger generation. In Ethiopia only, from a total of a hundred million people, more than twenty million use Facebook in different languages including the Amharic language which the official language of the country[14].

Even though social media usage in the country is going pervasively, it is extremely difficult to believe that the users are media literate as there is a clear knowledge gap in this virtual society to properly analyze, evaluate and create messages across a variety of contexts[15].
In addition to that, the irrational and frivolous communication exchanges on social media platforms illustrate many users have a weak understanding of the power of the media. As a result, many of them seem to fail to consciously manage the information overflow on social media platforms.

2.3 Fake News Detection Techniques

Researchers from different disciplines have been trying to demystify the clues which can be important to decide whether a given content is deceptive or not. Aside from the broader and groundwork studies on Fake News, studying how machines could computationally sense features from the data and generalize on the likelihood of the content to be in a given class is recently growing. To achieve that, different mechanisms are employed ranging from detecting based on the content of the news to using the social context and hybrid of both.

2.3.1 Content-based Approaches

Detecting deception from its content is established on either linguistic features extracted from the actual message or visual clues from supplementary contents like images and videos. Even though detecting Fake News from its content needs a lot of data to train and increase the load on the model to be trained, it has shown great potential to detect misinformation and intervene before it's spreading and causing destruction to society[16].

2.3.1.1 Linguistic Signal

This approach is based on the linguistic-based features that are taken out of the textual content. The contextual features could be extracted from the smaller components like characters and words to sentence and document level. The straightforward approach to harvest on linguistic cue is to represent the text as bag-of-word and n-gram, while some other researchers also make use of the lexical and syntactic features such as PoS, punctuation, and shallow parsing[17].

The other method to detect deception by capturing the writing style of news content. The idea evolved to computational science from psychology, and it’s on a paper[18] that the first style-based text categorization was proposed. Besides the earlier mention technique, the linguistic cue can be also attained by a knowledge-based approach. Here, deceptive contents are extracted and examined to decide its facticity by querying existing external knowledge[19].

2.3.1.2 Visual-Based

This approach focuses on the characteristics of distortion captured from visual contents such as pictures and videos. In addition to these characteristics, incorporating some statistical features also shown an improvement over verification of news as it can capture image distribution patterns qualitatively.

2.3.2 Social Context-based Approaches

To see the effect of using social engagement data for detecting Fake News, a paper [20] presented a technique to which utilizes the interaction of news stance, publisher bias, and important user engagements simultaneously from the news ecosystem. The researchers got the collected auxiliary information that could be useful in detecting deception that is more likely missed by the content-based approach[21], though getting quality engagement data is difficult and early detection is somehow compromised. This method can be categorized as a stance-based and propagation-based detection, while the previous is built on the users’ viewpoints from relevant posts to deduce the validity of original news articles, and the latter is based on the interrelations of relevant social media posts.

2.3.3 Hybrid Approaches

By combining the technique from social context-based and content-based, the hybrid method can be favored from the supporting information from different perspectives. Papers [22],[20] indicate integrating both techniques from the social context-based and content-based could improve the accuracy of automated prediction.

2.4 Feature Extraction Methods for Text Classification

2.4.1 Bag of Word

it is the simplest yet effective enough feature extraction method in text classification which merely considers if a known word occurs in a document or not. Using a bag of words is based on an assumption that the word count for similar documents tends to be similar. In other words, the more similar the words in two documents, the more similar the documents can be.[23]

Drawbacks of BOW:

1. ***Semantic meaning***: Word sequence and its syntactic and semantic content are ignored as it does not consider the meaning of the word in the document. The context of a word is completely ignored in BOW.
2. ***Vector size***: As the size of the data we have increased, obviously the number of vocabulary in the text increase, results in a lot of computation time and very sparse representation.

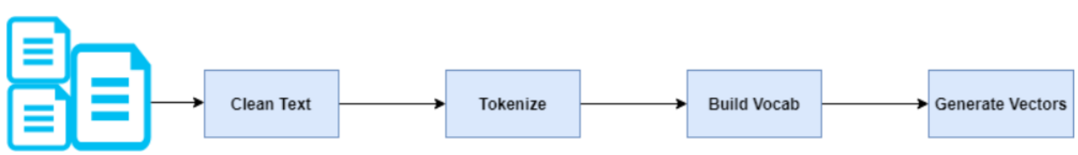


Figure 2.1[1] Bag of Word (BOW)

[1] https://www.i2tutorials.com/wp-content/media/2019/05/Bag-of-words-1-i2tutorials.jpg

2.4.2 TF-IDF Term Weighting

Tf-idf stands for *term frequency-inverse document frequency* and has been widely used in the fields of information retrieval and text mining to evaluate the relationship for each word in the collection of documents. A word significance increases as the number of incidence in the documented increase and is constrained by the frequency of the word in the corpus.

To calculate Tf-idf two values are required: the first one is Term Frequency and computed as a quotient of the number of times a word occurred in a document divided by the total count of a word in that document; the second term is the IDF, computed as dividing the total number of documents by the number of documents containing the term in the corpus.

T.F.: Term Frequency, is a mechanism to up-weight a word based on the frequency of that term in the document. The term frequency should be divided by the total number of words/terms in the document, as each document may have a different length. And, this could make bias as a term would appear much more time in long documents than shorter ones.

$$TF(t) = \frac{(Number\ of\ times\ term\ t\ appears\ in\ a\ document)}{(Total\ number\ of\ term\ in\ the\ document)} \qquad 2.1$$

IDF: Inverse Document Frequency, Because Term Frequency treats all words equally, we need to find a way to calculate the significance of a term as regards the whole document. To limit the unwanted effect of stop words and more frequent terms, we need to weigh down repeatedly occurring words while up-weighting the rare ones.

$$IDF(t) = \log_e\left(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ term\ t\ in\ it}\right) \qquad 2.2$$

2.4.3 Word Embedding

Word Embedding is a distributional representation of words with similar meanings to be understood by machine learning models. Word to vector mapping in the embedding process is can be achieved by probabilistic models, neural networks, or dimension reduction on the word co-occurrence matrix

The first modern approach for distributional representation was proposed by Mikolov, Chen, et al[24] who made eventually popularize word embeddings after they released word2vec. Word embeddings became the latest in natural language processing after the release of the word2vec toolkit and sparked a huge amount of interest in the topic.

GloVe, by Pennington et al.[25], is an unsupervised learning algorithm for word representation in vector space. It uses the advantage of the skip-gram model of word2vec and collaborative filtering algorithms aka matrix factorization. To its simple understanding, GloVe merely generates a word-word co-occurrence matrix from the overall document used for training and map each word into a semantically appropriate point in space where the distance between similar words is kept small. Eventually, after GloVe came out, word embedding became more familiar with the natural language processing community.[26]

2.4.3.1 Predictive Models in a Word Embedding

After Mikolov et al. proposed word2vec, a N.N. model for word representation, it has been shown that Neural Network architectures are more convenient and efficient because of their simple projection layer instead of multiple hidden layers. To achieve a good quality word representation Word2vec architecture is designed to be trained over a fairly large amount of data. Word vector representation of Mikolov et al. has two neural network architectures, namely CBOW and skip-gram model.

Continuous Bag-of-Words (CBOW) Architecture

In CBOW architecture, the vector representation of a given word is determined by its neighboring words, from where the semantics of the word can be captured. The output layer of the architecture holds the vector representation of a word after the model trained over the average vectors of accompanying words. Here the architecture doesn't influence by the order or location of words in the document as it uses the principle of bag-of-word.

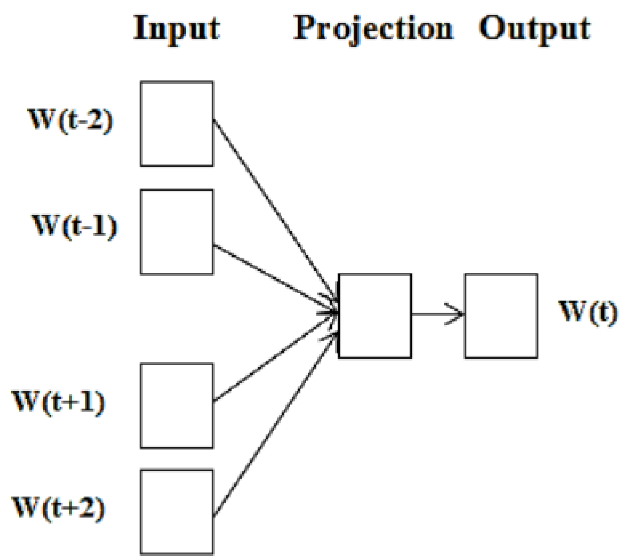


Figure 2.2 CBOW model Architecture

Continuous skip-gram Architecture

This model, instate of predicting the vector of target words, works in a reverse way of the continuous Bag-of-Word model. It predicts the accompanying words from the given word, which appears in the middle of the sequence relative to its position in the input words[27]. Even though CBOW architecture is preferable for smaller datasets, the continuous skip-gram model is more efficient for a large amount of data to be seen.

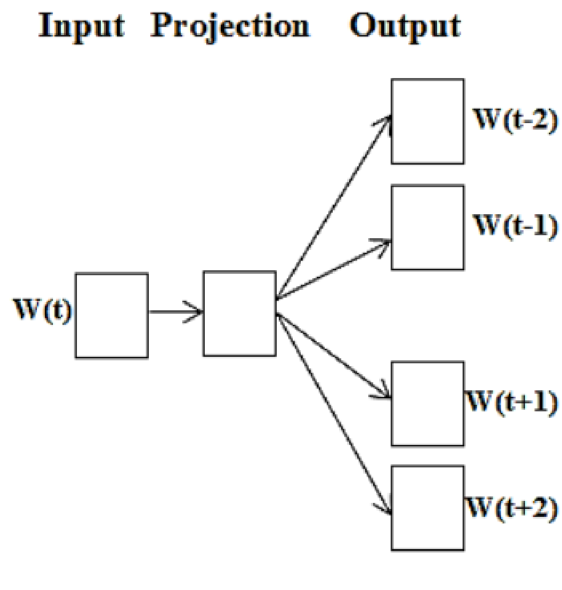


Figure 2.3 Continuous skip-gram model architecture

2.5 Deep Learning Models

2.5.1 Neural Networks

Biological Model

Neural networks are a pattern recognizable model fascinated by the interworking principle of the brain, especially the human brain. In a biological setting, one brain cell or neuron receives a signal from its extensions of a nerve cell so call dendrites or dendritic trees, and if the signal is strong enough, it will pass through an axon and link to a dendrite from another neuron. Neurons communicate with each other via electrical events called 'action potentials' only become connected when the link of an axon from one neuron and dendrite from the others are stimulated.

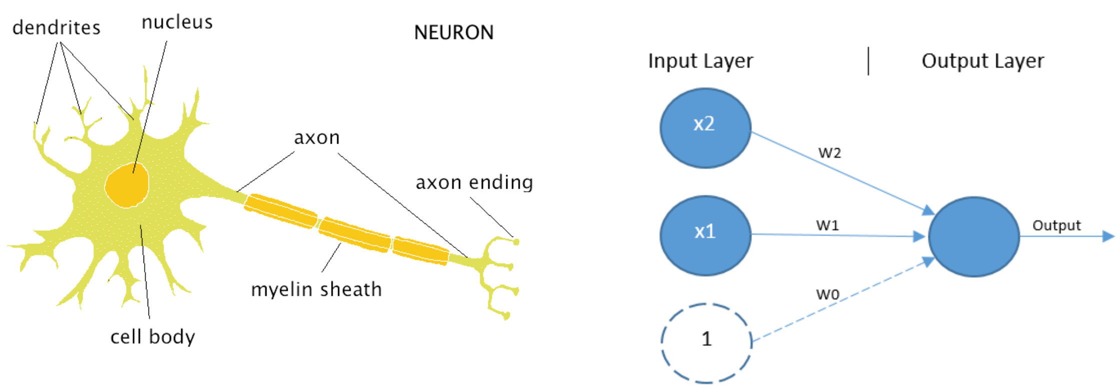


Figure 2.4 A Single Biological Neuron [2](Left) and Artificial Neuron[3] (Right)

Computational Model

Nodes:

Nodes are the building blocks in Artificial neural networks. Like the biological neuron build the overall nervous system in our brain, aggregation of Nodes results in a computational model that resembles a biological one.

---

[2] http://webspace.ship.edu/cgboer/theneuron.html

[3] https://dev.to/jbahire/demystifying-the-xor-problem-1blk

Connections:

The connection between the different neurons is represented by the edge connecting two nodes in the graph representation of the artificial neural network. They are called weights and are typically represented as W_{ij}.

Layers:

The word layer in deep learning is then used to call each of the stacked aggregations of neurons.

The layer which introduces features to the model is called the input layer. This layer is responsible for the initialization of the first weights, where its output is propagated forward to the consequent layers. The last layer receives the name of the output layer. Intuitively, it will provide the output resulting from all the computation performed by each of the neurons to the inputs. All the intermediate layers between the input and the output layer are called hidden layers. These layers

By creating a hierarchic representation, the above mention layers can learn the representation of the given data, so-called representational learning.

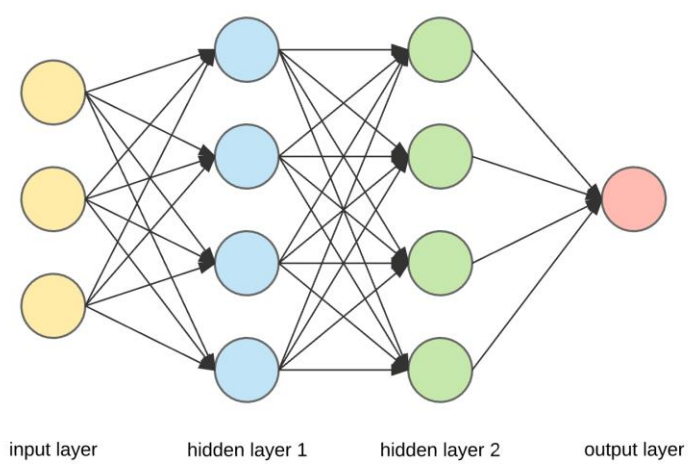


Figure 2.5[4] Neural networks with 2 hidden layers

[4] https://latticenepal.com/machinelearning1/

2.5.2 Recurrent Neural Networks (RNN)

Unlike the vanilla neural network, Recurrent Neural Networks (RNN) is designed to be trained with sequential data where the output from the previous step is consumed by the current step. To achieve that, RNNs should remember the information about a sequence and have a construct called a hidden state. The hidden state memory is responsible to hold all the weights calculated. For each input, the memory unit uses the same parameters as it performs recursively on the same task for all the hidden layers. Because of that, it has fewer complex parameters than the other type of neural networks.

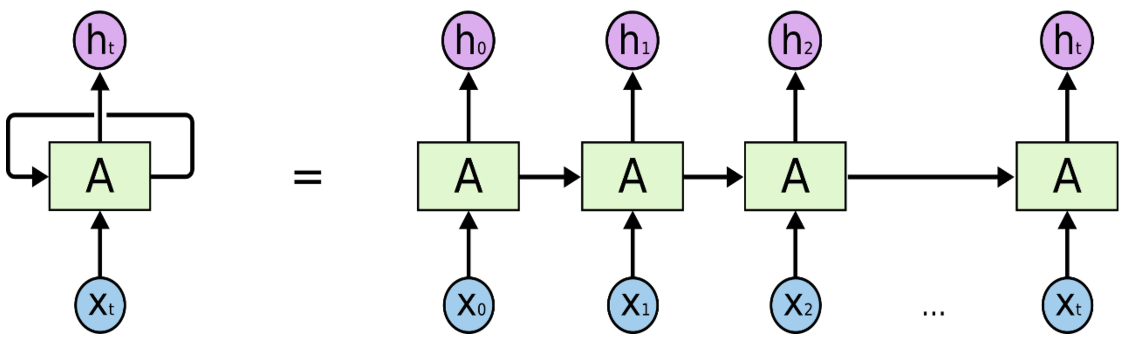


Figure 2.6[5] Recurrent Neural Network

Given input a_t and hidden state of previous step h_{t-1}, new hidden state and output at time step t are computed as:

$$h_t = \sigma_h (W_h x_t + U_h h_{t-1} + b_h)$$
$$y_t = \sigma_y (W_y h_t + b_y \qquad 2.3$$

where:

- x_t is input vector at a time step, h_t is hidden layer vector, y_t is output vector at time step t.
- W,U,b are parameter matrices and vectors.
- σ_h, σ_y are activation functions.

[5] https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent Neural Networks are essentially useful in modeling sequential data where inputs are not fed into the networks all at once but are broken down into small pieces which are later passed into the network cell one after another.

Despite being designed to deal with mimic and work on the sequence nature of some kinds of data, it is proved that RNNs have limitations in capturing long dependencies. As a result, the Long Short-Term Memory Network, a modified version of RNN with gating mechanisms, is devised to get over the limitation of traditional RNNs.

2.5.3 Long Short-Term Memory Networks (LSTMs)

LSTM is denoted in Figure 7 It models the word sequence x as follows:

$$
\begin{aligned}
i_t &= \sigma(x_t U^i + h_{t-1} W^i + b_i) \\
ft &= \sigma(x_t U^f + h_{t-1} W^f + b^f) \\
qt &= tanh(x_t U^q + h_{t-1} W^q + b_q) \\
pt &= f_t * p_{t-1} + i_t * q_t \\
ht &= o_t * tanh(p_t)
\end{aligned}
\tag{2.4}
$$

LSTM has three gates: input gate i_t, forget gate f_t, and output gate o_t. All gates are generated by a sigmoid function over the ensemble of input x_t and the preceding hidden state h_{t-1}. To generate the hidden state at current step t, it first generates a temporary result qt by a tanh nonlinearity over the ensemble of input x_t and the preceding hidden state h_{t-1}, then combines this temporary result qt with history p_{t-1} by input gate it and forget gate ft respectively to get an updated history pt, finally uses output gate o_t over this updated history pt to get the final hidden state h_t.

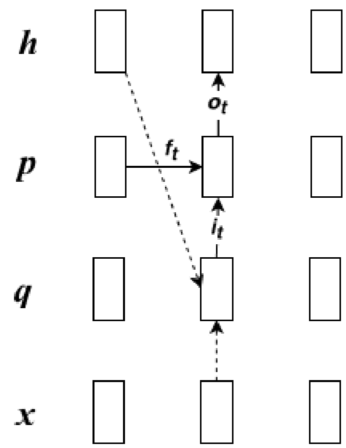


Figure 2.7 Long Short-Term Memory Networks

2.5.4 Attention Mechanism

Even though the encoder-decoder network performs quite well in machine translation for smaller input length, Cho et al[28] pointed out that it is difficult for the model to maintain its performance as the input gets larger. To deal with the loss a context over a large sentence, Bahdanau et al.[29] The proposed attention model in 2015, "consists of a bidirectional RNN as an encoder and decoder that emulates searching through a source sentence during decoding a translation" [26, p 3] The research intends to mimic a selective concentration capability of a human brain while ignoring the irrelevant one. Because the method was inspired by the nature of our brain, the attention mechanism also gained acceptance in other problems rather than machine translation, including computer vision, speech processing, etc.[30]

2.5.5 Convolutional Neural Networks

Convolutional Neural Networks, LeCun et al[31] sometimes called Convolutional networks(ConvNets) is a deep learning architecture specially designed for data that can be represented in a grid-like matrix form. For instance, time-series and textual data can be represented in the 1D vector and a 2D matrix can be used to represent pixels in image data. The architecture got its name "convolution neural networks" from the mathematical operation called Convolution, where a linear 2operation takes place of the ordinary matrix multiplication on at least one of the neural networks. Because of this convolution operation, the algorithm is very effective in detecting spatial and temporal dependency by applying appropriate filters and also it helps in parameter reduction such that the model can be trained over large size data without losing its important features[32]. The other quality of ConvNets is that it needs the minimum effort to preprocess data relative to the other traditional methods, which need data to be processed in a hand-engineered way. So given enough training data, ConvNets can learn filters or characteristics even considering single-layer architectures[33].

2.6 Amharic Language

Amharic (Amharic: አማርኛ, Amarəñña) is a state language in Ethiopian, and its the most widely spoken language after Arabic among the Semitic languages. Furthermore, the language has been serving as a lingua franca for many communities in business and administrative duties all over the country. Amharic uses its alphabet which is adopted from the Ge'ez script that makes the language rich in its literary history[26].

2.6.1 Word Formation

Fidäl (Amharic: ፊደል) is an alphabet in Amharic which essentially consists of consonant with vowel characters in different sequences. In the Amharic writing system, unlike English, vowels are not always written explicitly, instead, it uses fused consonant-vowel letters. The only exceptions are the letters አ-A and ዐ-A where each of them represents the vowel sound alone. For example, a word ነገረ could be written in its root form with its vowels as (ን-ኧ-ግ-ኧ-ር-ኧ 'n-ea-g-ea-r-ea')[34]. So with such combinations to address all the sounds in the language, Amharic has more than 270 characters each represent a consonant + vowel sequence.

1st	2nd	3rd	4th	5th	6th	7th	8th
ሀ [hə]	ሁ [hu]	ሁ [hi]	ሃ [ha]	ሄ [he]	ህ [hɨ]	ሆ [ho]	ሇ [hoʊ]
በ [bə]	ቡ [bu]	ቢ [bi]	ባ [ba]	ቤ [be]	ብ [bɨ]	ቦ [bo]	ቧ [boʊ]
ከ [kə]	ኩ [ku]	ኪ [ki]	ካ [ka]	ኬ [ke]	ክ [kɨ]	ኮ [ko]	ኯ [koʊ]

Figure 2.8 Example Amharic letters with their pronunciation

Character Redundancy and Affixation:

Amharic text could be written in a different way to represent the same sound and this characteristic of the language also extends the complexity to the use of different bound morphemes such as suffixes in word-formation. Such variability makes the language difficult to be processed and often makes the language need more data for effective representation in different natural language tasks.

2.7 Related Work on Fake News Detection

Text classification research for content written in Amharic had been started earlier though now it flaws concerning the use of state-of-the-art methods and making resources available at large to be used in different deep learning pipelines. Samuel Eyassu and Bjorn Gamback in their research paper[35] tried to tackle the Amharic news classification problem with Self-Organizing Maps (SOM), which is an unsupervised type neural network originally developed for multi-dimensional vectors on a reduced dimensional space. They used a corpus with a size of 206 new articles from Walta Information Center where 101 instances of the corpus are used for training and the rest 105 for the test. They experiment on user queries of 25 and got 60% precision on clustering unseen data and a precision of 69.5% on classifying the data.

Worku Kelemework on research[36] aimed at classifying Amharic news automatically using NNs, he used the Learning Vector Quantization (LVQ) algorithm with TF and TF-IDF feature extraction methods. When TF feature extraction was used, the researcher got an accuracy of 94.81%, 61.61%, and 70.08% on three, six, and nine classes task respectively, which is around 75% accuracy on average. In contrast, applying the exact experiment with TF-IDF results in an accuracy of 69.63%, 78.22%, and 68.03%, an average of 71.96%.

In a paper[37] by Md Zobaer Hossain et al., the researchers proposed a dataset of 50 thousand instances for one of low resourced language Bangla, a widely spoken language in Bangladesh. They collected legitimate news from 22 different mainstream media in Bangladesh and for collecting fake news they used the most popular satirical news publishers, fact-checking websites, and sites that potentially do clickbait. In their research, they treated satirical, fabricated, and clickbait contents as fake news. After they created the dataset, they have also evaluated the dataset with linear classifiers and neural network-based models and got traditional linguistic features to perform better than the neural network-based models. Besides that, they also find character-level features are important than the word-lever one.

Yang Yang et al.[5], proposed a model named TI-CNN (Text and Image information based Convolutional Neural Network). The researchers used a dataset size of 20,015, where 8,074 of the record are REAL and 11,941 are labeled as FAKE. The dataset column name consists of title, text, image, author, and website but researches selected (title, text, image) value to emphasize intrinsic difference b/n Fake and Real news. They undergo analysis of the text in a different aspect and got good insight from their visualization. For example, Fake news tends to not have a title and more have capital characters to attract the readers' attention. Also, a sentence of real news is shorter than that of fake news but fake news has fewer words than real news on the overall content. On the other hand from a cognitive perspective, their analysis showed that a legit writer use negations (like 'no', 'not') and exclusive words (such as, 'but', 'without', 'however') more frequently than deceivers. And fake news generators use fewer first-person, fewer second-person, and more third-person pronouns. Besides the textual data, an image in Fake and Real news also has distinctive behavior in their finding. For instance, Real news has more human faces than real Fake news and mostly Fake news incorporates irrelevant images such as animals and cartoons. The resolution of real news is also relatively better. After they made this analysis, the researchers build am a model that contains two major branches, such that the text branch and image branch. Explicit and latent features are extracted by taking the textual or visual data as inputs for each branch. Trining of the model done on 80% of the data, 10% reserved as test data, and the rest kept for validation purposes. Then they experimented and their model shows strong expandability, which can easily absorb other features of news. Besides, the CNN model gets trained faster than LSTM and other RNN models as it can see the whole input at once.

Yong Fang et al.[16] proposed Self Multi-Head Attention-based Convolutional Neural Networks for fake news detection. The researchers used a publicly available dataset with around 24,000 news, collected between October 26 and November 25 of the year 2016, during the U.S. presidential election period. Among the dataset collected there are 9,762 real news and 12,228 fake news. The proposed model to be trained on this dataset is the result of the conventional neural network and self multi-head attention mechanism which is an appropriate mechanism to obtain the internal spatial relationship in word. So by concatenating the matrix of words from embedding with the attention matrix, they got a new extended matrix that represents an article better. This collective matrix improves the word representation with contextual information between non-consecutive words. After the vector representation has gotten, the researcher used convolution(1D) kernels to extract features effectively and applied a max-pooling technique to decrease the noise coming from the out of the convolutional layer. Finally, a fully connected layer is applied to learn from the features learned lately using a non-linear softmax activation function. To prove its validity, the researchers conducted experiments on their dataset and achieved a precision rate of 95.5% with a recall rate of 95.6% under the 5-fold cross-validation. The experimental result indicates that the model is more effective at detecting fake news.

In this paper[38], by Yang Liu et al., the researcher tied to design fake news detection models using user characteristics rather than merely depend on linguistic features. To construct the propagation path from the given news story, they identified who was involved in the propagation of news and reserve the sequence in a multivariate time series variable. After the multivariate propagation was constructed, it was transformed into a fixed-length multivariate to ensure the input is compatible to be used in the RNN and CNN-Based Propagation path classification model. To evaluate the newly designed model, the researchers used three real-world datasets: namely Twitter15, Twitter16, and Weibo which are well-known social media platforms in the USA and China respectively. After experimenting on these different datasets, they got an accuracy of 85% on Twitter and 92% on Sina Weibo in 5 minutes after it starts to disseminate, which they claim is significantly faster than state-of-the-art baselines.

2.7.1 Summary of Related Works

Table 2.1 Summary of Related Work

Authors & year	Title	Feature extraction	Method & result
Rohit Kumar Kaliyar et at. (2020) [39]	FNDNet-A Deep Convolutional Neural Network for Fake News Detection	GloVe	CNN:91.50 , LSTM: 97.25, FNDNet:98.36 (ACC)
Álvaro Ibrain Rodríguez et al. (2019) [1]	FAKE NEWS DETECTION USING DEEP LEARNING	Word2Vec	LSTM: 0.91, CNN: 0.937, BERT: 0.98 (ACC)
Kai Shu et al. (2019) [40]	dEFEND:Explainable Fake News Detection	GloVe	text-CNN: 0.653, HAN: 0.837, dEFEND: 0.904 (ACC)
Junaed Younus Khan et al. (2019) [41]	A Benchmark Study on Machine Learning Methods for Fake News Detection	GloVe Embedding Character Embedding	CNN:0.58, LSTM:0.54, Char-level C LSTM:0.56 Conv HAN:0.59 (ACC)
Yaqing Wang et al. (2018) [42]	EANN: Event Adversarial Neural Networks for Multi-Modal Fake News Detection	Pre-trained word-embedding	VQA: 0.631, NeuralTalk: 0.610, att-RNN: 0.664, EANN: 0.715 (ACC)

2.8 Summary of the Chapter

This chapter commences by introducing terms and definitions related to fake news. Apart from presenting different definitions from different perspectives considering the controversiality of the term “Fake News”, a categorization of such contents is also covered. Then, the importance of social media literacy in helping the society to get accurate and unbiased information with its context in Ethiopia also stated.

The other things mentioned in this section are different approaches sought in automatically classify a given content as misinformation or not. Various mechanisms like using linguistic and visual information, social context and engagement data, and a hybrid of the content-based and propagation-based approaches are overviewed.

The last part of this section mentions different strategies used in feature extraction from the text content and deep learning architectures used in text classification, as detecting fake news from its content is basically a text classification problem. In the feature extraction part, various extraction mechanisms ranging from bag-of-words to predictive models are covered. Finally, the characteristics of the target language, its hindrance to the automation of fake news classification, and research papers from other languages also reviewed.

As shown in the above table, researchers design various architectures for a different language to capture any explicit or latent feature which may help the models generalize on the input dataset. Detecting fake news from its content rather than social engagement data or its propagation path is an active research area as it is ideal in detecting misinformation as they appear without waiting for metadata related to the story being published. Even though there is several of literature for well-resourced language like English, detecting fake news for the Amharic language is still an open problem.

CHAPTER THREE

3. RESEARCH METHODOLOGIES

This chapter oversees the tools and techniques used to carry through the research objectives proposed. The first section is dedicated to how the data is acquired, prepared, and annotated to be used by various deep learning models. The second section is concerned with the actual procedure used in building and training models, including data preprocessing, feature extraction methods, and model architectures. In the latter part, the model evaluation method used in the research is stated.

3.1 Dataset

Although Amharic is the official language in Ethiopia and has an immense amount of literature either in a digitized or a paper format, it is still a very low-resource language for NLP research. It has limited computational linguistic resources such as datasets and pre-trained models. As the case regarding dataset scarcity in studying Amharic Fake News detection is not different from the above-mentioned problem, it's needed to build a new dataset.

The process of building the dataset for fake news detection has two main steps:

1. Scraping the Amharic posts/news data from a social media platform.
2. Cleaning and filtering each datum to reduce the negative impacts of impurities on the classifier model, then
3. Consolidating and annotating to finalize it on one file as a single dataset.

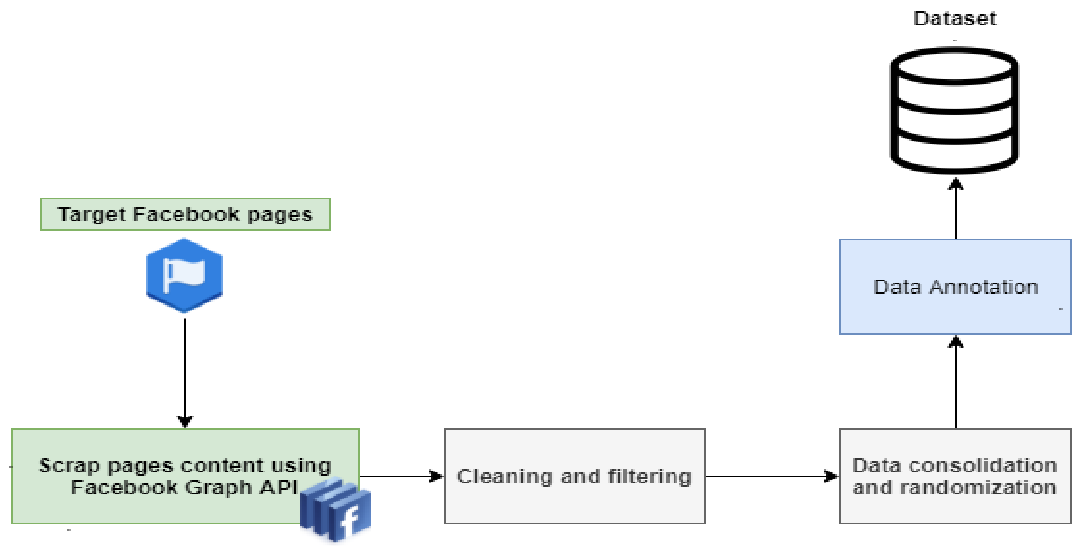


Figure 3.1 Data collection strategy

3.1.1 Data Source

Facebook is the most widely used social networking platform as it helps its users in getting informative content that they care about, in parallel with entertaining videos, memes, and other stuffs. We gathered all the data from it and to get the data out of the platform, we used a Facepager which is Facebook Graph API based data scraping tool. Facebook Graph API is an HTTP-based API used to programmatically query data and performs a wide variety of other tasks.

In gathering data, finding legit public pages is more straightforward than the Fake ones, as the platform gives an authentication mechanism for already known pages and users. But the case is different when it comes to fake pages and users. To optimize the cost in finding potential fake news in the overwhelmingly large social media data, we considered criteria for the page content to be pulled if:

- ✓ At least one of its stories reported as a fake by well-reputed fack-checkers.
- ✓ It has a redirecting link, which leads to a YouTube channel or any other splashy website.
- ✓ It has followers exceeding 10000.
- ✓ It is more concerned about current issues in politics and ethnicity.

3.1.2 Dataset Preparation

The collected data should pass through some data filtering, cleaning, and consolidation processes to alleviate the data quality issue, which may have an undesirable effect on building classification models.

The data items that need to be filtered and cleaned are:

- ✓ Entries that miss some fields.
- ✓ Poorly formatted entries.
- ✓ Duplicated entries.
- ✓ Non- Amharic posts/ entries

After cleansing, all posts from different Facebook pages are integrated into one file to be annotated.

3.1.3 Dataset Annotation

To get most of the authentic news, we used four mainstream media and one trusted social media based news page in Ethiopia. Namely, those are BBC News Amharic, DW Amharic, Ethiopian Broadcasting Corporation, FBC (Fana Broadcasting Corporate S.C.), and DireTube. Even though the selected pages are believed to be authentic, their content was reviewed by our fellow journalists for any ambiguous reports that should no be annotated as authentic news.

Table 3.1 Pages which are potentially a source of authentic news

Page Name	Number of Likes	No of Followers	No of the stories harvested
DireTube	3,026,362	3,049,111	4157
Ethiopian Broadcasting Corporation	1,803,003	2,048,680	4644
FBC (Fana Broadcasting Corporate S.C.	2,053,465	2,324,711	2302
DW Amharic	1,021,398	1,079,027	1950
BBC News Amharic	537,071	611,132	2254
Total number of potentially legit stories collected			20660
Total number of stories labeled as True			6000

For collecting fake news, we tried to include different types of stories as fake news is not monolithic and comes in all shades of gray. This includes:

- ✓ False Context: Any news that contains genuine information with the wrong context
- ✓ Clickbait: Story that uses tricky headlines to create an information gap on the audience aiming at luring the reader to the publisher's website.
- ✓ Satire/Parody: News stories that are deliberately untrue and which are intended for fun and parody

- ❖ A guideline from the news literacy project is adopted for the effectiveness and uniformity of the dataset annotation (
- ❖ **Appendix** A:)

Facebook pages selected as a potential source of fake news are:

Table 3.2 Pages which are potentially a source of Fake News

Page Name	**Number of Likes**	**No of followers**	**No of the stories harvested**
Yeshegertube.net /የሸገር ቲዩብ	320,110	320,911	2833
Tobia Tube/ጦቢያ ቲዩብ	551,979	551,979	3310
AddisTube አዲስ ቲዩብ	218,324	218,324	3027
Yegna Tube - የኛ ቲዩብ	2,304,853	2,300,431	3616
Haro Tube - ሐሮ ቱብ	1,764,636	1,760,583	4546
Tikvah eth	12,083	12,546	343
ምስለ-ዜና Misle-Zena	101,127	101,733	716
ኢኢትዮጵያዊ እውነታዎች - Ethiopian activists a	60,464	60,532	423
የኢትዮጵያ ብልፅግና ፓርቲ-Ethiopian Prosperity Party	14,919	15,147	470
Yegna Tikuret - የኛ ትኩረት	856,778	858,088	2045
Zena24	252,262	252,567	3380
Yehabesha	1,940,979	2,169	1665
Amhara News/ የአማራ ዜና	42,700	43,539	1439
Ethio Media	224,076	232,213	1944
ethiopia24news	697,548	725,391	3554

Ethiopianege	34,012	34,671	1125
Gihon Newspost	10,492	10,593	785
Prosperity News	105,359	107,983	401
ShegaMedia.com	60,194	60,224	1242
Vote abiy	99,251	224,883	547
	Total number of potentially Fake stories		**37411**
	Total number of stories labeled as fake		**6000**

3.2 Fake news Detection Modeling

3.2.1 Preprocessing

Even though Preprocessing steps are essential, applying all the commonly used steps may put the detection model in trouble. For example, stopword removal could be advantageous in document classification as the text's central idea can be recognized even if the stopword is removed; and which definitely leads to improved performance.

Figure 3.2 The most frequent words from the dataset labeled as authentic

But as the problem under study is different and as the data visualized in the word cloud above shows; stopwords appeared to be common in authentic news peculiarly, which could be an important feature in fitting the deep learning models for better classification result. So by considering such an adverse effect of some preprocessing steps, we adhered to apply only other preprocessing techniques such as cleaning, normalization, and tokenization.

Figure 3.3 The most frequent words from the dataset labeled as Fake

3.2.1.1 Cleaning

To remove unnecessary content and make the data more representable by the word embedding to be employed, we shall go over some cleaning steps. This cleaning procedure will get rid of all irrelevant special characters, symbols, and emojis. Also, the process removes all non-Amharic characters.

OBTAIN: *unprocessed dataset*

OUTPUT: *clean dataset*

INIT:

1: Read the dataset

2: WHILE*(it is not the end of file):*

> ***IF*** *text contain special characters and* stmbols [',', '.', '"', ':', ')', '(', '-', '!', '?', '|', ';', '"', '$', '&', '/', '[', ']', '>', '%', '=', '#', '*', '+', '\\', '•', '~', '@', '£',
>
> '·', '_', '{', '}', '©', '^', '®', '`', '<', '→', '°', '€', '™', '›', '❤', '←', '×', '§', '″', '′', 'Â', '█', '½'] ***THEN***
>
> *Eliminate characters*
>
> ***IF*** *text contain emojis*[😀, 😄, 🙈, 😛...] ***THEN***
>
> *Eliminate emojis*
>
> ***IF*** *text contain* [tabs,extra white space] ***THEN***
>
> *Eliminate*
>
> ***IF*** *text contain* [a-z A-Z] [0-9] &[[፩ ፪ ፫ ፬ ፭ ፮ ፯ ፰ ፱ ፲ ፳ ፴ ፵ ፶ ፷ ፸ ፹ ፺ ፻] ***THEN***
>
> *Eliminate English word, Arabic number, and Ethiopic numbers*
>
> ***Return*** *processed text*
>
> ***ENDIF***

3:HALT

3.2.1.2 Normalization

Amharic got a complex inflectional morphology. Especially when verbs are constructed, as they employ unpredictable prefixes and suffixes, it leads to a different spelling for the same word. For example, "ሰርቁዋለ" and "ሰርቁአል" could be normalized to "ሰርቋል".

Besides that, redundant letters in Amharic scripts can also make it difficult to represent each differently spelled word. While the word's contextual meaning is still the same, different people write in another way. For example, "ጸሀይ", "ጸሃይ", and "ጸኀይ" could be normalized to "ፀሀይ"

OBTAIN: unprocessed dataset

OUTPUT: Normalized dataset

INIT:

1: Read the dataset

***2: WHILE**(it is not the end of file):*

IF text contains [ሃኀኃሐሓኻ]' ***THEN*** *replace with* 'ሀ'

IF text contains [ሑኁዅ] ***THEN*** *replace with* 'ሁ'

IF text contains [ኂሒኺ] ***THEN*** *replace with* 'ሂ'

IF text contains [ኄሔዄ] ***THEN*** *replace with* 'ሄ'

IF text contains [ሕኅ] ***THEN*** *replace with* 'ህ'

IF text contains [ኆሖኾ] ***THEN*** *replace with* 'ሆ'

IF text contains [ዓኣዐ] ***THEN*** *replace with* 'አ'

IF text contains [ሠ ሡ ሢ ሣ ሤ ሥ ሦ] ***THEN*** *replace with* [ሰ ሱ ሲ ሳ ሴ ስ]

IF text contains [ዑ ዒ ዓ ዔ ዕ ዖ ኣ] ***THEN*** *replace with* [ኡ ኢ አ ኤ እ ኦ አ]

IF text contains [ጸ ጹ ጺ ጻ ጼ ጽ ጾ] ***THEN*** *replace with* [ፀ ፁ ፂ ፃ ፄ ፅ ፆ]

IF text contains [ሉ[ዋአ]] ***THEN*** *replace with* [ሏ]

IF text contains [ሙ[ዋአ]] ***THEN*** *replace with* [ሟ]

IF text contains [ቱ[ዋአ]] ***THEN*** *replace with* [ቷ]

IF text contains [ሩ[ዋአ]] ***THEN*** *replace with* [ሯ] . . .

Return *processed text*

ENDIF

3 HAlT

3.2.1.3 Tokenization

After some preprocessing techniques clean the dataset, features in the data set should take vectors form. Here the next process is applied, tokenization. The input to deep learning models should be tokenized as text $T = \{t_1, \ldots t_N\}$, where each word is represented by the fastText embeddings.

3.2.2 Feature Extraction Methods

Feature extraction methods based on state-of-the-art text mining; techniques will be applied for reducing redundant detection problems, features, and dimensionality.

Word embedding

Word embeddings are a feature representation mechanism that allows words with similar contexts to have similar values in the vector space. The mechanism word embedding employ makes them ideal for the current state-of-the-art deep learning algorithms, and this brought an improvement in the performance of many NLP challenges.

Even though word embeddings are good at representing textual data, the amount of data and computational resources needed is very high. Especially for low-resourced language, like the Amharic one, training custom embeddings with a little resource may end with non-representative vectors for each word in the corpora.

fastText

To lessen the problem faced by such low-resourced language, this study preferred to use fastText. fastText is a library for the learning of word embeddings and text classification created by Facebook's A.I. Research (FAIR) lab. The model allows us to create unsupervised learning or supervised learning algorithms for obtaining vector representations for words. Facebook makes available pre-trained models for 294 languages. fastText uses a neural network for word embedding.

Although fastText can be used as a classification model, this study used only embeddings to evaluate the performance of classification on other deep learning algorithms. fastText utilize both bag-of-words and bag of n-grams. To produce efficient outputs in using a bag of n-grams, a hashing technique to map n-grams is used.

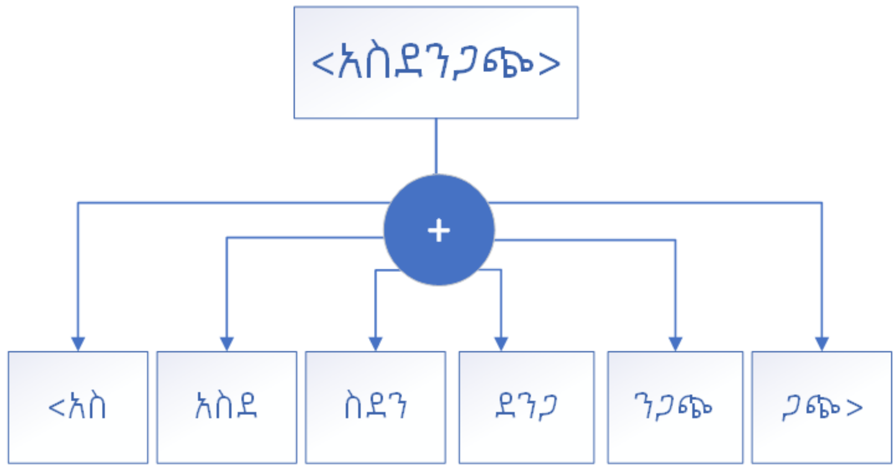


Figure 3.4 3-gram character representation in fastText

fastText is a robust embedding based on a continuous skip-gram model[43][26], which uses subword information, while other word embedding models such as GloVe and Word2vec utilize the entire word. Because GloVe and Word2vec models need a word to exist in the embedding vocabulary, the unseen word is ignored. To alleviate this problem and take into account the morphology of words and the parameter sharing of morphologically rich languages like Amharic, there must be a way to incorporate subword information for better representation.

Consequently, fastText roots its learning to the lower level n-grams so that each word is taken as a collective sum of n-gram characters, and these n-grams are the defining entity of the word to be embedded.

For example, taking an Amharic word "ወሳኝ", "ተለቀቀ", "አስታወቁ", and 3-gram representation of a word, embedding could be depicted as:

- ✓ ወሳኝ: <ወሳ,ወሳኝ,ሳኝ> and the unique sequence <ወሳኝ>
- ✓ ተለቀቀ: <ተለ,ተለቀ,ለቀ,ቀቀ> and the unique sequence <ተለቀቀ>
- ✓ አስታወቁ: <አስ,አስታ,ስታወ,ታወቁ,ወቁ> and the unique sequence <አስታወቁ>

Here, the word itself and the bag of subword are treated as the n-grams[43]. Because fastText explores subword, it appears to Fill the Vacuum created by other word embedding models like GloVe and Word2Vec. So, this will help in the representation of out-of-vocabulary(OOV) words and help better to handle the morphological richness of the Amharic language.

3.2.3 Models

After the dataset is preprocessed and represented in numeric form using pre-trained Amharic fastText word embeddings, the data is given to different deep learning classification models to evaluate each model's performance as shown in the figure below.

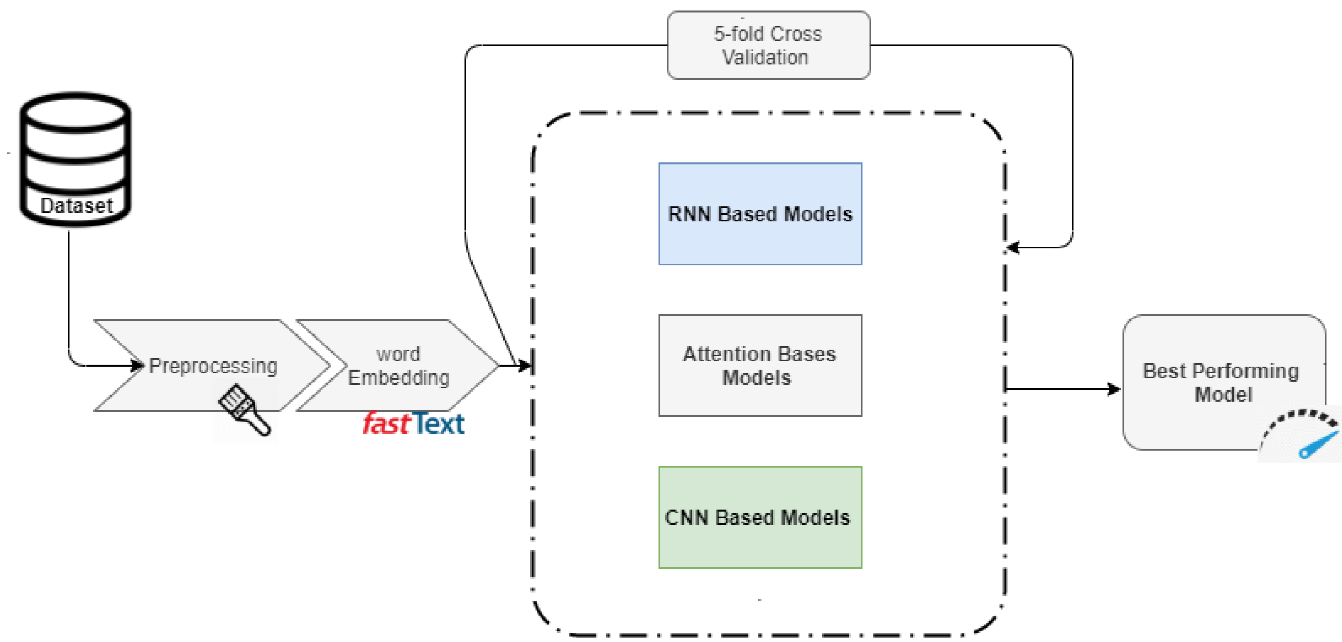


Figure 3.5 Amharic Fake News Detection Modeling

3.2.3.1 RNN-Based Models

The most well-entertained approach to NLP in Deep Learning is based on a recurrent model. RNN-based models are good in interpreting the text as a sequence of words, which makes them perfect for capturing word dependencies and text categorization. Nevertheless, vanilla RNN models do not perform admirably and may sometimes underperform a simple feed-forward neural network. Among numerous variations of RNNs, Long Short-Term Memory (LSTM) is the foremost prevalent variant, which is outlined to superior capture long term conditions[44]. The secret behind the LSTM model is its capability to address the gradient vanishing and exploding issues endured by the vanilla RNNs. By presenting a memory cell to keep values over arbitrary time intervals, and three gates (input gate, forget gate, output gate), LSTM can control the stream of information in the cell state.

The other competent variant is Gated recurrent units (GRUs). GRUs are fast to train and do well on a considerably small amount of training data. Besides LSTMs ad GRUs, in this study, an extension of these traditional models such as bidirectional LSTM and GRU is accessed to see if the performance on Amharic Fake News detection is improved.

3.2.3.2 Attention-Based Models

Even though RNN-based models have shown excellent performance on tasks related to text classification, However, these models are not intuitive enough for poor interpretability[45]. The attention mechanisms have shown a performance improvement with interpretability and which makes it favored in some cases.

3.2.3.3 Convolution Neural Networks Based Models (CNNs)

Convolution Neural Networks (CNNs), originally invented for computer vision, has shown superb performance on text classification tasks as well as other traditional Natural Language Processing (NLP) tasks, even when considering relatively simple one-layer models[33]. Besides its performance, researchers also have tried to understand the inner working of the network that makes it preferable in some NLP tasks; especially in text classification. In a paper[33] by Alon Jacovi et al., the researchers examined a hypothesis that claims filters accompanied by global max-pooling could be used as ngram detectors. From their finding, they able to conclude that different semantics of ngrams can be captured by filters with different activation patter and global max-pooling.

3.3 Model Evaluation and Prototyping

To see if the proposed model is performing well on unseen data and trust its prediction, a model evaluation should be done. Evaluating a mode is an important step in the machine learning pipeline to assess whether the model is merely memorizing the data or not. And more importantly to evaluate the generalization capability of the model on new out-of-sample data, which the model doesn't see in the process of fitting/ training the model.

There are two approaches to evaluate the given model performance. The first and straightforward method is the holdout method, where the evaluation strategy is based on testing a model on data that was not seen by the model at training time. Even though the approach is advantageous in terms of simplicity, speed, and flexibility; it has a downside in resulting differences in estimating accuracy as a result of its high variability among the training, validation, and test sets. The second and widely used approach is k-Fold Cross-Validation. This method is known for its skill in the less biased or less optimistic estimation over the previous simple train/split approach[46]. And also significantly reduces variance. As a result, k-Fold Cross-Validation is selected as a model evaluation method in this research.

3.3.1 k-Fold Cross-Validation

In this approach, the entire dataset is partitioned into k equal-sized groups or folds. And each fold is treated as a validation set, while the rest k-1 is used as a training set to fit the model[47]. Here the parameter k decides the number of fold that a given data sample is to be split into.

The general procedure for k-fold cross-validation is as follows:

1. Data randomization.
2. k fold splitting of the data
3. For each group:
    a. Take the group test data set
    b. Take the left groups as a training data set
    c. Fit the model
    d. Hold the evaluation score and drop the model
4. Summarize the overall evaluation scores

3.3.2 Classification Metrics

To evaluate the performance of our model, there are a variety of classification metrics such as Confusion matrix, Accuracy, precision, Recall/Sensitivity, and F1-score. The last four listed metrics are merely dependent on the confusion matrix and the numbers inside it.

3.3.2.1 Confusion Matrix

A confusion matrix is a table that is often used to describe the correctness and accuracy of the model. The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing. Basic terms are:

- ✓ ***True positives (T.P.)*:** These are cases in which we predicted Fake (the news articles are Deceptive), and they actually are.
- ✓ ***True negatives (T.N.)*:** We predicted Real, and the articles are actually legit.
- ✓ ***False positives (F.P.)*:** We predicted Fake, but the news articles are not actually deceptive (Also known as a "Type I error.")
- ✓ ***False negatives (F.N.)*:** We predicted Real, but the news articles are actually deceptive. (Also known as a "Type II error.")

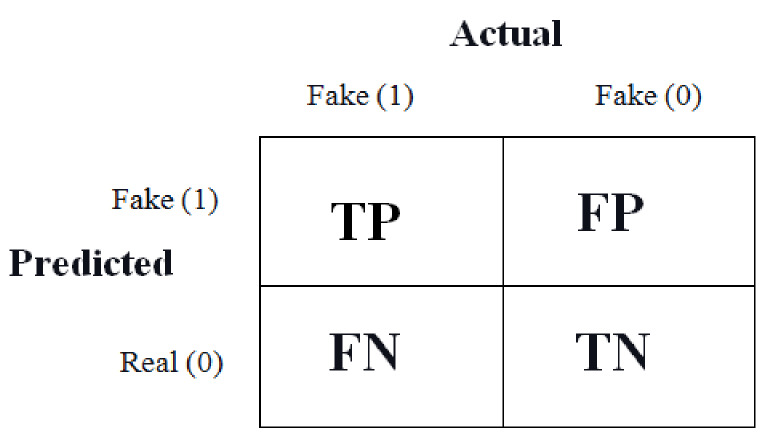


Figure 3.6 Confusion matrix

3.3.2.2 Accuracy

As our dataset class distribution is balanced, using accuracy as an evaluation metric is fairly harmless in portraying how the model is good for detecting the positive and negative class. Accuracy is computed as the sum of True Positive and True Negative divided by the overall number of instances in the dataset.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad 3.1$$

3.3.2.3 Precision

Precision is a measure of the likelihood of getting correct positive class classification. It is computed as the number of True Positives divided by the number of True Positive plus False Positive.

$$Precision = \frac{TP}{TP + FP} \qquad 3.2$$

3.3.2.4 Recall

The recall is the measure of how sensitive our model is in identifying the positive class. A recall is computed as the number of True Positive divided by the number of True Positives plus False Negatives.

$$Recall = \frac{TP}{TP + FN} \qquad 3.3$$

3.3.2.5 F1-score

F1-score is a harmonic mean or weighted average of the model's precision and recall. F1-score is more preferable in measuring Accuracy when False Negatives and False Positives are crucial than the True Positives and True Negatives.

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \qquad 3.4$$

3.3.3 Prototyping

After the models are trained and validated on each sample of cross-validation, the best performing classifier is deployed on the webserver. Before the trained model is being used by the users to check news whether fake or real, it should be saved in a format that can be accessed from the server. In our case, we saved it in a .hdfs file format. Then, the saved model is loaded on the Flask web server so that it became reachable through the WSGI server to the users.

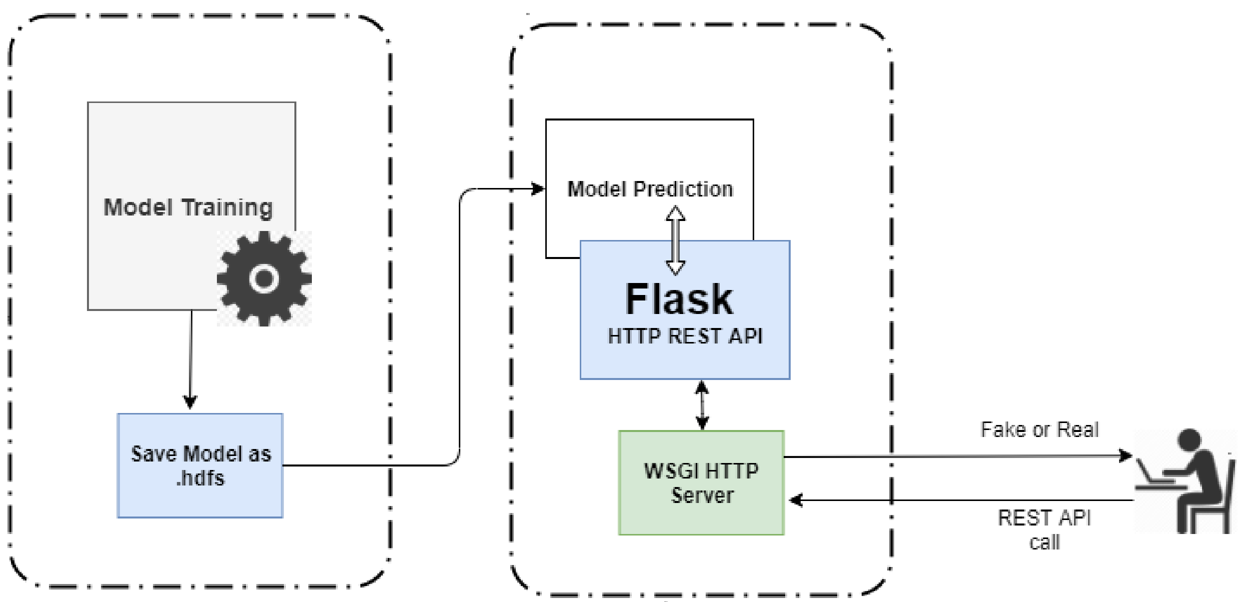


Figure 3.7 Deployment Architecture for Amharic Fake News Detection

CHAPTER FOUR

4. DESIGN AND IMPLEMENTATION OF FAKE NEWS DETECTION MODEL

In this chapter, we discuss how the models to be evaluated are implemented and the tools needed for experimentation. Before showing data preprocessing and actual implementation of the models, we try to list the software packages and tools used and why a specific one is chosen.

4.1 Tools

4.1.1 Data preparation and processing Tools

Facepager 4.0

FacePager is an open-source tool for collecting public data from social media platforms such as Facebook, Twitter, or YouTube. It is used in the data collection process to build the dataset. The collected data could be stored on a standard database like SQLite and can be extracted in a CSV file.

LibreOffice Calc 6.4.1.2

LibreOffice Calc is Used in cleaning filtering tasks like managing Missing entries, removing duplicates, labeling the dataset, and integrating all separate files into one dataset.

Numpy[6] 1.19.1

Numpy is a python library that adding support for large, multi-dimensional arrays and matrices along with a capability to operate on these data structures. Since data manipulation is a core component in data sciences and machine learning, NumPy is used as a backbone in many other different libraries also.

[6] *https://numpy.org/*

pandas[7] 1.1.1

pandas is a toolkit based on python, which presents a diverse range of utilities, ranging from parsing multiple file-formats to converting an entire data table into a NumPy matrix array.

scikit-learn[8] 0.23.2

scikit-learn is a free machine learning library for Python based on SciPy: A python-based ecosystem of open-source software for mathematics, science, and engineering. In this study, we used scikit-learn to train our model with k-fold cross-validation and the generation of classification report based on each fold's metrics results.

4.1.2 Package manager and Environments

Anaconda Navigator[9] 1.9.12

Anaconda is a distribution of packages for data science which makes package and environment management less bothering. As it helps with creating different environments, training with different python and package versions is simple.

Spyder[10] 4.1.5

Spyder a.k.a (The Scientific Python Development IDE) is a very customizable and interactive open-source cross-platform IDE for scientists, data analysts, and engineers.

Colab

Colab is Google's Jupyter notebook environment that runs on the cloud which allows clients to use free GPUs and TPUs. It helps to run very resource and time consuming deep learning codes efficiently. Moreover, it doesn't need to install any package explicitly.

---

[7] *https://pandas.pydata.org/*

[8] *https://scikit-learn.org/stable/*

[9] *https://anaconda.org/*

[10] *https://www.spyder-ide.org/*

CUDA

CUDA is a platform for parallel computing and programming, which makes GPUs simpler and more effective in use for general-purpose computing.

4.1.3 Modeling Tools

Python[11] 3.7

As of the StackOverflow analysis[48], python is getting more attention in recent years because of its easiness and fast experimentation. Its easiness, especially for the research community, is very helpful as it allows the researchers to focus on their objective instead of learning complex language and documentation. Being an interpreted language and increased productivity also gets python to be likable by most researchers and programmers. Besides that, the availability of thousands of stable third-party libraries and an active community makes it ideal for research. For this thesis, we used a stable version of python (which is 3.7).

TensorFlow[12] 2.1.0

TensorFlow is a machine learning framework from Google Brain Team, with a comprehensive, easily integrable ecosystem of tools, packages, and community. Because TensorFlow has an accessible and readable syntax, but at the same time with more flexibility, it's preferred over the other deep learning platforms. Aside from its flexibility, TensorFlow is given the researcher more network control and understanding of operations done by a specific model. TensorFlow version 2.1.0 is used for this study.

Keras[13] 2.3.1

Keras is a fast experimentation deep learning API written in python. It runs on top of TensorFlow more focusing on backing the research process by easily interfacing the researcher with artificial neural networks[49]. Because of its consistent & simple high-level API, working with Keras is speedier and allows us to do more experiments with less strain. However, Keras is also Flexible at the same time as integrates deeply with low-level

[11] *https://www.python.org/*

[12] *https://www.tensorflow.org/*

[13] *https://keras.io/*

TensorFlow core functionality. In this research, Keras 2.3.1 is used which comes with its backend TensorFlow.

cuDNN[14]

cuDNN is a GPU-accelerated library that provides highly tuned implementations for standard routines such as forward and backward convolution, pooling, normalization, and activation layers. Using such a library allows the researcher to spend more time on training the model rather than spending on low-level GPU performance tuning.

Pre-trained fastText[15]

fastText is Facebook's word embeddings and text classification library. It exploits subword information to construct word embeddings, which are a high-dimensional array of numbers represented in the vector space. Training word embedding models is a costly process in terms of both computational resources and time. To alleviate the problem the Facebook's AI Research lab makes available pre-trained models for more than 290 languages and the Amharic pre-trained model is one of them.

4.1.4 Hardware Tools

To see results from the experiments we use a personal computer with an Intel® processor Core™ i5-5200U CPU @2.20GHz, 8 Gigabyte of physical memory, GPU: NVIDIA GeForce GTX 950M with dedicated 2GB of RAM, 1 Gigabyte hard disk storage capacities, and 64 bits Windows 10 Pro operating system.

4.1.5 Deployment Tools

Flask 1.1.2 [16]

After the experiments are done the best performing model is deployed on a web server, where it can be used to make a real-time prediction by providing the story needed to be verified. To realize this, we used a flask micro web framework. Since we are deploying a deep learning

[14] *https://developer.nvidia.com/cudnn*

[15] *https://fasttext.cc/docs/en/crawl-vectors.html*

[16] https://palletsprojects.com/p/flask/

model that is based on a simple HTTP request–reply in text form, Flask is more favorable over other Heavyweight frameworks.

4.2 Preprocessing and Embedding Matrix Preparation

4.2.1 Loading Dataset

Before doing any pre-processing on the data, the data should be made available to the python environment we are doing. To achieve this goal, we use Pandas as it supports to load data as DataFrames. DataFrames are known for high-speed computation rather than other formats. The pandas method read_csv reads file in the commas separated value file and load it as DataFrames, where instances of the data are accessible via column name.

```
import pandas as pd
df = pd.read_csv('D:/datasets and models/datasets/FAKE NEWS/FB data/\
                  sample-processed/12000_ds/train-test/\
                      Fake_News_All_c.csv', sep=',', header=0)

y = df['label'].values
posts = df['posts'].tolist()
```

Figure 4.1 Implementation of data loading using pandas

4.2.2 Cleaning the data

```
from tqdm import tqdm
from keras.preprocessing.text import Tokenizer
from nltk.tokenize import RegexpTokenizer

tokenizer = RegexpTokenizer(r'\w+')
clean_this=([',', '.', '"', ':', ')', '(', '-', '!', '?',\
        '|', ';', "'", '$', '&', '/', '[', ']', '>', \
        '%', '=', '#', '*', '+', '\\', '•', '~', '@', '£',\
        '·', '_', '{', '}', '©', '^', '®', '`', '<', '→', '°', '€',\
           '™', '›', '♥', '←', '×', '§', '″', '′', 'Â', '█', '½', 'à', '…',\
        '“', '★', '”', '–', '●', 'â', '►', '−', '¢', '²', '¬', '░', \
           '¶', '↑', '±', '¿', '▾', '═', '¦', '║', '―', '¥', '▓', '—', '‹', '─',\
        '▒', '：', '¼', '⊕', '▼', '▪', '†', '■', '’', '▀', '¨', '▄',\
           '♫', '☆', 'é', '¯', '♦', '¤', '▲', 'è', '¸', '¾', 'Ã', '⋅', '‘', '∞', \
        '∙', '）', '↓', '、', '│', '（', '»', '，', '♪', '╩', '╚', '³', \
           '・', '╦', '╣', '╔', '╗', '▬', '❤', 'ï', 'Ø', '¹', '≤', '‡', '√',\
        'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P',\
           'Q','R','S','T','V','U','W','X','Y','Z','a','b','c','d','e','f',\
              'g','h','i','j','k','l','m','n','o','p','q','r','s','t','v','w','x','y','z'

processed_post_train = []
for doc in tqdm(posts):
    tokens = tokenizer.tokenize(doc)
    filtered = [word for word in tokens if word not in clean_this]
    processed_post_train.append(" ".join(filtered))
```

Figure 4.2 Implementation of data cleaning

4.2.3 Normalization

```
def replaceMultiple(mainString, toBeReplaces, newString):
    # Iterate over the strings to be replaced
    for elem in toBeReplaces :
        # Check if string is in the main string
        if elem in mainString :
            # Replace the string
            mainString = mainString.replace(elem, newString)

    return  mainStrin
```

```
url_regex = r'(https?:\/\/(?:www\.|(?!www))[a-zA-Z0-9][a-zA-Z0-9-]+[a-zA-Z0-9]\.[^\s]{2,}|www\.[a-zA-Z0-9][a-zA-Z0-9-]+\
[a-zA-Z0-9]\.[^\s]{2,}|https?:\/\/(?:www\.|(?!www))[a-zA-Z0-9]+\.[^\s]{2,}|www\.[a-zA-Z0-9]+\.[^\s]{2,})'
text = re.sub(url_regex, "<URL>", posts)
Normalized_post = replaceMultiple(text, ['ሃ','ኅ','ኃ','ሐ','ሓ','ኻ'] , "ሀ")
Normalized_post = replaceMultiple(text, ['ሑ','ኁ','ዅ'] , "ሁ")
Normalized_post = replaceMultiple(text, ['ኂ','ሒ','ኺ'] , "ሂ")
Normalized_post = replaceMultiple(text, ['ሕ','ኅ'] , "ህ")
Normalized_post = replaceMultiple(text, ['ኆ','ሖ','ኾ'] , "ሆ")
Normalized_post = replaceMultiple(text, ['ኆ','ሖ','ኾ'] , "ሆ")
Normalized_post = replaceMultiple(text, ['ዓ','ኣ','ዐ'] , "አ")
Normalized_post = replaceMultiple(text, ['ሠ','ሡ','ሢ','ሣ','ሤ','ሥ','ሦ'] , ['ሰ','ሱ','ሲ','ሳ','ሴ','ስ'])
Normalized_post = replaceMultiple(text, ['ዑ','ዒ','ዓ','ዔ','ዕ','ዖ','ኣ'] , ['ኡ','ኢ','አ','ኤ','እ','ኦ','አ'])
Normalized_post = replaceMultiple(text, ['ጸ','ጹ','ጺ','ጻ','ጼ','ጽ','ጾ'] , ['ፀ','ፁ','ፂ','ፃ','ፄ','ፅ','ፆ'])
```

Figure 4.3 Implementation for normalizing Amharic characters

4.2.4 Tokenization

The data that has been cleaned and normalized should be tokenized so that the model could understand the context. To achieve this we used the Keras Tokenizer by specifying the maximum number of words to be indexed. The fit_on_texts Methos from the Tokenizer class generates the vocabulary index based on word frequency. Next, the method texts_to_sequence creates a sequence of representative numbers based on the word index generated from the fit_on_texts method. Finally, as neural networks require inputs with the same shape and size, we need to pad the sequence.

```
from keras.preprocessing.text import Tokenizer
from keras.preprocessing import sequence
MAX_NB_WORDS = 100000

tokenizer = Tokenizer(num_words=MAX_NB_WORDS, lower=True, char_level=False)
tokenizer.fit_on_texts(processed_post_train)
word_seq_train = tokenizer.texts_to_sequences(processed_post_train)
word_index = tokenizer.word_index
print("dictionary size: ", len(word_index))
word_seq_train = sequence.pad_sequences(word_seq_train, maxlen=max_seq_len)
```

Figure 4.4 Implementation for Tokenization

The maximum number of sequences used for padding is calculated from the overall word distribution as depicted in figure 17.

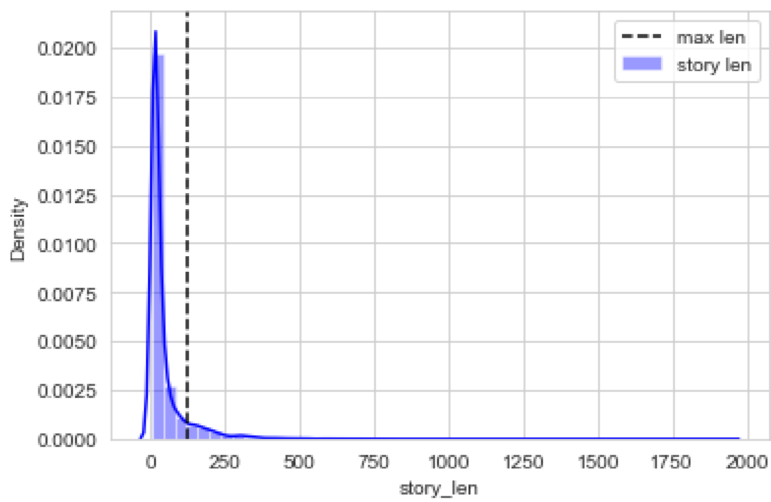


Figure 4.5 Word Distribution

4.2.5 Embedding Matrix Preparation

As computers can't understand a text in a sense as we humans do, we should find a way to represent our pre-processed dataset in a numerical form. In this study, we used a pre-trained Amharic word embedding from fastText. So, before the pre-trained vectors are assigned to the available word indexes from our dataset, the embedding must be loaded to memory as shown in figure 18.

```
import numpy as np
import  codecs
from tqdm import tqdm

embeddings_index = {}
f = codecs.open('D:/datasets and models/trainded models/fastText/amharic/cc.am.300.vec', encoding='utf-8
for line in tqdm(f):
    values = line.rstrip().rsplit(' ')
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    embeddings_index[word] = coefs
f.close()
```

Figure 4.6 Implementation for Loading Word Embedding

Now the coefficients of every embedding can be accessed by the word itself as a key from the embedding_index dictionary. The next step is to prepare the embedding matrix for all words in the word index from our data set. Each tokenized word form word index is cross-checked with the embedding_index to see if its match is found in the pre-trained one. Then if the match exists the corresponding coefficients will be assigned to the word otherwise it all-zero will be assigned. After all word in the iteration finished, an embedding matrix which can be used by the deep learning model is generated.

```
import numpy as np
MAX_NB_WORDS=100000
embed_dim=300

words_not_found = []
nb_words = min(MAX_NB_WORDS, len(word_index))
embedding_matrix = np.zeros((nb_words, embed_dim))
for word, i in word_index.items():
    if i >= nb_words:
        continue
    embedding_vector = embeddings_index.get(word)
    if (embedding_vector is not None) and len(embedding_vector) > 0:
        embedding_matrix[i] = embedding_vector
    else:
        words_not_found.append(word)
```

Figure 4.7 Implementation for Embedding Matrix Preparation

4.3 Model Implementation

After we got a pre-processed data and embedding matrix, which the deep learning model requires to be trained, now it is possible to implement all the architectures considered in this study.

4.3.1 Recurrent-based Design

Embedding layer: Totally we got 84770 words from the process of tokenizing the cleaned data, which needs to be represented in a dense continuous vector space as texts are a very spare form of data to be analyzed by deep learning models. To achieve this, we use the Keras Embedding layer by specifying its parameter to:

`Embedding(nb_words,embed_dim,weights=[embedding_matrix],input_length= max_seq_len, trainable=True))` where `nb_words` set to the size of vocabulary in the Amharic Fake News dataset and `embed_dim` is 300 which is the same as the dimension of the pre-trained fastext dimension to represent each word. We also specified the input`_length` to 124 which is a maximum sequence length that should be supported by the Embedding layer. The maximum sequence length is calculated by taking the mean of all stories length in the dataset and adding a standard deviation value to make sure of not losing important words. We let the layer learn the context by setting `trainable` parameter to `True`, as it updates the pre-trained weight according to the training dataset.

SpatialDropout1D:

Here, using the standard dropout technique for our dataset may not be sufficient regarding the reduction of co-adaption between neurons so that the model will no overfit and behave undesirably on unseen data. To alleviate the problem we used `SpatialDropout1D`, which is originally introduced for object localization problems by Jonathan Tompson et al.[50]. Because of the intra-class imbalance among the clickbait and other forms of fake news in our dataset, using spatial dropout may help the network to not overly learn from the imbalanced instances in a way that affects the generalization of the model. As shown in figure 4.8 ignoring randomly some of the features from frequently occurring word sequences in catchy stories may encourage the model to not memorize the pattern and not classify genuine stories with a word like “ሰበር” automatically to an inappropriate class without treating it exclusively.

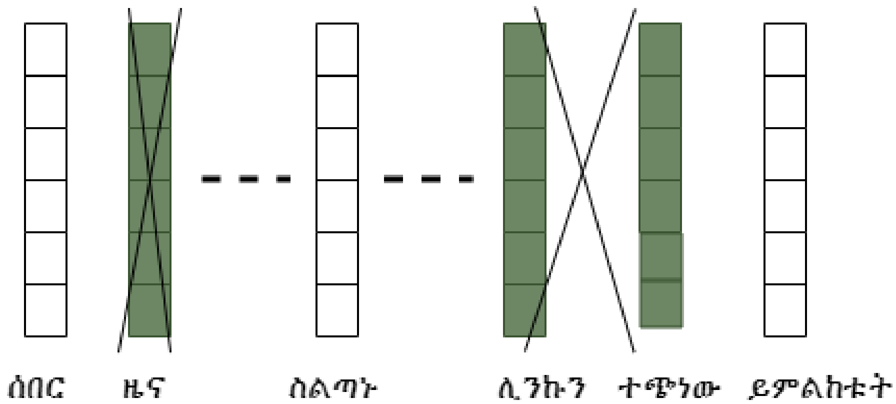


Figure 4.8 Spatial dropout

Recurrent layer:

As depicted in figure 4.8, different variants of the recurrent neural network are experimented with the collected dataset. We used Keras recurrent APIs (`SimpleRNN, LSTM, Bi-LSTM, GRU, Bi-GRU).` We set the dimensionality of the output to 128 by letting the other argument to default.

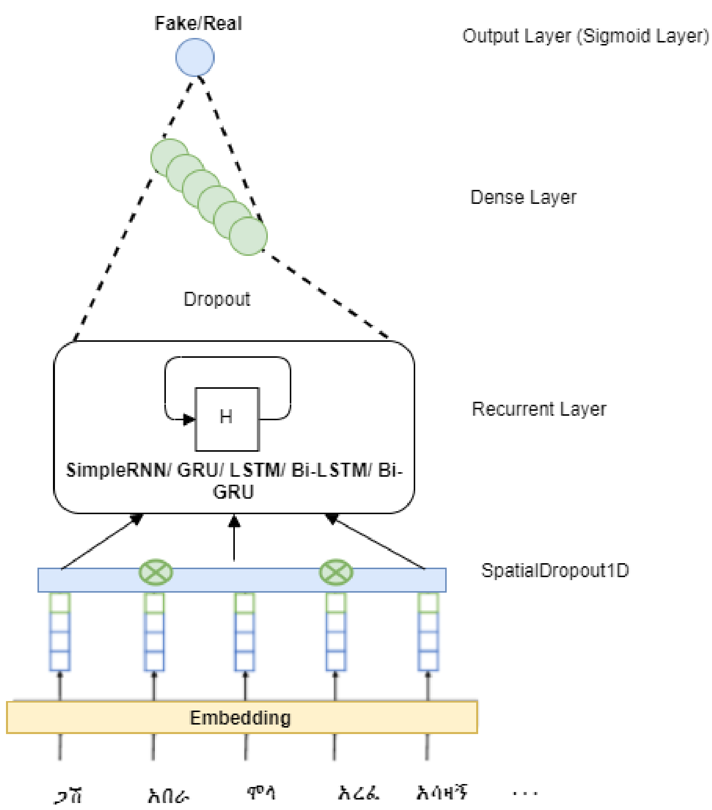


Figure 4.9 Recurrent-based Model Architecture for Amharic Fake News Detection

Dropout Layer and Dense Layer:

As recurrent networks specially LSTMs are prone to overfitting, a `Dropout` layer with a 50% rate is added after the output from LSTM units to regularization it. With this regularization technique, a possible adverse consequence of vanishing gradient may also be prevented. To learn the actual patterns gotten from the previous LSTM layer, a `Dense` layer with one output is imployed. The output of the last neuron is calculated with activation of the `sigmoid` function as the problem at hand is a binary classification.

4.3.2 Attention Mechanism

In this architecture, everything goes parallel like the recurrent-based model from the embedding layer to the bidirectional `LSTM/GRU` layer as shown in figure 4.11. where the first layer is responsible to embed the news stories in a proper representation that the model could use, then followed by `SpatialDropout1D` to lessen co-adaption among adjacent neurons and finally bidirectional `LSTM/GRU` layer which helps the model to get high-level features. After the recurrent layer, we add an attention mechanism to lower the negative impact of biased attentive behavior of the traditional recurrent model while addressing subsequent words.

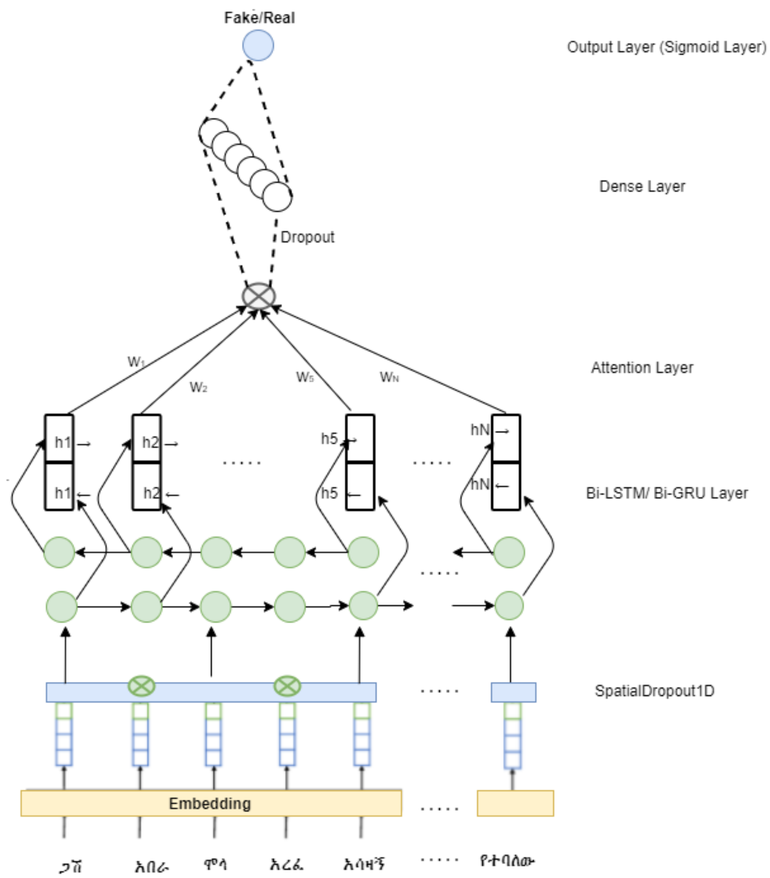


Figure 4.10 Attention-based Model Architecture for Amharic Fake News Detection

Attention layer:

As recurrent networks work by adding the current node hidden state to the next one at each time step, the hidden states closer to the end of news stories hold more momentous information that will lead to a favored distribution of weight to the last words in the sequence. To mitigate the issue with traditional RNN caused by assigning larger weights to the later sequences, an attention mechanism is employed. Instead of relying on the last sequence, the attention mechanism works by giving a focus to the more significant sequences by using bias alignment. This capability is achieved by initializing weights to each input and then by updating the weights after seeing the correlation of the input with the final prediction.

4.3.3 Convolutional Neural Networks

Convolution layer:

To get a feature vector that ideally represents the text at training time, we applied a one-dimensional convolution kernel of size 7 which will be a dimension of 7X300 and 64 different channels of the same size by specifying parameters to Keras Conv1D as:

```
Conv1D(num_filters, 7, activation='relu', padding='same'))
```

Besides that `relu` is specified to apply an activation after convolutions are performed and to make sure the convolution output has the same size as the input, it padded to `same.`

MaxPooling layer and convolution over local features :

After a feature vector, which represents the local features in a sequence of word embedding, is gotten from the convolution step, one-dimensional maxpooling is applied. A pooling size of 2 is used to form a high-level feature representation among two consecutive words regardless of the location across the larger input sequence by discarding less-relevant local information. After local features are mapped, the same convolution applied earlier on inputs is used.

GlobalMaxpooling1D and Dense Layer:

Following the convolution, a global max-pooling is applied to the entire sequence filters one by one and the pooled values are fed to the fully connected dense layer with a drop-out rate of 0.5. Finally, a `sigmoid` function is used as an output neuron to classify whether a given story is fake or real.

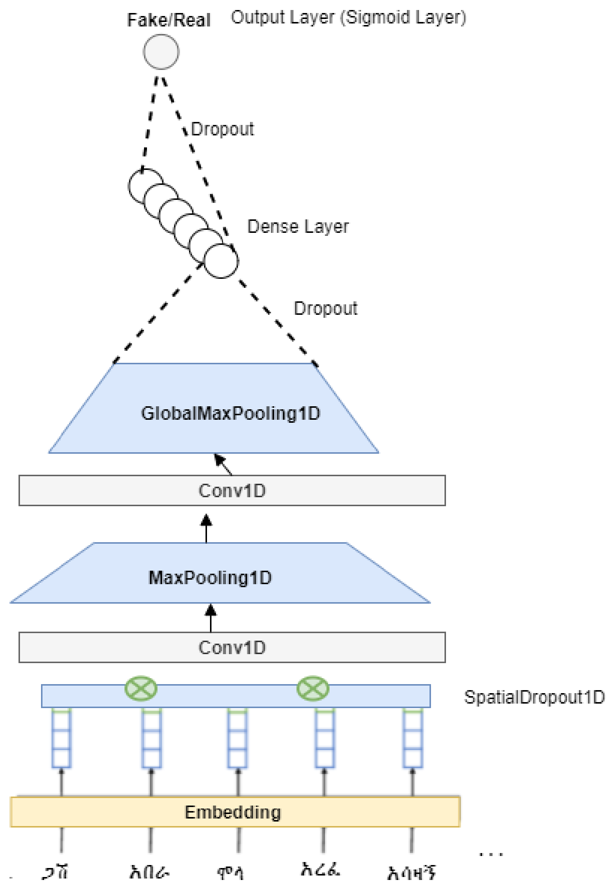


Figure 4.11 CNN-based Model Architecture for Amharic Fake News Detection

CHAPTER FIVE

5. RESULT AND DISCUSSIONS

In this chapter, the experimental results are presented. To access our models, 5-fold cross-validation is used and based on the validation result a confusion matrix and classification report is generated. And also the effect of Normalizing the data is accessed on the top two performing models. We used a sklearn library for the model evaluation reports. Finally, hyperparameter tuning is done on Convolutional Neural Network, as it exhibits a better performance in detecting Amharic Fake News.

5.1 Performance of a Classification

5.1.1 Confusion Matrix for Recurrent Based Models

Different variants of the recurrent models ranging from the Vanilla RNN to Bidirectional Recurrent networks are evaluated.

SimpleRNN:

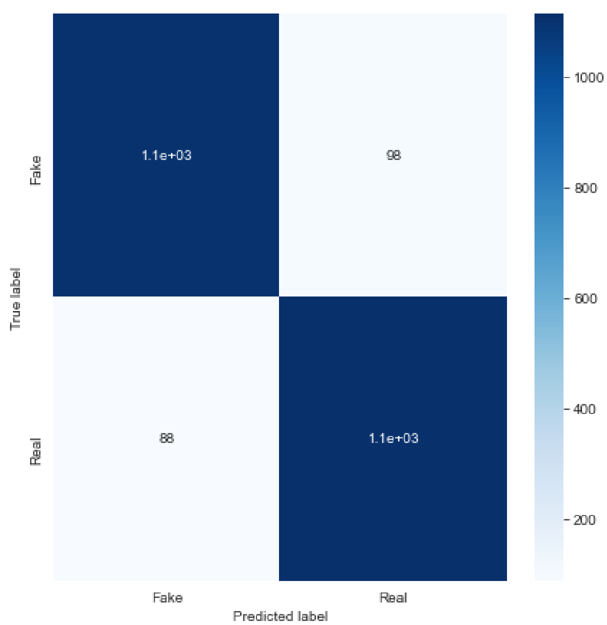


Figure 5.1 Confusion matrix for SimpleRNN

As can be seen from Figure 5.1 above, we first examined our dataset with the vanilla recurrent neural network that does classify 2214 of the news correctly out of the averaged fold size of 2400, where only 186 of the news are classified incorrectly. The model also exhibits an average accuracy of 92.27% and an f1-score of 0.92%.

LSTM-based Models:

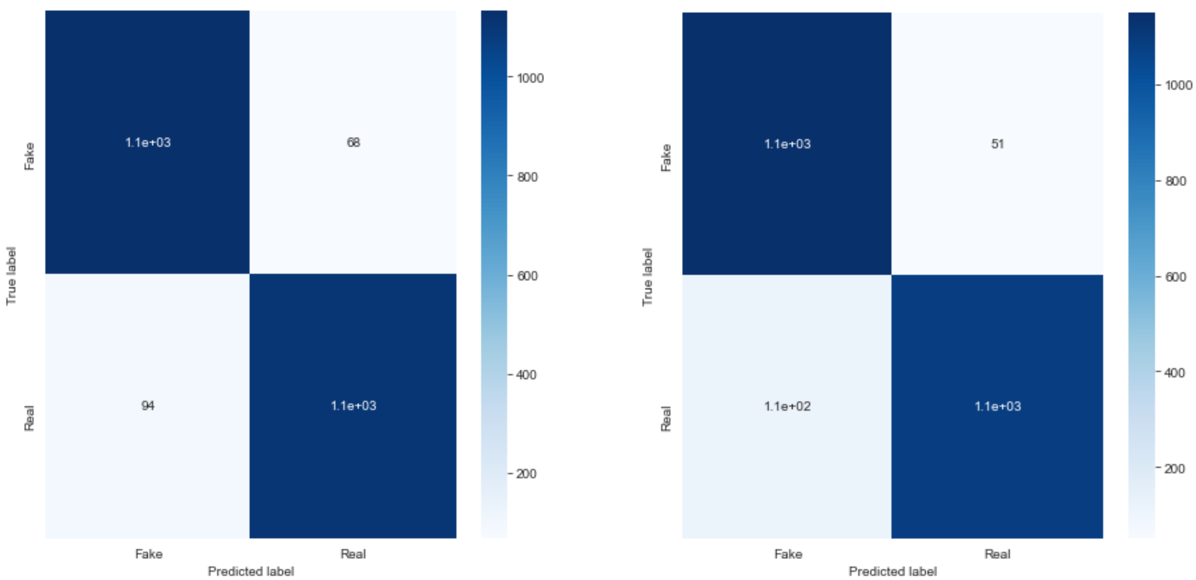


Figure 5.2 Confusion matrix for LSTM(left) and Bi-LSTM(right)

Then, to see how our data fits on a neural network that depends on the historical context of inputs rather than merely the last input, we applied two variants of LSTM models on it. The first model is an LSTM model where 2238 of the news among 2400 are correctly classified and the other 162 are wrongly classified with an accuracy of 93.25 % and f1-score of 0.93%. The second model is a bidirectional LSTM, which is an extension of traditional LSTMs which is assumed to improve model performance as it can preserve information from both past and future**.** Using bidirectional LSTM we got an accuracy of 93.16 % and an f1-score of 0.93%.

GRU-based Models:

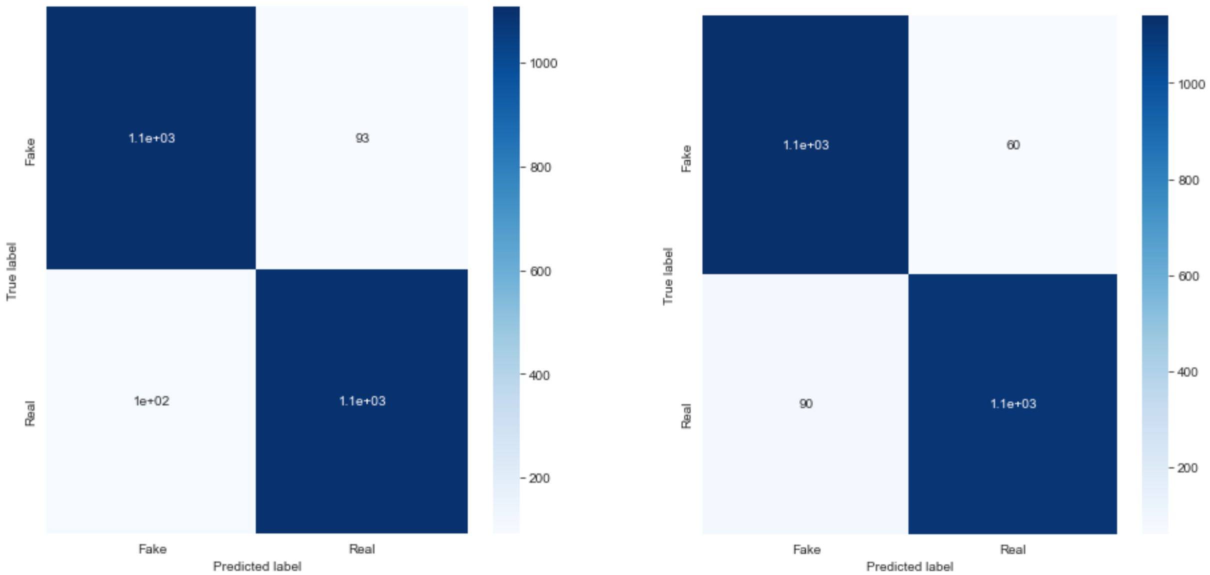


Figure 5.3 Confusion matrix for GRU(left) and Bi-GRU(right)

The other LSTM-like but the more computationally efficient model we experiment with is GRUs(Gated recurrent units) and the traditional GRUs classified 2204 of the news as a True Negative and True Positive and the rest 196 as Fase Negative and False Positive among the all averaged fold size of 2400 with an accuracy of 91.8% and f1-score of 0.92%. A bidirectional GRU is also trained and, ended up with an accuracy of 93.76% and an f1-score of 0.94%.

5.1.2 Confusion Matrix for Attention Based Models

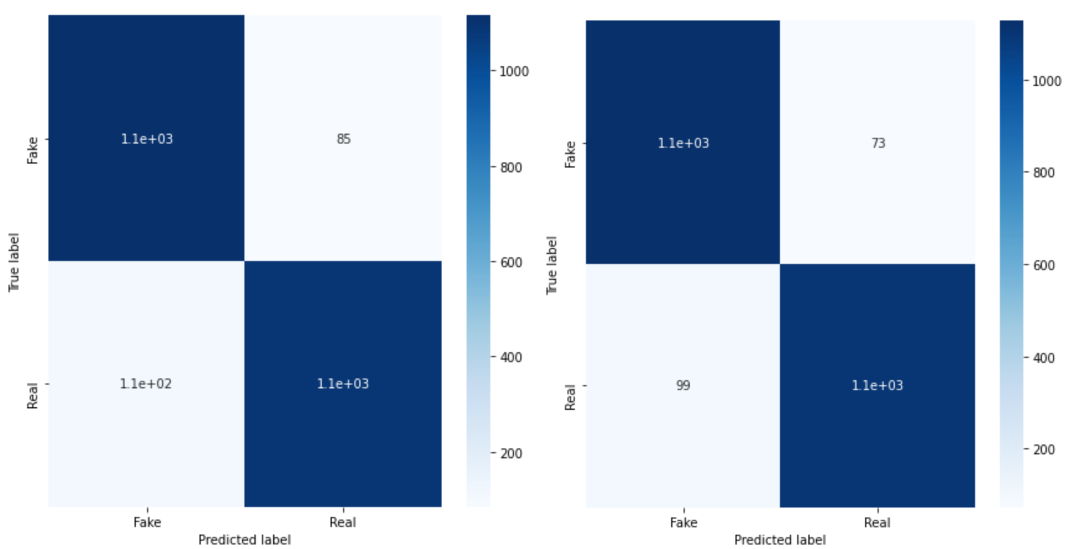


Figure 5.4 Confusion matrix for Attention-based Bi-LSTM(left) and Bi-GRU(right)

Attention mechanism also employed with bidirectional LSTMs and GRUs. The first model achieves an accuracy of 92.09% and 0.92% f1-score. The second, GRUs with attention, gets an accuracy of 92.84% and 0.93% of f1-score.

5.1.3 Confusion Matrix for CNN Model

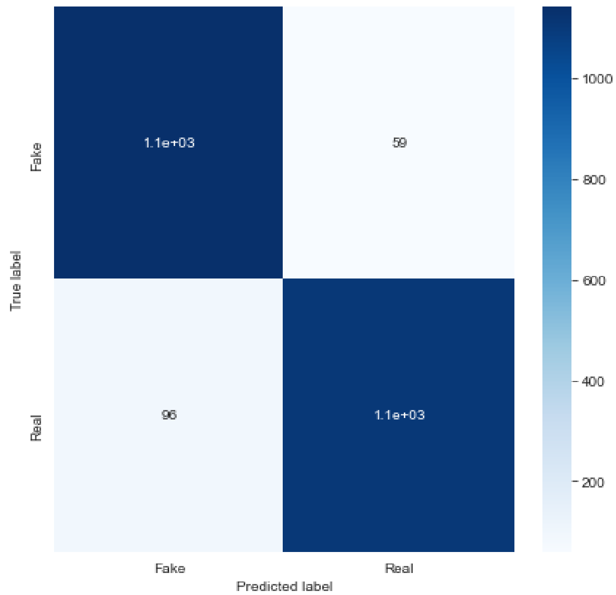


Figure 5.5 Confusion matrix for CNN

Lastly, Convolutional Neural Network which is common in computer vision and more recently in NLP is trained with our data and only 155 of the news are misclassified while the other 2245 of the data are correctly classified. With this model, we achieved accuracy and f1-score of 93.92% and 0.94% respectively.

Table 5.1 Summary for Classification Performance of The Models

Models	Accuracy	Loss	precision	recall	f1-score
simpleRNN	92.26666689	0.845165	**0.93**	0.92	0.92
LSTM	93.24999928	0.485913	0.92	0.94	0.93
Bi-LSTM	93.15833211	0.267965	0.91	**0.96**	0.93
GRU	91.81666613	0.672682	0.91	0.92	0.92
Bi-GRU	93.75833273	0.396515	**0.93**	0.95	**0.94**
Bi-LSTM + Attention	92.09166527	0.317551	0.91	0.93	0.92
Bi-GRU + Attention	92.84166574	0.851223	0.92	0.94	0.93
CNN	**93.91666651**	0.414159	**0.93**	0.95	**0.94**

5.1.4 Normalizing the data

To see the effects of normalizing morpheme on the outcome of our classification models, we choose the top two performing models and applied normalization to suffixes that are ambiguous across different writers. We got normalization in the context of the Amharic fake news data is not helpful in terms of performance improvement. As shown in the classification reports below, the CNN model dropped its f1-score, accuracy, and precision measures by 2%, 1.92%, and 3%, respectively.

```
              precision    recall  f1-score   support

           0       0.90      0.93      0.92      6000
           1       0.93      0.90      0.91      6000

    accuracy                           0.92     12000
   macro avg       0.92      0.92      0.92     12000
weighted avg       0.92      0.92      0.92     12000
```

Figure 5.6 Classification report for CNN with Normalization

Similarly, Bi-GRU with the normalized dataset got a decline in performance by 2%, 1.75%, and 2% regarding f1-score, accuracy, and precision, respectively.

```
              precision    recall  f1-score   support

           0       0.91      0.92      0.92      6000
           1       0.92      0.91      0.92      6000

    accuracy                           0.92     12000
   macro avg       0.92      0.92      0.92     12000
weighted avg       0.92      0.92      0.92     12000
```

Figure 5.7 Classification report for Bi-GRU with Normalization

5.1.5 Tuning Hyperparameters

Even though using automatic hyperparameter tunning markedly reduces the exhaustion to the researcher, it is often a very costly move concerning the computational resource. So to mitigate this problem we choose to tune on manually set up hyperparameter values. The purpose of this manual hyperparameter tuning is to end up with a model that has a low generalization error with a small runtime and memory budget we have. We arranged for a total of 12 trials as shown in Table 5.1 below and trained our CNN model.

Table 5.2 Hyperparameter chose for tuning

#Trial	***Learning rate***	***No. of filters***	***Kernel-size***	***Epochs***	***Batch size***	***Dropout***	***No. of hidden unit***	***No. of hidden layer***
1	0.001	16	7	4	128	0.5	160	3
2	0.001	112	13	10	256	0.02	128	3
3	0.02	80	16	4	64	0.05	32	1
4	0.01	48	5	10	128	0.2	64	2
5	0.002	16	13	20	256	0.02	128	5
6	0.001	64	4	4	32	0.1	32	1
7	0.5	160	8	4	256	0.02	64	2
8	0.2	96	3	20	512	0.01	64	2
9	0.001	64	8	10	256	0.02	128	5
10	0.003	112	2	4	128	0.1	96	3
11	0.03	128	5	4	64	0.2	128	5
12	0.1	32	10	20	128	0.5	96	3

Among the trials experimented, none of them exceed the performance exhibited before by convolutional neural network. As shown in the figure below, the maximum accuracy and f1-measure attained remain the same.

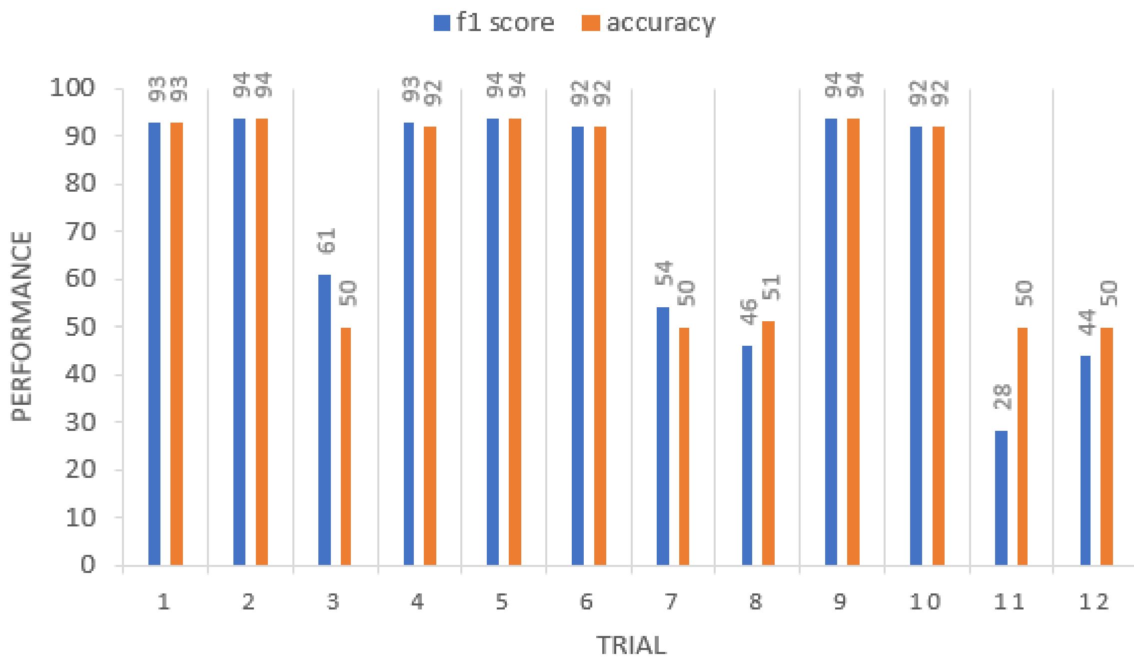


Figure 5.8 Performance evaluation for the differently tuned hyperparameters

Even if the performance of the model doesn't improve by the process of hyperparameter tuning, we got some consistencies regarding the hyperparameters we tweaked and performance measurement of the trained model. Despite stated in the research papers[51][52] that setting a higher number of filters yields greater performance in image and sentence classification problems, our result shows the case for the Amharic Fake News dataset is different. For example, the later paper suggests the use of 100-600 filters to achieve an optimum performance but, adversely as shown clearly in figure 5.11 below, almost all top f1-measures are recorded when the number of filters is tuned below 64.

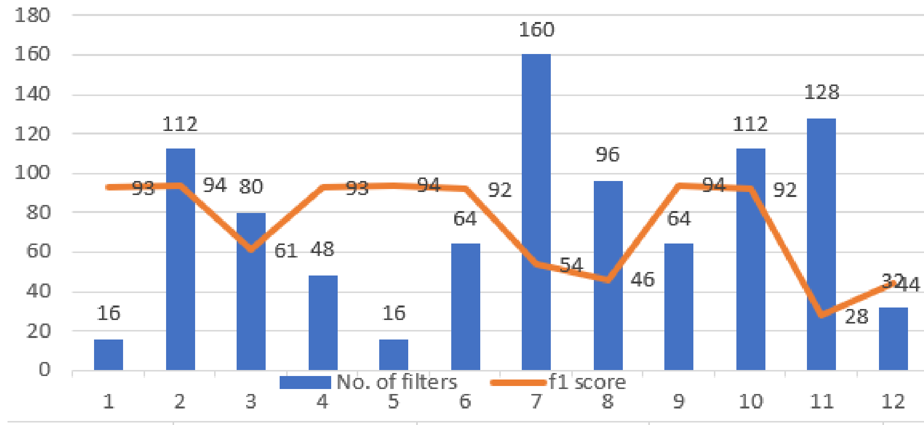


Figure 5.9 Consistency among the number of filters and performance

The other most important hyperparameter that shows some regularity is the learning rate. This study observed is that setting the learning rate to the third decimal point (0.001,0.002,0.003) is way better than updating the estimated network weights with a percent greater than 10 by letting the learning rate to the second decimal point. As shown in figure 5.12 and expected, the result from hyperparameter tuning formed some linearity that the more we make the learning rate high the more we lose performance as the weights may get diverges.

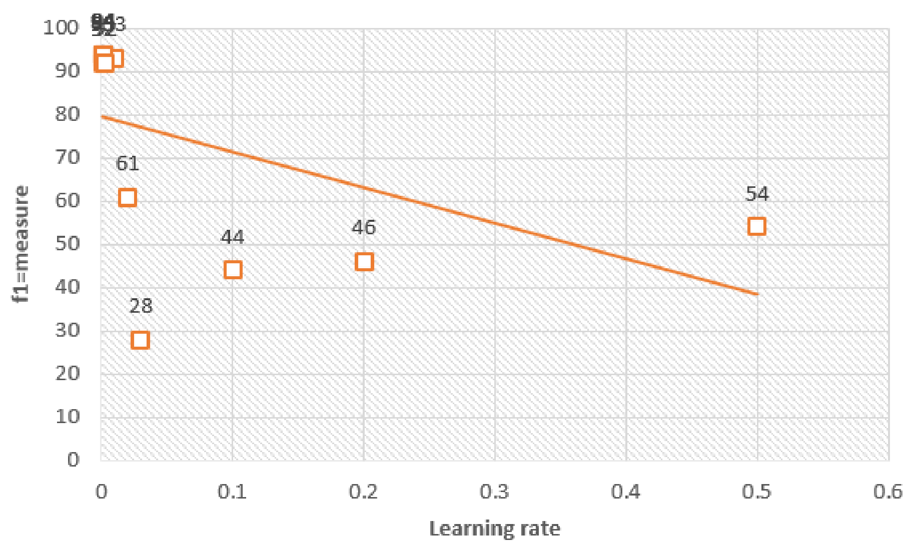


Figure 5.10 Consistency among learning rate and performance

Moreover, the experiment also revealed that setting out the number of the hidden unit beyond 128 results in a better f1-measure, and also specifying the hidden layer of 5 is an optimum for the dataset we have used. More on the result can be seen in **Appendix B.3:** Hyperparameter tuning results

5.2 Discussion

To our knowledge, this study is the first to explore the use of deep learning to detect "Fake News" in the Amharic language. We undergo several experiments on the newly collected dataset and got promising results. The study examined different deep neural networks ranging from networks that see the data as a sequence in time to model that extracting local and position-invariant features across space. As summarized in Table 3, Convolutional Neural Network(CNN) showed a better performance in terms of f1-score, precision, and accuracy mercies. Besides that, CNN also takes less training time compared to Bi-GRU, which is the second well-performing model.

Contrary to expectations, this study did not find a significant difference in terms of accuracy between models with and without an attention mechanism. Although Attention has been proven successful in many natural language processing and its interpretability is investigated in a paper by Xiaobing Sun et al.[53], none of our experiments showed a statistically significant difference in the experiment with our dataset. By contrast, with a closer inspection of table 3 below, adding the attention mechanism has shown a slight negative impact on the bidirectional LSTM and GRU models' performance**.**

Even though morphological normalization has shown a performance improvement in machine translation and some other classification problems, our experiments indicate that applying this technique to our Amharic Fake News dataset has a negative influence rather than promoting the performance of our models. For the reason that the morphologically rich language like Amharic encodes grammatical information at the beginning and end of the word, Fake News contents are more often littered with spelling and grammar errors that may help the deep learning algorithms to figure out them. And the possible explanation for the fall-off performance in some metrics may be as a result of flattening such features from the dataset.

As mentioned in the literature review and our experimental result shows, a Convolutional Neural Network got a promising performance with its less complex architecture relative to the sequence processing and attention mechanisms we had employed. This observation may support the hypothesis that the more complex model we use, we end up with an overfitted model such that the model does well familiarized with the training data but an ability to figure out the general relationship in the overall data is declined. Finally, we chose 12 different trials of hyperparameters to tune the convolutional neural network, and after testing each case we learned that the performance of the classification model gets lower as the number of filters used on a convolution operation increases. In addition to that, as obviously expected, we also could observe that using a very low learning rate leads to a great performance all the time.

‘

CHAPTER SIX

6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

The present study was designed to detect Amharic Fake News using a deep learning algorithm. To accomplish our study, we used a newly collected dataset as there has been no previously made available resource regarding the area that we want to explore. After we collected the data from the Facebook social media platform using the graph API, the dataset was annotated by two journalists. Guidelines from the news literacy project were also adopted to make sure the data is consistently annotated across different annotators. So, this process resulted in an annotated dataset of 12000 stories with a binary class. To alleviate a complication with an imbalance between the number of instances in each class and to be reliable on classification reports, we used equal-sized class instances: 6000, each for fake and real class.

To make the stories we collected understandable by the deep learning models, we used a word embedding method called fastText which is an extension of Word2Vec. We choose fastText as it performs better than other embedding techniques in representing rare words and the availability of the pre-rained model for the language under study. Following the representation of our dataset with the fastText embedding model, we exposed our data to the different deep learning models and evaluated each model with a 5-fold cross-validation technique, as we do have limited data samples and a way to reduce the after-effect of the overfitting problem.

This study generally has found that the Convolutional Neural Network is the best performing model. besides that, contrary to our expectation, the attention mechanism employed over the sequential models shows lesser performance than that of its baseline model. The other thing this study reveals is that morphological normalization in the Amharic Fake News data set is not necessarily useful to improve the models' performance**.**

6.2 Future Work

Even though we have experimented with the most fundamental deep learning techniques in Natural Language Processing, it could be a good move to assess other algorithms from other areas like Capsule Networks. Capsule Networks are showing a better performance in the computer vision world and adapting their strength to the NLP task may help in improving the Amharic Fake News detection model. Besides that, we suggest researchers who are interested in this area to train their own embeddings with domain-specific data to get a more semantically strong embedding model, which may lead to better detection.

* * *

This research was funded by Adama Science and Technology University with a grant number ***ASTU/SM-R/089/19****.*

REFERENCES

[1] D. H. Lee, Y. R. Kim, H. J. Kim, S. M. Park, and Y. J. Yang, “Fake news detection using deep learning,” *J. Inf. Process. Syst.*, vol. 15, no. 5, pp. 1119–1130, 2019.

[2] “Facebook expands third-party fact-checking to Ethiopia, more African countries -.” [Online]. Available: https://waltainfo.com/42502/. [Accessed: 14-Jan-2021].

[3] “Ethiopia cabinet approves bill to combat fake news, hate speech | Africanews.” [Online]. Available: https://www.africanews.com/2019/11/11/ethiopia-cabinet-approves-bill-to-combat-fake-news-hate-speech/. [Accessed: 13-Jan-2021].

[4] M. P. Goswami and M. Pratim Goswami Assistant Professor, “Fake News and Cyber Propaganda: A Study of Manipulation and Abuses on Social Media,” no. December, 2018.

[5] Y. Yang, L. Zheng, J. Zhang, Q. Cui, Z. Li, and P. S. Yu, “TI-CNN: Convolutional Neural Networks for Fake News Detection,” 2018.

[6] A. Hassan and A. Mahmood, “Efficient deep learning model for text classification based on recurrent and convolutional layers,” *Proc. - 16th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2017*, vol. 2017-Decem, no. December 2017, pp. 1108–1113, 2017.

[7] R. K. Kaliyar, “Fake news detection using a deep neural network,” *2018 4th Int. Conf. Comput. Commun. Autom. ICCCA 2018*, pp. 1–7, 2018.

[8] S. Ananth, K. Radha, S. Prema, and K. Nirajan, “Fake News Detection using Convolution,” pp. 49–63, 2019.

[9] “FAKE NEWS | meaning in the Cambridge English Dictionary.” [Online]. Available: https://dictionary.cambridge.org/dictionary/english/fake-news. [Accessed: 13-Feb-2020].

[10] E. C. Tandoc, Z. W. Lim, and R. Ling, “Defining ‘Fake News’: A typology of scholarly definitions,” *Digit. Journal.*, vol. 6, no. 2, pp. 137–153, 2018.

[11] E. C. Tandoc, “The facts of fake news: A research review,” *Sociol. Compass*, vol. 13, no. 9, pp. 1–9, 2019.

[12] "A human rights-based approach to disinformation | Global Partners Digital." [Online]. Available: https://www.gp-digital.org/a-human-rights-based-approach-to-disinformation/. [Accessed: 13-Feb-2020].

[13] M. Bulger and P. Davison, "The Promises, Challenges, and Futures of Media Literacy," *J. Media Lit. Educ.*, vol. 10, no. 1, pp. 1–21, 2018.

[14] M. A. Adamu, "Role of social media in Ethiopias recent political transition," *J. Media Commun. Stud.*, vol. 12, no. 2, pp. 13–22, 2020.

[15] G. A. Dagnaw and S. E. Tsigie, "CURRENT STAT OF THE ART DIGITAL LITERACY IN ETHIOPIA : ARTICLE REVIEW CURRENT STAT OF THE ART DIGITAL LITERACY IN ETHIOPIA :," no. September, 2019.

[16] Y. Fang, J. Gao, C. H. Id, H. Peng, and R. Wu, "Self Multi-Head Attention-based Convolutional Neural Networks for fake news detection," pp. 1–13, 2019.

[17] L. Ball and J. Elworthy, "Fake or real? The computational detection of online deceptive text," *J. Mark. Anal.*, vol. 2, no. 3, pp. 187–201, 2014.

[18] S. Argamon-Engelson, M. Koppel, and G. Avneri, "Style-based text categorization: What newspaper am I reading?," *Proc. AAAI Work. Learn. Text Categ.*, 1998.

[19] N. J. Conroy, V. L. Rubin, and Y. Chen, "Automatic deception detection: Methods for finding fake news," *Proc. Assoc. Inf. Sci. Technol.*, vol. 52, no. 1, pp. 1–4, 2015.

[20] K. Shu, S. Wang, and H. Liu, "Exploiting Tri-Relationship for Fake News Detection," *WSDM 2019 - Proc. 12th ACM Int. Conf. Web Search Data Min.*, pp. 312–320, 2019.

[21] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake News Detection on Social Media: A Data Mining Perspective," *ACM SIGKDD Explor. Newsl.*, vol. 19, no. 1, pp. 22–36, 2017.

[22] N. Ruchansky, S. Seo, and Y. Liu, "CSI: A hybrid deep model for fake news detection," *Int. Conf. Inf. Knowl. Manag. Proc.*, vol. Part F1318, pp. 797–806, 2017.

[23] "An introduction to Bag of Words and how to code it in Python for NLP." [Online]. Available: https://www.freecodecamp.org/news/an-introduction-to-bag-of-words-and-how-to-code-it-in-python-for-nlp-282e87a9da04/. [Accessed: 18-Feb-2020].

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *1st Int. Conf. Learn. Represent. ICLR 2013 - Work. Track Proc.*, pp. 1–12, 2013.

[25] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," *EMNLP 2014 - 2014 Conf. Empir. Methods Nat. Lang. Process. Proc. Conf.*, no. January 2014, pp. 1532–1543, 2014.

[26] "Exploring Neural Word Embeddings," 2019.

[27] A. Basirat, *Principal Word Vectors*, no. October. 2018.

[28] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *EMNLP 2014 - 2014 Conf. Empir. Methods Nat. Lang. Process. Proc. Conf.*, pp. 1724–1734, 2014.

[29] D. Bahdanau, K. H. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–15, 2015.

[30] "A Comprehensive Guide to Attention Mechanism in Deep Learning." [Online]. Available: https://www.analyticsvidhya.com/blog/2019/11/comprehensive-guide-attention-mechanism-deep-learning/. [Accessed: 24-Feb-2020].

[31] Y. LeCun *et al.*, "Backpropagation applied to digit recognition," *Neural computation*, vol. 1, no. 4. pp. 541–551, 1989.

[32] "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way." [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. [Accessed: 24-Feb-2020].

[33] A. Jacovi and O. S. Shalom, "Understanding Convolutional Neural Networks for Text Classification," 2016.

[34] N. Alemayehu and P. Willett, "Stemming of Amharic Words for Information Retrieval," 2002.

[35] S. Eyassu, "Classifying Amharic News Text Using Self-Organizing Maps," 2000.

[36] W. Kelemework, "Automatic Amharic text news classification: Aneural networks

approach,” *Ethiop. J. Sci. Technol.*, vol. 6, no. 2, pp. 127–137, 2013.

[37] Z. Hossain, A. Rahman, S. Islam, and S. Kar, “BanFakeNews: A Dataset for Detecting Fake News in Bangla.”

[38] Y. Liu and Y. F. B. Wu, “Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks,” *32nd AAAI Conf. Artif. Intell. AAAI 2018*, pp. 354–361, 2018.

[39] R. K. Kaliyar, A. Goswami, P. Narang, and S. Sinha, “FNDNet- A Deep Convolutional Neural Network for Fake News Detection,” *Cogn. Syst. Res.*, 2020.

[40] K. Shu, L. Cui, S. Wang, D. Lee, and H. Liu, “Defend: Explainable fake news detection,” *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 395–405, 2019.

[41] J. Y. Khan, M. T. I. Khondaker, A. Iqbal, and S. Afroz, “A Benchmark Study on Machine Learning Methods for Fake News Detection,” pp. 1–14, 2019.

[42] Y. Wang *et al.*, “EANN: Event adversarial neural networks for multi-modal fake news detection,” *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 849–857, 2018.

[43] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching Word Vectors with Subword Information,” *Trans. Assoc. Comput. Linguist.*, vol. 5, pp. 135–146, 2017.

[44] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, “Deep Learning Based Text Classification: A Comprehensive Review,” vol. 1, no. 1, pp. 1–42, 2020.

[45] Q. Li *et al.*, “A Survey on Text Classification: From Shallow to Deep Learning,” vol. 37, no. 4, 2020.

[46] “Introduction to Machine Learning Model Evaluation - Heartbeat.” [Online]. Available: https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f. [Accessed: 28-Feb-2020].

[47] G. Casella, S. Fienberg, and I. Olkin, *An Introduction to Statistical Learning*. 2013.

[48] “Why is Python Growing So Quickly? - Stack Overflow Blog.” [Online]. Available:

https://stackoverflow.blog/2017/09/14/python-growing-quickly/. [Accessed: 13-Oct-2020].

[49] "About Keras." [Online]. Available: https://keras.io/about/. [Accessed: 13-Oct-2020].

[50] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using Convolutional Networks," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 648–656, 2015.

[51] W. S. Ahmed and A. A. A. Karim, "The Impact of Filter Size and Number of Filters on Classification Accuracy in CNN," *Proc. 2020 Int. Conf. Comput. Sci. Softw. Eng. CSASE 2020*, pp. 88–93, 2020.

[52] Y. Zhang and B. Wallace, "A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification," 2015.

[53] X. Sun and W. Lu, "Understanding Attention for Text Classification," no. 1999, pp. 3418–3428, 2020.

APPENDICES

Appendix A: The news literacy project guideline to identify Fake News

the news literacy project

www.thenewsliteracyproject.org

checkology™

www.checkology.org

TEN QUESTIONS FOR FAKE NEWS DETECTION

Use the questions below to assess the likelihood that a piece of information is fake news. The more red flags you circle, the more skeptical you should be!

START

1. Gauge your emotional reaction:
   Is it strong? Are you angry? Are you intensely hoping that the information turns out to be true? False? YES | NO
2. Reflect on how you encountered this. Was it promoted on a website? Did it show up in a social media feed? Was it sent to you by someone you know?
3. Consider the headline or main message:
   a. Does it use excessive punctuation(!!) or ALL CAPS for emphasis? YES | NO
   b. Does it make a claim about containing a secret or telling you something that "the media" doesn't want you to know? YES | NO
   c. Don't stop at the headline! Keep exploring.
4. Is this information designed for easy sharing, like a meme? YES | NO
5. Consider the source of the information:
   a. Is it a well-known source? YES | NO
   b. Is there a byline (an author's name) attached to this piece? YES | NO
   c. Go to the website's "About" section: Does the site describe itself as a "fantasy news" or "satirical news" site? YES | NO
   d. Does the person or organization that produced the information have any editorial standards? YES | NO
   e. Does the "contact us" section include an email address that matches the domain (not a Gmail or Yahoo email address)? YES | NO
   f. Does a quick search for the name of the website raise any suspicions? YES | NO

6. Does the example you're evaluating have a current date on it? YES | NO

7. Does the example cite a variety of sources, including official and expert sources? Does the information this example provides appear in reports from (other) news outlets? YES | NO

8. Does the example hyperlink to other quality sources? In other words, they haven't been altered or taken from another context? YES | NO

9. Can you confirm, using a reverse image search, that any images in your example are authentic (in other words, sources that haven't been altered or taken from another context)? YES | NO

10. If you searched for this example on a fact-checking site such as Snopes.com, FactCheck.org or PolitiFact.com, is there a fact-check that labels it as less than true?

YES | NO

REMEMBER:

- It is easy to clone an existing website and create fake tweets to fool people.
- Bots are extremely active on social media and are designed to dominate conversations and spread propaganda.
- Fake news and other misinformation often use a real image from an unrelated event.
- Debunk examples of misinformation whenever you see them. It's good for democracy!

Visit **www.checkology.org** for a comprehensive collection of news literacy e-learning experiences and other resources from NLP.

Appendix B: Supporting Result

Appendix B.1: 5-fold Training Log

```
------------------------------------------------------------------------
Score per fold
------------------------------------------------------------------------
> Fold 1 - Loss: 0.23917133100330829 - Accuracy: 91.50000214576721%
------------------------------------------------------------------------
> Fold 2 - Loss: 0.24412204643090565 - Accuracy: 91.66666865348816%
------------------------------------------------------------------------
> Fold 3 - Loss: 0.23732525666554768 - Accuracy: 91.70833230018616%
------------------------------------------------------------------------
> Fold 4 - Loss: 0.20201323563853898 - Accuracy: 92.75000095367432%
------------------------------------------------------------------------
> Fold 5 - Loss: 0.1810554325580597 - Accuracy: 93.70833039283752%
------------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 92.26666688919067 (+- 0.8451645639248165)
> Loss: 0.2207374604592721
```

simpleRNN

```
----------------------------------------------------------------------
Score per fold
----------------------------------------------------------------------
> Fold 1 - Loss: 0.2014720813371241 - Accuracy: 93.37499737739563%
----------------------------------------------------------------------
> Fold 2 - Loss: 0.2067101151868701 - Accuracy: 92.45833158493042%
----------------------------------------------------------------------
> Fold 3 - Loss: 0.1937111698836088 - Accuracy: 93.54166388511658%
----------------------------------------------------------------------
> Fold 4 - Loss: 0.20067816020299992 - Accuracy: 93.00000071525574%
----------------------------------------------------------------------
> Fold 5 - Loss: 0.18479100245982408 - Accuracy: 93.87500286102295%
----------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 93.24999928474426 (+- 0.4859134211591236)
> Loss: 0.1974725058140854
```

LSTM

```
----------------------------------------------------------------------
Score per fold
----------------------------------------------------------------------
> Fold 1 - Loss: 0.23571181030012667 - Accuracy: 93.4583306312561%
----------------------------------------------------------------------
> Fold 2 - Loss: 0.19339612691352764 - Accuracy: 92.87499785423279%
----------------------------------------------------------------------
> Fold 3 - Loss: 0.21525613715251288 - Accuracy: 93.04166436195374%
----------------------------------------------------------------------
> Fold 4 - Loss: 0.19726701352745293 - Accuracy: 93.50000023841858%
----------------------------------------------------------------------
> Fold 5 - Loss: 0.20540950072929262 - Accuracy: 92.91666746139526%
----------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 93.1583321094513 (+- 0.26796555339949973)
> Loss: 0.2094081177245825
----------------------------------------------------------------------
```


Bi-LSTM

```
----------------------------------------------------------------------
Score per fold
----------------------------------------------------------------------
> Fold 1 - Loss: 0.35675979488218823 - Accuracy: 91.12499952316284%
----------------------------------------------------------------------
> Fold 2 - Loss: 0.25255596606681746 - Accuracy: 92.12499856948853%
----------------------------------------------------------------------
> Fold 3 - Loss: 0.28523807709415755 - Accuracy: 91.12499952316284%
----------------------------------------------------------------------
> Fold 4 - Loss: 0.2505387891456485 - Accuracy: 91.79166555404663%
----------------------------------------------------------------------
> Fold 5 - Loss: 0.2632142965247234 - Accuracy: 92.91666746139526%
----------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 91.81666612625122 (+- 0.672681535518488)
> Loss: 0.281661384742707
----------------------------------------------------------------------
```

GRU

```
---------------------------------------------------------------------
Score per fold
---------------------------------------------------------------------
> Fold 1 - Loss: 0.22614535763238866 - Accuracy: 93.20833086967468%
---------------------------------------------------------------------
> Fold 2 - Loss: 0.2025081232190132 - Accuracy: 93.62499713897705%
---------------------------------------------------------------------
> Fold 3 - Loss: 0.18084459016720453 - Accuracy: 94.29166913032532%
---------------------------------------------------------------------
> Fold 4 - Loss: 0.21338530281248191 - Accuracy: 93.54166388511658%
---------------------------------------------------------------------
> Fold 5 - Loss: 0.1825415187391142 - Accuracy: 94.12500262260437%
---------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 93.7583327293396 (+- 0.3965149027697026)
> Loss: 0.2010849785140405
---------------------------------------------------------------------
```

Bi-GRU

```
---------------------------------------------------------------------
Score per fold
---------------------------------------------------------------------
> Fold 1 - Loss: 0.30437803268432617 - Accuracy: 92.95833110809326%
---------------------------------------------------------------------
> Fold 2 - Loss: 0.39478033781051636 - Accuracy: 91.29166603088379%
---------------------------------------------------------------------
> Fold 3 - Loss: 0.23451818525791168 - Accuracy: 92.75000095367432%
---------------------------------------------------------------------
> Fold 4 - Loss: 0.34638074040412903 - Accuracy: 93.04166436195374%
---------------------------------------------------------------------
> Fold 5 - Loss: 0.30769819021224976 - Accuracy: 90.41666388511658%
---------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 92.09166526794434 (+- 1.0519165598066842)
> Loss: 0.3175510972738266
---------------------------------------------------------------------
```

Bi-LSTM+Attention

```
---------------------------------------------------------------------
Score per fold
---------------------------------------------------------------------
> Fold 1 - Loss: 0.19530682265758514 - Accuracy: 93.50000023841858%
---------------------------------------------------------------------
> Fold 2 - Loss: 0.2162739634513855 - Accuracy: 92.79166460037231%
---------------------------------------------------------------------
> Fold 3 - Loss: 0.22698847949504852 - Accuracy: 93.04166436195374%
---------------------------------------------------------------------
> Fold 4 - Loss: 0.1901939958333969 - Accuracy: 93.62499713897705%
---------------------------------------------------------------------
> Fold 5 - Loss: 0.2558349370956421 - Accuracy: 91.25000238418579%
---------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 92.8416657447815 (+- 0.8512231427094302)
> Loss: 0.21691963970661163
---------------------------------------------------------------------
```

Bi-GRU+Attention

```
-----------------------------------------------------------------------
Score per fold
-----------------------------------------------------------------------
> Fold 1 - Loss: 0.2109899067428584 - Accuracy: 93.54166388511658%
-----------------------------------------------------------------------
> Fold 2 - Loss: 0.22272758038869747 - Accuracy: 93.4166669845581%
-----------------------------------------------------------------------
> Fold 3 - Loss: 0.20871950501576067 - Accuracy: 93.99999976158142%
-----------------------------------------------------------------------
> Fold 4 - Loss: 0.18216385412340363 - Accuracy: 94.0416693687439%
-----------------------------------------------------------------------
> Fold 5 - Loss: 0.17047209228388965 - Accuracy: 94.58333253860474%
-----------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 93.91666650772095 (+- 0.4141594459433388)
> Loss: 0.19901458771092195
-----------------------------------------------------------------------
```

CNN

Appendix B.2: Classification Reports

	precision	recall	f1-score	support
0	0.93	0.92	0.92	6000
1	0.92	0.93	0.92	6000
accuracy			0.92	12000
macro avg	0.92	0.92	0.92	12000
weighted avg	0.92	0.92	0.92	12000

simpleRNN

	precision	recall	f1-score	support
0	0.92	0.94	0.93	6000
1	0.94	0.92	0.93	6000
accuracy			0.93	12000
macro avg	0.93	0.93	0.93	12000
weighted avg	0.93	0.93	0.93	12000

LSTM

	precision	recall	f1-score	support
0	0.91	0.96	0.93	6000
1	0.96	0.91	0.93	6000
accuracy			0.93	12000
macro avg	0.93	0.93	0.93	12000
weighted avg	0.93	0.93	0.93	12000

Bi-LSTM

	precision	recall	f1-score	support
0	0.91	0.92	0.92	6000
1	0.92	0.91	0.92	6000
accuracy			0.92	12000
macro avg	0.92	0.92	0.92	12000
weighted avg	0.92	0.92	0.92	12000

GRU

	precision	recall	f1-score	support
0	0.93	0.95	0.94	6000
1	0.95	0.93	0.94	6000
accuracy			0.94	12000
macro avg	0.94	0.94	0.94	12000
weighted avg	0.94	0.94	0.94	12000

Bi-GRU

	precision	recall	f1-score	support
0	0.91	0.93	0.92	6000
1	0.93	0.91	0.92	6000
accuracy			0.92	12000
macro avg	0.92	0.92	0.92	12000
weighted avg	0.92	0.92	0.92	12000

Bi-LSTM+Attention

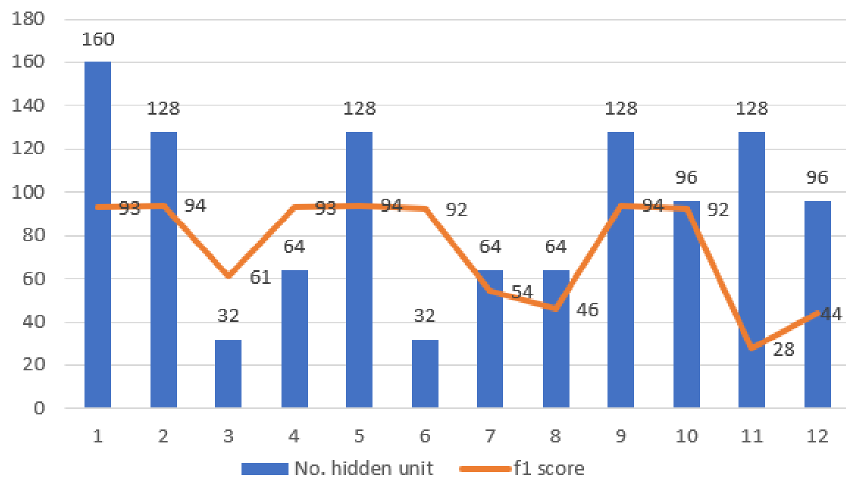
	precision	recall	f1-score	support
0	0.92	0.94	0.93	6000
1	0.94	0.92	0.93	6000
accuracy			0.93	12000
macro avg	0.93	0.93	0.93	12000
weighted avg	0.93	0.93	0.93	12000

Bi-GRU+Attention

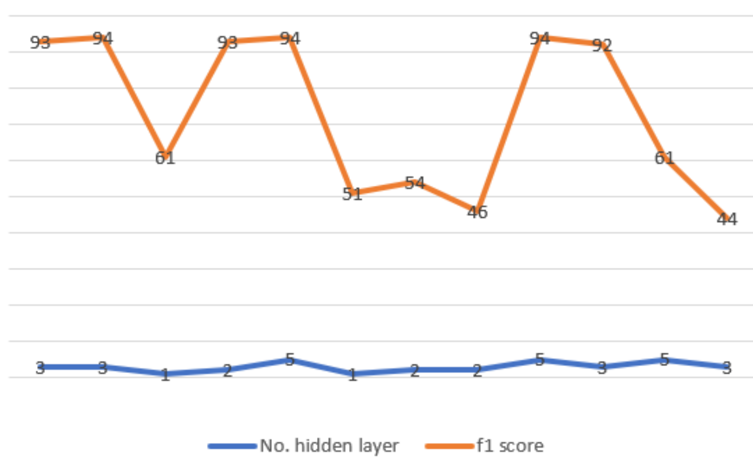
	precision	recall	f1-score	support
0	0.93	0.95	0.94	6000
1	0.95	0.93	0.94	6000
accuracy			0.94	12000
macro avg	0.94	0.94	0.94	12000
weighted avg	0.94	0.94	0.94	12000

CNN

Appendix B.3: Hyperparameter tuning results



Consistency among the number of hidden unit and performance



Consistency among the number of hidden layers and performance

Appendix C: Further about Prototyping

Before the prediction takes place, the news that needed to be verified should be acquired and passed to the webserver for the actual prediction processes. To manage this, we used a multi-line text input control with a button that has an id of "predict_now".

```
  <textarea class="form-control" id="name-input" rows="2" placeholder="ልጥፍዎን ያስገቡ"  ></textarea>
</div>

  <button  id="name-button" type="button" class="btn btn-primary">መርምር</button>
```

Then, after the user fills the textarea with the story to be checked, a javascript code will forward the story to the webserver script responsible for the prediction of the news as either fake or real.

```
<script src="http://code.jquery.com/jquery-3.3.1.min.js">  </script>
 <script>
 $("#predict_now").click(function(event){
     let message={
         post: $("#name-input").val()
         }
       $.post("http://127.0.0.1:5000/predict",JSON.stringify(message),function(response){


           if (response.pred == 0) {
             $('#success_alert').show()
             $('#danger_alert').hide()
            } else {
            $('#danger_alert').show()
            $('#success_alert').hide()
            }


           console.log(response);
           });

     });
</script>
```

After the server got the message forwarded from the user interface, it loads the keras model on a variable and a predict_classes() method is called to get the probability distribution. Finally, the server passes the response to the requesting HTML page.

```python
def predict():

    model = keras.models.load_model('CNN_model.hdfs')

    print("* Model loaded!")

    message = request.get_json(force=True)
    post = message['post']
    processed_post= preprocess_text(post)


    global graph
    model._make_predict_function()
    graph = tf.get_default_graph()
    with graph.as_default():

         prediction = model.predict_classes(processed_post).tolist()
    print(prediction)
    response= {
       'pred':prediction
       }
    return jsonify(response)
```

Scenario 1: Classifying story as fake

Scenario 2: Classifying story as real