Clickbait Detection for Amharic Language Using Neural Networks

Arefat Hyeredin Kedir

A Thesis Submitted to the Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master's in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

June, 2023

Adama, Ethiopia

Clickbait Detection for Amharic Language Using Neural Networks

Arefat Hyeredin Kedir

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 of the Requirement for the Degree of Master's in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

June, 2023

Adama, Ethiopia

APPROVAL PAGE

I, the advisor of the thesis entitled "**Clickbait Detection for Amharic Language Using Neural Networks**" and developed by Arefat Hyeredin Kedir, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

_____	_____	_____
Major Advisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis by Arefat Hyeredin Kedir have read and evaluated the thesis entitled "**Clickbait Detection for Amharic Language using Neural Networks**" and examined the candidate during open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

_____	_____	_____
Chairperson	Signature	Date
_____	_____	_____
Internal Examiner	Signature	Date
_____	_____	_____
External Examiner	Signature	Date

Finally, approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

_____	_____	_____
Department Head	Signature	Date
_____	_____	_____
School Dean	Signature	Date
_____	_____	_____
Office of Post Graduate Studies, Dean	Signature	Date

DECLARATION

I hereby declare that this Master's thesis entitled "**Clickbait Detection for Amharic Language using Neural Networks**" is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation.

_____	_____	_____
Name of the Student	Signature	Date

RECOMMENDATION OF ADVISORS

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis entitled “Clickbait Detection for Amharic Language using Neural Networks” prepared under my/our guidance by Arefat Hyeredin Kedir submitted in partial fulfillment of the requirements for the degree of Mater’s of Science in Computer Science and Engineering. Therefore, I/we recommend the submission of revised version of the thesis to the department following the applicable procedures.

_____	_____	_____
Major Advisor	Signature	Date
_____	_____	_____
Co-Advisor	Signature	Date

ACKNOWLEDGEMENT

In the name of Allah, the Most Gracious and the Most Merciful. Praise and thanks to the almighty God for all the blessings and strengths in helping me throughout this journey.

I am deeply grateful to my advisor, Dr. Mesfin Abebe, for his guidance, valuable feedback, and constant support throughout the research process. His expertise and mentorship have been instrumental in shaping the direction of this study.

I would like to acknowledge the assistance and cooperation received from Adama Science and Technology University (ASTU) and EthiopiaCheck, in providing access to the necessary facilities, resources and tools for this study. Their support has been essential in carrying out the experiments and analyses.

I would also like to take this opportunity to offer my heartfelt thanks to Mr. Kibrom Tadesse for his exceptional support, insightful discussions, and valuable feedback on the thesis.

I am very grateful to my family and friends for their understanding and encouragement throughout this time. Heartfelt thanks go to my parents, whose unwavering support and sacrifices have been the bedrock of my journey. Their belief in me and their encouragement have been a constant source of inspiration.

This work would not have been possible without the support and assistance of all the individuals and institutions mentioned above. I am sincerely grateful for their contributions and guidance, which have played a vital role in the successful completion of this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ACRONYMS AND ABBREVIATIONS

ALBERT	A Lite BERT
AMFTWE	Amharic FastText Word Embedding
ANN	Artificial Neural Network
API	Application programming interface
AUC-ROC	Area under Receiver Operating Characteristic Curve
BERT	Bidirectional Encoder Representations from Transformers
CBCNN	Clickbait Convolutional Neural Network
CBOW	Continuous Bag of Words
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CSV	Comma-separated Values
CTR	Click-Through Rate
CVD	Clickbait Video Detector
GBM	Gradient Boosting Machine
GPAC	General Purpose Amharic corpus
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
GUI	Graphical User Interface
IAA	Inter Annotator Agreement
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
LGBM	Light Gradient Boosted Machines
LR	Logistic Regression
LSTM	Long Short-Term Memory
MI	Machine Intelligence
ML	Machine Learning
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
OVCP	Online Video Clickbait Protector
POSAM	Part of Speech Analysis Module
RoBERTa	Robustly optimized BERT approach
RAM	Random Access Memory
RF	Random Forest
RNN	Recurrent Neural Network
SNE	Stochastic Neighborhood Embedding
SOM	Self-Organizing Maps
SSD	Solid State Drive
SVM	Support Vector Machine
TF-IDF	Term Frequency-Inverse Document Frequency
TPU	Tensor Processing Unit
URL	Uniform Resource Locator

ABSTRACT

With the increasing number of Ethiopians actively engaging on the Internet and social media platforms, the prevalence of clickbait content has become a significant concern. Clickbait, often utilizing enticing titles to tempt users into clicking, has become rampant for various reasons, including advertising and revenue generation. However, the Amharic language, spoken by a large population, lacks sufficient NLP resources for addressing this issue. In this research, we developed a model for classifying and detecting clickbait titles in Amharic. To facilitate this, we present the first Amharic clickbait dataset, comprising 53,227 posts collected from prominent social media platforms such as YouTube, Twitter, and Facebook. We established a baseline using traditional machine learning models like Logistic Regression (LR), Random Forest (RF), and Support Vector Machines (SVM) using TF-IDF and N-gram features. We then explored the effectiveness of neural network architectures, including Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN), combined with two kinds of word embedding techniques, namely word2vec and fastText. Our findings reveal that the CNN model with fastText word embedding achieves the highest performance, with an accuracy of 94.27% and an F1-score of 94.24%. This research provides valuable insights into combating clickbait content in Amharic and contributes to the advancement of natural language processing for resource-poor languages.

Keywords: *Clickbait Detection, Neural Networks, Amharic, CNN, Low Resource Language*

CHAPTER ONE

1. INTRODUCTION

1.1. Background of the Study

In Ethiopia, a growing number of individuals are reliant on the Internet to stay informed about local and global events. Platforms such as Facebook, Twitter, YouTube, and TikTok are frequently utilized for a range of purposes, including social communication, commerce, entertainment, and education. As a result of the immediacy and diversity of information available on these platforms, many people are turning to them as an alternative to conventional media outlets such as newspapers, radios, and televisions. The rapid growth of social media usage gave users the chance to produce and consume a variety of online content, facilitating ways to share their opinions and knowledge, advertise their products for sale, and keep up with current events. However, unethical content creators exploit these platforms to distribute deceptive, low-quality content that frequently spreads erroneous information, rumors, and hoaxes (Zannettou et al., 2019).

Digital media provides a variety of topics and diverse content to users but it is also filled with poor quality and dubious content (Fu et al., 2017). With a growing digital presence, from online publishing and social media activity to web advertisements, it has become a race to get more clicks, views, and likes. This led to the rise of what is known as clickbait. Clickbait is an advertised web content type designed to tempt readers into clicking a link (Potthast et al., 2016). Clickbait can be taken as online content of a misleading nature that aims to attract users' attention and bait them into a certain webpage (Agrawal, 2017).

According to Loewenstein's information gap - theory of Curiosity (Loewenstein, 1994), clickbaits stimulate the reader's curiosity by creating a cognitive gap that needs to be filled between what they already know and what they would like to know. Thus, they feel the impulse to click the link. This theory serves as the major foundation for clickbait studies. Clickbait appears in different forms, ranging from partial facts, exaggerated headlines, and catchy wordings (Fu et al., 2017) to enticing advertisements, scams, and misinformation campaigns (Al-Sarem et al., 2021). Clickbait also clogs up social media and news feeds while also violating journalistic codes of ethics (Naeem et al., 2020).

Digital literacy is not delivered formally to people at an early age, especially in Africa. To spice up their news articles and draw a huge audience, journalists have turned to the use of advertising and strategic communication techniques (Wanda et al., 2021). These problems added up to account for the problematic spread of clickbait. Technology should provide solutions to these challenges that unravel societies from within, whether it be by identifying misinformation or parties who solely do it for the money.

Clickbait advertisements have become a pervasive and controversial phenomenon in the digital advertising landscape worldwide. In Africa, there has been a surge in clickbait advertising due to the relative lack of legal control and enforcement. Clickbait ads are designed to generate as many clicks as possible by using sensational and often misleading headlines or images to entice viewers to click through to the advertiser's website. This practice can be particularly problematic in the Ethiopian context, where low levels of digital literacy and awareness can leave users vulnerable to scams and deceptive marketing tactics. Furthermore, the absence of regulatory frameworks and enforcement mechanisms can exacerbate the situation, as unscrupulous advertisers take advantage of loopholes and exploit vulnerable consumers.

The motives for producing and distributing misinformation, scams, and gossip tidbits extend from scaremongering to controlling opinion. Those who have a specific political agenda or stand to gain financially may be more inclined to utilize clickbait techniques. For instance, individuals who receive economic incentives for promoting clickbait content through platforms like Google AdSense or YouTube. Political actors may use clickbait to sway public opinion or manipulate election results, while economic actors may use clickbait to generate traffic to their websites and increase their revenue.

Even though the problem is a clear online threat, neither the government nor academics are taking any decisive action to identify Amharic clickbait. Furthermore, the lack of NLP resources and datasets for Amharic has made it difficult to automatically detect clickbait. Despite its growing popularity, limited literature has been published focusing on automated text analysis of Amharic documents. As such, there has been little focus on developing algorithms that can analyze and detect clickbait in Amharic text available online. This study's primary objective is to build an Amharic clickbait dataset and employ various types of techniques to detect clickbait computationally.

1.2. Motivation of the Study

Social media sites have ingrained themselves into our daily lives. Ethiopians are drawn to it on a massive scale where all kinds of information spread across communities. It is commonly observed now that clickbait behaviors that use sensationalized headlines, provocative advertisements, scams, and misinformation campaigns are expanding, due to many reasons that include financial gain, popularity in clicks or views, and fraudulent cases.

Even though clickbait is widely utilized across all Internet languages, little has been done to improve clickbait detection in languages other than English. Along with the contribution this work would make to the Amharic NLP landscape, it also contributes methods to identify clickbait presented in the Amharic. By focusing on information accessibility, the study aims to improve the quality of information available to Amharic-speaking users, promoting responsible content consumption in online platforms.

Clickbait is the major component used to fuel the spread of exaggerated facts, fake news, hateful speech, and rumors. Understanding the concerns of such behavior, it is causing manipulation, disappointment, loss, and further division in societies. The study recognizes the social impact of clickbait, aiming to mitigate its negative effects on public opinion, misinformation spread, and trust erosion in online platforms, fostering a more informed and responsible online society.

Some research has been done in related areas such as sentiment analysis, text classification, hate speech, and fake news detection for Amharic, but none in the Amharic clickbait detection domain has been found. Hoping to fill this gap and tackle the problem using technological means steered the research subject in this direction. Besides, we would also like to be among the first researchers to contribute to this subject to the best of our abilities.

1.3. Statement of the Problem

Clickbait is a growing concern on the Internet, particularly within social networks, such as Facebook, Twitter, and YouTube (C. Zhang & Clough, 2020). It is very challenging to identify these clickbait contents manually due to their vast number and wide distribution. It is essential to detect clickbait content on the Internet using automated means (Al-Sarem et al., 2021). The exploitation of social networks to spread clickbait is largely caused by inherent aspects of the Internet.

In Ethiopia, social media and Internet usage is growing rapidly every year. Clickbait is being extensively used in these digital platforms; mediums that share clickbait hide behind a financial goal achieved through reaching more people. The contents on these platforms are exaggerated and miss a foundational context. It has been weakening media credibility and promoting the online spread of rumors and false information (Awol & Gashaw, 2022).

Inherently, clickbait classification is a binary text classification task. But clickbait detection is difficult due to the dynamic nature of the language used to transmit it, as it uses hyperbolic language, an alluring tone, and intent to create viral content, which makes it harder to deal with. Amharic, a resource-poor language with a scarcity of NLP tools and annotated data brings a new breadth of challenges to the subject of clickbait detection. Thus, there is a gap to be filled in studying Amharic clickbait detection.

The problem of clickbait in the Amharic language is an important issue within the digital media environment. It has become increasingly difficult to distinguish between reliable and unreliable online news, advertisements, posts, or other content due to the many sensational headlines that deceive viewers into believing they are receiving genuine information (Gereme et al., 2021).

The high rate of content creation has made it difficult to collect and analyze such big data using traditional methods. This paper presents the application of neural networks in Amharic clickbait detection to reduce the challenges. It is aimed to address the need for such a high-level computational linguistic model to detect Amharic clickbait from social media texts.

1.4. Research Questions

This research tries to address the following research questions:

RQ1: What are the different methods that can be used for Clickbait detection?

RQ2: Which feature engineering and vectorization techniques can be used to detect clickbait content in Amharic?

RQ3: Which is the most effective neural network model that improves the overall accuracy of clickbait detection for Amharic?

1.5. Objectives of the Study

1.5.1. General Objective

The general objective of this study is to develop a clickbait detection model using neural networks for Amharic text.

1.5.2. Specific Objectives

The following specific objectives are implemented in order to achieve the general objective:

- We collected Amharic text data from YouTube, Twitter, and Facebook.
- We designed and implemented a preprocessing technique to ensure the data's quality.
- We developed annotation guidelines and curated an annotated Amharic clickbait dataset.
- We trained various neural network models using the compiled dataset.
- We performed a series of experiments to evaluate the performance of the Amharic clickbait detection model.
- We fine-tuned the neural network models to improve their performance.

1.6. Scope and Limitations

1.6.1. Scope of the Study

The goal of the study is to apply neural network techniques to develop a clickbait detection model for the Amharic language. This work solely targets Amharic textual content in terms of the clickbait element when collecting data, other than the accompanying URL in English. The collection of datasets is conducted mainly on social media platforms based on the extent of clickbait occurrence, which includes YouTube video titles, Facebook posts, and Twitter tweets. Reputable news media feeds and online blogs are incorporated based on the Amharic content they publish. It uses standard practices for annotation and evaluation of the model.

1.6.2. Limitations of the Study

This study does not consider the parties publishing the clickbait content, the features are limited to text-based features, not user-centric features. Visual-centric features such as image or thumbnail, video, or graphical clickbait content are beyond the scope of this study. It only considers the written textual components.

Time would also need to be factored in as the data collection and annotation process is a lengthy one, with error-prone and redundant mechanisms. Financially demanding tasks like running high-end processors is also an issue in constraining the full-scale reach of the study.

1.7. Significance of the Study

Previous research addressed the need for automated clickbait detection; however, they largely used the English clickbait corpus for training (Anand et al., 2017; Potthast et al., 2018). This study contributes to Amharic clickbait dataset, approaches, and techniques for Amharic NLP tasks such as text classification, semantic analysis, and semantic role labeling where there is limited work.

The research findings can be utilized to promote media literacy and critical thinking among Amharic-speaking internet users. By detecting and flagging clickbait content, individuals can become more aware of the manipulative tactics used in online media and make informed decisions about the information they consume and share.

News and media organizations can benefit from the study's findings by integrating clickbait detection algorithms into their content verification processes. This can help improve the accuracy and credibility of news articles and prevent the dissemination of misleading or sensationalized information. It enhances news search and recommendation platforms.

The study's outcomes can be valuable for the advertising and marketing industry, particularly in Amharic-speaking regions. By identifying and filtering out clickbait advertisements, businesses can enhance the transparency and integrity of their marketing campaigns.

The research findings can inform policy and regulatory efforts aimed at combating clickbait and deceptive practices in the digital space. Governments and regulatory bodies can utilize the study's insights to develop guidelines and regulations that promote responsible online behavior and protect users from the negative effects of clickbait.

The significance of this research extends to the realm of cybersafety, where it plays a crucial role in mitigating the risks associated with clickbait, including the propagation of viruses, fraud, and harmful information. By implementing effective clickbait detection mechanisms, this study aims to fortify the online environment, safeguard users from potential threats, and enable them to make informed choices amidst the intricacies of the digital landscape.

By assisting people, institutions, and social networks in sifting appropriate messages and deterring content creators from using clickbait, automatic clickbait detection offers a solution for moderation and pushes for advertisement guidelines. Generally, create healthy and high-quality content delivered in local languages.

1.8. Organization of the Study

The organization of this study is divided into six chapters, which are outlined below:

*Chapter One – Introduction***:** The first chapter serves as an introduction to the research and covers the background, the motivation behind the study, the statement of the problem, the research questions to be addressed, the scope and limitations of the study, and the objectives to be achieved.

*Chapter Two – Literature Review and Related Works***:** The second chapter presents a comprehensive literature review of clickbait and clickbait detection. This chapter explores different approaches and techniques used in clickbait detection, drawing from related works conducted in this research area. The review provides a critical analysis of the existing literature and identifies gaps that the current research aims to address.

Chapter Three – Research Methodology: The third chapter discusses the research methodology employed in the study. It describes the general approaches that have been taken by the study which include the data collection and preparation steps. The chapter also discusses the design tools, frameworks and evaluation methods used in the study.

*Chapter Four – Proposed Solution***:** Chapter Four provides a detailed explanation of the proposed solution to address the problem. This chapter includes the design and implementation, along with conforming mathematical descriptions, to provide comprehensive information on how the proposed solution addresses the problem. It also includes code snippets for the proposed solution, along with the environment setup required for the implementation.

Chapter Five – Results and Discussion: Chapter Five presents the results and discussion from the conducted experiments. This chapter includes a detailed analysis and discussion of the experimental outcomes utilizing figures, graphs, and tables, summarizing the performance of the clickbait detection models. The results are interpreted, and key findings are highlighted. The chapter also includes reports on hyperparameter tuning performed to optimize the model's performance.

Chapter Six – Conclusion and Future Works: The last chapter, Chapter Six concludes the research by summarizing the main findings and results. It offers recommendations based on the study's outcomes and identifies potential future directions for further research in related areas.

CHAPTER TWO

2. LITERATURE REVIEW AND RELATED WORKS

2.1. Introduction to Clickbait

Around the mid-2010s, researchers started to notice an Internet phenomenon unique to the publishing world that was creating a lot of noise in the booming digital era. Linguists like Vijgen (2014) investigated the idea of a major clickbait type known as "listicles", which are articles containing a list of things under a particular topic; for instance, titles such as "Top 10 ways to lose Weight". This writing style is referred to have given birth to the intriguing writing style that has become to be known as Clickbait.

The term "Clickbait" has a range of interpretations due to its dynamicity. Merriam-Webster's Unabridged Dictionary defines the word as "something (such as a headline) designed to make readers want to click on a hyperlink especially when the link leads to content of dubious value or interest." Similar definitions from other scholarly works present it as an advertised web content type designed to tempt readers into clicking a link (Potthast et al., 2016). Clickbait can be taken as online content of misleading nature which aims to attract users' attention and bait them into a certain webpage (Agrawal, 2017).

The nuance of clickbait language arises from the gripping tone that it uses to allure readers into wanting to know more about the case by triggering their curiosity. Exaggerated and humorous tones are characteristically and knowingly present to deliver the content as bait (Zheng et al., 2018). Clickbait does not necessarily mean false, presenting a factual statement in a highly loaded and exaggerated manner giving it an excessive connotation would likely result in a clickbait statement. It typically presents a headline or title that is misleading or exaggerated compared to the actual content of the article (Cao et al., 2017).

In recent times, a significant proportion of the information that people receive and disseminate is uncorroborated, and often accepted as valid. This phenomenon constitutes the root cause of societal upheaval with particularly acute ramifications when there are political changes in power. The proliferation of sensationalist headlines has been instrumental in propagating deceptive content pertaining to ethnic, political, and religious matters thereby amplifying instability throughout the country. Its easily shareable format only compounds this problem further.

The extensive usage of the Internet and social media has also led to the growing popularity of online advertisements, which has been accompanied by a disturbing spread of clickbait content. (Al-Sarem et al., 2021) Advertisement revenue is one of the driving factors that content creators focus on, chasing how many clicks they get while depreciating the quality of content they produce and share. This behavior and practice lead to users feeling deceived, irritated, and unsatisfied (Mowar et al., 2021). Most of the content on websites is supported by online advertising, which is an inevitable part of the current Internet. The economy of online advertising has become a de-facto standard. They are run under an economic model known as Clickthrough rate (CTR). Since advertisers are paid per click, clicking on ads presents ethical issues (Zeng et al., 2020).

In February 2017, First Draft, a nonprofit coalition company working on ethical guidance of the social web information published a spectrum of information disorder categories grouped into seven, mainly based on the misinformation or disinformation intensity. These categories include Misleading Content, Fabricated Content, False Connection, Imposter Content, Satire or Parody, False Context, and Manipulated Content. All these groups are inherently relevant in forming clickbait content.

The above-mentioned seven categories in terms of presenting clickbait can be expressed as follows:

- *Misleading content* – using misleading information to frame a particular issue, it can be a case of where advertised catchy headline misleads user to unrelated content.
- *Fabricated content* – a newly created false content prepared for the purpose of deception and baiting users into clicking a link.
- *False connection* – when headlines, captions and visual thumbnails don't support one other regarding a certain content.
- *Imposter content* – impersonating a genuine source and falsely reporting on their behalf, used to fuel hoaxes and rumors.
- *Satire or Parody* – a sarcastic presentation of a content where it may potentially fool people without proper disclaimers.
- *False context* – sharing genuine information with false or exaggerated contextual additives.
- *Manipulated content* – manipulating or disfiguring genuine information or imagery to deceive and gain traction with false basis.

As a practice used by news outlets to attract clicks using sensational language that ultimately disappoints readers. This type of pollution has negative long-term effects on people's relationship with news. While the need for web traffic and clicks makes it unlikely that clickbait techniques will disappear, their use of polarizing language is connected to broader issues such as the public's perception and engagement with journalism over time.

2.2. Digital Literacy

A rising corpus of studies points to the importance of digital literacy in how people consume online content, particularly clickbait news. Social media literacy is becoming increasingly important as more and more people turn to these platforms as their primary source of news and information. Digital literacy encompasses a range of skills, including the ability to critically evaluate the accuracy and reliability of online sources, to effectively search for and access relevant information, and to protect oneself from online threats like malware and phishing scams. It has been shown that individuals with higher levels of digital literacy are less likely to fall prey to clickbait headlines and are better equipped to navigate the complex landscape of online media (Munger et al., 2018). Web tracking data has further confirmed that digital literacy, in addition to ideology, strongly predicts the propensity to consume misinformation, highlighting the importance of promoting digital literacy skills among all Internet users.

2.2.1. Digital Literacy in Ethiopia

An article from K-flip Knowledge Hub, an Ethiopian newsletter platform, reported that the adult literacy rate in Ethiopia has improved to 51.8% in 2017, but a gender gap of 14.8% still exists. Ethiopia faces a long journey in developing digital skills due to limited data, including a lack of standardized digital skills assessment systems and exclusion from several reports. According to the World Economic Forum's Global Competitiveness Report for 2019, among 141 nations, Ethiopia ranks 137th in terms of ICT adoption and 100th in terms of digital skills, trailing Gabon, Zambia, Mauritius, Mali, Lesotho, Botswana, and Uganda.[1]

According to the “Digital 2023: Ethiopia” report, as of early 2023, Ethiopia had witnessed significant digital adoption and usage trends. The country recorded 20.86 million Internet users, indicating an Internet penetration rate of 16.7 percent. Additionally, Ethiopia had 6.40 million social media users, accounting for 5.1 percent of the total population. Furthermore,

[1] https://kflip.info/2023/01/14/digital-literacy-in-ethiopia/

the country boasted 66.80 million active cellular mobile connections, representing 53.5 percent of the population. These figures demonstrate the growing presence of digital technologies and connectivity in Ethiopia. In January 2023, Ethiopia had a total of 20.86 million Internet users, reflecting an Internet penetration rate of 16.7 percent among the country's population. The report citing Kepios also states that Internet users in Ethiopia grew by 520 thousand individuals, indicating a 2.6 percent increase between 2022 and 2023.[2]

Ethiopian citizens face a number of obstacles when trying to develop or improve their digital skills, including low literacy rates, a lack of proficiency in the English language, inadequate electricity coverage, poor connectivity, and limited access to the Internet.

The Ethiopian government has acknowledged the significance of digital literacy in propelling the country's economy, despite the absence of a comprehensive national digital skills framework. The Ministry of Innovation and Technology (MiNT) is developing a digital skills framework and action plan under the Digital Ethiopia 2025 initiative that prioritizes an inclusive digital ecosystem with a focus on marginalized communities, including rural areas and women.

In Ethiopia, low digital literacy levels and limited access to technology are identified as major obstacles to the development of digital skills. As web tracking data has shown, low digital literacy is a crucial moderator of online behavior such as the consumption of clickbait news. Thus, the country's low levels of digital literacy and lack of legal framework to address the issue contribute to a higher likelihood of the public consuming misleading and exaggerated headlines or titles presented in clickbait form.

[2] https://datareportal.com/reports/digital-2023-ethiopia

2.3. Clickbait Detection

Clickbait detection is a task that falls under the broad umbrella of Natural Language Processing (NLP). NLP is an interdisciplinary field of computer science and linguistics that focuses on the interaction between human language and computers. Its primary aim is to enable machines to understand, process, and generate natural language data in a way that is meaningful to humans. Clickbait detection involves using NLP techniques to automatically identify and classify clickbait content in online social media posts, news articles, and headlines (Klairith & Tanachutiwat, 2018). It has recently gained traction in academic research due to the growing prevalence of clickbait content in the digital world and its potential to manipulate public opinion and spread misinformation.

Clickbait can be detected using lexical similarity algorithms that analyze the similarity between the advertised headlines and the actual content. The semantic meaning of the headline and the content should be closely related because headlines are meant to inform readers about the subject of the article. If not, the headline may contain attractive information to engage readers' attention. Thus, such uncorrelated relationships are the hallmark of clickbait articles (Zheng et al., 2018).

2.3.1. Clickbait Detection Approaches

To address the issue of clickbait, researchers have developed various approaches for detecting clickbait content as well as a combination of them. The following section discusses an overview of the fundamental techniques used for clickbait detection.

Content-based methods are one of the most widely used techniques for clickbait detection. These methods analyze the content of an article or post to identify features that distinguish clickbait from non-clickbait content. Some common features used include the length of the headline, the use of superlatives or hyperbolic language, and the inclusion of certain keywords or phrases (Geçkil et al., 2018). A content-based approach mainly utilizes linguistic-cues for clickbait detection. This method analyzes the language used in the headline or post to identify patterns and characteristics that are typical of clickbait content. Some common linguistic features used in these methods include lexical and syntactical analysis features that consider the use of emotionally charged language, the use of question and rhetorical formats, and the use of specific pronouns and superlative adjectives (Indurthi & Oota, 2017). The strategy involves detecting baiting or provocative content, which typically has a writing style that is meant to attract readers, in contrast to legitimate content.

Researchers have employed different data representation models and manually created characteristics to identify and analyze news articles and social media posts. This method focuses on extracting linguistic characteristics solely based on the content of the dataset, without any external knowledge or social context (Awol & Gashaw, 2022).

Social Context-based methods are another approach to clickbait detection. These methods analyze the social context and engagement metrics of an article or post, such as the number of shares and likes it receives, to identify patterns that are typical of clickbait content. For example, these methods may look for posts that have a high number of shares or likes in a short period of time. Kaothanthong et al., (2021) discussed how context-based features, such as behavior-based analysis, are utilized to remove clickbait posts from social media platforms. Facebook, for instance, analyzes the click-to-share ratio and the period of time that the users spend on the post. On the other hand, Twitter identifies clickbait tweets through a Naïve-Bayes classifier, along with analyzing the users' behavior (Potthast et al., 2016).

Visual-based methods analyze the visual and graphical elements of an article or post, such as the images or videos used, to identify features that are associated with clickbait. For example, these methods may look for images that are unrelated to the content of the article or images that are designed to evoke strong emotions. More intricate analysis in this method includes speech-title similarity, engagement scores such as the number of likes and dislikes (Varshney & Vishwakarma, 2021).

A hybrid approach combines supplementary information from many perspectives with techniques, methods, and features from content-based methods and social context-based methods. Researchers like Kumar et al., (2018) and Chawda et al., (2019) attempted to combine different methodologies and features in order to effectively detect clickbait on the web.

2.3.2. Data Representation Methods

To train a machine learning model from the processed text, the textual data must be presented in numerical form. The most common text representation techniques include Bag of words, TF-IDF and word embedding. According to studies, text representation analysis, specifically lexicon-based analysis, is commonly used to identify clickbait in content-based approaches.

a) TF-IDF Word Representation

TF-IDF is a technique that evaluates the significance of a word in a particular document based on its frequency of occurrence (Term Frequency) and adjusts it based on how commonly it appears in all documents (Inverse Document Frequency).

Adelson et al., (2017) has put forward stating that $S_w^{(d)}$ be the TF-IDF score of a given word w in document d. $tf_w^{(d)}$ is the frequency of the word w in document d, and df_w is the number of documents containing word w. For a body of D documents:

$$S_w^{(d)} = tf_w^{(d)} \times -\log\frac{df_w}{|D|} \qquad (2.1)$$

The idea behind using this method for clickbait data representation is that clickbait titles often contain words that are essential to their meaning but are not necessarily common among other content, such as hyperbole and sensational terms. It is considered a baseline feature for this experiment.

b) Bag of Words Representation

The frequency of each word in the text is used as a feature for training a text classification model. This approach is simple and easy to implement, but can be limited in capturing the context and meaning of words. To overcome this limitation, techniques such as n-grams and stop word removal can be employed to improve accuracy. Additionally, this method can be combined with other representation techniques such as TF-IDF or word embeddings to enhance the performance of the models.

c) Word Embedding Representation

Word embedding is a technique where words from a list or vocabulary are mapped and represented by a vector of real numbers. It is a technique that represents words in a high-dimensional vector space, where each dimension corresponds to a different feature of the word. The vectors are learned from a large corpus of text data using unsupervised machine learning algorithms, such as Word2Vec or fastText. The resulting vectors capture the semantic and syntactic relationships between words, which makes them useful for text classification tasks. Word Embedding can be used to represent the text of articles as a matrix of word vectors. The matrix can then be fed into a machine learning algorithm, such as a neural network or a decision tree, which learns to classify articles as clickbait or not based

on the patterns in the word vectors (Li et al., 2015). By using Word Embedding, researchers can improve the accuracy of their models.

Distributed word embeddings are a feature learning method where words from the vocabulary are mapped to vectors of real numbers. These representations group similar words together, which allows learning algorithms to better understand the relationships between words and concepts. This can lead to improved performance on a variety of NLP tasks, such as text classification, machine translation and sentiment analysis (Mikolov et al., 2013).

Mikolov et al., (2013) also introduced the Skip-gram model, a neural network architecture for extracting high-quality vector representations of words from large unstructured text data. The Skip-gram model is more efficient than previous neural network architectures for learning word vectors because it does not require dense matrix multiplications. This makes it possible to train the Skip-gram model on large datasets in a short amount of time.

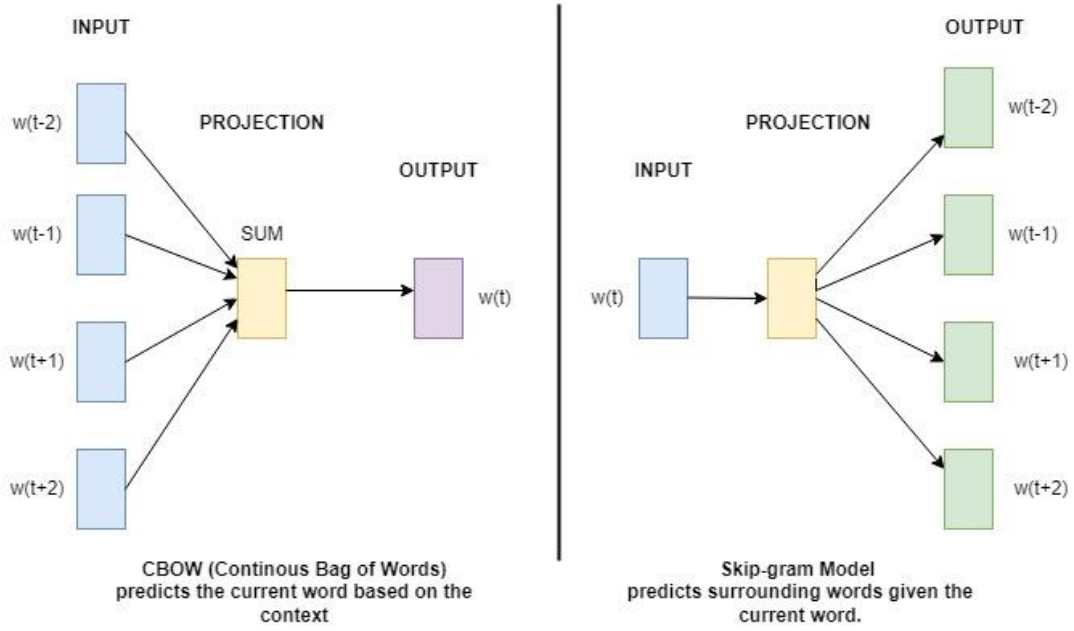


Figure 2.1 CBOW and Skip-gram Word Representation Models

The continuous Bag of Words (CBOW) is a predictive model that predicts a target word based on its context. The context is defined as the words that surround the target word. CBOW is a shallow neural network that has a single hidden layer. The input layer of the network is the context words, and the output layer is the target word. The hidden layer learns to represent the semantic meaning of the context words, and the output layer learns to predict the target word (Mikolov et al., 2013).

2.4. Types of Machine Learning and Neural Network Approaches

Numerous studies have put forth diverse methodologies employing a range of classic machine learning algorithms, including Logistic Regression (Geçkil et al., 2018), Support Vector Machine (Chakraborty et al., 2016), Decision Tree (Klairith & Tanachutiwat, 2018), Random Forest (Potthast et al., 2016), as well as deep learning techniques such as Recurrent Neural Network (Dimpas et al., 2018), Long Short-Term Memory (Manjesh et al., 2017), Gated Recurrent Unit (Dong et al., 2019) and Convolutional Neural Network (Fu et al., 2017; Zheng et al., 2018) for the study of clickbait detection.

2.4.1. Logistic Regression

Logistic Regression is a method that takes a linear combination of weights and features where it performs a nonlinear transformation. It is a technique that uses predictive analysis; the likelihood or probability of an outcome based on observable traits; to decide whether the content is clickbait or not. It builds a model using labeled data from both clickbait and non-clickbait texts to check for certain text characteristics that are associated with clickbaiting.

2.4.2. Support Vector Machines

SVMs or support vector machines, attempt to categorize the points in a dataset. SVMs attempt to separate the data based on their features and use this separation to make judgments rather than attempting to estimate a function. It is a technique based on a classification algorithm that can be used to classify clickbait by detecting text features, such as word and punctuation usage. The model produces a hyperplane to separate data points into two or more classes and classifies new data points by using the hyperplane's orientation.

2.4.3. Random Forest

Random forest is an ensemble learning method that consists of a large number of decision trees. The predictions of the many decision trees are combined to provide the final prediction after each tree has been trained on a random portion of the training data. Here are some of the areas where Random Forest has been used for text classification:

- Spam filtering: to develop spam filters that can identify and block spam emails.
- Sentiment analysis: to identify the sentiment of text as positive, negative, or neutral.
- Topic classification: identify the topic (politics, sports, or art) of a piece of text.

2.4.4. Neural Networks

Neurons are specialized cells in the human brain that communicate with one another, forming a complex biological neural network. The functioning of these interconnected nerve cells has served as inspiration for addressing complex data problems through the development of artificial neural networks.

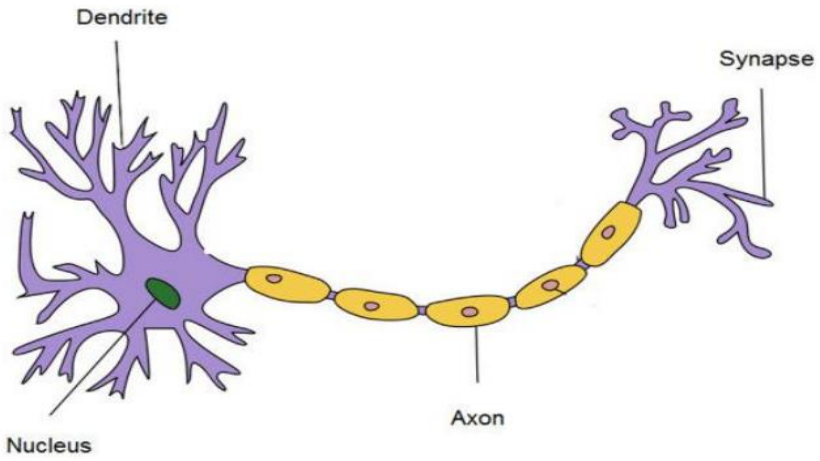


Figure 2.2 Biological Neuron [3]

In the context of neural networks, the units within the network are referred to as nodes. The initial layer of nodes, which is closest to the input, is known as the input layer or input nodes. Conversely, the final layer of nodes is referred to as the output layer or output nodes. The output layer performs an aggregation function and may also incorporate a transfer function. The transfer function is responsible for scaling the output to fit within the desired range. In combination with the aggregation and transfer functions, the output layer executes an activation function (Kotu & Deshpande, 2018). This basic architecture, depicted in Figure 2.3, consists of multiple layers: an input layer, an output layer and two hidden layers. The perceptron represents the most elementary form of an artificial neural network (ANN). It operates as a feed-forward network, with data flowing in a single direction.

Deep learning focuses on the training and utilization of neural networks with multiple layers. It is characterized by the ability of neural networks to automatically learn and extract hierarchical representations of data, leading to increasingly complex and abstract features. It has revolutionized various fields, including computer vision, natural language processing, and speech recognition, by enabling models to learn directly from raw data without the need for explicit feature engineering.

[3] https://towardsdatascience.com/mcculloch-pitts-model-5fdf65ac5dd1

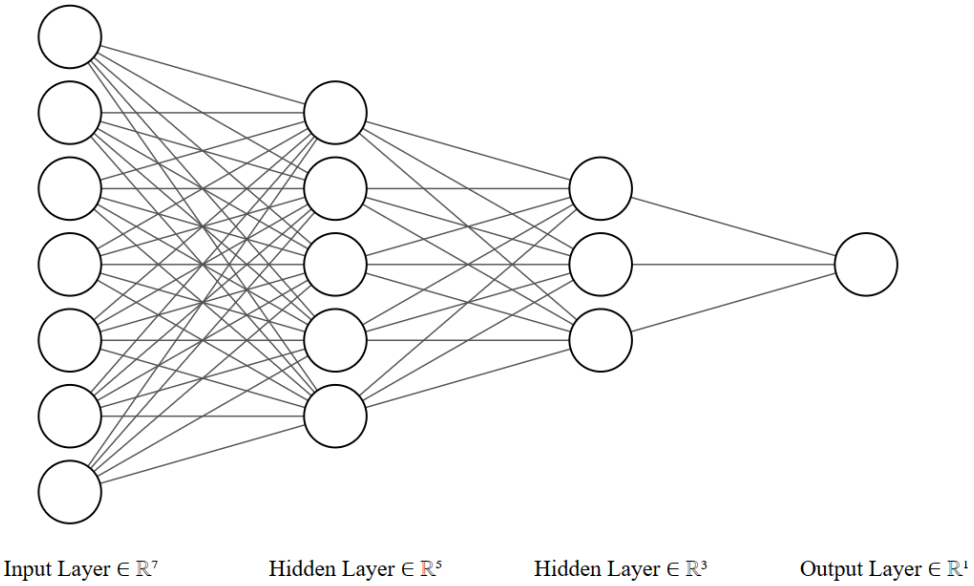


Figure 2.3 Artificial Neural Network [4]

A simple linear model can be represented as a neural network with one input layer, one output layer, and several nodes in each layer. Hidden layers processes input data by performing mathematical operations and applying activation functions to produce output for subsequent layers. The input layer nodes represent the input variables, the output layer node represents the output variable, and the weight of each connection represents the coefficient for the corresponding input variable.

2.4.5. Recurrent Neural Networks

Recurrent Neural Networks are a type of neural network that can process sequences of data. They are made up of a series of nodes, each of which has a weight and a bias. The weights and biases are adjusted during training to minimize the error between the predicted and actual output. A standard RNN has one internal state where its output at each time-step is dependent on that of the previous time-steps (Anand et al., 2017). Mathematically, given an input sequence x_t, an RNN computes its internal state h_t by:

$$h_t = g\left(U h_{t-1} + W_x x_t + b \right) \quad (2.2)$$

[4] http://alexlenail.me/NN-SVG/index.html

2.4.5.1. Long-Short Term Memory (LSTM)

Long short-term memory (LSTM) networks are a type of recurrent neural network (RNN) that are well-suited for tasks that require long-term memory, such as natural language processing and speech recognition. The gate mechanism consists of three parts: an input gate, a forget gate, and an output gate.

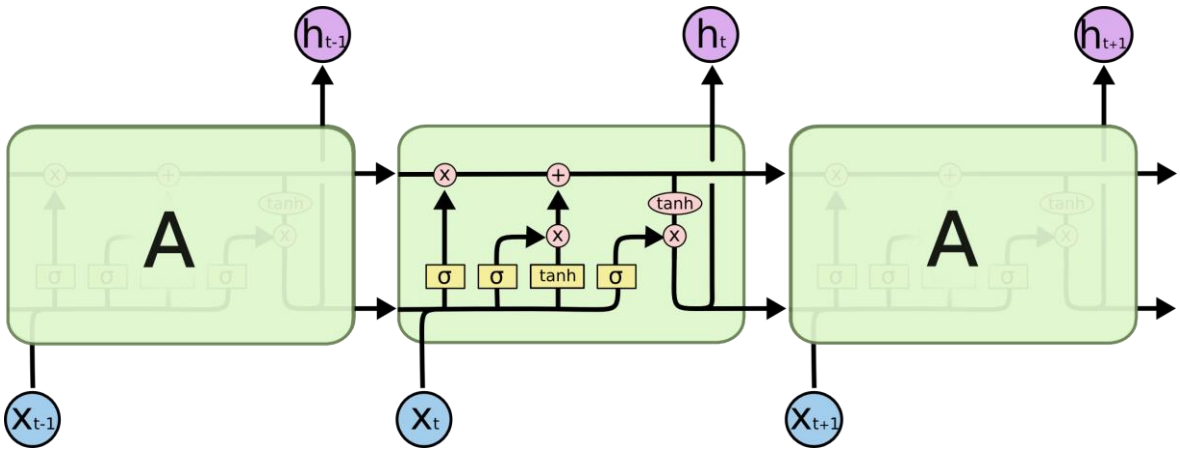


Figure 2.4 Long Short-Term Memory Networks[5]

LSTMs are well suited for capturing long-term dependencies among chunks of text information contained within a single news article title or post description to distinguish between clickbaiting words among all other words present in the text (Dimpas et al., 2018). LSTMs are particularly helpful when dealing with highly visible topics like entertainment celebrities because they capture relationships across different types of contextual information contained within an individual title or post description quickly and accurately.

$$
\begin{aligned}
f_t &= \delta_g\left(W_f x_t + U_f h_{t-1} + b_f\right) \\
i_t &= \delta_g\left(W_i x_t + U_i h_{t-1} + b_i\right) \\
o_t &= \delta_g\left(W_o x_t + U_o h_t + b_o\right) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \delta_c\left(W_c x_t + U_c h_{t-1} + b_c\right) \\
h_t &= o_t \odot \delta_h\left(c_t\right)
\end{aligned}
$$

(2.3)

Where $f_{t,} i_{t,} o_t$ *and* c_t are forget gate, input gate, output gate as well as memory activation vector time respectively. δ represents the sigmoid function with a hyperbolic tangent across an element-wise vector product.

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

2.4.5.2. Gated Recurrent Unit

GRUs are a different type of recurrent neural networks that can learn long-term dependencies. They do this by using a gate mechanism that controls the flow of information through the network. The gate mechanism consists of two parts: an update gate and a reset gate. The update gate controls how much of the previous state is updated with the current input. The reset gate controls how much of the previous state is forgotten (Klairith & Tanachutiwat, 2018).

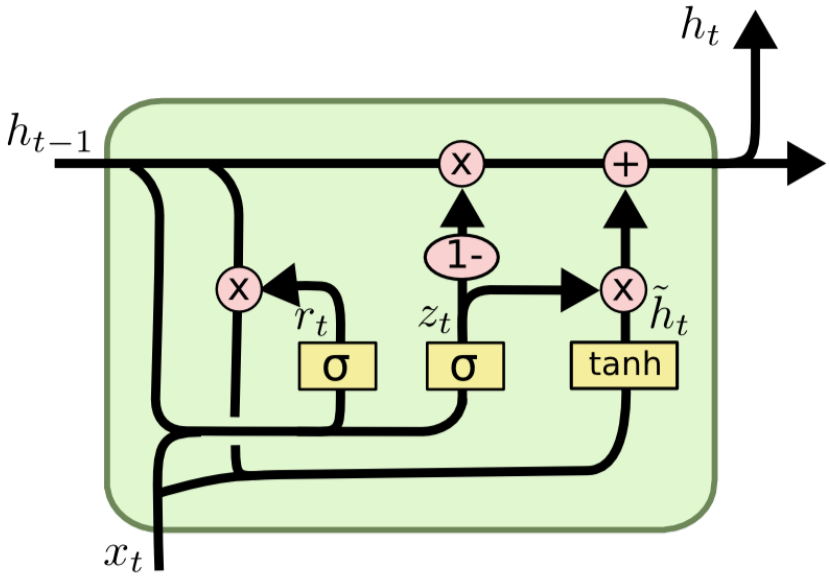


Figure 2.5 Gated-Recurrent Units Model [6]

A GRU cell calculates its internal state using the following iterative format:

$$
\begin{aligned}
z_t &= \delta\left(W_z x_t + U_z h_{t-1} \right) \\
r_t &= \delta\left(W_r x_t + U_r h_{t-1} \right) \\
\tilde{h}_t &= \tanh h\left(W_h x_t + U\left(r_t \odot h_{t-1} \right) \right) \\
h_t &= \left(1 - z_t \right) \tilde{h}_{t-1} + z_t h_t
\end{aligned}
$$

(2.4)

Where *z*, *r*, *h* are respectively update gate, reset gate and candidate activation with *W and U* memory cell activation, those are parameters of the GRU with an element-wise vector product.

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

2.4.6. Convolutional Neural Networks

Convolutional Neural Networks (CNNs) have primarily been used for image and visual data processing tasks. However, they have also shown promising results in text classification tasks. CNNs can effectively capture local patterns and dependencies within sequences of text, making them suitable for tasks such as sentiment analysis, document classification, and spam detection (Kotu & Deshpande, 2018).

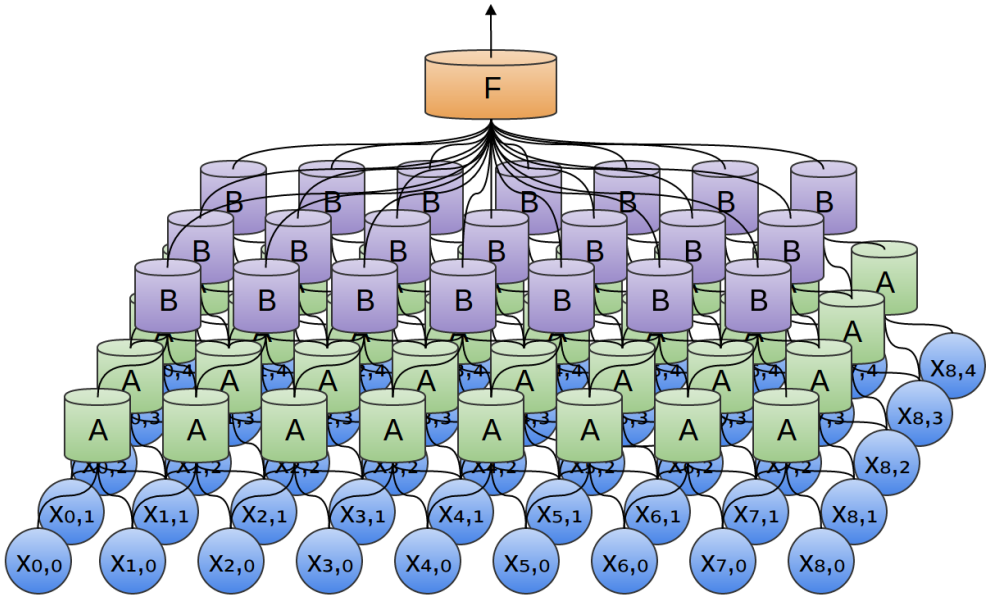


Figure 2.6 2D Convolutional Neural Network [7]

In text classification, CNNs operate on the premise that words and phrases within a text have local dependencies and contribute to the overall meaning and context. A CNN for text classification typically consists of an embedding layer, convolutional layers, pooling layers, and fully connected layers. The embedding layer maps words in the text to dense vectors, which serve as the input for subsequent layers. The convolutional layers apply filters of different sizes to the embedded text, sliding them over the sequence to extract local features.

A fully connected linear layer is then applied to this new representation to perform classification or regression. Briefly, a convolution is a transformation that takes a weighted matrix $\theta \, \varepsilon \, \mathbb{R}^{m \times n}$and slides it over a larger target matrix x, collapsing the product between the two into an entry in a new matrix (Adelson et al., 2017). Properly, the new entry matrix is represented as:

$$a = \sum_{1=1}^{m} \sum_{j=1}^{n} \theta_{ij} \, x_{ij} \qquad (2.5)$$

[7] https://colah.github.io/posts/2014-07-Conv-Nets-Modular/

It employs several weight matrices to combine and condense the input data into a transformed vector representation. This transformed representation is subsequently fed into a fully connected linear layer for the purpose of classification or regression tasks.

CNNs for text classification have demonstrated good performance, especially when combined with techniques like pre-trained word embeddings such as Word2Vec or GloVe. These pre-trained embeddings capture semantic and syntactic relationships between words, providing valuable contextual information to the CNN model (Indurthi & Oota, 2017).

2.5. Semantic Analysis Approach

Semantic analysis plays a crucial role in clickbait detection studies, as it enables the exploration of the underlying meaning and context of textual content. Researchers employ machine learning techniques to analyze advanced semantic features such as sentiment, topic, context, relationship, emotion, and readability. In the context of clickbait detection, semantic analysis is crucial for identifying the underlying message and intent of clickbait content. It helps in distinguishing clickbait from genuine content by uncovering linguistic cues, tones, sensational language, and misleading information.

According to a recent study by Bronakowski et al., (2023), a novel approach combining semantic analysis and machine learning techniques was proposed for accurately categorizing clickbait and non-clickbait headlines. The research explored thirty unique semantic features and evaluated six different machine learning classification algorithms, both individually and as ensembles. Notably, top-performing models, such as the support vector machine and gradient-boosted decision tree algorithm, achieved an accuracy of 98%.

2.6. Multimodal and Multilingual Approach

The realm of clickbait detection study expands to the multilingual and multimodal aspects. Clickbait exists in various languages and incorporates different modes of communication, such as text, images, and videos. Research by Fakhruzzaman & Gunawan, (2021) emphasizes the importance of considering the multilingual nature of clickbait for effective detection where they implemented a pretrained Multilingual-BERT model.

Additionally, the study by Varshney & Vishwakarma, (2021) highlights the significance of analyzing multimodal features, including visual, auditory, and textual cues, for accurate clickbait detection where they designed a model to detect clickbait from YouTube videos taking the audio, video, and user engagement metrics as features.

2.7. Amharic Language

Amharic (Amarəññа, አማርኛ) employs the Ethiopian script known as Ethiopic or Fidel, which is the only script of African origin. It stands as the second most widely spoken Semitic language globally, after Arabic, and serves as the official working language of the Ethiopian government (Gereme et al., 2021). Amharic is classified within the Ethiopian Semitic branch of South Semitic languages, which has a well-established presence in Ethiopia. Alongside the other eleven Ethiopian Semitic languages, Amharic showcases typical Semitic characteristics, including its inflectional and derivational morphology patterns. Additionally, it features distinct attributes such as a subject-object-verb (SOV) word order, likely influenced by prolonged interaction with Cushitic languages (Zitouni, 2014). However, Amharic is classified as a "low-resource" language, indicating its limited availability of tools necessary for natural language processing (NLP) and other computational linguistic applications (Gereme et al., 2021).

2.7.1. Amharic Orthography

Amharic employs the Ethiopic script (Fidel) as its writing system. It encompasses 26 consonant phonemes, with an additional /v/ sound utilized in foreign loan-words, and 7 vowels. The writing system of Amharic distinguishes four consonants that were present in Ge'ez but have been lost in Amharic. It should be noted that there is no universally agreed-upon transliteration method for Amharic. Therefore, the fundamental character set comprises a single character for each possible combination of 33 consonants and 7 vowels. Additionally, there are 37 additional characters that represent labialized versions of the consonants followed by specific vowels which are ኣ(a), ኡ(u), ኢ(i), ኣ(a), ኤ(e), እ(ə), ኦ(o). In total, the complete system consists of 268 characters. Ge'ez numerals also exist; however, in modern times, these are gradually being substituted by the Arabic numerals commonly used in European languages (Zitouni, 2014).

	Character order						
	1	2	3	4	5	6	7
ሀ	ሀ(ha)	ሁ(hu)	ሂ(hi)	ሃ(hā)	ሄ(hé)	ህ(he/h)	ሆ(ho)
ለ	ለ(la)	ሉ(lu)	ሊ(li)	ላ(lā)	ሌ(lé)	ል (le/h)	ሎ(lo)
ሐ	ሐ(ha)	ሑ(hu)	ሒ(hi)	ሓ(hā)	ሔ(hé)	ሕ(he/h)	ሖ(ho)
መ	መ(ma)	ሙ(mu)	ሚ(mi)	ማ(mā)	ሜ(mé)	ም(me/h)	ሞ(mo)

Figure 2.7 Amharic Letters (Fidel) sample with Transliteration

2.7.2. Amharic Morphology

Derivational morphology in Amharic involves the addition of affixes or modifications to the root to create new words with different meanings or lexical categories. This process allows for the formation of nouns, verbs, adjectives, and adverbs. Inflectional morphology in Amharic primarily involves the modification of word forms to indicate grammatical features such as tense, aspect, mood, gender, and number. Nouns, on the other hand, can be inflected for gender and number (Zitouni, 2014).

Understanding the derivational and inflectional morphology of Amharic is crucial for natural language processing tasks such as part-of-speech tagging, morphological analysis, and machine translation. However, the complex nature of Amharic morphology poses challenges in terms of resource availability, such as annotated corpora, which are necessary for developing effective NLP systems (Mossie & Wang, 2018).

2.7.3. Amharic Semantics

The semantics of the Amharic language involve producing accurate representations of meaning in utterances. Traditional methods of semantic analysis in Amharic have relied on lexicons, grammars, and limited language features. However, a study by Demlew, (2021) introduced an Amharic semantic parser that utilizes an attention-enhanced sequence-to-sequence neural model. Their model encodes Amharic utterances and generates their corresponding logical forms. The existing pre-trained models in the public domain are primarily designed to serve as multilingual semantic models, making them ill-suited for addressing the specific requirements of low-resource languages. Yimam et al., (2021) introduced a set of distinct semantic models for Amharic.

2.7.4. Amharic language on the Web

The use of Amharic language on the web has gained significant traction, especially within the Ethiopian community. Amharic is a well-known language with a large speaking population that provides a platform for expression, connection, and communication online. The growing use of social media sites like Facebook, Twitter, and YouTube has given Amharic speakers access to an online community to share their thoughts, ideas, and experiences (Yimam et al., 2020). Users can create and engage in content written in Amharic, including posts, comments, and messages, fostering a sense of community and facilitating communication among Amharic speakers across different geographical locations.

2.8. Related Works on Clickbait Detection

Early clickbait detection studies heavily relied on feature engineering methods. Chen et al., (2015) discussed that to identify clickbait, different techniques can be employed such as lexical features, image-based features, and user-behavior features.

Following then, computer scientists studied clickbait in social media like Twitter where they encountered a massive number of backlinked tweets with catchy phrases. They created a handcrafted features model to identify clickbait. There are 203 text features in their study, including the tweeter tags, emotional polarity, stopword number, and number of punctuation marks. A supervised classification algorithm received these features, which produced 0.76 precision and 0.76 recall (Potthast et al., 2016).

2.8.1. Clickbait Detection using Machine Learning

Several research groups have used machine learning techniques to detect clickbait. Chakraborty et al., (2016) implemented a machine-learning classifier to detect clickbait. To train their classifier, they used fourteen sets of components including linguistic analysis, N-gram, and word patterns. They also attempted to develop a browser add-in or extension named "Stop clickbait". They achieved 89% accuracy using an SVM classifier to detect and circumvent clickbait.

Another work by Biyani et al., (2016) regarding clickbait detection and classification was performed, where they first categorized the different types of clickbait into eight common groups (Ambiguous, Exaggeration, Formatting, Graphic, Inflammatory, Teasing, Bait-and-switch, Wrong). They used document informality and reading difficulty as a metric. Their approach utilized the classification algorithm known as gradient-boosted decision trees using four features – Content and Informality, Forward Reference, Similarity, and URL. They have also made comparisons based on the density of the clickbait dataset.

To accommodate some features outside of the linguistic analysis, Zheng et al., (2017) proposed a model that considers user-behavior features to detect clickbait. Their model computes the clickbait likelihood in accordance with user behaviors noticed under specific circumstances using the results of the traditional feature models as inputs.

2.8.2. Clickbait Detection using Neural Networks

Handcrafted lexical and syntactical features are limited as they are heavily dependent on expert knowledge. Fu et al., (2017) introduced a general end-to-end Convolutional Neural Network based method that automatically detects clickbait. It was able to induce several useful features for the end task without depending on any peripheral properties. They ran their experiments on an English dataset to show that their approach achieved steady results.

Agrawal, (2017) was also one of the first to apply a deep learning technique-based approach specifically using CNN that performs well on the classification of clickbait and non-clickbait headlines with 0.90 accuracy, 0.85 precision and a 0.88 recall on the English Twitter dataset. Similarly, Anand et al., (2017) proposed a neural network architecture based on Recurrent Neural Network (RNN) that showed satisfying results for clickbait detection.

Different types of blog articles, advertisements and headlines tend to use different ways to draw users' attention, and a pre-trained Word2Vec model cannot differentiate these diverse approaches. To address this issue, Zheng et al., (2018) propose a clickbait convolutional neural network (CBCNN) to consider not only the overall characteristics but also the specific characteristics of different writing styles.

Kumar et al., (2018) brought a multi-strategy approach to the clickbait detection problem. Their multimodal approach considers both textual and image features, to score the classified headlines using a bidirectional LSTM with an attention mechanism attaining a 65.37% F1 score.

It is also worth noting works by Khater et al., (2018) that proved that it is possible to identify clickbaits using the entire post while requiring a minimum number of features, and Shu et al., (2018) that proposed to generate artificial headlines from original documents with style transfer.

Another study on this subject was also conducted by Dong et al., (2019). The study measured the similarity between clickbait titles. They proposed a deep similarity-aware attentive model to capture and represent such similarities with better expressiveness. It implemented the Bi-GRU algorithm as a classification method to determine the correlations between titles and content. It attained an accuracy above 85% across two different datasets.

Among the recent studies to explore the use of deep learning techniques for clickbait detection, Wu et al., (2020) proposed an approach with style-aware title modeling and co-

attention. The researchers used Transformers to acquire the representations of the title and body in the content and derive two content-based clickbait scores for the title and body. They used a character encoder to learn styles of title representation to capture the artistic patterns of the title.

Naeem et al., (2020) presented a deep-learning framework for clickbait detection. The authors trained a model to detect the underlying structure and intrinsic characteristics of clickbait to train the model for knowledge discovery and then used the model in their LSTM decision-making apparatus called POSAM (Part of Speech Analysis Module). They were able to classify headlines as clickbait or legitimate news up to 97% performance. Repeatedly recategorizing by finetuning the machine learning models, and the features was an approach yielding better results.

Pujahari & Sisodia, (2021) proposed a hybrid categorization technique for separating clickbait and non-clickbait articles. Its architecture followed the integration of different features like sentence structure and clustering. t-Stochastic Neighborhood Embedding approach was applied for clustering along with word vector similarity.

All the above-mentioned works are focused on identifying, classifying, and detecting clickbait elements using various methods in an English language setting. Despite the fact that clickbaits are widespread among all languages used on the Internet, limited effort has been made for non-English languages compared to English. (Fu et al., 2017)

2.8.3. Clickbait Detection for non-English Languages

The works reviewed below are conducted in non-English and resource-poor languages in the clickbait detection domain. There has been little research published targeting other languages recently. Some notable clickbait detection studies in non-English languages include Filipino (Dimpas et al., 2018), Thai (Klairith & Tanachutiwat, 2018; Kaothanthong et al., 2021). Indonesian (William & Sari, 2020; Siregar et al., 2021; Fakhruzzaman & Gunawan, 2021), Hungarian (Vincze & Szabó, 2020), Chinese (C. Zhang & Clough, 2020), Telugu (Marreddy et al., 2021), Arabic (Al-Sarem et al., 2021) and Turkish (Genç & Surer, 2021).

Klairith & Tanachutiwat, (2018) proposed six different sets of neural networks - BiLSTMs with word-level embedding models for detecting clickbait in Thai language. They crowdsourced 30,000 headlines, and the model performed well achieving an accuracy rate

and F1 score of 0.98. They noticed that deep learning methods can be adopted in the Thai language without the need for traditional NLP feature engineering.

In the Philippines, Dimpas et al., (2018) collected English and Filipino headlines from social media and classified clickbaits using a neural network architecture and Word2Vec for word embedding. They were able to achieve a 91.5% accuracy level in their experiments. They suggested more improvements can be made with more layers and a larger dataset.

After the 2020 surge in clickbait in Indonesia, where it created mass confusion in the country. Studies on clickbait detection were often investigated. William & Sari, (2020) introduced the CLICK-ID dataset for Indonesian news, comprising 15,000 annotated texts. Fakhruzzaman & Gunawan, 2021) presented a web application using BERT and a RESTful API that divested the computing resources required to train the model on the cloud server. Their study proposed the design of a web-based application to detect Indonesian clickbait using IndoBERT as a language model. They were able to achieve a mean performance of ROC-AUC of 89%. Another study on this subject involved using data from the most popular online news sites to automatically identify clickbait headlines in Indonesian. They used the RNN based on LSTM model with word embedding to represent text data as vector data. It resulted in an accuracy of 83% in classifying clickbait. (Siregar et al., 2021)

A proof-of-concept study by Genç & Surer, (2021) expanded the Turkish clickbait dataset by building ClickbaitTR, which is an open-source clickbait dataset of 48,000 tweets. They applied different models such as Artificial Neural Network (ANN), Logistic Regression, Random Forest, LSTM, BiLSTM and Ensemble Classifier to the dataset, and compared their results in experiments. They concluded that LSTM, ANN, and Ensemble Classifier have achieved similar performances.

Furthermore, a more recent and broadened work that discussed overcoming NLP challenges in resource-poor languages for clickbait detection, while using Telegu as the target language was done by Marreddy et al., (2021). They stated that clickbait-related tasks cannot be effectively handled with machine translation as a cross-language experiment because the meaning of clickbait content in the source language may alter, and the curiosity component might be lost in translation. One of their contributions is a large, annotated clickbait dataset. They were also able to generate three pre-trained language models RoBERTa, ALBERT, and ELECTRA for the Telugu language. Marreddy et al., (2021) also developed a benchmark system for detecting clickbait headlines written in Telugu. They also explored

the feasibility of various neural architectures and pre-trained transformer models. Using their large local clickbait dataset, the Light Gradient Boosted Machines model attained a 0.94 F1-score for clickbait headlines.

2.8.4. Related Works in Amharic Language

Regarding studies of clickbait detection in Amharic, there is no publicly available research on the subject. Here, we try to review related matters to the subject to gain more perspective and build better insight into achieving a clickbait detection model for Amharic. Some research in areas like text classification, hate speech detection, fake news detection and sentiment analysis for Amharic has been conducted. These works have been an input for developing a working model for clickbait detection for Amharic. Some of those works are reviewed as follows.

In their study on Amharic text classification, (Asker et al., 2014) employed machine learning techniques and investigated the impact of operations like stemming and POS tagging on classification performance. They utilized a medium-sized, manually annotated Amharic corpus and observed that stemming did not significantly affect the performance of text classification in Amharic. Their approach employed a bag-of-words methodology for text classification, which inherently suffers from high sparsity due to the nature of word representations. Consequently, the model encountered challenges in effectively utilizing limited information within a vast vector-space. Additionally, utilizing this approach resulted in difficulties such as the probability of encountering out-of-vocabulary words and the absence of contextual information.

Kelemework, (2013) conducted a study on automatic Amharic news classification using a neural network approach with Learning Vector Quantization on a dataset containing 1,762 items across nine categories. However, a drawback of using LVQ is that it may require more hidden units, which can slow down the processing time needed for classification.

A study around Hate Speech detection by (Mossie & Wang, 2018) proposed the utilization of Apache Spark. The authors developed a model to classify Amharic Facebook posts and comments as either hate speech or non-hate speech. For learning, Random Forest and Naïve Bayes algorithms were employed, while Word2Vec and TF-IDF techniques were utilized for feature selection - with 10-fold cross-validation, an accuracy of 79.83% was observed.

Yimam et al., (2020) studied sentiment analysis for Amharic social media texts, while also building annotation tools and classification models to support their work and future ones as well. They employed the FLAIR deep learning text classifier with network embeddings that are calculated from a distributional thesaurus. They additionally examined and concluded that the challenges in building a sentiment analysis system for Amharic emanate from the extensive use of sarcasm and figurative speech.

Several contributions are presented, including an Amharic fake news detection model, a general-purpose Amharic corpus (GPAC), a novel Amharic fake news detection dataset (ETH_FAKE), and Amharic fastText word embedding (AMFTWE) (Gereme et al., 2021). Amharic fake news dataset is crafted from verified news sources and various social media pages and six different machine learning classifiers Naïve bayes, SVM, Logistic Regression, Random Forest, and Passive aggressive Classifier models are built (Tazeze & R, 2021).

Hailemichael, (2021) aimed to address the detection of fake news in Amharic by leveraging deep learning algorithms and a newly curated dataset. It was observed that Bi-GRU and CNN surpassed other recurrent and attention-based models, yielding an impressive accuracy of 93.92% and an f1-score of 94%.

Minale & Assabie, (2021) utilized Stacked Bidirectional Long Short-Term Memory Networks to create a hate speech detection system, achieving F1-core accuracy of 94.8%. They categorized text into racial, religious, gender, and normal speech categories using a custom annotation tool. They noted that challenges arise from the scarcity of annotated datasets, the morphological complexity of Amharic, and the absence of previous studies.

Recently, Awol & Gashaw, (2022) studied lexicon-stance based Amharic fake news detection by utilizing a newly curated dataset of Amharic fake news sourced from Facebook. The experimental results demonstrated that the integration of hybrid features (combining lexicon and stance) yields significant improvements over previous lexicon-based detection approaches, with an increase in accuracy, precision, recall, and F1-score by 4.1%, 3%, 4%, and 4%, respectively.

Table 2.1 Summary of Related Works

Title, Authors, Year	Feature Extraction	Methods used	Limitations
Clickbait Detection (Potthast et al., 2016)	Teaser message, link, meta-information	Random Forest	Only the English Twitter dataset was considered, and crude feature engineering techniques were used.
Clickbait Detection using Deep Learning (Agrawal, 2017)	Tweets, Word2Vec	CNN	Only the English dataset from Twitter and Reddit was used, Lack of source rating and advert identification
A Convolutional Neural Network for Clickbait Detection (Fu et al., 2017)	SVM, Word2Vec	CNN	The detection time is long, and the dataset is limited to English news content.
Identifying clickbait: A multi-strategy approach using neural networks (Kumar et al., 2018)	Siamese Network Text Embedding	Bi-LSTM with Attention mechanism	The dataset was limited to English news headlines. Due to the integration of two deep learning systems, it was also computationally challenging.
A Deep Learning Framework for Clickbait Detection on Social Area Network using Natural Language Cues (Naeem et al., 2020)	Word2Vec and POSAM (Part of Speech Analysis Module)	LSTM	The dataset was limited to English news headlines. Part of Speech Analysis module also limited to English. Incapable of discovering the clickbait advertisement
Combating Fake News in "Low-Resource" Languages: Amharic Fake News Detection Accompanied by Resource Crafting (Gereme et al., 2021)	Amharic fastText Word Embedding	CNN	The dataset was limited to Amharic news headlines. No consideration for clickbait links, advertisement, or posts. Limited to the attribute similarity scoring of headlines.
Clickbait Detection in Telugu: Overcoming NLP Challenges in Resource-Poor Languages using Benchmarked Techniques (Marreddy et al., 2021)	GloVe and FastText Word Embedding	LightGBM with BERT Pre-trained Language Models	The dataset was limited to Telugu language. Benchmark only applies to Dravidian language family. Pre-trained BERT models were limited in identifying new vectors. Scope doesn't extend to advertisements.

CHAPTER THREE

3. RESEARCH METHODOLOGY

3.1. Chapter Overview

This section describes the methodology that was used to carry out this study. It presents tools and techniques applied to meet the objective of the study. It conveys the techniques for data collection, data preprocessing, feature extraction, model training, and evaluation. Ethical considerations for the entire study are also disclosed in this section. Figure 3.1 shows the general steps followed for this research study.

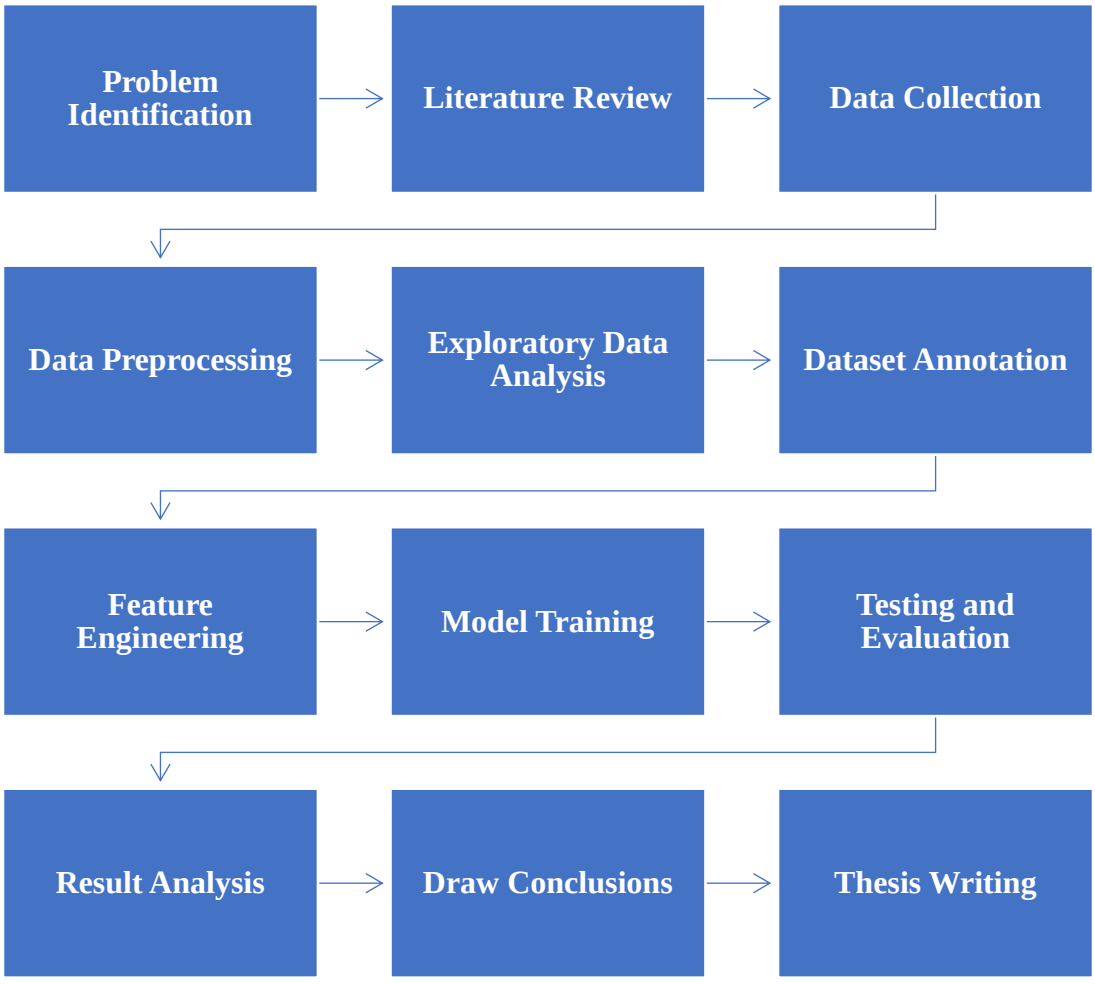


Figure 3.1 Steps of the Research Process

3.2. Research Design

This study has implemented an experimental research method. This approach allowed for the systematic investigation of clickbait detection techniques and their effectiveness in the Amharic language context. The study employed both qualitative and quantitative aspects, combining the strengths of these research approaches.

Qualitative aspects were incorporated through the collection and analysis of textual data from social media platforms, such as YouTube, Twitter, and Facebook. The qualitative analysis focused on understanding the characteristics of clickbait content in the Amharic language, including linguistic patterns, rhetorical strategies, and manipulative techniques used to engage users.

Quantitative aspects were employed to evaluate the performance of various neural network models in clickbait detection. Metrics such as accuracy, precision, recall, and F1-score were used to assess the effectiveness of the models. The quantitative analysis provided objective measures of the models' performance and their ability to classify clickbait and non-clickbait content accurately.

By combining qualitative and quantitative approaches, the study offered a comprehensive understanding of clickbait detection in the Amharic. The qualitative analysis shed light on the linguistic and cultural aspects of clickbait, while the quantitative evaluation strengthened with the exploratory data analysis provided empirical evidence of the effectiveness of the neural network models. This integration of qualitative and quantitative aspects contributes to a more holistic and robust examination of Amharic clickbait detection, enhancing our understanding and ability to combat clickbait in this specific linguistic context.

3.3. Dataset

Machine learning models like neural networks heavily depend on data, which they would use to turn a dataset into a model. NLP research needs natural language data to process and build computational linguistic models. Existing studies on non-English resource-poor languages focus on dataset creation from various sources to build the clickbait detection model. Amharic has a few pre-trained models and datasets. Thus, building the dataset is one of the primary tasks for this study.

3.3.1. Dataset Collection

The process of building the dataset would require crawling various web platforms. This is achieved with a set of tools such as web crawlers and scrapers. The data is gathered from web sources. To obtain the relevant text data for the work, the primary source is social media where people create an immense amount of text data and share it on the Internet. Social media provides relevant text data for the study where Amharic content is dispersed on a massive scale.

The creation of the dataset for clickbait detection involved the following procedures:

- Collecting Amharic tweets, titles, and posts from a social media through API calls or scraping, includes Twitter tweets, YouTube video titles, and Facebook posts.
- Cleaning each record to minimize the adverse effects of impurities on the model.
- Annotating the data and consolidating it into a dataset file.

3.3.2. Data Source

Platform Selection

When selecting social media platforms for clickbait detection research in Ethiopia, several considerations were taken into account. By selecting these major social media platforms, the research aims to obtain a comprehensive dataset that represents different forms of clickbait prevalent in Amharic language content and offers a deeper understanding of the clickbait landscape in Ethiopia. The selection of these major social media platforms as data sources is rationalized by their extensive usage and influence in the Ethiopian digital landscape.

- ***Facebook:*** Facebook holds a prominent position as the most popular social media platforms in Ethiopia. As of January 2023, Ethiopia had approximately 6 million Facebook users, which accounts for around 5 percent of the total population. Facebook's popularity can be attributed to its wide range of features, including sharing posts, news articles, and multimedia content, making it an ideal ground for clickbait in the Amharic language. Digital 2023: Ethiopia[8] presents that the potential ad reach on Facebook saw a slight increase between October 2022 and April 2023. Primarily the Facebook posts by the target users are an input to the dataset, commonly accompanied by a URL, accessed via Facebook's Graph[9] API.

[8] https://datareportal.com/reports/digital-2023-ethiopia
[9] https://developers.facebook.com/docs/graph-api/

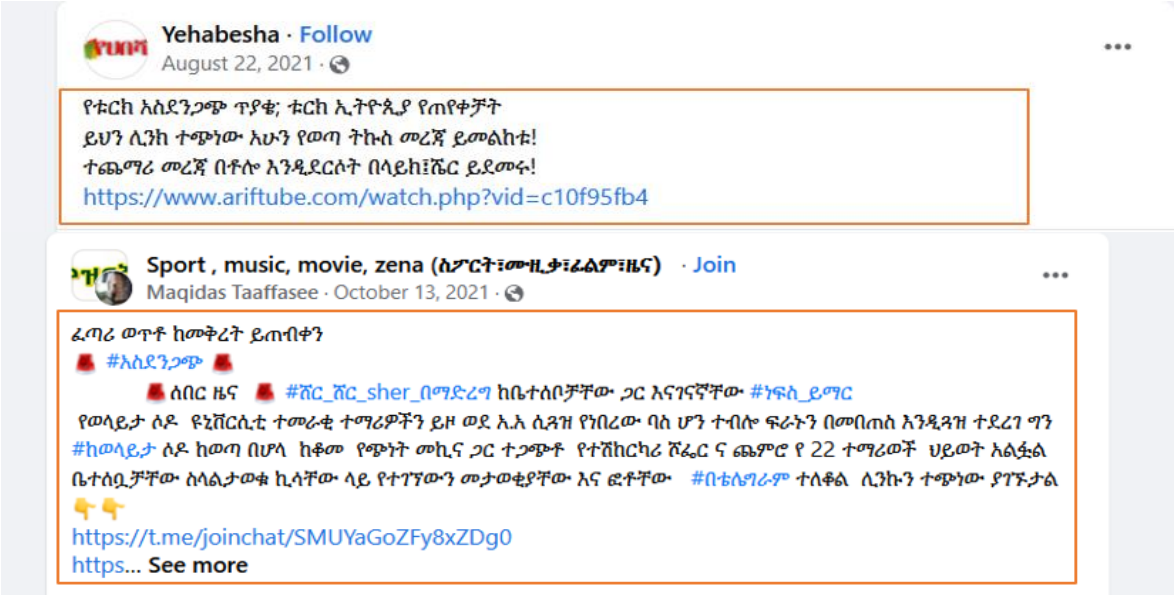


Figure 3.2 Clickbait Samples from Facebook Page and Group

- ***Twitter:*** Twitter also maintains a substantial user base of nearly 100 thousand in Ethiopia. While specific statistics may vary, Twitter serves as a platform for sharing short-form content, including tweets, hashtags, and links. As far as the data gathering is concerned, it targets Amharic tweets. Its real-time nature and brevity make it an important source for capturing timely and concise information. The data from Twitter can provide valuable insights into clickbait content and trends in Amharic. Tweets from the target accounts is considered for the dataset, commonly with a web link attached, accessed via Twitter API[10] v2.

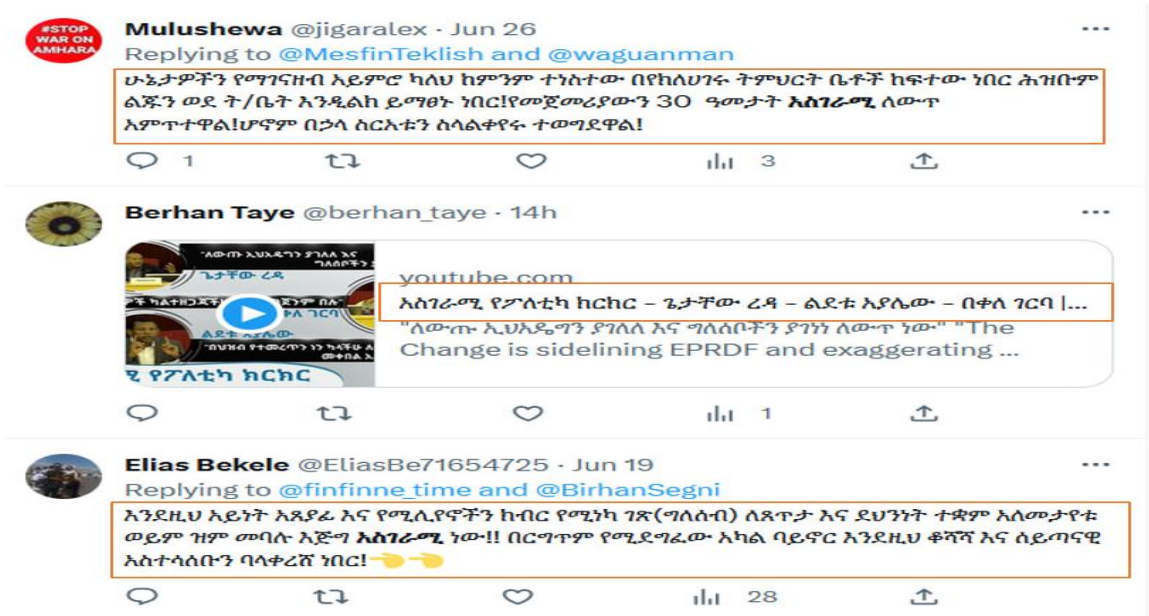


Figure 3.3 Clickbait Samples from Twitter

[10] https://developer.twitter.com/en/docs/twitter-api

- ***YouTube:*** YouTube, as a video-sharing platform, has gained significant traction in Ethiopia, with many users accessing and consuming video content regularly. With over 66.80 million cellular mobile connections active in the country, YouTube serves as a vital medium for disseminating clickbait content in Amharic. Analyzing YouTube video titles can provide valuable data for clickbait detection research. It was accessed via YouTube's Data[11] API.

Figure 3.4 Clickbait Samples from YouTube

Publisher Selection

When it comes to selecting publishers from the platforms discussed above, engagement metrics are a critical factor. Publishers with a large number of followers or subscribers and high engagement rates such as likes, comments, replies, and shares are more likely to generate clickbait content or authentic content. On the other hand, publishers with a smaller following and a lower engagement rate may be more likely to produce legitimate content. Therefore, it is crucial to take a holistic approach to publisher selection, considering factors such as topics and themes, frequency of posting, and overall credibility and reputation.

The study applied both purposive and systematic sampling methods to collect a representative dataset. Purposive sampling was utilized to select platforms and publishers known for their prevalence in clickbait content, ensuring a diverse range of sources and engagement patterns. Systematic sampling was then implemented to systematically select

[11] https://developers.google.com/youtube/v3

posts at predetermined intervals and genre, avoiding bias and providing a comprehensive representation of clickbait content across different time periods and sources.

During the data collection process for clickbait, distinguishing genuine public pages from clickbaiting ones poses a specific challenge. To streamline the analysis of extensive social media data for potential clickbait content, specific criteria were employed to select pages for inclusion in the dataset. These criteria include:

- *Language*: Posts written in the Amharic language were targeted to align with the focus of the study.
- *Timeframe*: To ensure a representative dataset, posts were sampled across different time periods from 2018 to 2023.
- *Flagged Pages*: Pages flagged as clickbait by reputable fact-checkers, specifically EthiopiaCheck[12], an organization that works on media literacy, fact-checking and training activities focused on Ethiopian media.
- *Redirecting Links*: Pages containing redirecting links to flashy websites or YouTube channels were selected.
- *Follower Count and Content Volume*: We selected pages with followers exceeding 1000 and accounts with a minimum of 100 published contents (tweets, posts, titles).
- *Content Focus*: Pages primarily focusing on trending topics, entertainment, current political issues, and advertisements were included.

The timeframe chosen for data collection, spanning from 2018 to 2023, was justified to ensure a representative dataset for the. The selected timeframe allowed for the inclusion of posts and content across different years, capturing the evolving nature of clickbait strategies and language patterns over time as well as account for the growing Internet and social media userbase. Additionally, the choice of a multi-year timeframe was influenced by previous events in Ethiopia (Ayele et al., 2022), such as the 'June' events, which had significant implications for social and political turmoil and dynamics in the country (5Js, 2018-2022). By considering posts from different years, the study aimed to account for temporal variations and provide a comprehensive understanding of clickbait in the Amharic language context. By utilizing these criteria, the research endeavors to enhance the identification of potential sources of clickbait. Sources are categorized as reputable or suspicious based on their susceptibility to share legitimate or clickbait content.

[12] EthiopiaCheck - https://ethiopiacheck.org/

Potential Publishers of Legitimate Content

The process of collecting data from authentic and legitimate sources involved selecting mainstream media outlets and trusted social media pages usually presenting a verified badge and official corroboration with the organization. These organizations include Ethiopian Broadcasting Corporation (EBC), Walta TV, VOA Amharic, Reporter News, BBC News Amharic, DW Amharic, and FBC (Fana Broadcasting Corporate).

These sources were chosen based on their reputation and credibility in providing reliable content. The content from these sources was reviewed and backed by EthiopiaCheck as a verified legitimate and authentic sources of quality content with proper standards.

Potential Publishers of Clickbait Content

The process of collecting data from Amharic clickbait sources can be challenging due to the diverse forms that clickbait can take. Clickbait texts are the primary target for the data collection, but they might be presented as image ads, embedded in the text, or in any other form. Clickbait comes in various forms, each with its own characteristics and intentions.

1. *False Context*: Clickbait with false context presents information out of its original context or distorts facts to attract attention and generate clicks.
2. *Satire:* Satirical clickbait uses humor and irony to criticize or mock individuals, events, or society. It often employs exaggerated elements to grasp attention.
3. *Parody:* Parodic clickbait imitates the style or content of genuine news articles to entertain or ridicule. It intentionally mimics reputable sources but with altered information or satirical twists.
4. *Fabrication:* Fabricated clickbait includes completely fabricated stories or events that are presented as factual. It aims to deceive readers by inventing stories to generate engagement and clicks.
5. *Propaganda:* Clickbait with a propagandistic agenda spreads biased information or ideologies to manipulate public opinion. It often uses emotional language, sensationalism, or conspiracy theories to persuade readers.
6. *Advertisement:* Clickbait in the form of advertisements aims to promote products, services, or events. It uses catchy headlines, captivating visuals, and enticing promises to attract clicks and drive traffic to specific sites.

Each of these clickbait forms employs specific tactics and strategies to entice users into clicking on the content. Understanding these different forms was crucial for effectively collecting data from Amharic clickbait sources as it helps researchers identify and categorize the various techniques used to manipulate and engage users.

Table 3.1 Potential Sources of Amharic Clickbait Content

Target Name	**Platform**	**Target Name**	**Platform**
Mereja today	YouTube	Tewnet	Facebook
ዓባይ ዜና - Abay News	YouTube	Yegna Tube̎ የኛ ቲዩብ	Facebook
Tikus Neger	YouTube	Tobia Tube - ጦቢያ ቲዩብ	Facebook
Berbir Mereja	YouTube	Zena24	Facebook
ብሩህ እይታ / Beruh Eyita	YouTube	BreakingNewsEthiopia	Facebook
Addis Kememoch	YouTube	Ethiopia24h	Facebook
Dallol Entertainment	YouTube	Sheger Media	Twitter
Axum Tube / አክሱም ቲዩብ	YouTube	Asaye Derbie	Twitter
Kalu media-ቃሉ ሚዲያ	YouTube	Zehabesha	Twitter
Inspire Ethiopia	YouTube	Adonis Lo	Twitter
Ethio Mereja	YouTube	Habtamutm	Twitter
Feta Link ፈታ ሊንክ	YouTube	Maleda	Twitter
Mehal Media	YouTube	Abrar Suleiman	Twitter

The data collection process presented various challenges and biases. These includes selection bias, as the manual selection of publishers and platforms based on reputation and past activity may introduce biases towards specific types of clickbait or non-clickbait content. Language bias is also a factor, limiting the generalizability of findings to other languages and cultures. Reputational bias arises from relying on fact-checkers and historical records, which may have their own biases and limitations in identifying clickbait. Platform limitations contribute to challenges in capturing a comprehensive sample of clickbait content across platforms. Additionally, annotation bias can arise during the labeling process, leading to inconsistencies and subjective judgments.

3.2.3. Dataset Preparation

The data collection process resulted in a total raw dataset instance of 53,227 gathered from the mentioned platforms. The statistics of the data instances across platform and clickbait nature is given in the following Table 3.2.

Table 3.2 Statistics of the Dataset

Platform	Clickbait	Non-clickbait	Total
YouTube	14,378	11,061	25,439
Twitter	7,136	9,432	16,568
Facebook	5,164	6,056	11,220
Total	26,678	26,549	**53,227**

Preparing the collected raw Amharic clickbait dataset involves several important steps to ensure its quality and usability for analysis. Firstly, the dataset needs to undergo a cleaning process to remove any irrelevant information. This includes eliminating irrelevant features that do not align with the objective of clickbait analysis.

Formatting is another crucial aspect of dataset preparation. It involves standardizing the structure and organization of the data to make it consistent and easy to work with. This is done by converting the dataset into a uniform file format, such as CSV or JSON, and ensuring that each data entry follows a consistent structure, including relevant metadata such as timestamps, sources, and labels.

Dealing with duplicates is also essential to maintaining the integrity of the dataset. Duplicate entries can skew the analysis results and introduce biases. Therefore, it is important to identify and remove any duplicate posts or articles, ensuring that each instance is unique and representative of distinct clickbait content.

One significant challenge in an Amharic clickbait dataset is the presence of non-Amharic text. Given that Amharic is the primary language of interest, any non-Amharic text needs to be addressed. This can involve removing non-Amharic entries entirely or in some rare cases translating them into Amharic to maintain consistency within the dataset.

Overall, the dataset preparation process for an Amharic clickbait dataset involves cleaning the data such as removing missing fields, standardizing its format, removing duplicate fields, addressing the non-Amharic text.

By following these steps, researchers can ensure the dataset's quality and suitability for clickbait analysis in the Amharic language. After these steps, the records from various sources are consolidated into an organized structures folder and file for annotation.

Table 3.3 Examples of Amharic Clickbait with English Translations

No.	Amharic Clickbait Sample Statements with English Translations
1	ከሽንብራ በጣም የሚያስገርም ጥቅም በቀላል ወጪ ፊቶት ላይ ላለዉ ችግር መፍትሄ አያምልጦት!
	Don't miss out on the amazing benefits of chickpeas for the problems on your face at a simple cost.
2	በየቀኑ ክሊክ የሚደረጉ 95 ስራዎች አሉት። እያንዳንዳቸው 67.00 ብር ያስገኛሉ።
	It has 95 jobs to be clicked every day. They reward 67.00 Birr each.
3	የኢትዮጵያ አንድ ብር 2,000 ብር እየተሸጠ ይገኛል። ሳንቲሙ እጃችሁ ላይ የሚገኝ አሁኑኑ ሽጡ። አድራሻ ሊንኩ ላይ አለ።
	One Ethiopian Birr is being sold at 2,000 Birr. Sell the coin now if you have it. Address is at the link.
4	ልታዩት የሚገባ ምርጥ የአማረኛ ፊልም እንዳያመልጦ
	Don't miss the best Amharic movie you should watch
5	አስደንጋጭ ዜና... ከአይሮፕላን ላይ ወድቆ ሞተ
	Shocking news... he died after falling from the plane
6	ስለ ፍሬህይወት ታምሩ አስገራሚና ያልተሰሙ እውነታዎች
	Surprising and unheard facts about Firehiwot Tamiru
7	ሰበር - ግብጽ ለየላት! ደ/ሱዳን ናይልን ልገድብ ነው አለች |ህውሃት የውጭ ዜጋ ገደለ |አውሮፕላኑ በመብረቅ ተመታ!
	Breaking – Egypt is done! South Sudan announced to dam Nile | TPLF killed a foreigner | The plane was struck by lightning!
8	የሚስቴን ጉድ ስራ ቦታ ድረስ ሄጀ አየሁ። አይታቹህ ፍረዱኝ።
	I went to my wife's workplace and saw the shame. Watch and judge for yourself.
9	በምድራችን ላይ ተደርገዋል ብላቹ የማታስቧቸው የአልባሳት ውድድሮች
	Clothing contests that you won't believe are held on Earth
10	ትምህርት ቤቶች ውስጥ ያለው ጉድ ለማመን አስቸጋሪ የሆኑ
	Hard to believe shameful acts in schools
11	ከጥቂት ወራት በኋላ ኢትዮጵያ ውስጥ በየቤቱ ግልፅ ጦርነት ሊጀመር ነው
	After a few months, an open war is about to start in every Ethiopian home

3.2.4. Data Annotation

There are certain guidelines to be followed by human annotators to label a given textual content as clickbait or non-clickbait. An Annotation Guideline and Inter-Annotator Agreement is prepared, which expresses features and qualities of clickbait which would help recognize one. The appeal of the statements is related to one's own personal feelings and curiosity. The Annotation guideline is presented on the Appendix A Section. The Inter-Annotator Agreement is provided under Appendix B section.

3.2.4.1.Annotation Strategy

The process of annotating sentences for Amharic clickbait detection is a subjective task. In the guidelines provided to the annotators, we have defined clickbait headlines' attractiveness as the ability to generate interest, pique curiosity with suspenseful language, and motivate users to click on the link. Furthermore, the appeal of the sentences is influenced by individual interests and curiosity.

Some of the consideration basis for the annotation task:

- *Content analysis:* This method looks at the text to classify as clickbait or non-clickbait to determine the speech's main semantic components and targets. To find more general patterns and trends, these components can then be categorized and subjected to quantitative approaches.
- *Discourse analysis:* This method places the text within its broader political and socio-economic context to gain an understanding of the underlying thoughts and justifications that determine whether it should be considered clickbait or not.
- *Automated approach:* This approach involves leveraging automated methods to analyze large volumes of text from various sources using systematic means. We followed certain rules to automatically label instances specifically for non-clickbait based on the source's reputation and reported cases.

The level of agreement between annotators is computed using the kappa (κ) statistic. A κ value less than 0 indicates agreement less than chance, while a value between 0.01 and 0.20 signifies slight agreement. Fair agreement falls within the range of 0.21 to 0.40, moderate agreement between 0.41 and 0.60, substantial agreement between 0.61 and 0.80, and almost perfect agreement between 0.81 and 0.99. The specific interpretation of what constitutes "good" agreement may vary, but a commonly cited scale suggests that a κ value of 0.57 falls within the moderate agreement range, indicating a higher level of agreement. It is important to note that a κ value of 1 represents perfect agreement, while 0 indicates chance agreement (Li et al., 2015).

To conduct the annotation, we employed three annotators, two expert news aggregators and one postgraduate linguist, where they worked on a smaller dataset for verification purposes. The Fleiss's kappa score, a measure of inter-annotator agreement, was determined to be 0.94. Subsequently, the annotation process was carried out on a larger dataset, resulting in inter-annotator agreement score of 0.89.

3.2.4.2.Annotation Tools

The process of annotation is a challenging and laborious task in the development of machine learning components. In countries where technology utilization is limited, manual labor is often employed for conducting surveys or annotations, typically using printed. However, for machine learning approaches, these questionnaires need to be digitized, which can be a labor-intensive process and may introduce encoding errors (Yimam et al., 2020).

This study explored some options for the annotation tools to use, that include using a spreadsheet in a tabular format to label, and a social media-based annotation tool using Telegram chatbot.

	A	B	C
1	text	clickbait	
2	ለማመን የሚከብዱት የበፊቷ ሶቪየት የአሁኗ ሩሲያ የጦር መሣሪያ	1	
3	ማንም በእውን አለም ይኖራሉ ብሎ የማይገምታቸው ድንቅ ቦታዎች	1	
4	ምድር ላይ በሰዎች ስህተት የተፈጠሩ ውድ ኪሳራዎች	1	
5	ስለ ኢትዮጵያ - ስለ ኢትዮጵያ የፓናል ውይይት በ አርባ ምንጭ Etv	0	
6	በርካቶች ለማየት ያልታደሏቸው አስገራሚ የምድር ክስተቶች	1	
7	በአንድ ጊዜ ባገኙት ግዙፍ ወርቅ የአለም ባለሀብት መሆን የቻሉ አድለኛ ሰዎች	1	
8	በካሜራ የተያዙ ለማመን የሚከብዱ ድንቅ ምስለሥ#ethiopia #አስገራሚ #denklejoch,ZenaAddis ዜናአዲስ	1	
9	በፈንጂ እንዲፈራርሱ የተደረጉት ለማመን የሚከብዱ ግዙፍ ህንፃዎች	1	
10	ብዙ ኢትዮጵያውያን ልጅነታቸውን ያሳለፉባቸው የህንድ ቅንጅ ተዋናዬች አሁን የት እና በምን አይነት ሁኔታ ይገኛሉ	1	
11	ነፀብራቅ - በሳምንቱ የነበሩ ዋና ዋና ጉዳዮች ዙሪያ የተደረገ ውይይት Etv	0	
12	ኢቲቪ. ወቅታዊ- ባለፉት አራት ዓመታት የተከናወኑ የለውጥ ሥራዎች Etv	0	
13	ኢትዮ ቢዝነስ በዚህ ሳምንት በጅትና ግሽን አስመልክቶ የሚቀርብ ዝግጅት ቅዳሜ ምሽት etv ዜና ይጠብቁን Etv	0	
14	ከ30 በላይ ሴት ባለሃብቶችን በሚሊዮን የሚቆጠሩ ብሮችን በማጭበርበር የተከሰሰው ደምሰው ዘሪሁን Etv	0	
15	ከፍተኛ ገንዘብ እየተከፈላቸው ፊልም ላይ የሚተውኑ ውድ ተከፋይ ተዋናዬች	1	
16	የልቦና ውቅር፡- ማህበራዊ ኢኮኖሚ Etv	0	
17	የቤተዘመድ ነገር . . . አጭር ድራማ Etv	0	
18	የትምህርት ገጽ የዩኒቨርሲቲ ማንባታ የወልቂጤ የጅማ የቦንጋ ዩኒቨርሲቲዎች Etv	0	
19	የኛ ጉዳይ - ሀገራዊ አካታች የምክክር ሂደት እና የሚዲያ ሚና Etv	0	
20	የአለማችን አስገራሚዎቹ ጫማዎች	1	
21	ዲፕሎማሲያችን - የተባበሩት መንግስታት ድርጅት እና የኢትዮጵያ ግንኙነት ክፍል - 2 Etv	0	
22	ጤናአዳም የያዘው የማይታመን ጥቅም እና ብዙዎች ያልጠበቁት ዘግናኝ ጉዳቱ	1	
23	ፈራርሰው ከተሞችን ያጠፉ ትልልቅ ግድቦች	1	
24	ፕሬዝዳንት ሳህለወርቅ ዘውዴ በአዲስ አበባ ዩኒቨርሲቲ የተማሪዎች የምረቃ ሥነ- ሥርዓት ላይ ያደረጉት ንግግር Etv	0	

Figure 3.5 Spreadsheet as an Annotation Tool

Telegram Bot[13] employs a client-server architecture implemented over its multi-platform applications, facilitating direct communication between users with Telegram accounts and the Telegram server. We have developed a Telegram chatbot that interacts with annotators through various endpoints. Annotators interact with buttons on the chatbot, Clickbait or Non-clickbait, where it will be populated in the working spreadsheet file as 1 or 0 respectively.

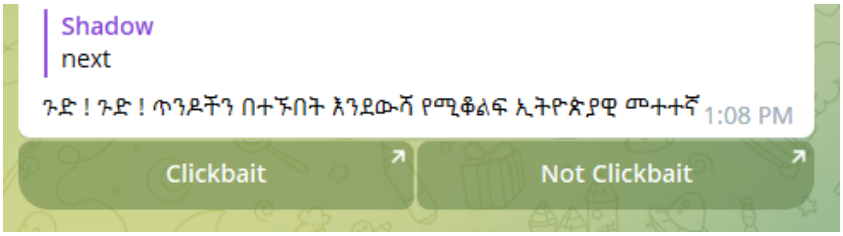


Figure 3.6 Telegram Bot as an Annotation Tool

[13] https://core.telegram.org/bots/api

3.2.5. Ethical Consideration

Ethical consideration for this study includes the ethical data collection, the privacy of participants such as annotators, and the accuracy of the results collected. It is also important to be aware of how research results may be used beyond the scope of academic research, such as when data is sold or used elsewhere than intended. Such considerations are addressed with those parties as part of informed consent before using the data.

All data collection steps like web scrapping are transparent and follow ethical considerations as such: a public API is used where available, data is passed through the user-agent string for identification, data is scraped at a reasonable and throttle-controlled rate, and data is never scraped from private sources. Also, it is essential that the data collected be pseudonymous so as not to reveal personal or organizational private information.

The effectiveness of clickbait detection measures must also be evaluated rigorously in order to ensure accuracy and reliability, which can only be done via extensive testing against a variety of potential scenarios.

Finally, it is essential for the researchers conducting the study to remain impartial and objective throughout all stages of their research to prevent bias from inching into the experiments or results.

We would like to emphasize that no conflict of interest exists in this research. All authors were committed to performing quality research without bias, and none of the authors or their related institutions have a financial stake in any clickbait detection technology. Therefore, all authors are independent and receive no benefit outside of intellectual stimulation from the proposed research herein.

3.3. Development Tools

3.3.1. Hardware Tools

A personal computer was used for the purposes of this study, where the experiments took place. The specifications of the machine are AMD Ryzen7-5700G CPU with NVIDIA GeForce RTX 3060 6GB dedicated graphics RAM of GPU, running 3.30 GHz Processor and 8.00 GB of RAM on x64-bit Windows 11 Pro Operating System.

3.3.2. Software Tools

For this research study, several types of software applications are used to design and implement the proposed thesis work's experiments. These tools include different software tools/platforms, modeling packages, and libraries. Table 3.3 briefly discusses these tools.

Table 3.4 Software Tools

No.	Software Tool	Version	Description
Modelling Language and Libraries			
1	**Python**[14]	**3.7/3.10**	Python is a versatile and widely used programming language in the machine learning and NLP communities. It offers a rich libraries and frameworks for data manipulation, scientific computing, and machine learning.
2	**Keras**[15]	**2.12.0**	Keras is a high-level neural networks library written in Python. It provides interface for designing, training, and evaluating deep learning models. Keras abstracts away low-level implementation details and simplifies the process of building complex neural networks.
3	**TensorFlow**[16]	**2.12.0**	Google's TensorFlow is an open-source deep learning framework. It offers a complete ecosystem for developing and deploying machine learning models. TensorFlow is compatible with both high-level APIs.
4	**NLTK**[17] **(Natural Language Toolkit)**	**3.8.1**	NLTK is a popular library for natural language processing in Python. It provides a wide range of functionalities and tools for tasks such as tokenization, stemming, part-of-speech tagging, parsing, and sentiment analysis.

[14] https://www.python.org/
[15] https://keras.io/
[16] https://www.tensorflow.org/
[17] https://www.nltk.org/

5	**cuDNN (CUDA Deep Neural Network library)[18]**	**8.9.1**	cuDNN is a GPU-accelerated library developed by NVIDIA. It provides highly optimized implementations of deep neural network primitives, such as convolutional and recurrent layers, for efficient computation on NVIDIA GPUs.
6	**NumPy[19]**	**1.24.2**	NumPy is a Python numerical computation package. It includes high-performance multidimensional array objects as well as a set of functions for data manipulation and calculation.
7	**scikit-learn[20]**	**1.2.2**	scikit-learn is a popular machine learning library in Python that provides tools and algorithms for data mining, analysis, and modeling.
8	**pandas[21]**	**2.0.0**	pandas is a versatile library for data manipulation and analysis in Python. It provides data structures and functions for handling structured data, such as tabular data, time series, and more.
Environment and Package Managers			
9	**Anaconda Navigator[22]**	**2.3.1**	Anaconda Navigator is a graphical user interface (GUI) that comes with the Anaconda distribution. It allows users to manage environments, install packages, and launch applications for data science and machine learning.
10	**Jupyter[23]**	**6.5.4**	Jupyter is a web-based interactive computing environment that supports various programming languages. It allows users to create and share documents called notebooks that contain live code, equations, visualizations, and text.
11	**Google Colab[24]**		Colab is a cloud-based notebook environment provided by Google. It allows users to write and execute Python code in a browser without the need for local installation. Google Colab provides free access to GPU and TPU resources.
Data collection and processing tools			
12	**Facepager[25]**	**4.5.0**	Facepager is a tool used for data scraping and data collection from various online sources, and social media platforms like Twitter, Facebook, and YouTube.

[18] https://developer.nvidia.com/cudnn
[19] https://numpy.org/
[20] https://scikit-learn.org/
[21] https://pandas.pydata.org/
[22] https://docs.anaconda.com/free/navigator/index.html
[23] https://jupyter.org/
[24] https://colab.research.google.com/
[25] https://github.com/strohne/Facepager

13	**Apify**[26]		Apify is a powerful tool that facilitates the collection and scraping of data for different purposes. It provides a user-friendly and efficient platform for automating the extraction of data from various sources, including websites, APIs, and other online platforms.
14	**ModernCSV**[27]		ModernCSV is a library or tool that provides efficient and streamlined processing of CSV files. It offers functionalities for handling large datasets, manipulating data, and performing operations like filtering and aggregation,
NLP Tools and Models for Amharic Language			
15	***Amharic-Word-Embedding-Word2Vec***[28]		It is a pre-trained distributed word representation model designed to support Amharic natural language processing researchers. It offers freely available resources for training custom word embeddings using users' own datasets and computing similarity between words or phrases.
16	***fastText***[29]		It is developed by Facebook AI Research, extends the concept of word embeddings to subword units called character n-grams. fastText-Amharic-embedding-Vectors[30] model introduced by (Eshetu et al., 2020) extends the concept of word embeddings word vectors for both CBOW and skip-gram model.
17	***HornMorpho***[31]		HornMorpho is a Python program designed to perform morphology analysis and word generation in three languages spoken in the Horn of Africa: Amharic, Oromo, and Tigrinya. By breaking down words into their constituent morphemes and generating new words based on root or stem inputs and grammatical structures, it enables linguistic analysis and language processing.

[26] https://apify.com/
[27] https://www.moderncsv.com/
[28] https://github.com/gashawdemlew/Amharic-Word-Embedding-Word2vec
[29] https://fasttext.cc/docs/en/crawl-vectors.html
[30] https://github.com/Abe2G/FastText-Amharic-Embedding-Vectors
[31] https://github.com/hltdi/HornMorpho

3.4. Evaluation Model

Evaluation for clickbait detection and overall NLP study is very important for accurately assessing the efficacy and performance of any given model or algorithm. To evaluate the model's quality and ensure that it is operating optimally, evaluation metrics are utilized. Two commonly used approaches for model evaluation are holdout and k-fold cross-validation.

3.4.1. k-Fold Cross-Validation

k-fold cross-validation is a more robust evaluation technique that overcomes the limitations of holdout validation. In k-fold cross-validation discussed by (Cao et al., 2017), the dataset is divided into k equally sized folds. The model is trained and tested k times, each time with a different fold serving as the testing set and the remaining folds serving as the training set. This process ensures that each data point is used for both training and testing. The performance metrics are then averaged over the k iterations to provide a more reliable estimate of the model's performance. The most common choices for k are 5 and 10, however they might vary depending on the size of the dataset and processing resources. The use of k-fold cross-validation results in a more stable and representative evaluation of the model's performance, reducing the impact of data variability and offering a more realistic estimate of its generalization capacity. A general procedure for k-fold cross-validation followed in this study is as follows:

1. Split the dataset: Divide the dataset into k equally sized folds. Each fold should contain a representative distribution of the data.
2. Initialize performance metrics: Set up variables to store the performance metrics you are interested in evaluating, such as precision, recall, accuracy or F1 score.
3. Iterate over the folds: For each fold in the dataset:
    a. Use the current fold as testing set.
    b. Use the remaining folds for training.
    c. On the training set, train the clickbait detection model.
    d. Evaluate the model: Apply the trained model and calculate the metrics.
    e. Update the overall performance metrics: Aggregate the performance metrics.
4. Calculate average performance
5. Interpret the results: Analyze the performance metrics obtained to assess the model's performance and generalization capabilities.

3.4.2. Classification Metrics

To give a qualitative analysis and a quantitative metric for evaluating text classification tasks such as clickbait detection models, there are various evaluation metrics. The commonly used metric to measure effectiveness is accuracy, which measures how often the classifier predicts the correct label (e.g., whether a given text content is clickbait). Other evaluation metrics include precision, recall, F1-score, and confusion matrix.

3.4.2.1.Confusion Matrix

Confusion Matrix is a widely used classification metric in clickbait detection and other NLP tasks. It provides a tabular representation of the performance of a classification model, allowing us to analyze the model's predictions and assess its accuracy. The matrix is divided into four categories: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).

- *True Positive (TP):* This indicates the number of clickbait instances correctly classified as clickbait by the model.
- *True Negative (TN):* This indicates the number of non-clickbait instances correctly classified as non-clickbait by the model.
- *False Positive (FP):* This indicates the number of non-clickbait instances incorrectly classified as clickbait by the model. These are the instances where the model predicted clickbait, but the actual label was non-clickbait.
- *False Negative (FN):* This indicates the number of clickbait instances incorrectly classified as non-clickbait by the model. These are the instances where the model predicted non-clickbait, but the actual label was clickbait.

Table 3.5 Confusion Matrix

		Predicted	
		Non-Clickbait	**Clickbait**
Actual	**Non-Clickbait**	TN	FP
	Clickbait	FN	TP

3.4.2.2.Accuracy

Accuracy is a classification metric that measures the general correctness of a model's estimates. Accuracy is calculated by dividing the number of correctly classified instances (both True Positives and True Negatives) by the total number of instances in the dataset. However, accuracy alone may not provide a whole view of the model's performance, particularly in imbalanced datasets where one class may dominate the other. Therefore, when evaluating clickbait detection models, it is recommended to consider additional metrics such as precision, recall, and F1 score, which consider the different types of classification errors (Cao et al., 2017).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})} \tag{3.1}$$

3.4.2.3.Precision

Precision is a classification metric that calculates the percentage of accurately predicted positive cases (True Positives) out of all positive examples anticipated. It quantifies the accuracy of positive predictions made by a model. A higher precision indicates that the model has a lower rate of falsely classifying non-clickbait instances as clickbait.

$$Precision = \frac{TP}{(TP + FP)} \tag{3.2}$$

3.4.2.4.Recall

Recall or sensitivity is a classification metric that measures the proportion of correctly predicted positive instances (True Positives) out of all actual positive instances. It quantifies the ability of a model to identify positive instances correctly. A higher recall indicates that the model has a lower rate of falsely classifying clickbait instances as non-clickbait.

$$Recall = \frac{TP}{(TP + FN)} \tag{3.3}$$

3.4.2.5.F1-score

F1-score is a classification metric that combines both precision and recall into a single value, providing a balanced evaluation of a model's performance. It is particularly suitable when an imbalance is observed between the number of positive and negative instances.

$$F1 - score = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)} \tag{3.4}$$

CHAPTER FOUR

4. PROPOSED AMHARIC CLICKBAIT DETECTION MODEL

In this chapter, we delve into the proposed model, the experimental environment, and the detailed implementation of the clickbait detection model. This chapter provides a comprehensive overview of the model architecture, the tools employed, an exploratory data analysis finding, the experimental setup, and the step-by-step implementation process, highlighting the key elements necessary for successfully detecting clickbait in Amharic text.

4.1. High-Level Architecture of the Model

The following illustration in Figure 4.1 presents the proposed high-level architecture of the proposed Amharic clickbait detection model, with general procedures carried out in the design and implementation of the model.

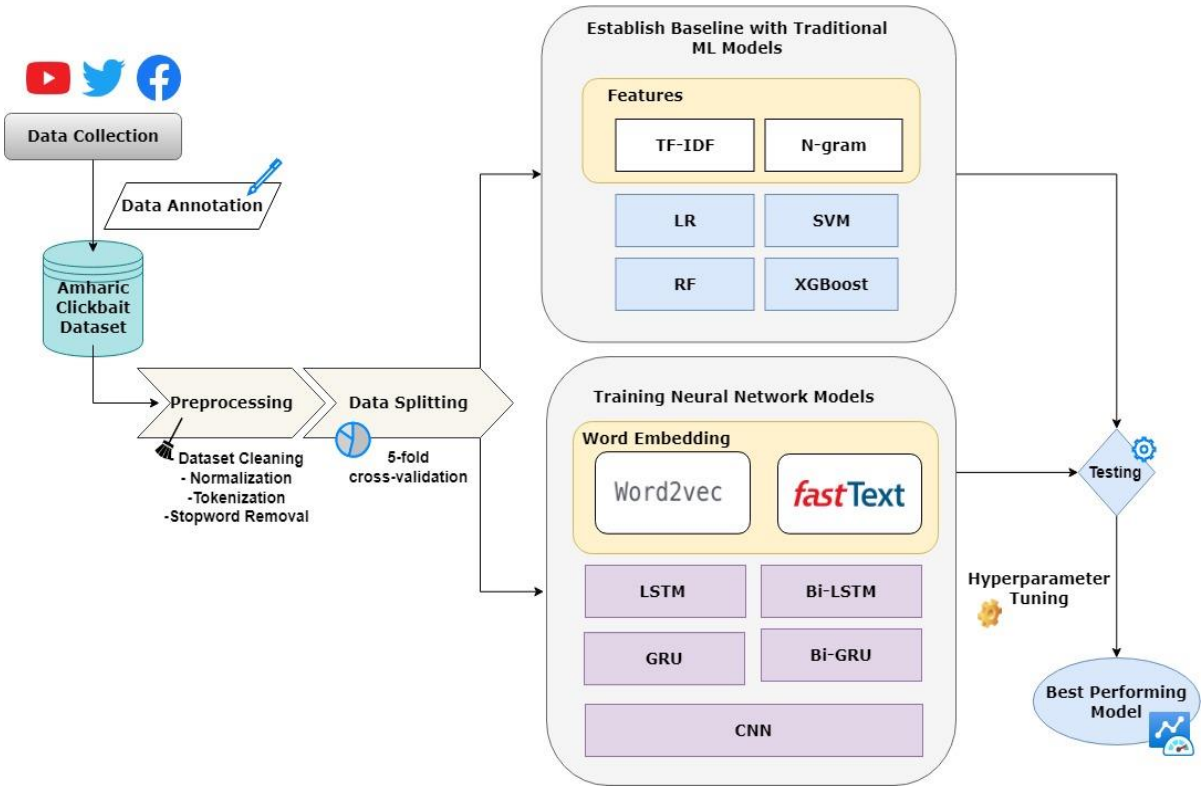


Figure 4.1 Architecture of the Amharic Clickbait Detection Model

4.2. Proposed Model

4.2.1. Problem Definition

The problem of Clickbait Detection can be framed as a binary text classification task. We define the task as follows: Let *C* be the set of two classes *{clickbait, non-clickbait}*, where one class represents clickbait sentences and the other represents non-clickbait sentences. Given a dataset *D* consisting of sentences, our objective is to train a model that can classify each sentence into one of the two classes. The training set *t* consists of labeled sentences *{t,C}*, where each sentence is associated with the corresponding class label $\{t,C\} \in D \times C$. The goal is to learn a function f that can effectively map the sentences in the dataset to their respective classes $f : D \rightarrow C$, enabling accurate classification of clickbait and non-clickbait sentences.

4.2.2. Model Selection

This study experimented with Logistic Regression, Support Vector Machine Random Forest and XGBoost, with TF-IDF and n-gram as features to establish baseline.

a) Logistic Regression (LR)

Logistic Regression is an algorithm applies a nonlinear transformation to a linear combination of weights and features. It is a technique that uses predictive analysis; the likelihood or probability of an outcome based on observable traits; to decide whether the content is clickbait or not. Several studies have shown the effectiveness of LR in clickbait detection. (Cao et al., 2017) used Logistic Regression and reported good results, where (Adelson et al., 2017) utilized LR with TF-IDF and n-gram features to establish baseline and they achieved competitive results.

b) Support Vector Machines (SVMs)

SVMs categorize the points in a dataset. SVMs attempt to separate the data based on their features and use this separation to make judgements rather than attempting to estimate a function. It is a technique based on a classification algorithm that can be used to classify clickbait by detecting text features, such as word and punctuation usage. The model produces a hyperplane to separate data points into two or more classes and classifies new data points by using the hyperplane's orientation. Previous studies, such as works by (Chakraborty et al., 2016; Geçkil et al., 2018) have demonstrated the effectiveness of SVM in clickbait detection.

c) Random Forest (RF)

Random Forest leverages ensemble learning techniques and the strength of decision trees to effectively classify clickbait and non-clickbait instances. Its ability to handle noisy data and capture non-linear relationships makes it a suitable candidate for our research. RF models have shown promise in clickbait detection, as highlighted in the study by (Potthast et al., 2016; Shu et al., 2018) where they achieved the best results.

d) XGBoost

XGBoost (Extreme Gradient Boosting) is a machine learning algorithm that belongs to the group of gradient boosting algorithms. XGBoost operates by sequentially building a strong ensemble model consisting of multiple weak decision trees. This iterative process allows XGBoost to capture complex relationships and interactions within the data, leading to robust and accurate predictions. Study carried out by (Shu et al., 2018) demonstrates the efficacy of XGBoost and other Gradient Boosting algorithms like LightGBM (Marreddy et al., 2021) in accurately identifying clickbait content. It is particularly adept at handling unstructured text data by using techniques such as bag-of-words or TF-IDF representations.

The proposed model first creates the data corpus by collecting clickbait and non-clickbait texts. Then these textual contents are converted into word embeddings which finally serve as input to the different neural network models.

Mossie & Wang, 2018 utilized word2vec to detect Amharic hate speech in social networks and achieved sound results. Word2vec word-level embedding creates a 300-dimensional vector using a continuous bag of words architecture. Two arrays would be required for the indices of the words: word2vec (C) and word-to-index (R). For the embedding layer, a 2D array: $S = R$ x C

fastText model considers the bag of character n-grams to represent each word, it allows us to compute rare word representations. In a study by Gereme et al., (2021), the authors explored the use of fastText for Amharic fake news detection. They found that fastText outperformed traditional word embedding methods in capturing the semantics of Amharic words, even with limited training data.

The study also experimented with Recurrent Neural Networks like LSTM and GRU, and Convolutional Neural Networks with two different kinds of word embedding vectorization techniques. Word2Vec and FastText word-level embeddings are utilized in this case.

d) LSTM (Long Short-Term Memory Networks)

LSTMs are a type of recurrent neural network well suited for capturing long-term dependencies among chunks of text information contained within a single news article title or post description to distinguish between clickbaiting words among all other words present in the text. LSTMs are particularly helpful when dealing with highly visible topics like entertainment celebrities because they capture relationships across different types of contextual information contained within an individual title or post description quickly and accurately. In a study by Manjesh et al., (2017), LSTM models were employed for clickbait pattern detection in news headlines. Additionally, Dimpas et al., (2018) explored the use of Recurrent Neural Networks like Bi-LSTM for clickbait detection in Filipino and English language.

e) GRU (Gated Recurrent Units)

GRU is designed to address the vanishing gradient problem often encountered in traditional RNNs, allowing it to capture long-term dependencies in sequential data effectively. Unlike standard RNNs, GRU incorporates gating mechanisms that selectively update and reset information within the hidden state, making it more efficient and capable of capturing relevant contextual information. By analyzing the sequential nature of textual information, GRU can effectively capture important patterns and dependencies within the text, enabling accurate classification. We observed from the works of Klairith & Tanachutiwat, (2018); Dong et al., (2019) where they used GRU and BI-GRU models in clickbait detection and yielded reliable results.

f) Convolutional Neural Networks (CNN)

Multiple weight matrices are combined using a convolutional neural network (CNN), which then creates a new vectorized representation of the input. A fully connected linear layer is then applied to this new representation to perform classification or regression. This architecture, combined with the utilization of word embeddings and attention mechanisms, allows CNNs to effectively capture both local and global dependencies in text data as it was demonstrated with the works of (Fu et al., 2017; Zheng et al., 2018), leading to robust and accurate clickbait classification results. Recent work by Abebaw et al., (2022) demonstrated the effectiveness of CNN text classification approach with word2vec embedding for detecting Amharic hate speech, which lead us to select CNN model as the primary proposed model for Amharic clickbait detection.

4.3. Data Preprocessing

4.3.1. Stopword Removal

While it is crucial to perform preprocessing steps, it is important to exercise caution when applying all commonly used techniques, as they may have unintended consequences for this model. Take stopword removal, for instance. While it can be beneficial in document classification by allowing the recognition of the central idea even without stopwords, it is essential to consider the potential impact on performance. The use of stop word removal as a preprocessing step in clickbait detection experiments can be a topic of debate. Stop words are commonly occurring words in a language that are often removed to reduce noise and improve computational efficiency in NLP tasks (Awol & Gashaw, 2022).

In the context of clickbait detection, the removal of stop words may have both positive and negative implications. On one hand, removing stop words can potentially eliminate noise and improve the detection of important keywords or phrases that contribute to the clickbait nature of a text. This can help in capturing the essence of the clickbait content more effectively. On the other hand, removing stop words may also lead to the loss of some contextual information that could be relevant for clickbait detection. Stop words can provide important syntactic and semantic cues that contribute to the overall meaning and structure of a sentence. By removing them, there is a risk of losing some of the nuances and subtleties that are characteristic of clickbait.

It is advisable to conduct experiments with and without stopword removal to evaluate the impact on the detection performance and make an informed decision based on the results obtained, where in our case a number of irrelevant stopwords are removed. Other pre-processing procedures such as cleaning, normalization and tokenization is commenced.

4.3.2. Cleaning

Dataset cleaning is an essential preprocessing step in most NLP tasks. The cleaning process involves removing special characters, symbols, punctuation, emojis, and whitespace from the dataset. Non-Amharic text is also eliminated to ensure the dataset consists solely of Amharic content. Additionally, irrelevant numerals are removed, and long sentences are filtered out to maintain a fair document length. These cleaning steps help ensure that the dataset is properly formatted and contains relevant Amharic text, improving the accuracy and effectiveness of the subsequent steps.

Clickbait Dataset Cleaning Algorithm

Input: Text from the dataset
Output: Clean text
START:

1. WHILE (NOT the end of dataset file):
2. READ line:
   IF text has any special characters ['<', '@', '~', '^','$', '%','&', …] THEN:
   Remove special character
   IF text has any emoji [😂, 👏, ⭐, 🧒, 🎉,…] THEN:
   Remove emoji
   IF text has any punctuation marks [‘¤’,’፥’,’፣’,’፤’…] THEN:
   Remove punctuation mark
   IF text has extra whitespaces THEN:
   Trim the whitespace
   IF text contains more than 100 words THEN:
   Flag the text for manual processing
3. RETURN Clean text:

END

Algorithm 4.1 Clickbait Dataset Cleaning Algorithm

4.3.3. Normalization

Normalization is an important preprocessing step for datasets, as it standardizes the text by addressing different variations of writings into a constant form. The goal of normalization is to transform the text into a consistent and unified representation, reducing ambiguity and improving the accuracy of subsequent analysis.

To achieve normalization of Amharic words, Gambäck et al., (2009) discussed types of normalization issues in Amharic word tokens. The first issue involves identifying and replacing shorthand representations of words written with forward slashes ('/') or periods ('.'). For instance, "ም/ቤት" is replaced with "ምክር ቤት," and "ዓ.ም" is replaced with "ዓመተ ምህረት." The second normalization issue addresses the presence of homophone Fidels or spellings within the Amharic writing system. These Fidels have the same pronunciation but different symbols. Although they convey different connotations in Ge'ez (the parent language of Amharic), they have been used interchangeably in Amharic. Examples of such Fidels include አ and ዐ, ፀ and ጸ, ሰ and ሠ, ሀ, ሐ, and ኀ. For instance, the word "Artist" could be written as አርቲስት, ዐርቲስት, ዓርቲስት - all with the same pronunciation but different orthography. The other normalization issue to consider is of the delicate inflectional morphology Amharic exercises with prefixes and suffixes, leading to different formats of the word.

Clickbait Dataset Normalization Algorithm

Input: Unprocessed text from the dataset
Output: Normalized text
BEGIN:

1. WHILE (NOT the end of dataset file):
2. READ line:
   IF text has [ሃ, ኅ, ኃ, ሐ, ሓ, ኻ] THEN:
   Replace with 'ሀ'
   IF text has [ዓ, ኣ, ዐ] THEN:
   Replace with 'አ'
   IF text has [ሠ, ሡ, ሢ, ሣ, ሤ, ሥ, ሦ] THEN:
   Replace with [ሰ, ሱ, ሲ, ሳ, ሴ, ስ, ሶ]
   IF text has [ቱ-ዋአ], [ሩ-ዋአ] … THEN:
   Replace with [ቷ], [ሯ] …
   …
3. RETURN normalized text:

END

Algorithm 4.2 Clickbait Dataset Normalization Algorithm

4.3.4. Tokenization

Tokenization is another preprocessing step taken over the dataset, as it involves breaking down the text into individual tokens or units, typically words or subwords. Though Amharic formally uses punctuations to delimit words with other words, it is not commonly observed on the web. Thus, the given processed text is tokenized as $T = \{t1, t2, t3, \ldots tn\}$ for each word using space as a delimiter.

4.4. Feature Extraction

The process of turning raw text input into a numerical representation that machine learning algorithms can analyses is known as feature extraction, and it is crucial to NLP research. In the context of Amharic, where it is a computationally low-resource language and the script posing unique challenges, choosing appropriate feature extraction methods is key for capturing the semantic and syntactic information necessary for clickbait detection.

4.4.1. Word Embedding

One popular approach for feature extraction is word embedding, which represents words as dense vector representations in a continuous multi-dimensional space. Word embeddings capture semantic relationships between words based on their contextual usage, allowing algorithms to leverage this information for various NLP tasks, including clickbait detection.

Clickbait Sample **- ብዙዎችን ያስቆጣው አነጋጋሪው የሴቶች አሳፋሪ ቪድዮ ተለቀቀ**

Tokenization **- <ብዙዎችን><ያስቆጣው><አነጋጋሪው><የሴቶች><አሳፋሪ><ቪድዮ><ተለቀቀ>**

Stemming and Root word analysis with HornMorpho

word: ብዙዎችን
POS: noun stem: ብዙ
word: ያስቆጣው
POS: verb root: <qWT'> citation: አስቆጣ
word: አነጋጋሪው
POS: noun stem: አነጋገረ
word: የሴቶች
POS: noun stem: ሴት
word: አሳፋሪ
POS: adj stem: አሳፈረ
word: ቪድዮ
POS: noun stem: ቪድዮ
word: ተለቀቀ
POS: verb stem: ለቀቀ

Handling Out-of-Vocabulary Terms with Prefix, Affix and Suffix

<አነጋጋሪው> = <አ-ነጋጋሪ-ው> <አ-ነጋገረ-ው> <አ-ነገረ-ው>

Forming Embedding with n-gram character with fastText

ያስቆጣው: <ያስ-ያስቆ-ስቆጣ-ቆጣው-ጣው> **ተለቀቀ: <ተለ-ተለቀ-ለቀቀ-ቀቀ>**

fastText addresses this issue by generating embeddings for out-of-vocabulary words using their subword representations. Even if a specific word is unobserved during training, fastText can approximate its representation based on the shared subword units with other known words.

In practice, these word embeddings can be used as input features for clickbait detection models. The word embeddings can be either pre-trained on some large corpus or trained specifically on an Amharic dataset. The embeddings are then passed onto machine learning models, such as recurrent neural networks (RNNs) or convolutional neural networks (CNNs), which learn to classify clickbait based on the learned representations.

By leveraging word embeddings like Word2Vec and fastText, the clickbait detection models can capture the semantic meaning and contextual information in the language, improving the accuracy and effectiveness of the detection process.

Word2vec

Word2vec is a widely used word embedding technique that forms vector representations by training a shallow neural network on a large corpus of text. It generates dense word vectors where words with similar meanings or contexts are positioned closer to each other in the vector space. For Amharic language, word2Vec can be trained on a large Amharic text corpus to create meaningful word embeddings that capture the nuances of the language.

Word2Vec visualizes words as fixed-dimensional vectors that capture the semantic links between words depending on how frequently they occur together in a corpus. It operates at the word level and considers the context of words in a sentence.

fastText

fastText is another popular word embedding technique that extends Word2Vec by considering subword information. It represents words as a combination of character n-grams and their embeddings, which is particularly useful for morphologically rich languages like Amharic. FastText can capture the morphological variations and compound words present in Amharic, enhancing the performance of clickbait detection models.

fastText can handle rare and out-of-vocabulary words because it operates at the subword level. By leveraging character n-grams, it can generate representations for unseen words based on their subword components. fastText excels in morphologically rich languages, like Amharic, where words can have various forms and structures. By considering subword units, it captures the morphological nuances and can handle word variations effectively.

4.5. Environmental Setup

The hardware environment consisted of a high-performance computing system equipped with state-of-the-art processing capabilities and memory capacity to handle the computational requirements of the tasks. On the software front, a range of powerful tools and libraries were employed, including Python programming language. Previous sections 3.4 and Table 3.3 provided an overview of the hardware and software environment, ensuring a working environment for the subsequent implementation and evaluation of the Amharic clickbait detection model.

4.6. Experimental Setup

4.6.1. Initiating Experiment

The experiment began by importing the necessary libraries and frameworks required for the implementation of the Amharic clickbait detection model. Key libraries such as pandas, numpy, and scikit-learn were imported to handle data processing and manipulation tasks effectively. The Keras and TensorFlow libraries were employed for building and training deep learning models. Additionally, the nltk library was utilized for natural language processing tasks such as tokenization and stemming.

```
import pandas as pd
import string
import nltk
import keras
import sklearn
import time
from keras.models import Sequential
from keras.layers import Dense
import matplotlib.pyplot as plt
import numpy
import tensorflow as tf
import numpy as np
from time import time
from nltk.corpus import stopwords
from pandas import read_csv
from pandas.plotting import scatter_matrix
from matplotlib import pyplot
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import cross_val_score
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import StratifiedKFold
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from keras.callbacks import TensorBoard
from keras.preprocessing.text import Tokenizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Dense , Activation , Dropout
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelBinarizer
from sklearn.preprocessing import MultiLabelBinarizer
import keras.preprocessing.text
```

Figure 4.2 Implementation of Importing Libraries and Models

4.6.2. Loading Dataset

To load the dataset, the pandas' library was used to read two separate CSV files representing the clickbait and non-clickbait datasets. The CSV files were loaded into pandas dataframes, with the "text" column representing the textual content, and the "class" column indicating the label i.e., clickbait (1) or non-clickbait (0). By utilizing the read_csv function provided by pandas, the datasets were merged and efficiently loaded into memory, enabling easy access and manipulation of the data for further processing and analysis in the clickbait detection experiment.

```
data_clickbait = pd.read_csv ('clickbait-raw.csv' , sep = ',' , header = 0)
data_nonclickbait = pd.read_csv ('nonclickbait-raw.csv' , sep = ',' , header = 0)
data_clickbait['class'] = [1 for i in range(data_clickbait.shape[0])]
data_nonclickbait['class'] = [0 for i in range(data_nonclickbait.shape[0])]
data_final = pd.concat((data_clickbait , data_nonclickbait), ignore_index = True)
df = data_final.rename(columns = { 0 : "text" })
```

Figure 4.3 Implementation of Loading Dataset

4.6.3. Exploratory Data Analysis

As an exploratory data analysis phase, the clickbait dataset was examined before undergoing cleaning and preprocessing steps. Several features were extracted from the dataset to gain insights into the lexical nuances present in the data. These features included analyzing whether a title contained a question mark or an exclamation mark, determining the number of words in each headline, and identifying whether a title included numeric characters or not. By exploring these features, it was possible to uncover patterns and characteristics specific to clickbait headlines, providing a deeper understanding of the lexical composition of the dataset and potentially informing the subsequent preprocessing and modeling stages.

Figure 4.4 illustrates a notable disparity between clickbait and non-clickbait items where the number of interrogatory words and question mark are used. Clickbait titles tend to employ a higher number of questions, capitalizing on the human brain's inherent cognitive bias and evoking a sense of curiosity. Similarly, clickbait title exhibits a distinct sentence structure characterized by a greater number of words and a composition that incorporates an abundance hyperbolic language signified using the "!" exclamation mark, enticing readers with sensational, provoking, and alluring language. Both clickbait and non-clickbait items exhibit a similar presence of numbers, indicating that numerical information is found in both types of content to a comparable extent.

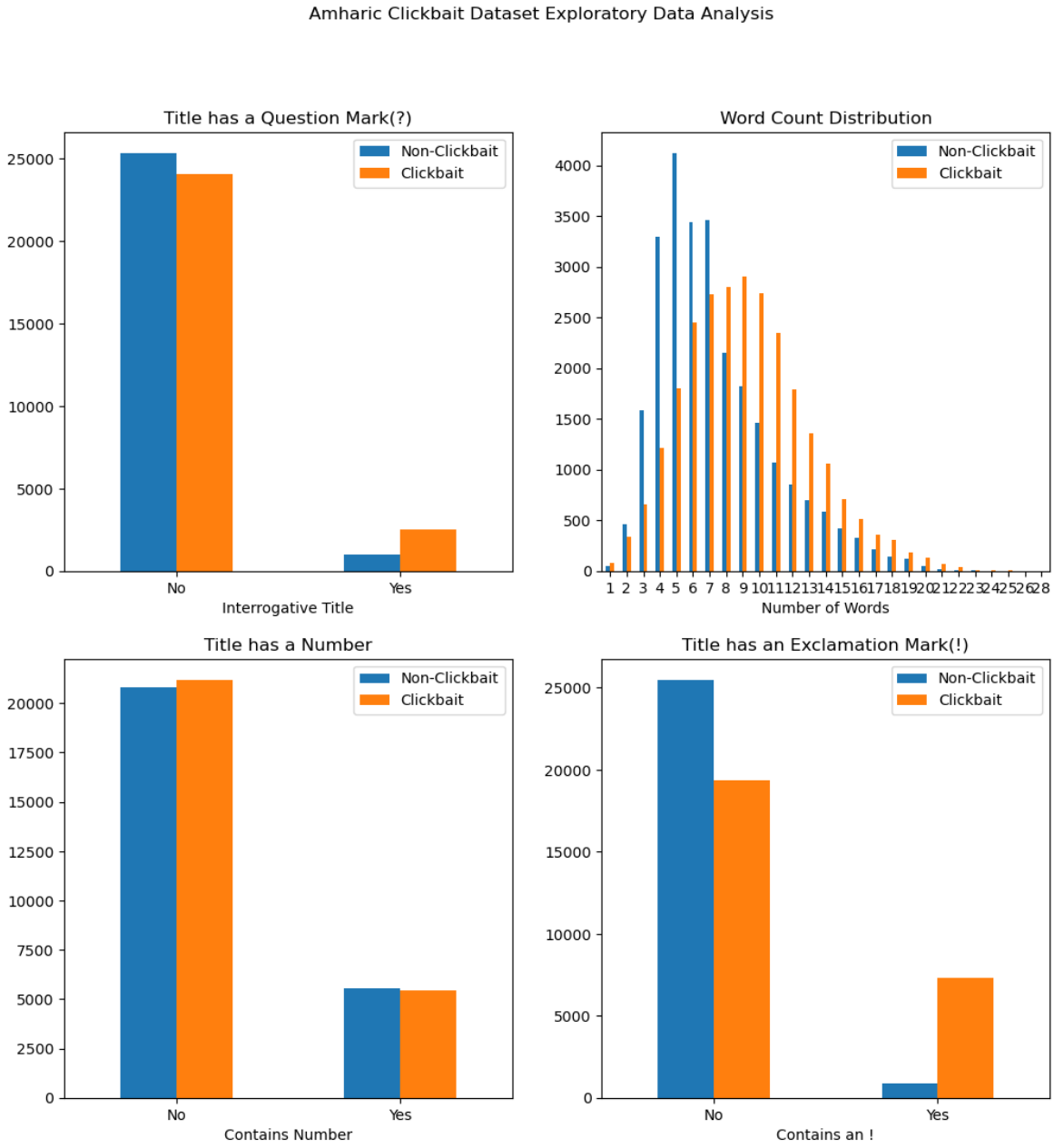


Figure 4.4 Exploratory Data Analysis of the Amharic Clickbait Dataset

The dataset used in this study contains a substantial number of emojis, which can be attributed to the nature of the content being derived from social media platforms. Emojis have gained significant popularity as a means of expressing emotions, sentiments, and reactions in online communication. Clickbait often aims to evoke strong emotional responses from readers to capture their attention. Emojis serve as a powerful tool in this regard, as they can convey a wide range of emotions in a concise and visually appealing manner. Emojis have become an essential part of modern communication, enabling individuals to convey their emotions and signify domain representations for such as sentiment.

4.6.4. Cleaning the Dataset

The dataset cleaning process involved the removal of special characters, punctuations, URL links, English alphanumeric characters, and Amharic numbers using regular expressions. These elements were considered noise in the dataset and could potentially interfere with the analysis and modeling of Amharic clickbait detection. By applying regular expressions, all occurrences of these elements were identified and eliminated, resulting in a dataset consisting of Amharic-only texts.

```
import re
def clean_text(text):
    '''remove special char, URLs, symbols, amh punctuation and numbers, english letters and numbers.'''
    text = re.sub('\w*\d\w*', ' ', text)
    text = re.sub('\n', ' ', str(text))
    text = re.sub(r'^https?:\/\/.*[\r\n]*', '', str(text), flags=re.MULTILINE)
    text = re.sub(r'\w+:\/{2}[\d\w-]+(\.[\d\w-]+)*(?:(?:\/[^\s/]*))*', ' ', str(text))
    text = re.sub('\[.*?\]', ' ', str(text))
    text = re.sub('[%s]' % re.escape(string.punctuation), ' ', str(text))
    text = re.sub(r'[^\w\s]',' ',str(text))
    text = re.sub('፨፡።፣፤፥፦፧፠“”‘’…‹›‹›·•"#$%^&@~`;◆-!*-',' ',str(text))
    text = re.sub(r'[a-zA-Z0-9]+','',str(text))
    text = re.sub('፩፪፫፬፭፮፯፰፱፲፳፴፵፶፷፸፹፺፻፼','',str(text))
    return text
```

Figure 4.5 Implementation of Cleaning Dataset

4.6.5. Normalizing the Data

To normalize the Amharic texts, two main techniques were applied: multi-word short form expansion and character level normalization. For short form representation, a list of short forms in the Amharic language was consulted to expand expressions from their short form to their long form. The character level normalization task focused on addressing the interchangeability of certain characters in Amharic writing and reading. Characters such as "ሀ," "ኅ" and "ሐ," or "ሰ" and "ሠ" may be used interchangeably. Similarly, characters like "ጸ" and "ፀ" or "አ" and "ዓ" can be used to represent the same meaning. For instance, "ጸሀይ" (sun) can also be written as "ፀሐይ." Additionally, Amharic words with suffixes, such as "ቷል," can be alternatively written as "ቱዋል." To ensure consistency, characters falling under these categories were normalized to a common canonical standard representation.

```
def normalize_char_level(input_token):
        #Normalizing similar character
        rep1=re.sub('[ሃኅኃሐሓኻ]','ሀ',input_token)
        rep2=re.sub('[ሑኁዅ]','ሁ',rep1)
        rep7=re.sub('[ሠ]','ሰ',rep6)
        rep8=re.sub('[ሡ]','ሱ',rep7)
        rep19=re.sub('[ዐ]','አ',rep18)
        rep20=re.sub('[ጸ]','ፀ',rep19)
        #...
        #Normalizing words with Labialized Amharic characters
        rep27=re.sub('(ሉ[ዋአ])','ሏ',rep26)
        rep28=re.sub('(ሙ[ዋአ])','ሟ',rep27)
        rep29=re.sub('(ቱ[ዋአ])','ቷ',rep28)
        rep30=re.sub('(ሩ[ዋአ])','ሯ',rep29)
        rep31=re.sub('(ሱ[ዋአ])','ሷ',rep30)
```

Figure 4.6 Implementation of Normalizing Dataset

4.6.6. Tokenization

Using the word_tokenize function from the NLTK library, the dataset's text was tokenized. Tokenization is the division of the text into discrete components known as tokens. In this case, the text was tokenized at the word level, where each word was separated and transformed into a list of tokens. By tokenizing the dataset, we obtain a structured representation of the text that can be further processed using feature engineering.

4.6.7. Stopword Removal

Stopwords, such as articles and certain verbs, are typically considered insignificant in determining the context or meaning of a sentence. These words can be removed from the dataset without adversely affecting the final model being trained. These common noisy words have limited importance in text feature extraction as they do not contribute significantly to the actual meaning of a sentence. Instead, they mainly contribute to feature dimensionality and can be discarded to enhance performance. Stopword removal was performed using predefined stopword list specifically designed for the Amharic language. With a few observations and expert's suggestion, a custom list of stopwords is added to the primary list to further clean out irrelevant words.

```
from nltk.tokenize import word_tokenize
#Tokenization
def tokenize(text):
    text = [word_tokenize(x) for x in text]
    return text

df.text = tokenize(df.text)

#Stopword Removal
stopwords_list = ['በሆነው', 'መሆኑን', 'ላይ', 'በአንድ', 'እንደዚህ', 'ስለ', 'የሚሆኑ', 'ለዚያው', 'የሆኑ', 'በመሆን', 'ይሁን',
                  'ላይም ', 'በማለት ', 'በእነዚህ', 'ከነዚህ' , 'እዚሁ', 'ሁኔታ', 'ከላይ', 'ባለ', 'ያህል', 'ቢሆንም',
                  'ሌላ', 'ሁሉ', 'ይህንን', 'ነበር', 'ከሆነና','ሌላውን', 'ሌላው', 'እነዚሁ', 'በዚህ', 'አለበት', 'አይሆንም', 'የሌሎች ', 'እንደሆኑ', 'እስከ',

df.text = df['text'].apply(lambda x: [item for item in x if item not in stopwords_list])

```

Figure 4.7 Implementation of Stopword Removal

4.6.8. Bag of Words and Term Frequency

A simple bag of words approach was used to analyze the most common terms. This approach involved creating a frequency distribution of terms from the dataset, where the occurrence of each term in the dataset was counted. By applying this technique, we were able to identify the most frequent terms occurring in the clickbait dataset. An illustration of term frequency for clickbait and non-clickbait terms is presented in Figures 4.8 and 4.9 respectively.

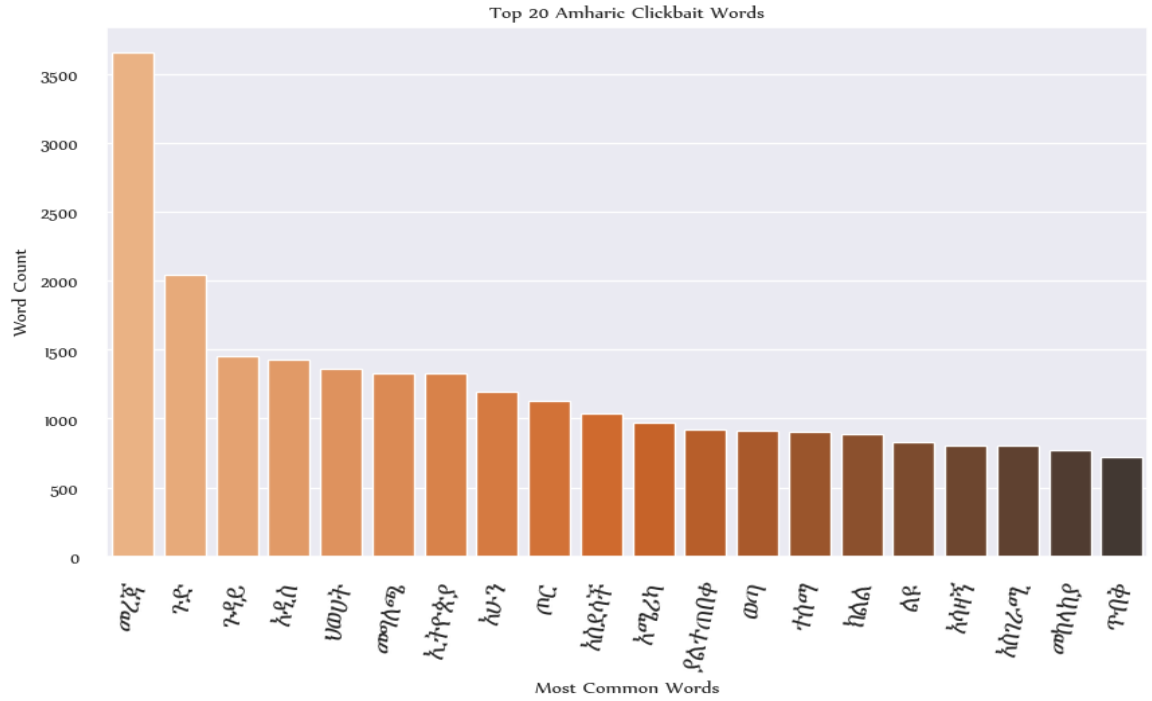


Figure 4.8 Top 20 Most Common Amharic Clickbait Words

As it can be observed in the Figure 4.8 above, the most common Amharic clickbait words predominantly revolve around current, trending, and hot topics. These words often employ hyperbolic language, provocative subjects, and sensationalized claims to capture the readers' attention. The presence of these terms indicates the clickbait content's intention to entice and engage readers by offering intriguing and exaggerated headlines.

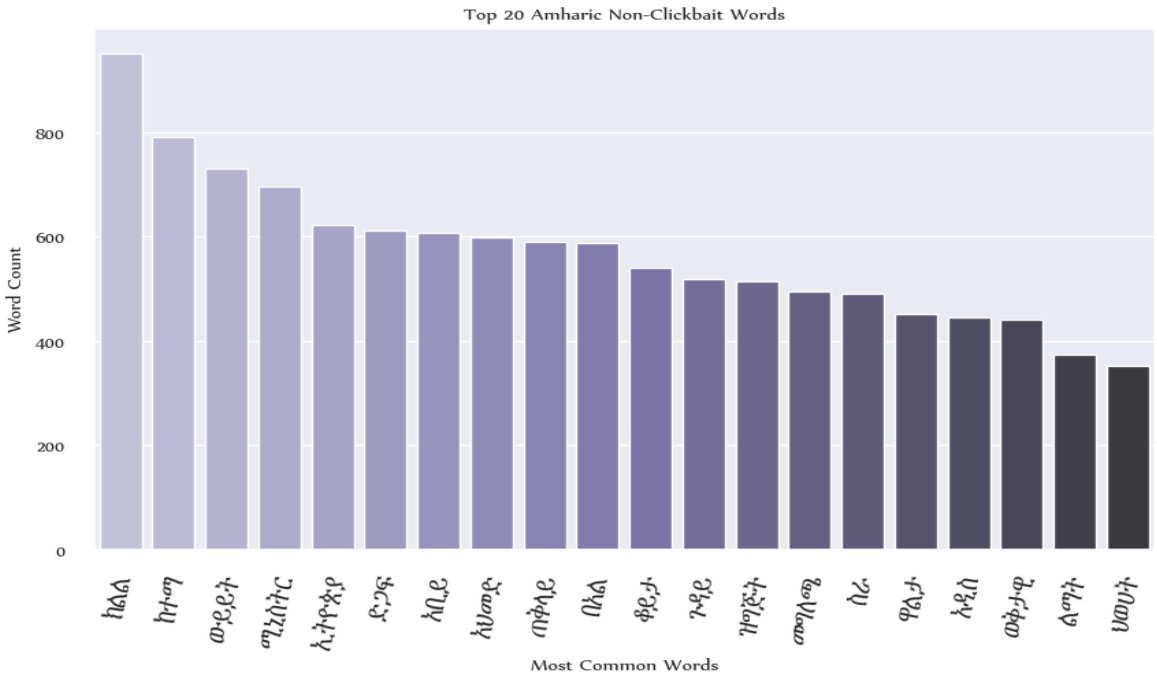


Figure 4.9 Top 20 Most Common Non-clickbait Words.

It is observed in Figure 4.9 that the most common non-clickbait words often revolve around topics such as government, officials, work, and development. Unlike clickbait content, these words convey a more measured and calm tone, focusing on factual information and highlighting achievements and progress in various areas. The prevalence of these non-clickbait terms suggests that there is a significant presence of news, informative articles, and reports related to government initiatives, official announcements within the dataset.

Word clouds play a significant role in enriching the dataset analysis by providing additional information about the textual content. By visualizing the frequency of words in a visually appealing manner, word clouds offer a quick and intuitive overview of the prominent terms within the dataset. They allow researchers to identify the most common and impactful words, enabling a deeper understanding of the underlying themes, sentiments, and language patterns present in the data.

Figure 4.10 Amharic Clickbait Terms Word cloud

The observations regarding the most common terms in both clickbait and non-clickbait classes can be further reinforced by visualizing the data using word clouds. By generating word clouds for clickbait and non-clickbait texts separately, we can observe a clearer and more comprehensive picture of the prevalent terms in each group.

Figure 4.11 Amharic Non-clickbait Terms Word cloud

4.6.9. Stemming

The dataset was then subjected to stemming using the HornMorpho library. Stemming is a process that aims to reduce words to their base or root form, which helps in standardizing the vocabulary and reducing the dimensionality of the dataset. The HornMorpho library has a module specifically designed for Amharic language processing and it provided the necessary methods for analyzing word morphology. By applying stemming, the available roots, stems, and lemmas of the given words were extracted and transformed into their standard form. This process enhances the efficiency of subsequent analyses and machine learning algorithms by reducing the complexity of the dataset. Stemming ensures that variations of the same word are treated as a single entity, enabling better comparison and identification of patterns in the clickbait data.

4.6.10. Feature Extraction

As this is a newly procured dataset and first of a kind experiment upon Amharic clickbait, we considered the traditional feature representations – BoW (Bag of Words) and TF-IDF (Term Frequency-Inverse Document Frequency) to establish baseline for the experiment. To incorporate the TF-IDF approach, the scikit-learn library's TF-IDF Vectorizer was employed. This tool transformed the raw text data into a numerical representation by converting each document into an array vector. The TF-IDF vectorizer ran under different

variations, such as unigrams (1-gram) and bigrams (2-gram) to capture both single words and combinations of adjacent words. This approach does not consider the semantic and syntactic information.

```
#TF-IDF Baseline
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import TfidfTransformer

#using uni-gram
tfidfv1=TfidfVectorizer(ngram_range=(1,1),use_idf=True, smooth_idf=True)
unigram=tfidfv1.fit_transform(stem_words).toarray()

#using bi-gram
tfidfv2=TfidfVectorizer(ngram_range=(2,2),use_idf=True, smooth_idf=True)
bigram=tfidfv2.fit_transform(stem_words).toarray()

```

Figure 4.12 Implementation of TF-IDF Baseline using unigram and bigram

Following the feature extraction stage, the dataset was split into training and testing subsets using the "train_test_split" class from the scikit-learn model selection package. It was used perform the data splitting. This code includes the parameter "random state=42", which ensures that we obtain the same train and test sets across different executions. This allows for reproducibility and consistency in evaluating the model's performance. Here, "x" refers to the extracted features, while "y" represents the class label.

```
L = []
for  i , token  in  enumerate (df['text'].astype('U')) :
    token = str(token)
    words = [w for w in token.split()]
    L.append(len(words))
sequence_size = max (L)

X = df['text']
y = df['class']

# Data Splitting
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X , y, test_size = 0.2, random_state = 42)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size = 0.2, random_state = 42)
```

Figure 4.13 Implementation of Tokenization and Data Splitting

4.6.11. Word Embedding

Word Embedding is a technique used to represent words as numerical vectors, enabling computers to understand and process textual data. In our experiment, two word embedding models, namely word2vec and fastText, were utilized.

Amharic word2vec model which has both stemmed and unstemmed skip-gram formats, using the gensim library was utilized to generate word embeddings specific to the Amharic language. By training the word2vec model on a large Amharic text corpus, it learned to represent words as vectors in a dimensional space, retaining their semantic and contextual

relationships. These word embeddings allow the model to extract the meaning and similarity between words, which is crucial for the subsequent machine learning tasks.

```
word2vec_model  = 'C:/Users/Arefat/Am-word2vec-skipgram300-stemmed.bin'
w2v = gensim.models.KeyedVectors.load_word2vec_format(WORD2VEC_VECTORS_BIN , binary = True)

embedding_dim = 300
def create_data(data, sequence_size, embedding_dim):
    embedding_data = np.zeros((len(data), sequence_size, embedding_dim))
    for i,sentence in enumerate(data) :
        sentence = sentence.replace('-', ' ')
        words = nltk.word_tokenize(sentence)
        j = 0
        for w in words:
            try :
                embedding_data[i , j] = w2v[w]
                j += 1
            except :
                pass
    return embedding_data

train_data = create_data(X_train, sequence_size, embedding_dim)
test_data = create_data(X_test, sequence_size, embedding_dim)
val_data = create_data(X_val, sequence_size, embedding_dim)

```

Figure 4.14 Implementation of word2vec Embedding.

To access the coefficients of the word embeddings, the word itself is used as a key to retrieve the corresponding embedding from the embedding_index dictionary. The next step involves preparing an embedding matrix of 300x300 dimension for all the words in the word index of our dataset. To check if a match exists in the pre-trained word embeddings, each tokenized word from the word index is compared with the embedding_index. Word2vec transforms texts into vectors as it is shown in Figure 4.15 below.

```
w2v["አስገራሚ"]
```

```
array([ 2.19328166e-03, -2.77252086e-02, -1.29083339e-02,  3.47509198e-02,
       -5.95358796e-02,  4.85551655e-02, -8.19950830e-03,  7.64514655e-02,
        9.57262889e-02, -3.31964903e-02, -7.24890009e-02,  2.01746989e-02,
        3.86186838e-02, -2.37429165e-03, -9.55067109e-03,  8.53477567e-02,
       -1.54164823e-04,  2.50162724e-02,  1.49386553e-02,  6.85853288e-02,
        9.39473510e-03, -1.24926306e-01, -1.20888390e-01, -2.33595930e-02,
       -1.32726533e-02,  1.12407595e-01,  8.67028758e-02,  6.32498506e-03,
        1.81709435e-02, -4.08096537e-02, -2.69676023e-03,  1.05976313e-01,
       -3.82349528e-02, -1.08297683e-01,  8.54445249e-02,  4.89874277e-03,
        4.27312916e-04, -5.23079485e-02, -8.30474347e-02, -2.24515311e-02,
       -1.83811912e-03,  1.14844264e-02,  1.44593511e-02,  1.80480964e-02,
        1.93812400e-02,  9.28979553e-03,  2.54408177e-02, -9.85527262e-02,
       -7.94856343e-03,  1.32884517e-01, -7.14199543e-02,  4.27177288e-02,
```

Figure 4.15 word2vec vector representation sample

In addition to word2vec, a pre-trained fastText Amharic model was utilized. It breaks words into smaller components called character n-grams, which enables it to handle out-of-vocabulary words and capture morphological variations in a language. The pre-trained fastText Amharic model provides word embeddings is expected to outperform the word2vec model. Another benefit of fastText is that it provides the ability to train an embedding model on a custom wordlist such as your tokens vocabulary which would provide better context and establish relevant relationship between words. A comparison of a general pretrained Amharic fastText model and custom-trained models with the clickbait token is presented below in Figure 4.16 below.

```
model = fasttext.load_model('cc.am.300.bin')
```

```
model.get_nearest_neighbors('አስቂኝ')
```

```
[(0.8644751906394958, 'አስቂኝም'),
 (0.8403595089912415, 'በአስቂኝ'),
 (0.8011069297790527, 'አስቂኝና'),
 (0.7737205624580383, 'ክትሞቱ'),
 (0.7293388843536377, 'ገጠመኝና'),
 (0.7265560626983643, 'እዉነተኛና'),
 (0.7011873126029968, 'አስገራሚም'),
 (0.6811816692352295, 'ገጠመኞቻቸዉና'),
 (0.6786954998970032, 'አስገራሚና'),
 (0.6768706440925598, 'ገጠመኞቹ')]
```

```
model.get_nearest_neighbors('ከተማ')
```

```
[(0.5735253691673279, 'ከመዲናይቱ'),
 (0.569017231464386, 'የተቆረቆረችው'),
 (0.5674211382865906, 'በሻሸመኔ'),
 (0.564568817615509, 'ከሻሸመኔ'),
 (0.5594919919967651, 'በምትገኝ'),
 (0.5561972260475159, 'የደብረማርቆስ'),
 (0.554186999797821, 'ከኮምቦልቻ'),
 (0.5525496602058411, 'በሃዋሳ'),
 (0.5512027740478516, 'ከተቆረቆረች'),
 (0.5505874752998352, 'በባሕርዳር')]
```

```
custom_model = fasttext.train_unsupervised
('clickbait-cleaned.txt')
```

```
custom_model.get_nearest_neighbors('አስቂኝ')
```

```
[(0.9009625315666199, 'ቀልድ'),
 (0.8945362567901611, 'ቀልዶች'),
 (0.8935312032699585, 'ሀዲይሳ'),
 (0.8892109394073486, 'አዝናኝና'),
 (0.8868719935417175, 'አዝናኝ'),
 (0.8853875398635864, 'የፅድቅ'),
 (0.8695741891860962, 'ከተመልካች'),
 (0.8598551750183105, 'ተመልካች'),
 (0.8547405004501343, 'ሞኖሎግ'),
 (0.8523733019828796, 'የተላኩ')]
```

```
custom_model.get_nearest_neighbors('ከተማ')
```

```
[(0.9164811968803406, 'ከተማና'),
 (0.878445029258728, 'ከተማን'),
 (0.8769887685775757, 'ክፍለ'),
 (0.8640891313552856, 'የከተማ'),
 (0.8586955666542053, 'የከተማዋ'),
 (0.8526677489280701, 'በከተማ'),
 (0.8524888753890991, 'ለከተማ'),
 (0.8023151755332947, 'ከተራ'),
 (0.7981350421905518, 'በከተማዋ'),
 (0.7946054339408875, 'ነዋሪዎች')]
```

Figure 4.16 Comparison of general (left) and custom (right) fastText Amharic models

4.7. Experiments

In this section, we present the implementation and evaluation of various machine learning and neural network models trained on the Amharic clickbait dataset. Our goal is to explore the effectiveness of different models in detecting and classifying clickbait in Amharic language. For the machine learning models, we have employed Support Vector Machines (SVM), Random Forest (RF), and Logistic Regression (LR), which are widely used in text classification tasks. Additionally, we have also experimented with deep learning models such as Long Short-Term Memory (LSTM) model, Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN) to harness their ability to capture complex patterns and sequential information in the text data.

4.7.1. Machine Learning Models Implementation

4.7.1.1. Logistic Regression

The implementation of Logistic Regression on the Amharic clickbait dataset is done using scikit-learn's LogisticRegression class, which is imported to create an instance of the logistic regression model. The model is configured with specific parameters such as 'C' set to 500, which controls the extent of regularization applied to the model, and 'tol' indicating the convergence tolerance. The model is then trained on the training data using the fit function. Predictions are made on both the training and testing sets and results are stored.

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(C=500, solver = 'liblinear', tol=0.0001)

lr.fit(X_train,y_train)

lr_train_preds = lr.predict(X_train)
lr_test_preds = lr.predict(X_test)

print(train_results(lr_train_preds))
print(test_results(lr_test_preds))

#plot confusion matrix on test set LR Classifier
sns.set()

cm_dc = confusion_matrix(y_test, lr_test_preds)
sns.heatmap(cm_dc.T, square=True, annot=True, cbar=False,cmap="inferno", xticklabels=['non-clickbait','clickbait']
plt.xlabel('true label')
plt.ylabel('predicted label')

```

Figure 4.17 Implementation of Logistic Regression Model

4.7.1.2. Support Vector Machine

When training SVM on the Amharic Clickbait Dataset, the first step was to import the necessary library using from sklearn. It imports the LinearSVC model from the sklearn SVM module. Next, we instantiate the SVM classifier as LinearSVC and specify any desired parameters. We then provide the training features (X_train) and the corresponding class labels (y_train) to the classifier using the fit method, initiating the training process of the machine learning model.

```
#SVM Model on Amharic Clickbait Dataset

svm_classifier = LinearSVC(class_weight='balanced', max_iter = 1500 )

svm_classifier.fit(X_train, y_train)

svm_test_preds = svm_classifier.predict(X_test)
svm_train_preds = svm_classifier.predict(X_train)

print(train_results(svm_train_preds))
print(test_results(svm_test_preds))

#plot confusion matrix for SVM Classifier
sns.set()

cm_dc = confusion_matrix(y_test, svm_test_preds)
sns.heatmap(cm_dc.T, square=True, annot=True, cbar=False, xticklabels=['non-clickbait','clickbait'],yticklabels=[

plt.xlabel('true label')
plt.ylabel('predicted label');
```

Figure 4.18 Implementation of Support Vector Machine

4.7.1.3. Random Forest

Experiment for Random Forest Classification began by creating an instance of the RandomForestClassifier model. The model is then trained using the training features (X_train) and the corresponding class labels (y_train) with the 'fit' method, which initiates the training process. Then we observed the results for the prediction for the training and testing set to assess the accuracy and recall of the model.

```
#Random Forest Classification on Amharic Clickbait
rf_classifier = RandomForestClassifier(class_weight = 'balanced' )
rf_classifier.fit(X_train, y_train)

rf_test_preds = rf_classifier.predict(X_test)
rf_train_preds = rf_classifier.predict(X_train)

print(train_results(rf_train_preds))
print(test_results(rf_test_preds))

#without Grid Search
```

Figure 4.19 Implementation of Random Forest Classifier

To find the best model parameters under the Random Forest classifier, a GridSearch was performed. GridSearch is a technique that exhaustively searches through a specified parameter grid to find the optimal combination of hyperparameters for a given model. In this

case, the RandomForestClassifier was used as the base model. The GridSearchCV class from scikit-learn was utilized, which automatically performs cross-validation on the training data to evaluate different hyperparameter combinations.

```
]: grid_rfc=GridSearchCV(rfc_grid,param_grid_rfc,cv=2,scoring='accuracy',n_jobs=-1,verbose=1)
   grid_rfc.fit(X_train,y_train)

   Fitting 2 folds for each of 4 candidates, totalling 8 fits

]: GridSearchCV
   GridSearchCV(cv=2, estimator=RandomForestClassifier(class_weight='balanced'),
                n_jobs=-1,
                param_grid={'max_depth': [200, 300], 'n_estimators': [800, 900]},
                scoring='accuracy', verbose=1)
     estimator: RandomForestClassifier
     RandomForestClassifier(class_weight='balanced')
       RandomForestClassifier
       RandomForestClassifier(class_weight='balanced')
```

Figure 4.20 Parameter Tuning using Grid Search algorithm.

By specifying a parameter grid containing different values for parameters such as the number of estimators and maximum depth, GridSearch systematically evaluates the performance of the Random Forest classifier for each combination. It was found that under max depth of 300 and around 900 estimators would result in the best performance for this Random Forest classifier, The best model is then selected based on the evaluation metric, such as accuracy or F1-score.

4.7.1.4. XGBoost

XGBoost is a popular gradient boosting framework that has gained attention for its effectiveness in various machine learning tasks. In this experiment, the XGBoostClassifier was used. The XGBoostClassifier object is then instantiated with default hyperparameters. Next, the model is trained using the training data by calling the fit method and passing the feature matrix (X_train) and the corresponding class labels (y_train).

```
1  #XGBoost Classifier
2  from xgboost import XGBClassifier
3
4  xgb_clf = XGBClassifier()
5  xgb_clf.fit(X_train, y_train)
6
7  xgb_test_preds = xgb_clf.predict(X_test)
8  xgb_train_preds = xgb_clf.predict(X_train)
9
10 print(test_results(xgb_test_preds))
11 print(train_results(xgb_train_preds))
```

Figure 4.21 Implementation of XGBoost Classifier

4.7.2. Neural Network Models Implementation

In this section, we discuss the implementation of various neural network models for the detection of clickbait in the Amharic language. We explored the effectiveness of recurrent neural network (RNN) architectures, including long short-term memory (LSTM) and gated recurrent unit (GRU), as well as their bidirectional counterparts, bidirectional LSTM (Bi-LSTM) and bidirectional GRU (Bi-GRU). Additionally, we experimented on the potential of convolutional neural networks (CNN) in clickbait detection, as it has been researched before by (Fu et al., 2017; X. Zhang et al., 2015). These models are trained on the Amharic clickbait dataset, aiming to capture the underlying patterns and linguistic nuances of clickbait content. Furthermore, we conducted hyperparameter tuning to optimize the model's performance by fine-tuning parameters such as learning rate, batch size, and regularization techniques.

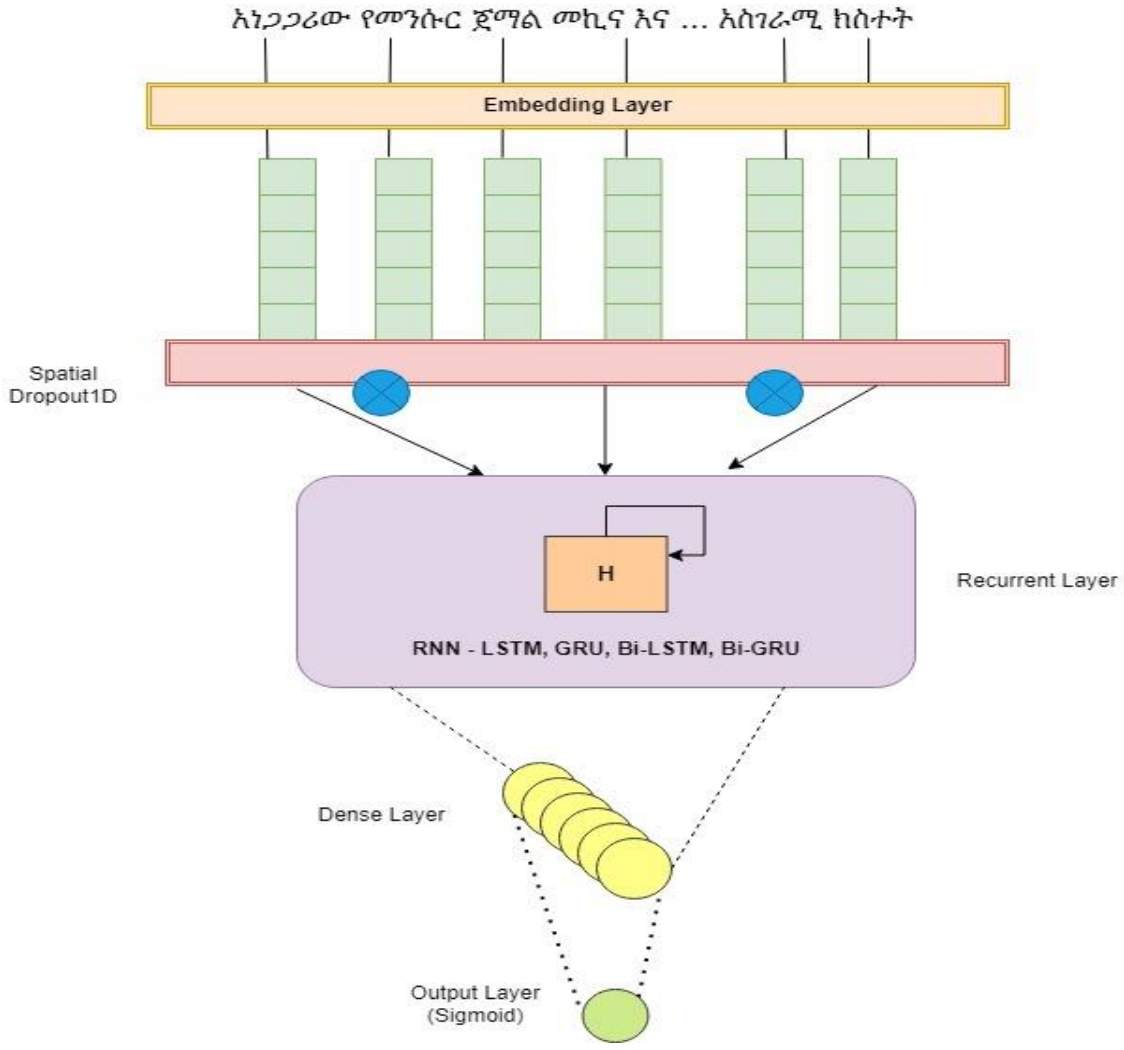


Figure 4.22 Recurrent Neural Networks Model Architecture for Clickbait Detection

4.7.2.1. Recurrent Neural Network Implementation

Neural networks applied on text data requires an Embedding Layer, which maps each word to a continuous vector representation, allowing the model to capture the semantic relationships between words. The weights of the embedding layer are learned during the training process, enabling the model to adapt to the specific task of clickbait detection. Thus, the design starts with an Embedding Layer, which maps each Amharic word to a vector representation which we have already done before, using both word2vec and fastText embedding models.

For these experiments, we use Keras' different variants of recurrent methods by importing the various layers, operations, regularizers and models we would use to train the model.

```
import keras
from keras.models import Model
from keras.layers import Layer, Input, merge, Dense, LSTM, Bidirectional, GRU, SimpleRNN
from keras.layers.merge import concatenate, dot, multiply, add
from keras.layers.core import Dense, Lambda, Reshape
from keras.layers.convolutional import Convolution1D
from keras.layers.merge import concatenate, dot
from keras.callbacks import ModelCheckpoint
from keras import regularizers, constraints
from keras.initializers import glorot_uniform
from keras.callbacks import ModelCheckpoint
from attention import AttentionWithContext
```

Figure 4.23 Importing Models for Recurrent Neural Network Implementation

To mitigate overfitting and enhance generalization, spatial dropout is applied after the embedding layer. Spatial dropout randomly drops entire 1D feature maps within the sequence, promoting regularization and preventing the model from relying too heavily on specific features or words as introduced by (Tompson et al., 2015). The spatial dropout layer is a type of regularization technique used in neural networks. It helps prevent overfitting by randomly ignoring (setting to zero) some of the features or channels from frequently occurring word sequences in catchy stories.

In the context of clickbait detection, catchy stories often contain certain words or phrases that are strongly associated with clickbait content. By randomly reducing some of these elements during training, the model is encouraged to focus on other relevant features and not solely rely on the occurrence of specific words. For example, the word "ሰበር" or "Breaking" is frequently present in clickbait headlines, using the spatial dropout layer can prevent the model from automatically classifying any story containing that word as clickbait without considering other contextual cues. By selectively ignoring certain features, the model is

encouraged to learn more generalized patterns and avoid memorizing specific word associations, which can lead to more robust and accurate classification results.

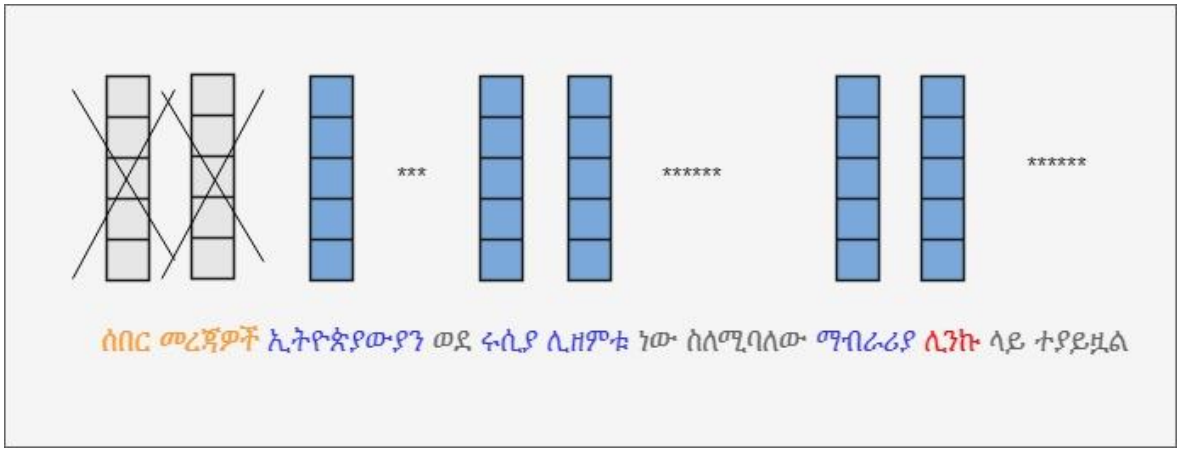


Figure 4.24 Spatial Dropout

Following the spatial dropout, a recurrent layer such as LSTM or GRU is used to model the sequential nature of the text. These recurrent layers can retain information from previous words and consider the contextual dependencies within the sequence. Finally, a dense layer with an appropriate activation function like sigmoid is used to make the final clickbait prediction based on the extracted features from the recurrent layer. The dense layer transforms the learned representations into the desired output format, classifying whether the input is clickbait or not.

4.7.2.2. Long-Short Term Memory (LSTM)

We experimented with Long Short-Term Memory networks (LSTM), an extension of Recurrent Neural Networks (RNN). LSTM incorporates forget or remember gates in each hidden layer, allowing it to retain information from previous stages throughout the entire learning process. This enables LSTM to actively maintain self-connecting layers, ensuring that important information from new inputs is preserved as the learning progresses to deeper layers (Vorakitphan et al., 2019).

This general class we defined which is the main class for Amharic clickbait detection for the various recurrent neural network (RNN) models we are implementing takes two parameters. They are `title_max`, representing the maximum number of words in the title, and `word_embedding_size`, representing the size of the word embeddings which is 300. It starts by defining an input layer for the word embeddings of the title, with the shape. The subsequent layers include multiple bidirectional LSTM layers, an attention layer, and several dropout and dense layers.

```
class Detector: #Main Class for Amharic Clickbait Detection = RNN models

    def __init__(self, title_max, word_embedding_size):
        """ Parameters: Maximum words, Size of the word embeddings"""
        self.model = None
        self.title_max = title_max
        self.word_embedding_size = word_embedding_size

    def set_params(self, activation):
        #Acitvation Function
        self.activation = activation

    def create_model(self):
        # Input - word embeddings of the title | Padding is applied if title falls short, or truncated if the title is long.
        # A LSTM, Bi-LSTM, GRU, BiGRU layer can be included here. | Sigmoid function is used.
        title_words = Input(shape=(self.title_max, self.word_embedding_size))

        # Layers for the Model 1
        lstm_layer = Bidirectional(LSTM(64, return_sequences=True))(title_words)
        lstm_layer = Bidirectional(LSTM(64, return_sequences=True))(lstm_layer)
        lstm_layer = Bidirectional(LSTM(64, return_sequences=True))(lstm_layer)
        attention_layer = AttentionWithContext()(lstm_layer)
        dropout1 = Dropout(0.2)(attention_layer)
        left_hidden_layer1 = Dense(64, activation=self.activation)(dropout1)
        dropout2 = Dropout(0.2)(left_hidden_layer1)
        left_hidden_layer2 = Dense(32, activation=self.activation)(dropout2)

        # Combines both the left and the right component
        combined1 = concatenate([left_hidden_layer1, left_hidden_layer2 ])
        dropout_overall1 = Dropout(0.2)(combined1)
        combined2 = Dense(32, activation=self.activation)(dropout_overall1)

        # Predicts
        output = Dense(1, activation='sigmoid')(combined2)
        self.model = Model(inputs=[title_words] , outputs=output)
        self.model.compile(optimizer='adadelta', loss='binary_crossentropy', metrics=['accuracy'])

        print self.model.summary()
```

Figure 4.25 Implementation of LSTM models on Amharic Clickbait Dataset

These layers help capture the contextual information and features of the input data. The activation function specified in the `activation` parameter is applied to the dense layers. The Bi-directional LSTM layer in this case, accepts the 'return_sequences' argument, which determines whether to return the full sequence of outputs or only the last output. Setting 'return_sequences' to True allows the subsequent layers to access the complete sequence information, which is useful in tasks where the sequential order is important, such as clickbait detection.

The final layers involve concatenating the outputs of the previous layers, applying dropout, and generating the prediction using a dense layer with sigmoid activation. The model is then compiled with the `adadelta` optimizer, binary cross-entropy loss function, and accuracy metric.

4.7.2.3. Gated Recurrent Units (GRUs)

Similarly, to the LSTM model, we adopt the GRU algorithm for our Amharic clickbait detection. The GRU extends the traditional RNN architecture by incorporating gating mechanisms, including forget and update gates, within each hidden layer. These gates enable the model to selectively remember or discard information from previous time steps

throughout the learning process. In our implementation, we utilize the Keras GRU layer by specifying the necessary parameters. The 'units' parameter determines the number of units in the GRU layer, while the 'activation' parameter defines the activation function to be applied. By setting 'return_sequences' to True, we ensure that the GRU layer returns the output sequence rather than just the final hidden state.

4.7.2.4. Convolutional Neural Network

In order to obtain an ideal feature vector that represents the text during training, we employ a one-dimensional convolutional kernel, resulting in a dimension of 300 and 32 channels of the same size. This is achieved by specifying the parameters in the Keras Conv1D layer. The 'relu' activation function is applied after the convolutions to introduce non-linearity, and the 'padding' parameter ensures that the output of the convolution has the same size as the input by padding it with zeros if necessary.

Once we obtain the feature vector representing the local features within a sequence of word embeddings through the convolution step, we apply one-dimensional max pooling. By taking into account pairs of subsequent words, irrespective of their precise placement within the wider input sequence, a high-level feature representation with a pooling size of 2 is produced. This allows us to discard less-relevant local information.

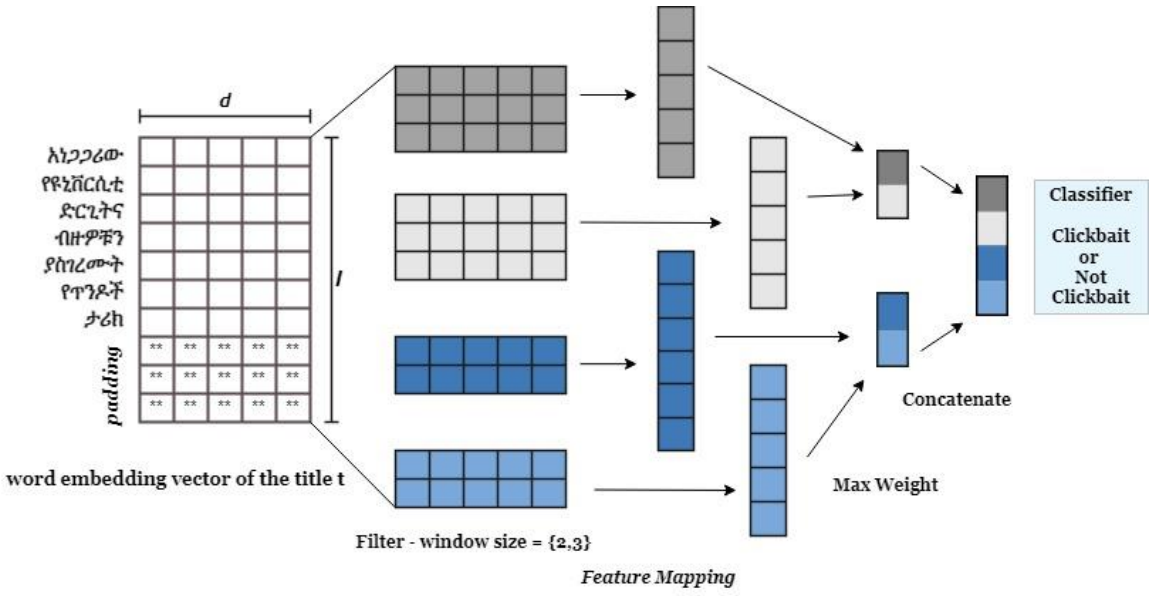


Figure 4.26 CNN Architecture for Feature Extraction in Amharic Clickbait Detection

After extracting the local features, we perform global max pooling over the entire sequence of filters, processing them one by one. Finally, a sigmoid activation function is applied in the output neuron to classify a given text as clickbait or non-clickbait.

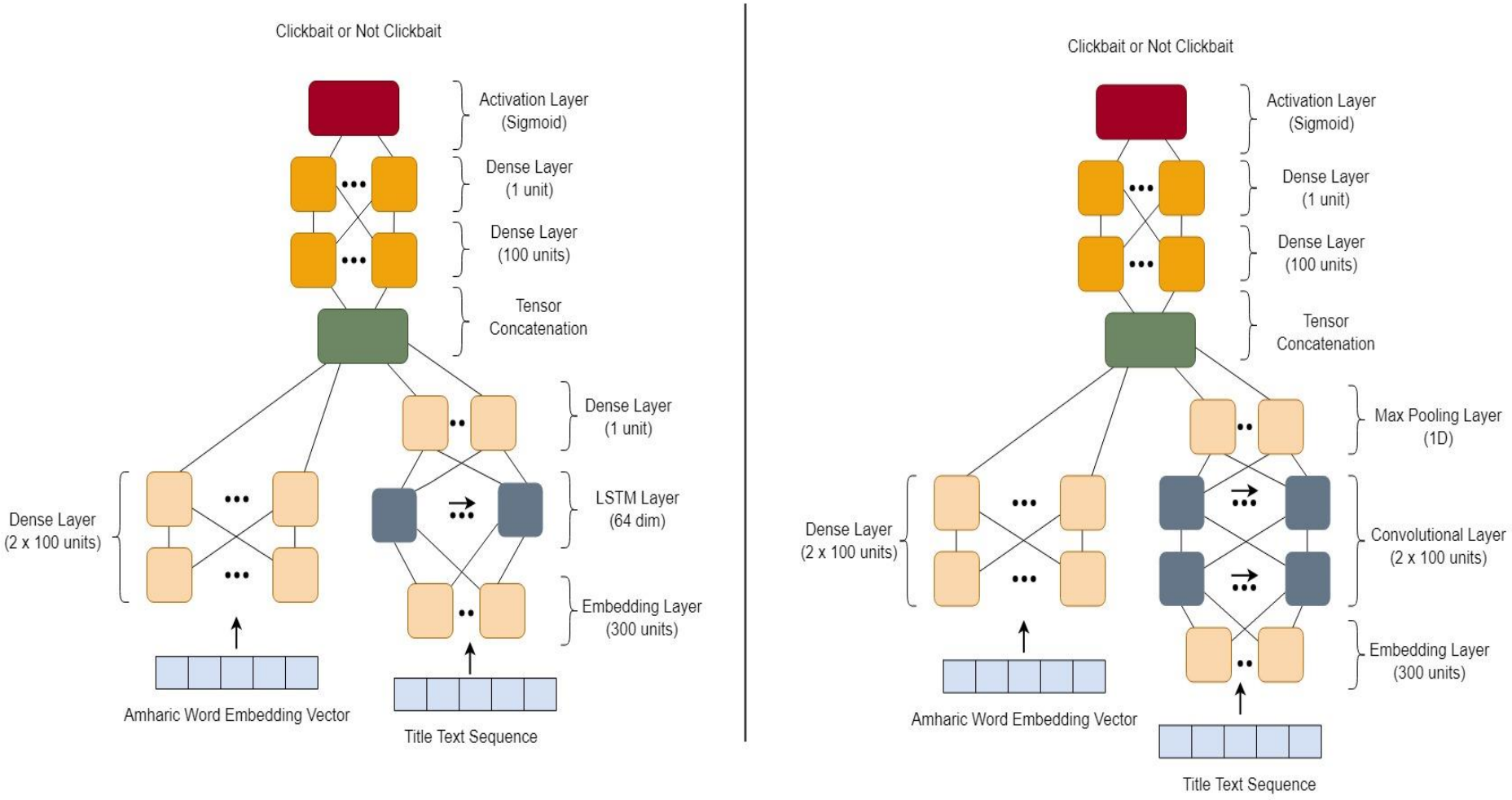


Figure 4.27 LSTM (left) and CNN (right) Models for Clickbait Classification

4.7.3. Tuning Hyperparameters

We can find the optimal hyperparameter settings to optimize the performance of the model by using the hyperparameter tuning procedure. We can fine-tune the models to deliver the best performance by methodically examining various hyperparameter combinations. The rationale for hyperparameter tuning is to find the optimal values that strike a balance between model complexity and generalization. By tuning the hyperparameters, we aim to prevent overfitting, where the model becomes too specialized to the training data and fails to generalize well to unseen data. On the other hand, underfitting happens when the model is overly straightforward and misses significant data patterns.

Hyperparameter tuning can be broadly categorized into two groups, specifically using automated means: Model-Based and Model-Free approaches. Within the Model-Free category, we have Grid Search and Random Search. Grid Search involves exhaustively searching over all possible combinations of parameters, but it can be computationally intensive and requires significant computational resources. The Model-Based category includes Bayesian Optimization and Evolutionary Algorithms (Pandey & Kaur, 2018). The hyperparameters that are commonly tuned include embedding dimensions, batch size, the number of epochs, activation function, and learning rate to identify the optimal set of hyperparameters that yield the highest accuracy.

We have used Grid Search for tuning the Random Forest classifier where it provided parameters for the max depth and number of estimators. However, it was very costly in terms of computational resources using this automated means. For tuning the CNN model, we have taken the manual approach and selected the common hyperparameters to study the effects, and several experiments were conducted. Those hyperparameters are described below:

- Number of epochs - determines how many times the entire dataset is used to assign or re-assign the neural network weights. Choosing the right number of epochs is crucial to avoid underfitting or overfitting.
- Window size - also known as n-gram, represents the number of surrounding words considered in the context.
- Word-vector dimension - the size of the dense vector that represents each word in a large dictionary. It impacts the efficiency of the features, as a high dimension can result in sparseness and the loss of semantic information.

- Modeling Setting or Technique - there are three variations of the CNN model: static, non-static, and random. The static model uses word vector's weights as the initial representation of all words and keeps them fixed. The non-static approach also starts with word vector's weights but adjusts them during training. The random variation assigns random weights to each word initially and updates them during model training.
- Learning rate - controls how frequently the model's weights are changed during training. It controls the speed of convergence and influences the model's ability to find the optimal solution.
- Batch size - is another hyperparameter that defines the number of training samples processed together before updating the model's weights. It affects the speed and stability of the training process. A larger batch size can provide faster training but may require more memory and computational resources.
- Activation function - determines how the output of a neuron or layer is calculated. The activation function brings nonlinearity into the network, allowing it to learn complicated patterns in the input. Common activation functions for hidden layers include sigmoid, which maps inputs to a range between 0 and 1; ReLU (Rectified Linear Unit), which sets negative values to zero; tanh (hyperbolic tangent), which maps inputs to a range between -1 and 1.

Table 4.1 Hyperparameters Selected for Tuning

Hyperparameters	**Variations**						
Number of epochs	5		10		20		
Window Size	2		2,3		2,3,4		
Word vector Dimension	100 (only fastText)				300		
Modelling Setting	Static		Non-static		Random		
Learning Rate	0.001	0.002	0.01	0.02	0.1	0.2	0.5
Batch Size	16		32		64		
Activation Function	Sigmoid		ReLU		Tanh		

CHAPTER FIVE

5. RESULTS AND DISCUSSIONS

This section presents the findings of the experimental study conducted to evaluate the performance of the Amharic clickbait detection model. In this chapter, we analyze and interpret the results obtained from the experiments, which include the application of a 5-fold cross-validation technique. The evaluation report provides comprehensive insights into the model's performance by examining key metrics such as accuracy, precision, recall and F1-score. Additionally, the chapter includes illustrations of the confusion matrix, which offers a deeper understanding of the model's classification performance.

5.1. Results for the Machine Learning Models

5.1.1. Results for Logistic Regression

Our first experimental result established a baseline using TF-IDF with unigram and bigram features. We examined the dataset with the Logistic Regression Model where its results are presented in the Confusion Matrix Figure 5.1 below. From the 5th fold of the dataset allocated for testing, it was able to classify 88% correctly. The model was able to achieve 0.87 recall and 0.88 precision, as well as an F1-score of 0.88.

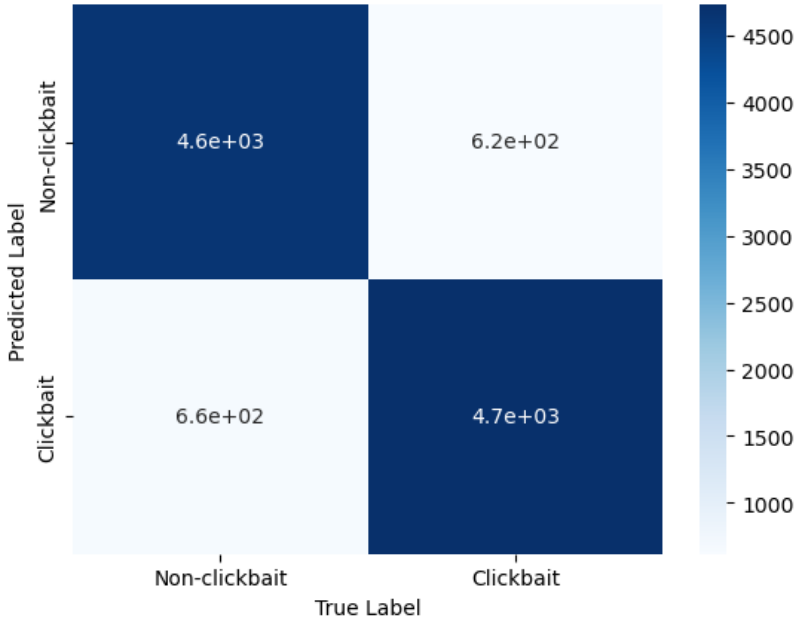


Figure 5.1 Confusion Matrix for Logistic Regression model

Another variation of the same experiment using TF-IDF with bigrams has scored 84% accuracy, with 0.83 recall and 0.84 precision. It was observed that the TF-IDF feature performed well when using unigrams but not with bigrams. This can be attributed to the nature of Amharic and the characteristics of clickbait content. Unigrams capture individual words, allowing the model to identify specific terms and their importance in distinguishing clickbait from non-clickbait. On the other hand, bigrams combine pairs of adjacent words, which may not capture the nuanced patterns and linguistic structures that are prevalent in clickbait headlines. Higher order n-grams like trigram have not yielded any comparable results.

5.1.2. Results for Support Vector Machines

From the results obtained using the Support Vector Machine (SVM) model with TF-IDF and unigram feature in this experiment, out of the total test instances, the SVM model correctly classified 4749 samples as clickbait and 4652 samples as non-clickbait, accounting for a significant portion of the test dataset. However, there were 617 instances misclassified as clickbait when they were actually non-clickbait, and 659 instances misclassified as non-clickbait when they were clickbait. In terms of percentages, the SVM model achieved an accuracy of approximately 88.31%. The precision is approximately 0.86 and recall result of 0.90. Figure 5.2 presents the confusion metrics along with training and testing accuracy recall observed for this experiment.

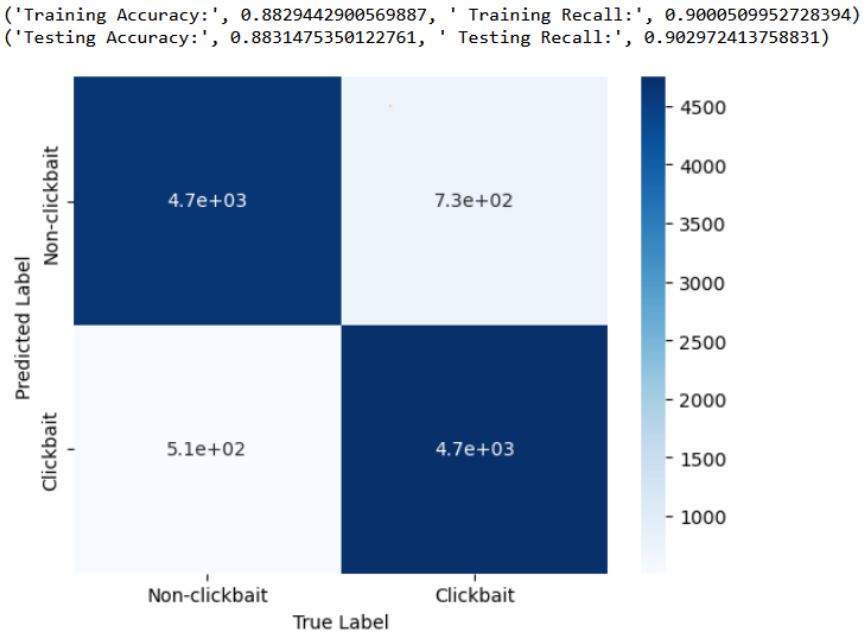


Figure 5.2 Confusion Matrix for Support Vector Machine

5.1.3. Results for Random Forest

The Random Forest experiment yielded promising results after conducting tuning with GridSearch, using a maximum depth of 300 and 900 estimators. Out of testing dataset, the RF model correctly classified 4871 samples as clickbait and 4627 samples as non-clickbait. This corresponds an accuracy rate of 89.22% of the instances being classified correctly. Examining precision, that measures the proportion of appropriately predicted clickbait out of all predicted clickbait instances, we find that the model achieved a precision rate of approximately 0.91.

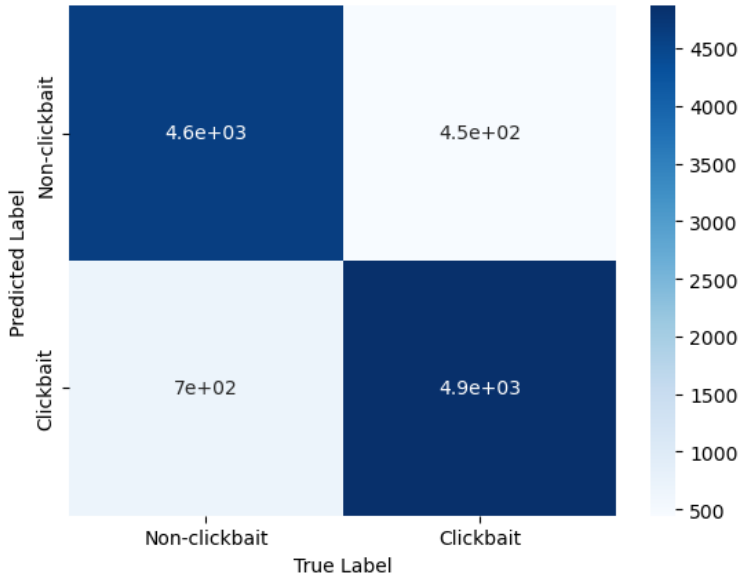


Figure 5.3 Confusion Matrix for Random Forest Classifier

The experiment achieved a recall rate of approximately 0.87, indicating that the model correctly identified a significant portion of the actual clickbait instances. Furthermore, the F1-score, which combines precision and recall, was approximately 0.89, indicating a good balance between the two metrics. The classification report generated for the experiment is provided in Figure 5.4 below.

Classification Report

	precision	recall	f1-score	support
0	0.9161	0.8752	0.8946	5244
1	0.8683	0.8740	0.8896	5402
accuracy			0.8922	10646
macro avg	0.8919	0.8740	0.8941	10646
weighted avg	0.9109	0.8788	0.8922	10646

Figure 5.4 Classification Report for Random Forest Classifier

5.1.4. Results for XGBoost

The XGBoost model achieved a high number of true positives and true negatives, with 4905 and 4768 respectively, indicating that it correctly classified a large proportion of both clickbait and non-clickbait instances. In terms of performance metrics, the XGBoost model showcased a high accuracy rate of 90% among the traditional machine learning models, correctly classifying a significant portion of the testing set. Additionally, the precision and recall values were 0.91 and 0.90 respectively.

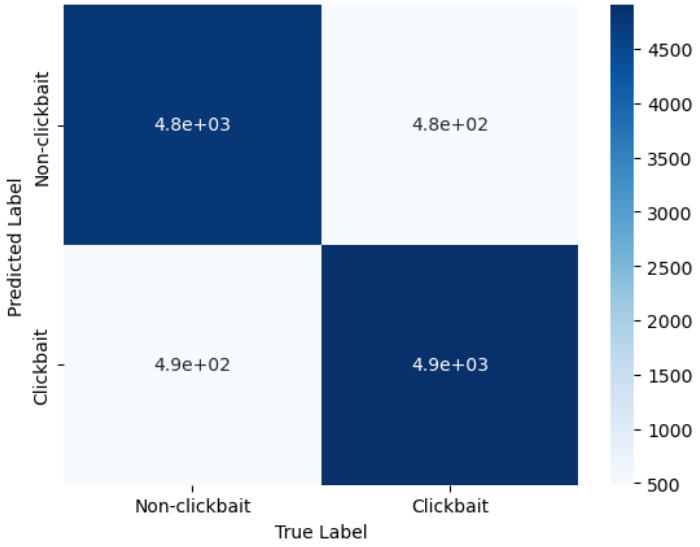


Figure 5.5 Confusion Matrix for XGBoost model

XGBoost utilizes gradient boosting, which involves iteratively adding new models to correct the mistakes made by previous models. Its iterative process allows XGBoost to concentrate more on misclassified instances, gradually improving its predictive performance. Classification report for XGBoost is illustrated in Figure 5.6.

```
Classification Report

              precision    recall  f1-score   support

           0     0.9100    0.9095    0.9098      5244
           1     0.9072    0.9077    0.9074      5402

    accuracy                         0.9086     10646
   macro avg     0.9100    0.9095    0.9098     10646
weighted avg     0.9100    0.9086    0.9085     10646

Confusion Matrix

[[4768    485]
 [488    4905]]
```

Figure 5.6 Classification Report for XGBoost

5.1.5. Summary of Machine Learning Models Results and Discussion

Table 5.1 Summary of Classification Performance with Machine Learning Models

Features	Model	Accuracy	Precision	Recall	F1-score
Unigram + TF-IDF	LR	0.8801	0.8847	0.8778	0.8812
	SVM	0.8831	0.8661	0.9029	0.8841
	RF	0.8922	**0.9161**	0.8740	0.8946
	XGBoost	**0.9086**	0.9100	**0.9095**	**0.9098**
Bigram + TF-IDF	LR	0.8425	0.8433	0.8362	0.8402
	SVM	0.8307	0.8217	0.8294	0.8255
	RF	0.8683	0.8652	0.8517	0.8643
	XGBoost	0.8617	0.8595	0.8778	0.8685

From the summary results on Table 5.1, we can observe that the unigram-based features outperformed the bigram-based features. Across all models, including Logistic Regression (LR), Support Vector Machine (SVM), Random Forest (RF), and XGBoost, the accuracy, precision, recall, and F1-score were consistently higher when using unigram features combined with TF-IDF.

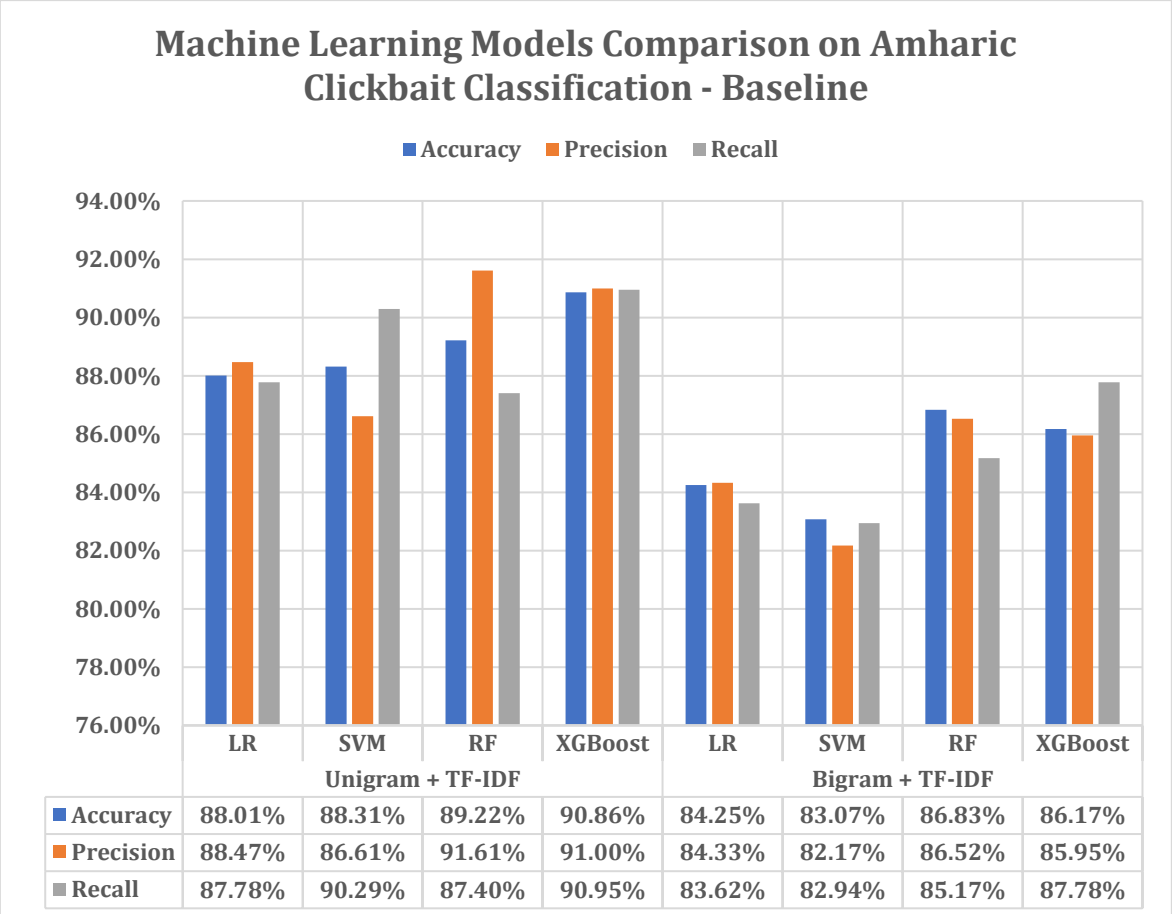
In terms of accuracy, XGBoost achieved the highest accuracy of 90.86%, among traditional ML models when using unigram features with TF-IDF, indicating its strong performance in classifying clickbait and non-clickbait instances in Amharic. XGBoost also exhibited high recall and F1-score values, consistently surpassing other models.

The rationale behind using XGBoost in this experiment lies in its ability to handle complex text patterns and capture non-linear relationships. By leveraging the power of ensemble learning and combining multiple weak classifiers, Gradient Boosting models such XGBoost and LightGBM have been proven to effectively model intricate clickbait detection patterns, leading to accurate predictions and improved performance (Shu et al., 2018).

It is noteworthy that the Random Forest (RF) classifier emerged as the second-best performing traditional model with a precision of 91.61%. It is important to note that the Random Forest model's performance was further enhanced through parameter tuning using grid search. By optimizing the number of estimators with a value of 900 and maximum depth up to 300, the model achieved improved precision and overall performance.

This tuning process allowed the Random Forest classifier to leverage the collective ensemble of multiple decision trees, leading to better generalization and classification accuracy in the Amharic clickbait detection task.

While the bigram features combined with TF-IDF still achieved reasonably good accuracy and performance metrics across all models, they were consistently lower than those obtained using unigram features. This suggests that for Amharic clickbait detection, unigram features provide more informative and discriminative signals for classification, capturing the important lexical nuances present in the language.



	Unigram + TF-IDF				Bigram + TF-IDF			
	LR	SVM	RF	XGBoost	LR	SVM	RF	XGBoost
Accuracy	88.01%	88.31%	89.22%	90.86%	84.25%	83.07%	86.83%	86.17%
Precision	88.47%	86.61%	91.61%	91.00%	84.33%	82.17%	86.52%	85.95%
Recall	87.78%	90.29%	87.40%	90.95%	83.62%	82.94%	85.17%	87.78%

Figure 5.7 Comparison of Results by ML models on Amharic Clickbait Classification

5.2. Results for the Neural Network Models

This section presents the findings of the experimental evaluation conducted on various neural network models for Amharic clickbait detection. It assesses the performance analysis of Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Bidirectional LSTM (Bi-LSTM), Bidirectional GRU (Bi-GRU), and Convolutional Neural Network (CNN) models. The neural network models were subjected to rigorous evaluation and hyperparameter tuning to maximize their classification accuracy and effectiveness.

5.2.1. Results for LSTM and Bi-LSTM

We experimented on the two variants of recurrent neural networks; LSTM and Bi-LSTM which mostly depends on the historical context of inputs rather than the last input. We used the two word embedding vectors we prepared, word2vec and fastText to feed the vector information along with the text sequence.

The LSTM model while using the word2vec embedding classified 9813 titles as clickbait and non-clickbait effectively, from the total 10,646 testing set. It yielded an accuracy of 92.18% and f1-score of 92.24%. The second variation of the LSTM model we experimented on, the bidirectional LSTM (Bi-LSTM), which is an extension of traditional LSTMs is assumed to enhance model performance as it can retain information from different occurrences.

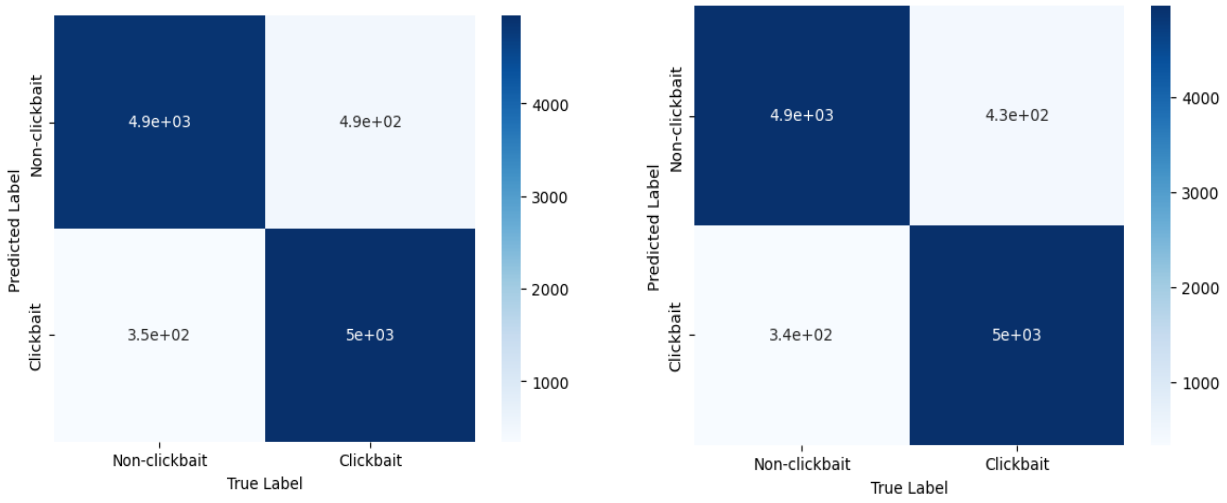


Figure 5.8 Confusion Matrix for LSTM (left) and Bi-LSTM (right) with word2vec

We have observed that the bidirectional LSTM under word2vec embedding resulted in an accuracy of 92.76% and an f1-score of 92.78%. The Bi-LSTM architecture, which incorporates bidirectional information flow, outperformed the LSTM model. The improvement in accuracy and f1-score suggests that capturing contextual information from both past and future tokens enhances the model's ability to capture the intricate language patterns in Amharic clickbait detection.

While using fastText pre-trained word embedding model on LSTM, the results of the experiments showed that both models achieved high accuracy and f1-scores, indicating their effectiveness in detecting clickbait content. The LSTM model achieved an accuracy of 92.65% and an f1-score of 92.72%, while the Bi-LSTM model achieved an accuracy of 92.60% and an f1-score of 92.67%.

Classification Report

	precision	recall	f1-score	support
0	0.9160	0.9346	0.9252	5244
1	0.9359	0.9177	0.9267	5402
accuracy			0.9260	10646
macro avg	0.9259	0.9261	0.9260	10646
weighted avg	0.9266	0.9266	0.9260	10646

Confusion Matrix

```
[[4876  341]
 [447  4982]]
```

Figure 5.9 Classification Report for Bi-LSTM model with fastText

5.2.2. Results for GRU and Bi-GRU

We also explored the use of GRUs (Gated Recurrent Units), which are computationally efficient models similar to LSTMs, but with a different architecture. The GRU model using the word2vec embedding vectors correctly classified 9841 instances of titles as True Positives and True Negatives, while misclassifying 805 instances as False Negatives and False Positives out of the averaged fold size of 10646. The model achieved an accuracy of 92.44% and an f1-score of 92.50%. Additionally, we trained a bidirectional GRU model, which considers information from both past and future contexts. This model achieved higher performance with an accuracy of 92.95% and an f1-score of 93.01%.

In addition to the experiments with LSTM and Bi-LSTM models using word2vec embeddings, we also utilized fastText embeddings for training the GRU and Bi-GRU models. The GRU model, leveraging fastText embeddings, achieved an accuracy of 92.82% and an f1-score of 92.89%. Furthermore, the bidirectional GRU model, incorporating fastText embeddings, demonstrated slightly improved performance, attaining an accuracy of 93.07% and an f1-score of 93.13%. These results highlight the effectiveness of employing fastText embeddings in conjunction with GRU and Bi-GRU models for Amharic clickbait detection.

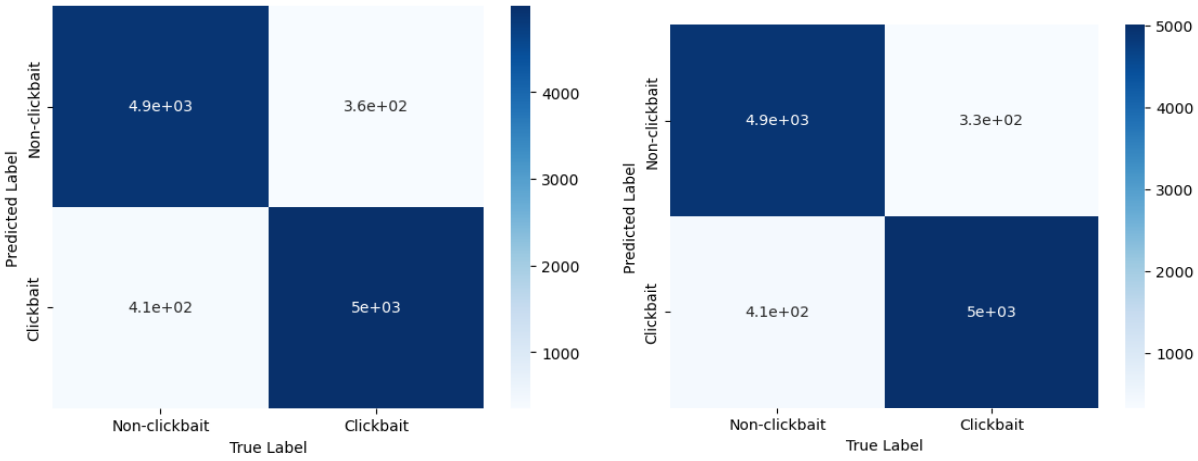


Figure 5.10 Confusion Matrix for GRU (left) and Bi-GRU (right) with fastText

5.2.3. Results for CNN (Convolutional Neural Network)

We also evaluated the performance of Convolutional Neural Network (CNN) models for Amharic clickbait detection using two different word embedding techniques: Word2Vec and fastText. When utilizing word2vec embeddings, the CNN model achieved an accuracy of 93.60%. The precision and f1-score were relatively high with the help of a few tuned parameters, with values of 93.90% and 93.66%, respectively.

On the other hand, the CNN model utilizing fastText embeddings exhibited even better performance. It achieved an accuracy of 94.20%. The precision and f1-score for fastText embedding were 93.82% and 94.23%, respectively, demonstrating the model's ability to accurately classify clickbait and non-clickbait text.

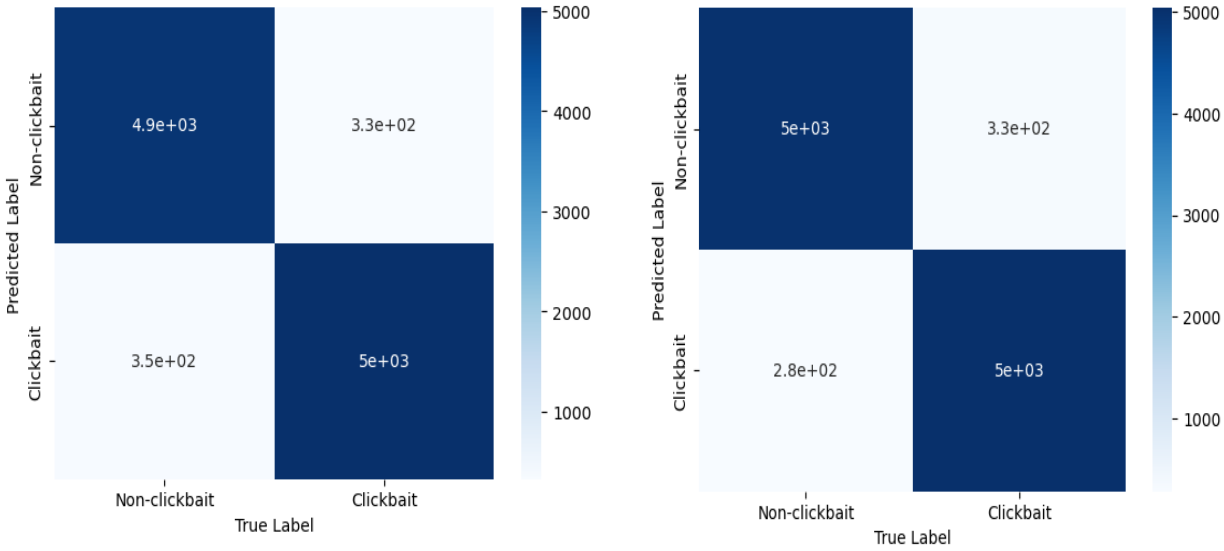


Figure 5.11 Confusion Matrix for CNN with word2vec (left) and fastText (right)

The model accuracy graph illustrated below in Figure 5.12 shows a constant upward trend for the training accuracy where it starts approximately at 87% and steadily increases with each epoch. The model's ability to accurately classify the training data improves significantly, reaching a high of 94% on the 4th epoch. This indicates that the model is effectively learning the underlying patterns and features in the dataset. Similarly, the validation accuracy values demonstrate a positive trend throughout the training process. The validation accuracy reaches a peak of 94% on the 10th epoch. This indicates that the model performs well not only on the training data but also on new, unseen data, suggesting that it can effectively classify clickbait and non-clickbait articles.

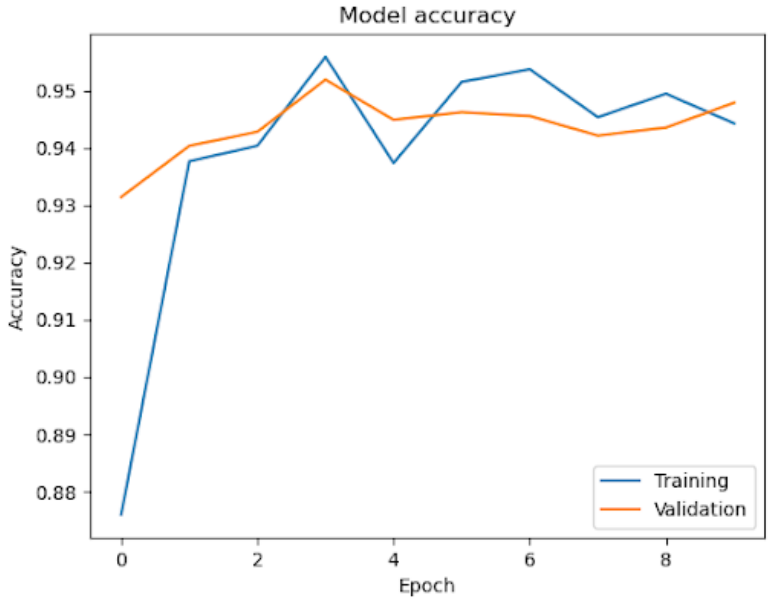


Figure 5.12 CNN Model Accuracy Graph (10 epochs)

5.2.4. Summary of Neural Network Models Results and Discussion

Table 5.2 Summary of Classification Performance with Neural Network Models

Embedding	Model	Accuracy	Precision	Recall	F1-score
word2vec	LSTM	0.9218	0.9104	0.9347	0.9224
	Bi-LSTM	0.9276	0.9360	0.9198	0.9278
	GRU	0.9244	0.9327	0.9174	0.9250
	Bi-GRU	0.9295	0.9221	0.9382	0.9301
	CNN	0.9360	0.9390	0.9343	0.9366
fastText	LSTM	0.9265	0.9193	0.9352	0.9272
	Bi-LSTM	0.9260	0.9359	0.9177	0.9267
	GRU	0.9282	0.9333	0.9246	0.9289
	Bi-GRU	0.9307	**0.9385**	0.9243	0.9313
	CNN	**0.9420**	0.9382	**0.9465**	**0.9423**

The experimental results from the neural network models for Amharic clickbait detection provide valuable insights into their performance. In terms of accuracy, the CNN model utilizing fastText embeddings achieved the highest accuracy of 94.20%, indicating that it correctly classified the highest proportion of clickbait and non-clickbait titles. This result demonstrates the effectiveness of the CNN architecture in capturing important patterns and features of clickbait, .

When examining the f1-score, which considers both precision and recall, the CNN model with fastText embeddings also yielded the highest f1-score of 94.23%. This indicates a balance between correctly classifying clickbait (recall) and minimizing false positives (precision). The high f1-score further underscores the strong performance of the CNN model with fastText embeddings in accurately detecting clickbait content in Amharic.

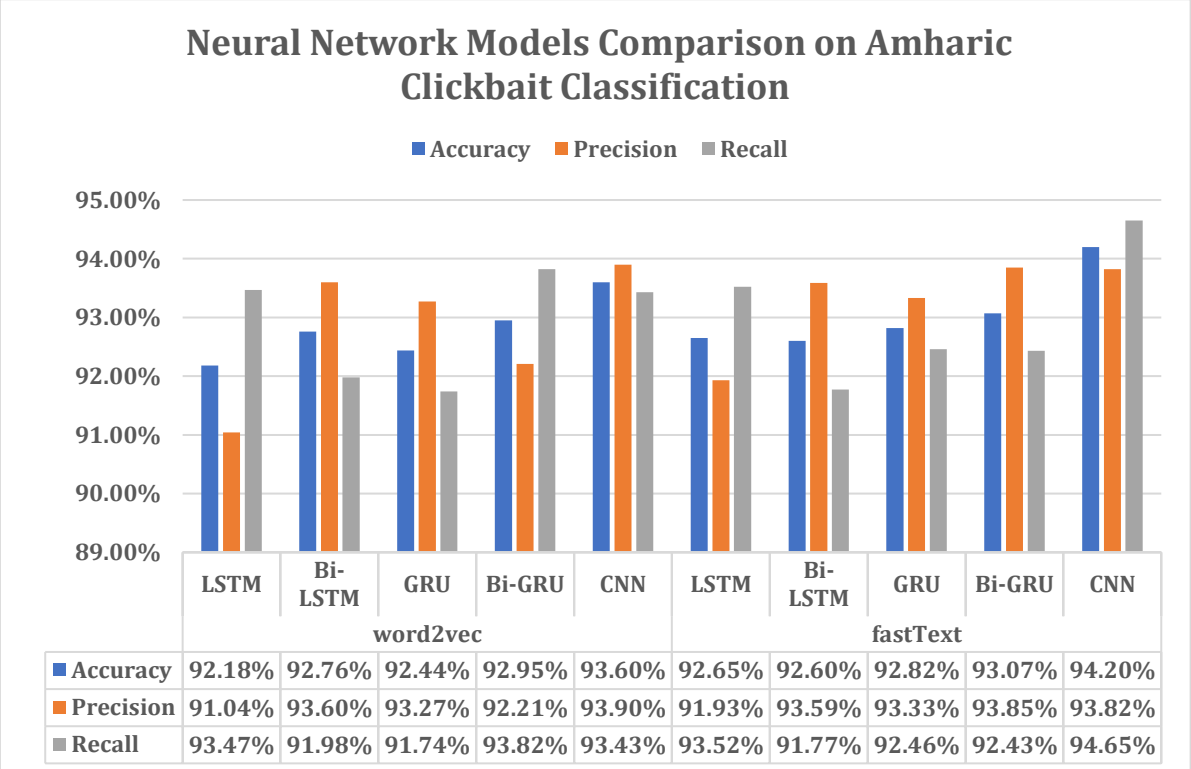
Comparing the precision values among the neural network models, the Bi-GRU model using fastText embeddings achieved the highest precision score of 93.85%. This implies that it had the lowest number of false positives, effectively distinguishing non-clickbait text from clickbait ones as it had similarly been observed in the work of Dong et al., (2019).

Comparing the results of the LSTM and Bi-LSTM models, it is evident that the Bi-LSTM model performs slightly better, achieving higher accuracy and f1-score. The bidirectional nature of the Bi-LSTM allows it to consider both preceding and succeeding words, enabling a more comprehensive understanding of the textual context.

When evaluating the other neural network models, such as LSTM, Bi-LSTM, and GRU, both word2vec and fastText embeddings yielded relatively similar accuracy and f1-score results. However, it is worth noting that the CNN model consistently outperformed these models in terms of accuracy and f1-score, showcasing its ability to capture and leverage contextual information effectively.

The results highlight the impact of the choice of embedding technique on the performance of neural network models. FastText embeddings, which capture subword information, appear to be more effective in capturing the nuances of Amharic language, resulting in higher accuracy and f1-score compared to word2vec embeddings.

In summary, the experimental results indicate that the CNN model with fastText embeddings achieved the highest accuracy and f1-score among the neural network models, demonstrating its effectiveness in Amharic clickbait detection. The Bi-GRU model showed the highest precision, emphasizing its ability to minimize false positives.



	word2vec					fastText				
	LSTM	Bi-LSTM	GRU	Bi-GRU	CNN	LSTM	Bi-LSTM	GRU	Bi-GRU	CNN
Accuracy	92.18%	92.76%	92.44%	92.95%	93.60%	92.65%	92.60%	92.82%	93.07%	94.20%
Precision	91.04%	93.60%	93.27%	92.21%	93.90%	91.93%	93.59%	93.33%	93.85%	93.82%
Recall	93.47%	91.98%	91.74%	93.82%	93.43%	93.52%	91.77%	92.46%	92.43%	94.65%

Figure 5.13 Comparison of Results by ANN on Amharic Clickbait Classification

5.3. Hyperparameter Tuning Results

As a binary classification task, it is practical to choose Sigmoid function as an output layer activation function. Tanh and sigmoid activation functions are better suited for recurrent layers like LSTM because they have a bounded output range (-1 to 1 for tanh and 0 to 1 for sigmoid). On the other hand, ReLU activation function is commonly used in CNNs because it introduces non-linearity and helps in capturing complex patterns in the vector data. ReLU does not suffer from the vanishing gradient problem and can provide faster convergence during training (Pandey & Kaur, 2018). The results shown in Table 5.3 gives the accuracy when the hyperparameter tuning of different activation functions with different epoch is performed at the hidden layers of LSTM and CNN models.

Table 5.3 Hyperparameter Tuning (Activation Function & Epoch) Results

Activation Function	Model	Epochs	Accuracy
Sigmoid	LSTM + word2vec	10	91.74%
		20	92.18%
	LSTM + fastText	10	92.17%
		20	92.65%
Tanh	LSTM + word2vec	10	90.87%
		20	91.82%
	LSTM + fastText	10	92.33%
		20	92.69%
ReLU	CNN + word2vec	10	93.28%
		20	93.60%
	CNN + fastText	10	93.86%
		20	**94.20%**

Despite testing different batch sizes, ranging from 16 to 64, the resulting performance metrics remained relatively stable. This suggests that the choice of batch size did not have a substantial impact on the model's ability to learn and make accurate predictions. Generally, it was showing consistency, indicating that the default batch size produced satisfactory results without the need for further adjustment.

We explored tuning the learning rate within the range of 0.001 to 0.5. The results showed that slower learning rates (0.001, 0.002, 0.003) produce better performance, however slower learning rates may require running more epoch cycles. After a certain threshold around 0.01, the performance of the model begins to deteriorate. This performance aligns with our expectations, as high learning rates can cause the weights to diverge and prevent the network from effectively learning and converging to an optimal solution.

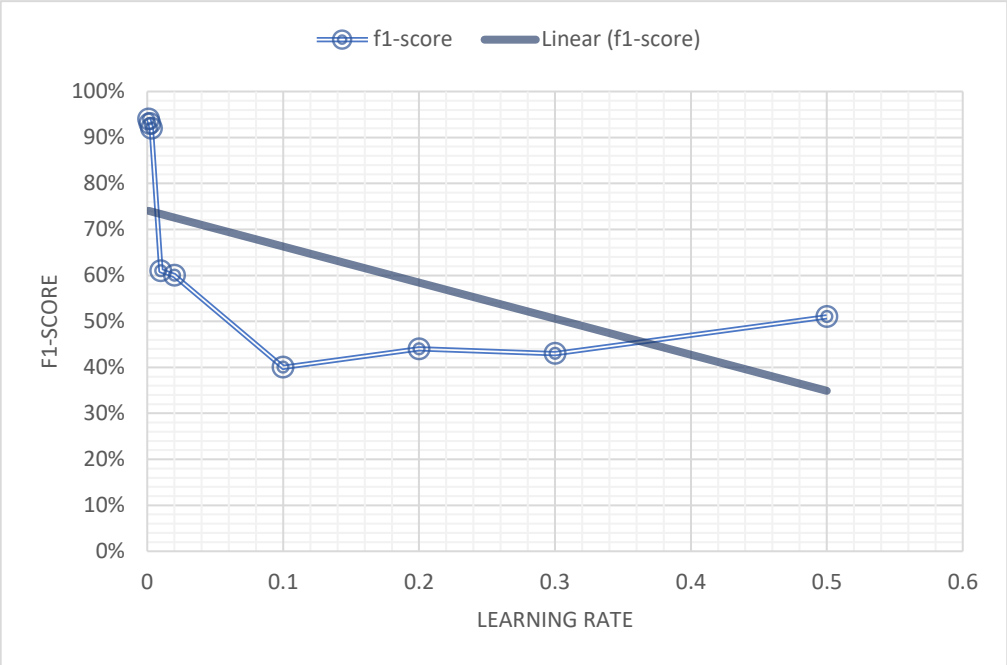


Figure 5.14 Performance (F1-score) Graph with Learning Rate

For our CNN modeling technique, we utilized the different hyperparameter settings to investigate the changes. Specifically, we experimented with static, non-static, and random variations. As word2vec provides a fixed 300 dimension for all vectors, it was ideal for tuning the model setting with word2vec embedding rather than fastText.

Table 5.4 Results of Applying Different CNN Modeling Techniques

Setting	Accuracy	Precision	Recall	F1-score
Static	93.50%	93.28%	93.44%	93.48%
Non-static	**93.60%**	**93.90%**	**93.43%**	**93.66%**
Random	92.19%	92.89%	92.74%	92.22%

The result in Table 5.4 shows that the non-static modeling technique achieves an accuracy of 93.6%. This can be attributed to the feature that non-static modeling enables the network to capture the subtle changes in word meanings and associations that may be specific to the clickbait (Kaothanthong et al., 2021). By updating the word embeddings during training, the model can better capture the evolving semantics and linguistic nuances present in the Amharic clickbait dataset.

The other two CNN hyperparameters we tweaked were window size and dimension of the word embedding vector. The window size, which determines the number of surrounding words considered as an n-gram, was explored using values of 2, 3, and 4. This allowed for an investigation into the impact of different context sizes on the performance of the models.

Additionally, the dimensions of the fastText word vectors were examined, with options of 100 and 300 dimensions, corresponding to the available pretrained models. word2vec only provides the 300 dimension in a pretrained format, which has been experimented on earlier.

Table 5.5 Results of Applying Different Window Size and Vector Dimension

fastText vector dimension	Window size	Accuracy	F1-score
100	2	92.85%	92.82%
	2,3	93.08%	93.11%
	2,3,4	92.27%	92.23%
300	2	94.20%	94.23%
	2,3	**94.27%**	**94.24%**
	2,3,4	93.12%	93.07%

In our experiment to find a suitable window size and vector dimension for the word embedding, the numbers of n-gram are {2,3,4} and vector dimensions 100 and 300 were applied. The experimental result can be found in Table 5.5. The result shows that the most appropriate window size is n = {2,3} and the 300-vector dimension which gives 94.27% accuracy, the highest accuracy we achieved with any of the experiments we have done on the Amharic clickbait dataset.

By leveraging the power of Amharic fastText embeddings and optimizing the CNN model through hyperparameter tuning, we were able to achieve outstanding results in clickbait detection. The combination of these factors contributed to the model's high accuracy and F1-score, making it the top-performing approach in the study.

In summary, through the process of hyperparameter tuning, several optimal settings were identified to enhance the performance of the CNN model for Amharic clickbait detection. Specifically, by combining a window size of n={2,3}, along with 300 dimensions of fastText word embedding, an epoch size of 20, and employing the non-static model setting of the CNN model with ReLU as the activation function, the highest accuracy of 94.27% and the highest f1-score of 94.24% were achieved.

CHAPTER SIX

6. CONCLUSION AND FUTURE WORK

6.1. Conclusion

This research study addressed the prevalent issue of clickbait in the online domain. This research marks the first exploration of clickbait classification and detection for Amharic language, contributing to the limited literature available in this area. To facilitate this study, the first Amharic Clickbait Dataset was developed and curated, which included data collected from popular platforms such as YouTube, Twitter, and Facebook.

The research encompassed several essential steps to prepare the dataset for analysis. These steps included rigorous preprocessing techniques such as text cleaning, normalization, stemming, stopword removal, and tokenization. Furthermore, exploratory data analysis was conducted to gain insights into the lexical nuances and characteristics of the clickbait data. The study primarily focused on a content-based approach, considering textual components like teaser messages, titles, and headlines.

To establish a baseline for performance comparison, traditional machine learning models such as Logistic Regression (LR), Support Vector Machines (SVM), Random Forest (RF), and XGBoost were applied to the newly procured dataset. Among these models, XGBoost demonstrated well, closely followed by Random Forest using unigram feature with TF-IDF.

The study proposed and implemented neural network models including Long Short-Term Memory (LSTM), Bidirectional LSTM (Bi-LSTM), Gated Recurrent Units (GRU), Bidirectional GRU (Bi-GRU), and Convolutional Neural Network (CNN). These models incorporated word sequence and word-level embeddings using Amharic word2vec and fastText. The experiments were conducted using 5-fold cross-validation, and after thorough hyperparameter tuning, the CNN model emerged as the top-performing model. Notably, when employing fastText embedding, the CNN model achieved an accuracy of 94.27% and an F1-score of 94.24%.

The findings highlight the superior performance of the CNN model with fastText embedding, providing a promising solution for clickbait detection in Amharic. This research lays the foundation for future studies in the field and contributes to the growing body of knowledge on clickbait classification and detection for low-resourced languages.

6.2. Recommendation

Based on the findings and insights gained from the study, we would like to put forth the following recommendations. It is recommended to focus on lower n-grams because they have demonstrated better performance. The utilization of fastText word embedding is highly recommended, as it consistently outperformed word2vec embedding across various evaluation metrics.

Employing social media-based annotation tools can greatly enhance the quality and efficiency of dataset annotation, enabling more accurate and comprehensive clickbait detection models. To improve the preprocessing phase, we suggest gathering a more extensive and tailored stopwords vocabulary specifically for the Amharic language.

Considering the prevalence of clickbait content on YouTube, we recommend prioritizing it as a major source for future clickbait research in the Amharic language. Moreover, stemming techniques have proven to yield significantly better performance, thus incorporating stemming processes into the preprocessing pipeline is advised.

It is also essential to invest in the development of curated and annotated corpora for Amharic. These resources can serve as valuable references for future research endeavors and contribute to the advancement of NLP research in the Amharic language.

6.3. Future Work

Several areas are open for exploration to further enhance the study and related areas. Incorporating attention models within deep learning algorithms can potentially improve the performance of the detection system, especially when combined with a more comprehensive dataset. Considering social-context and user-based features, along with engagement metrics, can provide valuable insights for clickbait detection. Exploring multimodal clickbait detection in images such as analyzing thumbnails, audio, and video, or domain-specific approach can also expand the scope of the research. It would be beneficial to train custom embedding models tailored to the Amharic language and explore the use of pre-trained models like BERT for a more advanced approach. Developing user-interactive components such as browser extensions can facilitate the identification and filtering of clickbait content. Extending the study of clickbait classification in multilingual sense which considers other local languages can contribute to a more comprehensive understanding of clickbait in diverse linguistic contexts.

SPECIAL ACKNOWLEDGMENT

This research project was funded by Adama Science and Technology University under the grant number - **ASTU/SM-R/851/23**

REFERENCES

Abebaw, Z., Rauber, A., & Atnafu, S. (2022). Multi-channel Convolutional Neural Network for Hate Speech Detection in Social Media. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, *411 LNICST*, 603–618. https://doi.org/10.1007/978-3-030-93709-6_41

Adelson, P., Arora, S., & Hara, J. (2017). *Clickbait; Didn't Read: Clickbait Detection using Parallel Neural Networks*. http://cs229.stanford.edu/proj2017/final-reports/5231575.pdf

Agrawal, A. (2017). Clickbait detection using deep learning. *Proceedings on 2016 2nd International Conference on Next Generation Computing Technologies, NGCT 2016*, 268–272. https://doi.org/10.1109/NGCT.2016.7877426

Al-Sarem, M., Saeed, F., Al-Mekhlafi, Z. G., Mohammed, B. A., Hadwan, M., Al-Hadhrami, T., Alshammari, M. T., Alreshidi, A., & Alshammari, T. S. (2021). An Improved Multiple Features and Machine Learning-Based Approach for Detecting Clickbait News on Social Networks. *Applied Sciences 2021, Vol. 11, Page 9487*, *11*(20), 9487. https://doi.org/10.3390/APP11209487

Anand, A., Chakraborty, T., & Park, N. (2017). We used neural networks to detect clickbaits: You won't believe what happened next! *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10193 LNCS*, 541–547. https://doi.org/10.1007/978-3-319-56608-5_46/COVER

Asker, L., Argaw, A., Gambäck, B., & Sahlgren, M. (2014). Applying machine learning to Amharic text classification. *Proceedings of the 5th World Congress of African Linguistics.* https://www.academia.edu/download/31053749/10.1.1.97.8241.pdf

Awol, I. N., & Gashaw, S. M. (2022). Lexicon-Stance Based Amharic Fake News Detection. *Researchgate.Net*. https://www.researchgate.net/profile/Ibrahim-Awol/publication/369203279_Lexicon-Stance_Based_Amharic_Fake_News_Detection/links/64105d84a1b72772e4f9308a/Lexicon-Stance-Based-Amharic-Fake-News-Detection.pdf

Ayele, A. A., Dinter, S., Belay, T. D., Asfaw, T. T., Yimam, S. M., & Biemann, C. (2022). The 5Js in Ethiopia: Amharic Hate Speech Data Annotation Using Toloka Crowdsourcing Platform. *2022 International Conference on Information and Communication Technology for Development for Africa (ICT4DA)*, 114–120.

Biyani, P., Tsioutsiouliklis, K., & Blackmer, J. (2016). "8 Amazing Secrets for Getting More Clicks": Detecting Clickbaits in News Streams Using Article Informality. *Thirtieth AAAI Conference on Artificial Intelligence*. https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11807

Bronakowski, M., Al-khassaweneh, M., & Al Bataineh, A. (2023). Automatic Detection of Clickbait Headlines Using Semantic Analysis and Machine Learning Techniques. *Applied Sciences.*, *13*(4), 2456.

Cao, X., Le, T., Jiasheng, J. (, Zhang,), & Lee, D. (2017). *Machine Learning Based Detection of Clickbait Posts in Social Media*. https://arxiv.org/abs/1710.01977v1

Chakraborty, A., Paranjape, B., Kakarla, S., & Ganguly, N. (2016). Stop Clickbait: Detecting and preventing clickbaits in online news media. *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2016*, 9–16. https://doi.org/10.1109/ASONAM.2016.7752207

Chawda, S., Patil, A., Singh, A., & Save, A. (2019). A novel approach for clickbait detection. *Proceedings of the International Conference on Trends in Electronics and Informatics, ICOEI 2019*, 1318–1321. https://doi.org/10.1109/ICOEI.2019.8862781

Chen, Y., Conroy, N. J., & Rubin, V. L. (2015). Misleading online content: Recognizing clickbait as "false news." *WMDD 2015 - Proceedings of the ACM Workshop on Multimodal Deception Detection, Co-Located with ICMI 2015*, 15–19. https://doi.org/10.1145/2823465.2823467

Demlew, G. (2021). Amharic semantic parser using deep learning. *Proceeding of the 2nd Deep Learning Indaba-X Ethiopia Conference.*

Dimpas, P. K., Po, R. V., & Sabellano, M. J. (2018). Filipino and english clickbait detection using a long short term memory recurrent neural network. *Proceedings of the 2017 International Conference on Asian Language Processing, IALP 2017*, *2018-January*, 276–280. https://doi.org/10.1109/IALP.2017.8300597

Dong, M., Yao, L., Wang, X., Benatallah, B., & Huang, C. (2019). Similarity-aware deep attentive model for clickbait detection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11440 LNAI*, 56–69. https://doi.org/10.1007/978-3-030-16145-3_5

Eshetu, A., Teshome, G., & Abebe, T. (2020). Learning Word and Sub-word Vectors for Amharic (Less Resourced Language). *International Journal of Advanced Engineering Research and Science (IJAERS)*, *7*(8), 358–366. https://doi.org/10.22161/ijaers.78.39

Fakhruzzaman, M. N., & Gunawan, S. W. (2021). *Web-based Application for Detecting Indonesian Clickbait Headlines using IndoBERT*. https://doi.org/10.48550/arxiv.2102.10601

Fu, J., Liang, L., Zhou, X., & Zheng, J. (2017). A Convolutional Neural Network for Clickbait Detection. *2017 4th International Conference on Information Science and Control Engineering (ICISCE)*, 6–10. https://doi.org/10.1109/ICISCE.2017.11

Gambäck, B., Olsson, F., Argaw, A., & Asker, L. (2009). Methods for Amharic part-of-speech tagging. *First Workshop on Language Technologies for African Languages*. https://www.diva-portal.org/smash/record.jsf?pid=diva2:1042595

Geçkil, A., Müngen, A. A., Gündogan, E., & Kaya, M. (2018). A clickbait detection method on news sites. *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 932–937.

Genç, Ş., & Surer, E. (2021). ClickbaitTR: Dataset for clickbait detection from Turkish news sites and social media with a comparative analysis via machine learning algorithms. https://doi.org/10.1177/01655515211007746

Gereme, F., Zhu, W., Ayall, T., & Alemu, D. (2021). Combating Fake News in "Low-Resource" Languages: Amharic Fake News Detection Accompanied by Resource Crafting. *Information 2021, Vol. 12, Page 20*, *12*(1), 20. https://doi.org/10.3390/INFO12010020

Hailemichael, E. N. (2021). Fake news detection for amharic language using deep learning. *Academia.Edu*. https://www.academia.edu/download/84664801/ERMIAS_20NIGATU.pdf

Indurthi, V., & Oota, S. R. (2017). *Clickbait detection using word embeddings*. http://arxiv.org/abs/1710.02861

Kaothanthong, N., Kongyoung, S., & Theeramunkong, T. (2021). Headline2Vec: A CNN-based Feature for Thai Clickbait Headlines Classification. *INTERNATIONAL SCIENTIFIC JOURNAL OF ENGINEERING AND TECHNOLOGY (ISJET)*, *5*(1), 20–31. https://doi.org/10.1016/J.INS.2019.05.035

Kelemework, W. (2013). Automatic Amharic text news classification: A neural networks approach. *Ethiopian Journal of Science and Technology*, *6*(2), 127–137. https://www.ajol.info/index.php/ejst/article/view/117217

Khater, S. R., Al-Sahlee, O. H., Daoud, D. M., & El-Seoud, M. S. A. (2018). Clickbait detection. *ACM International Conference Proceeding Series*, 111–115. https://doi.org/10.1145/3220267.3220287

Klairith, P., & Tanachutiwat, S. (2018). Thai clickbait detection algorithms using natural language processing with machine learning techniques. *ICEAST 2018 - 4th International Conference on Engineering, Applied Sciences and Technology: Exploring Innovative Solutions for Smart Society*. https://doi.org/10.1109/ICEAST.2018.8434447

Kotu, V., & Deshpande, B. (2018). *Data science: concepts and practice*. Morgan Kaufmann.

Kumar, V., Khattar, D., Gairola, S., Kumar Lal, Y., & Varma, V. (2018). Identifying clickbait: A multi-strategy approach using neural networks. *41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018*, 1225–1228. https://doi.org/10.1145/3209978.3210144

Li, Y., Xu, L., Tian, F., Jiang, L., & Zhong, X. (2015). Word embedding revisited: A new representation learning and explicit matrix factorization perspective. *Ijcai.Org*. https://www.ijcai.org/Proceedings/15/Papers/513.pdf

Loewenstein, G. (1994). The psychology of curiosity: A review and reinterpretation. *Psychological Bulletin*, *116*(1), 75–98. https://doi.org/10.1037/0033-2909.116.1.75

Manjesh, S., Kanakagiri, T., Vaishak, P., Chettiar, V., & Shobha, G. (2017). Clickbait Pattern Detection and Classification of News Headlines Using Natural Language Processing. *2nd International Conference on Computational Systems and Information*

Technology for Sustainable Solutions, CSITSS 2017, 1–5. https://doi.org/10.1109/CSITSS.2017.8447715

Marreddy, M., Oota, S. R., Vakada, L. S., Chinni, V. C., & Mamidi, R. (2021). Clickbait Detection in Telugu: Overcoming NLP Challenges in Resource-Poor Languages using Benchmarked Techniques. *Proceedings of the International Joint Conference on Neural Networks, 2021-July*. https://doi.org/10.1109/IJCNN52387.2021.9534382

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26. https://proceedings.neurips.cc/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html

Minale, S., & Assabie, Y. (2021). Hate Speech Detection and Classification System in Amharic Text with Deep Learning. *Unpublished, MSc Thesis Submitted to Addis Ababa University*.

Mossie, Z., & Wang, J.-H. (2018). Social network hate speech detection for Amharic language. *Computer Science & Information Technology*, 41–55.

Munger, K., Luca, M., … J. N.-… /uploads/2018/09/munger, & 2018, undefined. (2018). The effect of clickbait. *Kmunger.Github.Io*. http://kmunger.github.io/pdfs/clickbait.pdf

Naeem, B., Beg, M., Mujtaba, H., Khan, A., Mirza, ·, & Beg, O. (2020). A deep learning framework for clickbait detection on social area network using natural language cues. *Springer*, *3*(1), 231–243. https://doi.org/10.1007/s42001-020-00063-y

Pandey, S., & Kaur, G. (2018). Curious to click it?-Identifying clickbait using deep learning and evolutionary algorithm. *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE.*, 1481–1487.

Potthast, M., Gollub, T., Hagen, M., & Stein, B. (2018). *The Clickbait Challenge 2017: Towards a Regression Model for Clickbait Strength*. http://arxiv.org/abs/1812.10847

Potthast, M., Köpsel, S., Stein, B., & Hagen, M. (2016). Clickbait Detection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9626*, 810–817. https://doi.org/10.1007/978-3-319-30671-1_72

Pujahari, A., & Sisodia, D. S. (2021). Clickbait detection using multiple categorisation techniques. *Journal of Information Science*, *47*(1), 118–128. https://doi.org/10.1177/0165551519871822

Shu, K., Wang, S., Le, T., Lee, D., & Liu, H. (2018). Deep Headline Generation for Clickbait Detection. *Proceedings - IEEE International Conference on Data Mining, ICDM*, *2018-November*, 467–476. https://doi.org/10.1109/ICDM.2018.00062

Siregar, B., Habibie, I., Nababan, E. B., & Fahmi. (2021). Identification of Indonesian clickbait news headlines with long short-term memory recurrent neural network algorithm. *Journal of Physics: Conference Series*, *1882*(1), 012129. https://doi.org/10.1088/1742-6596/1882/1/012129

Tazeze, T., & R, R. (2021). Building a Dataset for Detecting Fake News in Amharic Language. *International Journal of Advanced Research in Science, Communication and Technology*, 76–83. https://doi.org/10.48175/IJARSCT-1362

Tompson, J., Goroshin, R., Jain, A., LeCun, Y., & Bregler, C. (2015). Efficient object localization using convolutional networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 648–656. http://openaccess.thecvf.com/content_cvpr_2015/html/Tompson_Efficient_Object_Localization_2015_CVPR_paper.html

Varshney, D., & Vishwakarma, D. K. (2021). A unified approach for detection of Clickbait videos on YouTube using cognitive evidences. *Applied Intelligence*, *51*(7), 4214–4235. https://doi.org/10.1007/S10489-020-02057-9/FIGURES/12

Vijgen, B. (2014). THE LISTICLE: AN EXPLORING RESEARCH ON AN INTERESTING SHAREABLE NEW MEDIA PHENOMENON. *Studia Universitatis Babes-Bolyai - Ephemerides*, *59*(1), 103–122.

Vincze, V., & Szabó, M. K. (2020). *Automatic detection of hungarian clickbait and entertaining fake news*. 58–69. http://real.mtak.hu/135208/1/2020.rdsm-1.6.pdf

Vorakitphan, V., Leu, F. Y., & Fan, Y. C. (2019). Clickbait detection based on word embedding models. *Advances in Intelligent Systems and Computing*, *773*, 557–564. https://doi.org/10.1007/978-3-319-93554-6_54

Wanda, J. F., Chipanjilo, B. S., Gondwe, G., & Kerunga, J. (2021). Clickbait-style headlines and journalism credibility in Sub-Saharan Africa: Exploring audience perceptions. *Journal of Media and Communication Studies*, *13*(2), 50–56. https://doi.org/10.5897/JMCS2020.0715

William, A., & Sari, Y. (2020). CLICK-ID: A novel dataset for Indonesian clickbait headlines. *Data in Brief*, *32*, 106231. https://doi.org/10.1016/J.DIB.2020.106231

Wu, C., Wu, F., Qi, T., & Huang, Y. (2020). Clickbait Detection with Style-Aware Title Modeling and Co-attention. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *12522 LNAI*, 430–443. https://doi.org/10.1007/978-3-030-63031-7_31

Yimam, S. M., Alemayehu, H. M., Ayele, A. A., & Biemann, C. (2020). *Exploring Amharic Sentiment Analysis from Social Media Texts: Building Annotation Tools and Classification Models*. 1048–1060. https://doi.org/10.18653/V1/2020.COLING-MAIN.91

Yimam, S. M., Ayele, A. A., Venkatesh, G., Gashaw, I., & Biemann, C. (2021). Introducing Various Semantic Models for Amharic: Experimentation and Evaluation with Multiple Tasks and Datasets. *Future Internet.*, *13*(11), 275.

Zannettou, S., Sirivianos, M., Blackburn, J., & Kourtellis, N. (2019). The Web of False Information. *Journal of Data and Information Quality (JDIQ)*, *11*(3). https://doi.org/10.1145/3309699

Zhang, C., & Clough, P. D. (2020). Investigating clickbait in Chinese social media: A study of WeChat. *Online Social Networks and Media*, *19*, 100095. https://doi.org/10.1016/J.OSNEM.2020.100095

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level Convolutional Networks for Text Classification. *Advances in Neural Information Processing Systems*, *28*, 648–657.

Zheng, H. T., Chen, J. Y., Yao, X., Sangaiah, A. K., Jiang, Y., & Zhao, C. Z. (2018). Clickbait Convolutional Neural Network. *Symmetry 2018, Vol. 10, Page 138*, *10*(5), 138. https://doi.org/10.3390/SYM10050138

Zheng, H. T., Yao, X., Jiang, Y., Xia, S. T., & Xiao, X. (2017). Boost clickbait detection based on user behavior analysis. *Lecture Notes in Computer Science (Including*

Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10367 LNCS, 73–80. https://doi.org/10.1007/978-3-319-63564-4_6/COVER

Zitouni, I. (2014). *Natural language processing of semitic languages*. Springer. https://link.springer.com/content/pdf/10.1007/978-3-642-45358-8.pdf

Appendix A - Annotation Guideline: Clickbait Detection

Please carefully review the given text content and assess whether it exhibits characteristics of clickbait or not. Consider the following factors and indicators when making your judgment:

1. Emotional Reaction: [ ] Clickbait [ ] Non-clickbait
- Does the content evoke strong emotions such as curiosity, fear, anger, or excitement?
- Is it designed to trigger an emotional response from the reader?

2. Promotional Nature: [ ] Clickbait [ ] Non-clickbait
- Does the content aim to promote a product, service, or agenda?
- Is there an obvious intention to sell or advertise something?

3. Excessive Punctuation: [ ] Clickbait [ ] Non-clickbait
- Are there excessive exclamation marks, question marks, or ellipses in the headline?
- Does the text use punctuation to grab attention or create suspense unnecessarily?

4. Secretive/Sensational Language: [ ] Clickbait [ ] Non-clickbait
- Does the content use phrase like "You won't believe," "This secret," or "shocking"?
- Does it create a sense of mystery or intrigue without providing substantial information?

5. Baiting: [ ] Clickbait [ ] Non-clickbait
- Does the content promise something irresistible or extraordinary?
- Does it tempt the reader with vague or exaggerated claims?

6. Source: [ ] Clickbait [ ] Non-clickbait
- Is the content from a reputable and reliable source?
- Consider the credibility and trustworthiness of the publication or website.

7. Directing to External Website/Content: [ ] Clickbait [ ] Non-clickbait
- Does the content urge the reader to click a link to access more information?
- Is there an explicit intention to redirect the reader to another webpage?

8. Propaganda: [ ] Clickbait [ ] Non-clickbait
- Does the content display a biased or manipulative viewpoint?
- Is there an attempt to influence or shape public opinion?

9. Exaggeration, Shocking tone or Polarizing: [ ] Clickbait [ ] Non-clickbait
- Does the content use hyperbole, alarming or make sensationalized claims?
- Does the headline provoke or use divisive or controversial language?

10. Withholding Information: [ ] Clickbait [ ] Non-clickbait
- Does the content intentionally withhold crucial details or key information?
- Does it create a sense of curiosity by leaving out important facts?

11. Piggybacking: [ ] Clickbait [ ] Non-clickbait
- Does the content rely on the popularity or relevance of a current event or trend?
- Is there an attempt to ride the coattails of something popular for attention?

12. Talking About Money or Sex: [ ] Clickbait [ ] Non-clickbait
- Does the content focus on financial gain or highlight monetary benefits?
- Is there a display of sexual behavior in the headline or text?

If the content demonstrates multiple clickbait characteristics, select "Clickbait." If none of the characteristics are present, select "Non-clickbait."
Thank you for your diligent.

Appendix B – Inter-Annotator Agreement (IAA)

To ensure consistency and reliability in the annotation process for clickbait detection, it is important to establish an Inter-Annotator Agreement (IAA) among the annotators. The IAA measures the level of agreement between different annotators and helps assess the reliability of the annotations. Please follow the guidelines below:

1.Familiarize Yourself: Before beginning the annotation process, carefully review the provided guidelines and instructions for clickbait detection. Understand the characteristics and indicators of clickbait content.
2. Independent Annotation: Each annotator should work independently without consulting or collaborating with other annotators during the annotation process. This helps maintain objectivity and reduces bias.
3. Annotation Methodology: Use the provided checklist as a guide and indicate on your annotation tool for clickbait occurrence in the given text content. Ensure that your annotations accurately reflect the presence or absence of clickbait tendencies.
4. Annotation Consistency: While annotating, focus on the specific characteristics outlined in the guidelines and evaluate the text content accordingly. Aim for consistency in your annotations and apply the guidelines uniformly across all instances.
5. Disputed Cases: In cases where the determination of clickbait is unclear or there is ambiguity, mark the content as "Uncertain" or "Not Determinable." These instances can be further reviewed or resolved through discussions with the research team.
6. Regular Discussions: It is advisable to have regular discussions and meetings among annotators and the research team to address any questions, concerns, or doubts that may arise during the annotation process.
7. Calculating Inter-Annotator Agreement: Once the annotations are completed, the Inter-Annotator Agreement can be calculated to measure the level of agreement among annotators. This can be done using established metrics such as Cohen's or Fleiss' Kappa.
8. Resolving Discrepancies: In cases where there are significant disagreements between annotators, the research team can review and discuss those instances to arrive at a consensus or resolve discrepancies. This collaborative approach ensures the highest possible agreement and reliability in the annotations.

By adhering to these guidelines and maintaining consistency throughout the annotation process, we can achieve reliable and accurate clickbait annotations. The Inter-Annotator Agreement helps assess the quality of the annotations and contributes to the overall validity of the research findings.

Appendix C – Additional Results and Illustrations

a) CNN Model Parameters and Settings

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 28, 300)]         0

 transformer (Transformer)   (None, 28, 300)           1340824

 flatten (Flatten)           (None, 8400)              0

 dense (Dense)               (None, 512)               4301312

 dropout_1 (Dropout)         (None, 512)               0

 dense_1 (Dense)             (None, 256)               131328

 dropout_2 (Dropout)         (None, 256)               0

 dense_2 (Dense)             (None, 1)                 257

=================================================================
Total params: 5,773,721
Trainable params: 5,773,721
Non-trainable params: 0
_________________________________________________________________
```

b) LSTM Training History with 5 epochs

```
1 history = model.fit(train_data, y_train, batch_size = 16, epochs = 5, validation_data = (val_data , y_val), verbose = 1)
```

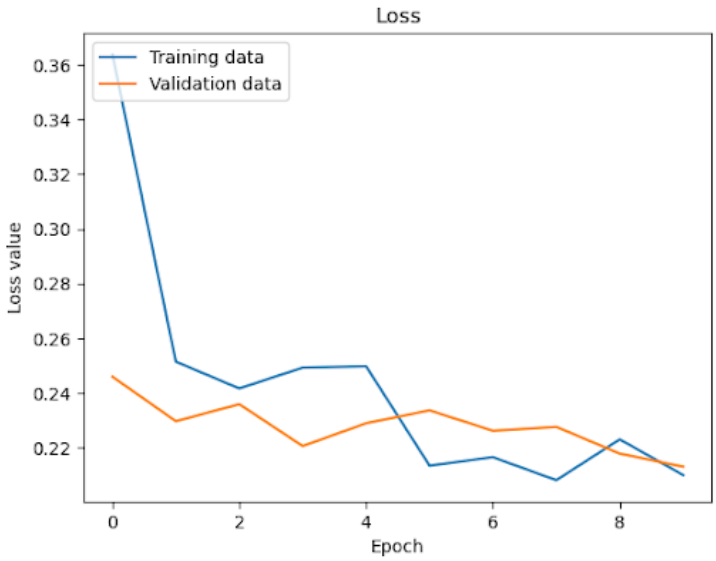
```
Epoch 1/5
2517/2517 [==============================] - 177s 69ms/step - loss: 0.2977 - accuracy: 0.9151 - val_loss: 0.2598 - val_accurac
y: 0.2295
Epoch 2/5
2517/2517 [==============================] - 174s 69ms/step - loss: 0.2636 - accuracy: 0.9216 - val_loss: 0.2595 - val_accurac
y: 0.2295
Epoch 3/5
2517/2517 [==============================] - 149s 59ms/step - loss: 0.2637 - accuracy: 0.9216 - val_loss: 0.2593 - val_accurac
y: 0.2295
Epoch 4/5
2517/2517 [==============================] - 148s 59ms/step - loss: 0.2636 - accuracy: 0.9216 - val_loss: 0.2592 - val_accurac
y: 0.2295
Epoch 5/5
2517/2517 [==============================] - 147s 58ms/step - loss: 0.2636 - accuracy: 0.9216 - val_loss: 0.2593 - val_accurac
y: 0.2295
```

c) LSTM Training History with 10 epochs

```
history = model.fit(train_data, y_train, batch_size = 64, epochs = 10, validation_data = (val_data , y_val), verbose
```

```
Epoch 1/10
630/630 [==============================] - 155s 238ms/step - loss: 0.2490 - accuracy: 0.9299 - val_loss: 0.2434 - v
al_accuracy: 0.9395
Epoch 2/10
630/630 [==============================] - 153s 242ms/step - loss: 0.2454 - accuracy: 0.9261 - val_loss: 0.2323 - v
al_accuracy: 0.9467
Epoch 3/10
630/630 [==============================] - 146s 232ms/step - loss: 0.2359 - accuracy: 0.9292 - val_loss: 0.2025 - v
al_accuracy: 0.9503
Epoch 4/10
630/630 [==============================] - 157s 249ms/step - loss: 0.2206 - accuracy: 0.9257 - val_loss: 0.2048 - v
al_accuracy: 0.9428
Epoch 5/10
630/630 [==============================] - 156s 248ms/step - loss: 0.2189 - accuracy: 0.9266 - val_loss: 0.2189 - v
al_accuracy: 0.9509
Epoch 6/10
630/630 [==============================] - 152s 242ms/step - loss: 0.2199 - accuracy: 0.9256 - val_loss: 0.2011 - v
al_accuracy: 0.9574
Epoch 7/10
630/630 [==============================] - 151s 239ms/step - loss: 0.2117 - accuracy: 0.9298 - val_loss: 0.2094 - v
al_accuracy: 0.9522
Epoch 8/10
630/630 [==============================] - 145s 230ms/step - loss: 0.2105 - accuracy: 0.9209 - val_loss: 0.2017 - v
al_accuracy: 0.9581
Epoch 9/10
630/630 [==============================] - 145s 230ms/step - loss: 0.2013 - accuracy: 0.9253 - val_loss: 0.2109 - v
al_accuracy: 0.9493
Epoch 10/10
630/630 [==============================] - 146s 231ms/step - loss: 0.2011 - accuracy: 0.9277 - val_loss: 0.2033 - v
al_accuracy: 0.9415
```

d) Training-Validation Loss Graph for CNN



e) Testing trained CNN model with new instances

```
In [43]: def TestingOwnData(sentence) :
             X = np.zeros((1,sequence_size,300))
             sentence = sentence.replace('-',' ')
             words = nltk.word_tokenize(sentence)
             j = 0
             for w in words:
                 try :
                     X[0,j] = w2v[w]
                     j += 1
                 except :
                     pass
             prediction = predict_function(model.predict(X))[0][0]
             if prediction == 1 :
                 return 'Clickbait'
             return 'Not A Clickbait'

In [44]: TestingOwnData('ይህ በጣም አስገራሚ እድል እንዳያመልጥዎ')
         1/1 [==============================] - 0s 24ms/step
Out[44]: 'Clickbait'

In [45]: TestingOwnData('የኩተማው ጸጥታ የተረጋገጠ ነው')
         1/1 [==============================] - 0s 19ms/step
Out[45]: 'Not A Clickbait'

In [46]: TestingOwnData('የወልድያ ኩተማ ሰላማዊ እንቅስቃሴ')
         1/1 [==============================] - 0s 19ms/step
Out[46]: 'Not A Clickbait'

In [47]: TestingOwnData('ጉድ!አዲሰ አስቂኝ ፊልም ሊለቀቅ ነው')
         1/1 [==============================] - 0s 19ms/step
Out[47]: 'Clickbait'
```