2022-07

IDIOMATIC EXPRESSION IDENTIFICATION FROM AMHARIC TEXTS USING DEEP LEARNING

TIRUEDLE, ASTERAYE TSIGE

BAHIR DAR UNIVERSITY

BAHIR DAR INSTITUTE OF TECHNOLOGY

SCHOOL OF GRADUATE STUDIES

FACULTY OF COMPUTING

MSc. Thesis On:

IDIOMATIC EXPRESSION IDENTIFICATION FROM AMHARIC TEXTS USING DEEP LEARNING

BY

TIRUEDLE ASTERAYE TSIGE

JULY 2022

BAHIR DAR, ETHIOPIA

BAHIR DAR UNIVERSITY

BAHIR DAR INSTITUTE OF TECHNOLOGY

SCHOOL OF GRADUATE STUDIES

FACULTY OF COMPUTING

IDIOMATIC EXPRESSION IDENTIFICATION FROM AMHARIC TEXTS USING DEEP LEARNING

BY

TIRUEDLE ASTERAYE TSIGE

A Thesis submitted to the school of Research and Graduate Studies of Bahir Dar Institute of Technology, in partial Fulfillment for the Degree of Master of Science in Information Technology in the Faculty of Computing.

Advisor: Tesfa Tegegne (PhD)

JULY 2022

BAHIR DAR, ETHIOPIA

Declaration

This is to certify that the thesis entitled **"Idiomatic Expression Identification from Amharic Texts Using Deep Learning"**, submitted in partial fulfillment of the requirements for the degree of Master of Science in Information Technology under Faculty of Computing, Bahir Dar Institute of Technology is a record of original work carried out by me and has never been submitted to this or any other institution to get any other degree or certificates. The assistance and help I received during the course of this investigation have been duly acknowledged.

Name of the Candidate	Signature	Date
Tiruedle Asteraye		17/02/2015

BAHIR DAR UNIVERSITY
BAHIR DAR INSTITUTE OF TECHNOLOGY
SCHOOL OF GRADUATE STUDIES
FACULTY OF COMPUTING

Approval of thesis for defense result

I hereby confirm that the changes required by the examiners have been carried out and incorporated in the final thesis.

Name of Student: Tiruedle Asteraye Tsige Signature: _____ Date: 20/08/2022

As members of the board of examiners, we examined this thesis entitled "**Idiomatic Expression Identification from Amharic texts using Deep Learning**. We hereby certify that the thesis is accepted for fulfilling the requirements for the award of the degree of Masters of Science in "Information Technology".

Board of Examiners

Name of Advisor Tesfa Tegegne (PhD)	Signature	Date 28/10/22
Name of External examiner Martha Yifiru (PhD)	Signature	Date 18/10/2022
Name of Internal Examiner Mekonnen Wagaw (PhD)	Signature	Date Oct. 27, 2022
Name of Chairperson Tamir Anteneh	Signature	Date Oct. 28, 2022
Name of Chair Holder Abdulkerim M (PhD)	Signature	Date 27. 10. 2022
Name of Faculty Dean Asegahegn E.	Signature	Date Oct. 28, 2022

To My Families

Acknowledgment

First, I would like to thank the Almighty GOD and St. Mary for giving me the strength to write this thesis. I am very grateful to thank Bahir Dar University for giving me the scholarship opportunity.

Next, I would like to express my greatest gratitude to my advisor Tesfa Tegegne (Ph.D.) for his continuous support, motivation, and patience. I am very proud of working under his supervision. Thank you for your unreserved guidance, support, encouragement, and constructive suggestion.

I would like to thank Melese Zegeye (Ph.D.), and Gashaw Arutie (Ph.D.) linguists at Bahir Dar University for their support and professional contribution to the data preparation and all of the linguist parts of my study. I would like to say thank you to the linguistics master's students at Bahir Dar University for their kind support to annotate the dataset.

I also would like to thank my family for their support, unconditional love, and patience. Lastly, I would like to say thanks to my friends and colleagues for their advice and support.

Abbreviations

AI	Artificial Intelligence
BiLSTM	Bidirectional Long Short Term Memory
BNC	British National Corpus
CBOW	Continuous Bag of Words
DL	Deep Learning
IEs	Idiomatic Expressions
KNN	K-Nearest Neighbor
LSTM	Long Short Term Memory
ML	Machine Learning
NB	Naïve Bayesian
NLP	Natural Language Processing
NVC	Noun Verb Construction
POST	Part of Speech Tag
RFC	Random Forest Classifier
SMT	Statistical Machine Translation
SVM	Support Vector Machine
TFIDF	Term Frequency Inverse Document Frequency
VNC	Verb Noun Construction
VNIC	Verb Noun Idiomatic Combination
WSD	Word Sense Disambiguation

Table of Contents

List of Figures

List of Tables

Abstract

Idiomatic expressions are a natural feature of all languages and are used frequently in our daily conversations. Idioms are difficult to comprehend because their meaning cannot be deduced immediately from the phrase from which they are derived. Idiomatic expressions are phrases or clauses that have a common meaning that is unrelated to the individual words' meanings. The existence of idiomatic expression in a text has an impact on different natural language processing studies like Machine translation, sentiment analysis, and semantics analysis are affected by idiom. Because idioms are one of the most important aspects of a language, having an algorithm and mechanism for detecting them is critical for NLP research. Several scholars conduct idiom recognition in different languages however, there is not enough work on Amharic language idiomatic expression identification. Models which are developed for other languages are not effective to process Amharic documents due to morphological, syntactical, and semantic differences. So, to address those gaps we developed a model for identifying the idiomatic expressions from Amharic texts. This study used a deep learning approach to design an idiomatic expression identification model and the corresponding idiomatic meaning for the Amharic language using the Anaconda Jupiter notebook python framework. We have experimented with CNN, LSTM, and BiLSTM deep learning algorithms using 4800 Amharic sentences for idiomatic expression identification. The accuracy of CNN, LSTM, and BiLSTM was 94%, 95%, and 99% respectively. We compared the proposed model with classical machine learning algorithms SVM and KNN. The experimental result shows that the proposed BiLSTM model performs better than SVM, KNN, CNN, and LSTM.

Keywords: Amharic Idiomatic Expression, Natural Language Processing, Deep Learning

CHAPTER ONE

1. Introduction

1.1. Background

Language is the most important tool for individuals to communicate with each other. Individuals can communicate their thoughts, facts, experiences, and knowledge with others using language (Ayu & Dian, 2021). Individuals use idioms to make phrases more beautiful when speaking or writing with one another. The idiom is made up of a unique set of words that cannot be translated word by word. Writers also use idioms to make conversations more fascinating and make the interlocutors more willing to read or listen.

Idiomatic expression is a type of expression that involves the use of metaphorical language. Idiomatic expressions are phrases or clauses that have a common meaning that is unrelated to the individual words' meanings. Idioms cannot be deduced immediately from the word from which they are derived (Owens, 2016).

Idiomatic expressions are a regular part of all languages and a common part of our daily conversation (Hinkel, 2017; Salton et al., 2017). Idioms are one of the most challenging and attractive aspects of the Amharic language. However, they are regarded as one of the most unusual aspects of all languages, on the other hand, they are difficult to understand for nonnative speakers (Hillert, 2004; Sprenger et al., 2006).

Even if we are using idioms in our daily conversation it is important to note that, everyone may not familiar with them because the meaning of an idiom cannot be determined from the meaning of the words which form the idiom (Cacciari & Glucksberg, 1991). An idiom's meaning isn't merely the sum of the individual terms' meanings. For example, the expression **ሆዱ ሰፊ** (“Wide stomach”) is used to express the idiomatic meaning of someone’s (Patience) rather than to express someone's stomach wideness. which cannot predictable by knowing the forming words **ሆድ** (“stomach”) or **ሰፊ** (“Wide”). Natural Language Processing (NLP) is a branch of computer science, particularly Artificial Intelligence (AI), concerned with enabling computers to interpret and process human language. The primary goal of NLP is to program computers to analyze and interpret large amounts of natural language data (Zaid, 2019).

Identifying idioms from literal necessitates a detailed understanding of both the language and the processes used to automate NLP expressions. The presence of idioms has influenced natural language processing studies. Question answering, sentiment analysis, machine translation, and semantics analysis have all been demonstrated to be affected by idioms (Callison-Burch et al., 2011; Farhadloo & Rolland, 2016). The existence of idioms gives bad results by translating the idiomatic expression into literal meaning. Because idioms are one of the most important aspects of a language, having an algorithm and model for detecting them is critical for NLP research. When it comes to the significance of idioms, languages become monotonous without them because words are the foundation of a language (Mantyla, 2004).

The Amharic language is a Semitic language that is mainly spoken in Ethiopia and it is one of the morphologically rich languages (Desalegn, 2015). The language has its grammatical structure which is subject-object-verb while in some special cases it follows the subject-verb-object agreement. Like other languages, the Amharic language is also one research area for many NLP applications (Gereme et al., 2021). Amharic documents can be organized with different idiomatic expressions which are easily understood by linguistics and language experts. While it is difficult to comprehend second language learners and literacy readers unless they get the idiomatic meanings for those expressions and make other NLP applications like machine translation function correctly. The identification of idiomatic expressions from a given text and giving its idiomatic meaning needs investigation, which is a challenging task (Vilar et al., 2006).

The existences of idiomatic expressions like **ሆድ ይፍጀው** (“Let the stomach glide”)፤ **ሁለት ልብ** (“Two Hearts”)፤ **አመድ አፋሽ** (“Ashes mouth”)፤ **ሁለት ምላስ** (“Two tongues”) in the document needs identification and its corresponding idiomatic meaning to improve the performance of different NLP applications (Farhadloo & Rolland, 2016).

Several scholars (Hashimoto et al., 2006; Peng & Feldman, 2016; Salton et al., 2017; Verma & Vuppuluri, 2015) conduct idiom recognition in Japanese and English languages respectively. However, there is not enough work on Amharic language idiomatic

expression identification to minimize the difficulty that happened to the existence of idioms.

Models which are developed for other foreign languages are not effective to process Amharic documents due to morphological, syntactical, and semantic differences. This study is conducted using a deep learning approach to build idiomatic expression identification and its corresponding idiomatic meaning model for the Amharic language.

1.2. Problem Statement

The nature of idioms has an impact on other NLP investigations, such as semantic analysis, statistical machine translation, and sentiment analysis (Yimam et al., 2021). Another property of idioms that makes them challenging to comprehend for NLP systems is that they have both idiomatic and literal (non-idiomatic) uses.

Statistical Machine Translation (SMT) systems, such as Google Translate, are one of the most important NLP applications that are badly affected by idioms (Callison-Burch et al., 2011). The basic SMT word-by-word technique is extended by phrase-based SMT systems (Fellbaum et al., 2006). As a result, these systems are restricted to a direct translation of phrases that are lacking any syntactic or semantic background. In standard phrase-based SMT systems, idioms are not modeled (Bouamor et al., 2011). Unfortunately, there hasn't been much research done using idioms to improve SMT (Popovíc et al., 2006). When an input sentence contains an idiom, it is well known in the Machine Translation field that the performance of SMT systems worsens since they frequently attempt to translate the idiom as “literal text”.

For example : - በአንድ ራስ ሁለት ምላስ (“Two tongues in one head”) ።።

From this sentence, the phrase “**ሁለት ምላስ**” (“Two tongues”) is used to express someone’s truthfulness and falsified/lie. However, when the machine gets this sentence simply it translated to its direct meaning **“two tongues in one head”**. So, the machine translates the given text in the wrong way because the translation model does not recognize idiomatic expressions and their corresponding meaning. As a result, we are motivated to develop an Amharic idiomatic expression identification model. Sentiment analysis is another NLP study that is influenced by idioms. The most frequent text

categorization tool is sentiment analysis, which analyzes an incoming text and determines if the underlying sentiment is positive, negative, or neutral (Ankit & Saleena, 2018; Farhadloo & Rolland, 2016).

Most sentiment analysis works by examining words individually, assigning positive points to positive words and negative points to bad phrases, and then adding these points together. The presence of idioms in a text makes the sentiment analyzer fail to categorize the text into the correct class.

For example: -

1. **የተማሪዎችን ምዝገባ በተገቢው መንገድ ለማካሄድ የሚያስችል ህግ መዘጋጀቱ ተገለፀ** (“Legislation to properly fund student enrollment revealed”) ። **Positive**
2. **አሁን የምንፈራበት ጊዜ አይደለም የምፀልይበት እንጂ** (“Now is not the time to be afraid, but to pray.”) ። **Negative**
3. **የዘንድሮው ፖ`ለቲካ ጉዳይ ሆድ ይፍጀው የሚያስብል ነው** (“The politics of the dragon are the stomach-y”) ። **Positive**

In the above sentence, the phrase “**ሆድ ይፍጀው**” (“Let the stomach glide”) is an idiomatic expression that leads the sentiment analyzer into the incorrect class. Since the presented expression is used to express “**ምንም መናገር አልፈልግም**” / (“Menem Menager alfelegem”) / (“I don’t want to say anything”) which should be categorized into neutral class. However, because of the existence of the idiomatic expression, the analyzer fails to categorize the given sentence into the correct class. So that to overcome such kinds of problems we proposed the Amharic idiomatic expression identification model from Amharic texts.

In the works (Fellbaum et al., 2006; Muzny & Zettlemoyer, 2013; Hashimoto et al., 2006) idiomatic expression recognition has been attempted for the English and Japanese languages respectively. However natural language processing application is language-dependent and it is known that the works which are done in English, Japanese or Chinese cannot be used for the Amharic language since they follow different morphology, semantics, and grammar (Cohen, 2013).

(Fenta, 2021) designed the Amharic idiom classification model at the phrase level by taking both idiom and literal expressions. Their work doesn't incorporate the corresponding meanings of the given idiomatic expressions.

Hence, we are going to develop an idiomatic expression identification model for Amharic texts by preparing a sentence-level dataset with their corresponding meanings to enhance the model's performance. Since the sentence is the base for the paragraph as well as the document dataset.

In this study, the following research questions are explored and answered:

RQ (1) Which DL algorithm (CNN, LSTM, and BiLSTM) gives better performance for the proposed model?

RQ (2) Which preprocessing techniques are useful for the Amharic idiomatic expression identification model?

RQ (3) To what extent can the deep learning approach improve the performance of the proposed model?

1.3. The objective of the study

1.3.1. General Objective

The main objective of this study is to design and develop an idiomatic expression identification model for Amharic texts.

1.3.2. Specific objective

- Conduct a literature review to understand idiomatic expression recognition approaches.
- Prepare a sentence-level dataset for idiomatic expressions in Amharic texts.
- Design and develop the proposed idiomatic expression identification model using Deep Learning.
- To determine the best model for idiomatic expression identification.
- Evaluate the performance of the proposed idiomatic expression identification model using appropriate evaluation metrics.

1.4. Scope of the study

Amharic Idiomatic expressions can be formed at a word, phrase, or clause level. So that this study contains those three forms of idiomatic expressions. From the given text the idiomatic expressions are identified and given their corresponding hidden meaning is done. So for our work, we used sentence-level idiomatic expression identification.

The main challenge in idiomatic expressions is that they can be interpreted both literally and idiomatically. Idiomatic expressions have the nature of pure or semi-pure.

All pure or semi-pure idiomatic expressions will be employed in our thesis. However, the identification of its nature whether it is pure or semi-pure is not considered in this study due to the difficulty of identifying them. This study considers only text files though audio, video, and image files will not employ.

1.5. Significance of the study

Writers (Amsalu & Dagnachew, 1992) stated that idiomatic expressions are regularly used in published Amharic fiction books, and different documents to attract the reader's attention and to maximize the degree of a message to be conveyed. For example: - the idiomatic expression "**ሆዱ ሰፊ**" ("Wide stomach") can be expressed as "**ቻይ**" ("patience") or "**ታጋሽ**" (patience). Though, the first expression is more powerful than the second expression.

Idiomatic expression identification algorithms for the Amharic language are required to clarify the concept presented in fiction, education books, and historical books to make it easy to understand. This demands the creation of a model which identifies Amharic idiomatic expression collections with their idiomatic meaning.

A reader may not capable to detect the combined phrases that form idioms and determining the existence of an Amharic idiom based on the expression at hand. As a result, the idiomatic expression of texts is taken literally by the readers including by machines. If the idiomatic expressions were translated word by word, the intended meaning would be lost.

The main importance of this study is the output from this research would be used as the starting point for further investigations. And the proposed model would clarify idiomatic

expressions and it can be incorporated into other NLP research to avoid the problems that happened to the existence of idiomatic expressions.

So, it would improve the performance of NLP applications like machine translation, semantic analysis, and sentiment analysis by identifying the idiomatic expressions and giving them hidden meanings.

As a methodological contribution since there is no study done before and our work is the pioneering work on sentence level in Amharic language idiomatic expression identification, we collected the dataset from the scratch. So this is one of our methodological contributions.

1.6. Organization of the research work

This research work is organized into five portions. The first chapter includes the introduction, problem statement, objective, scope, and significance of the study. Chapter two covers the overview of the Amharic language, writing system, punctuation marks, numbers, and the related works which are done before in idiomatic expression identification. The third chapter discusses the research methodology used, approaches, and tools used to develop the proposed model. The fourth chapter describes how to custom deep learning approach for idiomatic expression identification from Amharic texts. By comparing different selected algorithms with experiments and evaluating their performance. Finally, the conclusion, recommendation, and some future works have been presented in chapter five.

CHAPTER TWO

2. Literature Review

This chapter aims to demonstrate the overview of the Amharic language, Amharic Morphology, Amharic word classes, Amharic writing system, punctuation marks, numbers, and the related works of idiom recognition.

2.1. Overview of Amharic Language

Amharic (አማርኛ) is the second most-spoken Semitic language in the World, next to Arabic and it is the official working language of the Federal Democratic Republic of Ethiopia (Gereme et al., 2021). Amharic is also spoken in many Ethiopian states, notably Amhara and the multi-ethnic Southern Nations, Nationalities, and Peoples. Additionally, it is the functioning language of the Ethiopian Federal Government and some regional governments in Ethiopia, most documents and News in the country are produced in the Amharic language.

However, still, the Amharic language has few electronic resources on the web and little work has been done for different computer-based applications. It used the Fidel or abugida (**ፊደል**) writing system, which was adopted from the now-extinct Ge'ez language. It has only one African-derived script, Ethiopic Fidel; the script Fidel has 33 orders and seven forms, resulting in 231 different Fidels (**ፊደሎች**). There are even forty more with a specific trait that usually represents labialization, such as **ሟ**, **ሿ**, and so on. There are 275 characters in all (**ፊደሎች**, Fidels), but not all of them are required for the spoken language's pronunciation patterns; some were simply inherited from Ge'ez with no meaning or phonetic differentiation in modern Amharic. The 275 symbols of the script, only about 233 remain if the redundant ones are removed. There are no upper and lower case variations in the Amharic writing system. Amharic Characters are represented by Unicode in a computer, and Unicode provides a unique number for every single character using the program.

2.1.1. Amharic Morphology

Morphology studies the pattern of word formations including inflection, derivation, or compound word formation using different affixes to create derivational and inflectional morphemes (Hordofa, 2020). A phoneme or collection of phonemes forms a morpheme, which is the lowest expressive piece in a word (**ይማም**, 1989). For example, from the noun '**ልጅ** -' child' another noun **ልጅነት**-'childhood', from the adjective **ደግ** -' generous' the noun **ደግነት** - 'generosity', from root **ሄደ**, the noun **መሄድ**-'go' from infinitive verb **መስበር** -' to break' the noun **ስበር** - 'break' can be derived.

2.1.2. Amharic Word Classes

The Amharic Language has different word classes such as nouns, verbs, adverbs, pronouns, adjectives, and prepositions.

- **Noun (ስም)**

Nouns in Amharic can be either simple or derived (e.g., '**ቤት**' - 'house', '**መሬት**' - 'earth', and '**እሳት**' - 'fire'). Compound words can be used to make nouns (sometimes by affixing the vowels **ኧ** and **ኦ**): - Noun (**ብረት**) + Noun (**ድስት**) => **ብረት ድስት**; Noun (**ልብ**) + Verbal (**ወለድ**) => **ልብወለድ**; from Nouns by suffixing bound morphemes Nouns can be derived (Example: Noun (**መንገድ**) + morpheme (**ኛኛ**) => **መንገድኛኛ** => **መንገደኛ**).

In the Amharic Language noun class, there are two gender indicators called masculine and feminine. Masculine means male gender indicator word and feminine means female gender indicator. However, for things that are not naturally male or female, the gender tends to be used when the entity is small or adorable; the gender male is used otherwise. The feminine gender suffix (- it or yt, pathologically conditioned) is accustomed to marking femininess in cases that otherwise would be masculine. In addition to this expansion, Amharic nouns can also be expanded in number to form a plural. Consider the following example: - Example: - [**ካህን - ካህናት**], [**መምህር - መምህራን**], [**ልጅ-ልጆች**], [**ክፉ - ክፉዎች**], [**መነኩሴ - መነኮሳት**].

- **Verb (ግስ)**

Any word that can be used at the end of a sentence and accepts suffixes such as /ህ/, /ሁ/, / ሽ/, etc in the Amharic language is called **ግስ**(verb). Verbs are derived from verbal roots by affixing the vowel -ኧ. Example: **ስ -ብ -ር ኧብኧር -** [**ሰበር** -] may derive from compound words of stems and verbs. Example: - **ስብር + አለ= ስብር አለ**, **ፀጥ+ አደረገ =ፀጥ አደረገ**.

- Adjective (**ቅፅል**)

Any words that qualify a noun or an adverb that occurs before a noun, e.g. **ፈጣን ልጅ**, and after an adverb (**በጣም ፈጣን**). When pluralizing an adjective, it will repeat the previous letter of the word's last letter e.g., **ረኧዦም => ረዦኧዦም**, **አጭር => አጫጭር**, **ጥቁር => ጥቋቁር**, etc. By infixing vowels between consonants, adjectives can be formed from verbal roots.

- **Adverb (ተውሳክ ግስ)**

In Amharic Language, Adverbs qualify verbs by adding additional concepts to the sentence. Example:- **ገና**, **ዛሬ**, **ቶሎ**, **ምንኛ**, **ክፉኛ**, **እንደገና**, and **ልግምኛ**, etc.

- **Preposition (መስተዋድድ)**

A preposition is a word that can be used before a noun to conduct adverbial actions such as place, time, cause, and so on, but it can't take any suffix or prefix from the beginning to the end of the character and can't be used to make a new word. It consists of **ከ ፣ ለ ፣ ወደ ፣ ስለ ፣ እንደ** etc.

- **Pronoun (ተውላጠስም)**

This category can be further separated into deictic specifiers, which include **ይህ**, **ያ**, **እሱ**, **እስዋ**, **እኔ**, **አንተ**, **አንች** quantitative specifier, which includes **አንድ**, **አንዳንድ**,**ብዙ**, **ጥቂት**, **በጣም** and possession specifier, which include **አንድ**, **አንዳንድ**,**ብዙ**, **ጥቂት**, **በጣም** and possession specifier such as **የእኔ**, **የአንተ**, **የእሱ**,**እነሱ**, and so on.

- **Amharic Phrases (ሐረግ)**

A phrase (**ሐረግ**) is a collection of words that convey some meanings but do not make complete sense on their own. It is always a part of a sentence and group of words, often carrying a special idiomatic meaning; in this sense, it is synonymous with expression. In linguistic analysis, a phrase is a crowd of words (possibly a solo word) such as '**ና**' that functions as essential in the arrangement of a sentence, a single unit within a grammatical hierarchy.

- **Amharic sentence (አረፍተ ነገር)**

A sentence is a collection of words that expresses or conveys a complete meaning. The order of words inside Amharic sentences is different than in English. Generally, the verb goes after the sentence and the structure is Subject / Object / Verb. It can be a declaration used to announce, clarify, or argue an occasion (Sikdar & Gambäck, 2018). Example Abebe [subject] eat [verb] his lunch [object], In Amharic **አበበ** [subject] **ምሳዉን** [object] **በላ** [verb] as a phrase to express anything, a sentence may be incomplete. The following are two types of Amharic sentences: - sentences, both basic and complicated. A simple sentence has only one verb within a phrase. Whereas a complex sentence is categorized as a simple sentence and formed by complex phrases.

The subject-object-verb arrangement is common in Amharic sentences (SOV). However, OSV sentences do occur from time to time. For example:- the sentence '**አለሚቱ ሰለሞንን መታችዉ**'/alemitu selomonn Metachwu, which is in SOV order, can also be written as '**ሶሎሞንን አለሚቱ መታችዉ**' Solomon alemitu metachw' in OSV order. Unless the word (sentence object) includes the object marker, the meaning of a phrase might be changed depending on where words are placed in the sentence. '**ን**'/'-n'. For instance, '**ጅብ ውሻ ይበላል**' /'jb wusha ybelal'and '**ውሻ ጅብ ይበላል**' /'wuxa jb ybelal': Both phrases utilize the same words, but they have distinct meanings. '**ጅብ**' /'jib' and '**ውሻ**'/'wusha' are the subjects of the first and second sentences, respectively (**ይማም**, 1989). There are no subject markers or morphemes in Amharic nouns (affix). However, a subject can be identified from its place in a sentence. '**-ን**' /'-n' is a suffix that is used as a sign for Amharic objects.

2.1.3. Amharic Punctuation Marks

In Amharic, there are many punctuation marks. There are approximately seventeen punctuation marks in the Amharic language writing system (Tewodros, 2003). However, only a few of them are widely used. The most commonly used punctuations in both computer-generated and handwritten text are listed below.

- **Hulet Neteb (፡)** two square dots arranged like a colon: This is used to separate one Amharic word from the other in the Amharic writing system. However these days its function is replaced by spaces.
- **Four square dots** arranged in a four-sided pattern (።): is the basic one in the Amharic language which is used to represent the end of a sentence. It has the same use as a full stop in the English language.
- **Netela sereze (፣)**, an equivalent of a comma used for separate lists in Amharic text.
- **Derib sereze (፤),** which is the equivalent of a semi-colon, may also be found in use as a list separator.

The Amharic writing system has also borrowed additional punctuation marks from other foreign languages (?, ! , “ , ”, ‘, /, \, etc.) (Tewodros,2003).

2.1.4. Amharic Numbers

The Amharic Number system writing has 20 single characters which represent one (1/፩ up to 9/ ፱), tenths (ten/፲ to ninety/፺), hundred (፻), and ten thousand (፼). These characters are resulting from Greek letters and are modified to look like the Amharic character by adding a horizontal line on the top and bottom of each character.

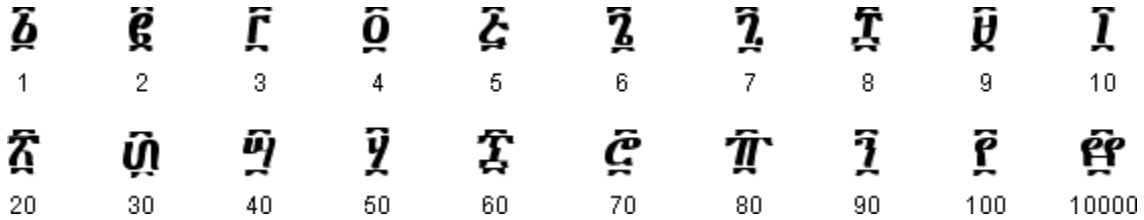


Figure 1: Amharic Numbers

2.2. Overview of Amharic Idioms

Amharic Idiomatic expressions are a phrase that cannot be interpreted by taking individual word meanings. It can be formed by a single term, phrase, or clause level. For example:

Single word: **ቦጨቀ** (“Bocheke”) ፤ **ስንዝሮ** (“Senzero”)፤ **ሰብቷል** (“Sebtwal”)

Phrases: **ሁለት ልብ** (“Hulet Lib”)፤ **አመድ አፋሽ** (“Amed Afash”)

Clauses: **ለአቅመ ሄዋን ደረሰች** (“She Reached Out to Iowa”)፤ **በረሀብ አለንጋ ተገረፈ** (“He was whiplified by a hunger whip”)

The complex nature of idiomatic expressions is both pure and semi-pure property. This means the idioms can have both literal and idiomatic meanings. Pure idiomatic expressions have only one meaning (Idiomatic) whereas semi-pure idiomatic expressions have two different meanings (Idiomatic & Literal). For example:- the Amharic idiomatic expression “**እንቶ ፈንቶ**” / (“A hundred funnels”)is interpreted in only one meaning which is “**ተራ፤ ዝባዝንኬ ነገር**” / (“Ordinary”, “Something straying”). Unlike pure idiomatic expressions, some idioms can be taken both idiomatically and literally. The Amharic idiomatic expression “**እጁ ሰባራ**” / (“A Broken Hand”) can be interpreted idiomatically as “**የማይረባ ስራ የሰራ**” / (“Doing nonsense”) and it can be interpreted as literally to express “**የተሰበረ እጅ**” / (“Broken hand”).

2.3. Approaches for Idiom identification

For idiom expression identification work, there are three categories of approaches which are rule-based, classical machine learning, and deep learning (Salton et al., 2017). Rule-based models depend on rewriting the rules to handle the semantic expression of the sentence, by using logic-based rules (Muzny & Zettlemoyer, 2013). In these traditional models, the representation and decision components are present. Rule-based techniques do not take the context and semantics of the words into consideration, which are context-less identification techniques. For idiom identification, it is difficult to use this technique because the idiom expression being idiom or non-idioms depends on the context of the words that occur in the text (Peng & Feldman, 2016).

2.3.1. Classical Machine Learning Approaches

Classical machine learning uses handcrafted text features. This feature can be lexical representation, syntactic representation, structural representation, and semantic representation (Kamath et al., 2018). Classical machine-learning approaches can solve most of the problems of rule-based approaches by learning the structure and semantics of idiomatic expressions.

K-Nearest Neighbor

KNN is a non-parametric instance-based learning approach that saves all existing data points and categorizes additional points of data depending on a similarity measure (Bzdok et al., 2018). The KNN technique assigns newly unclassified samples to the class containing the majority of their next K. While the number of samples in the training data set is big, this approach is particularly good in reducing misclassification mistakes.

The KNN is a prospective statistical classification method that is used to classify objects in feature space based on the next learning instances (Ankit & Saleena, 2018). The sluggish learning algorithm is another name for this algorithm which means that training data points are not generalized and all training data is required during the test phase. When the class of an instance is unknown, the algorithm calculates its closest K neighbors and assigns the class by selecting one of them. The training part of the KNN technique is very rapid, but the testing step is time and memory-intensive (Ankit & Saleena, 2018; Bzdok et al., 2018).

K-Nearest Neighbor (KNN) is an automatic learning method that determines the nearest neighbors for determining the given instance class based on the calculation of the least distance between the given point and the other points of the distances calculated with; Euclidean, Manhattan, Minkowski, Supremum, and Cosine Similarities (Bzdok et al., 2018). The test pattern is ordered by the number of votes received from neighbors K, and the sample is assigned to the most often used class among its neighbors K-Closer. K is a positive integer that is determined via a test-and-error process that yields the lowest error rate. In general, the classifier architecture is simple, but the classification time increases as the volume of training data grow.

Support Vector Machine

The support vector machine (SVM) is a statistical learning theory-based machine learning technique (Bzdok et al., 2018). A support vector machine constructs a hyper plane or a group of hyper planes in a high or infinite dimensional space for classification (Kumar et al., 2017). Any class's hyper plane with the largest distance to the closest training data point (functional margin) achieves good separation; in general, the higher the margin, the lower the classifier's generalization error. SVM employs a non-parametric technique that uses a binary classifier and is capable to process a large amount of data quickly. Performance and accuracy are affected by hyper plane selection and kernel setting.

The primary advantages of SVM are that it gives more flexibility in choosing the threshold form, it has a nonlinear transformation, it has a good generalization capability, and it eliminates the problem of overfitting. Complicated computations are reduced. Decision rule complexity and error frequency are easy to manage. The disadvantages of SVM include limited transparency, time-consuming training, a difficult-to-understand algorithm structure, and difficulty determining optimal parameters when there are nonlinearly distinguishable training data (Bzdok et al., 2018).

In SVM, both the computation and decision rule complexity are minimized. The amount of the learning data and the separation of the classes affect training speed in SVM. The learning process in KNN is free; no assumptions about the features of the ideas to be learned are required, and complicated concepts can be learned using basic processes by local approximation.

KNN can produce class borders that are less interpretable than linear SVM class boundaries. Despite having a huge amount of input variables, SVM can achieve decent prediction accuracy for new observations, however, KNN's classification performance rapidly deteriorates when searching for patterns with a high number of input variables since all variables are given equal weight.

When inferring the session of a new observation, SVM only requires a small portion of training points to build the classification rule, making it more memory efficient and computationally intensive. Alternatively, KNN often necessitates more processing and memory resources because it classifies new observations using all input variables and training samples.

The hyper plane with the biggest margin between the two groups is the preferred hyper plane for an SVM. The margin is the slab's thickest part that corresponds to the hyper-plane with no data points on the interior. A bigger distance from the boundary has a higher possibility of being classified, but a smaller space means a greater chance of misclassification (Bzdok et al., 2018).

The following equation draws a straight line in testing data to classify it into different classes as shown in Figure 2 (Source (Javatpoint, 2018)).

$G(x) = W^t X + b = 0$ ---------- equation (2.1)

Where

W is the line perpendicular to the hyper plane

b is the position of a hyper plane in a feature vector

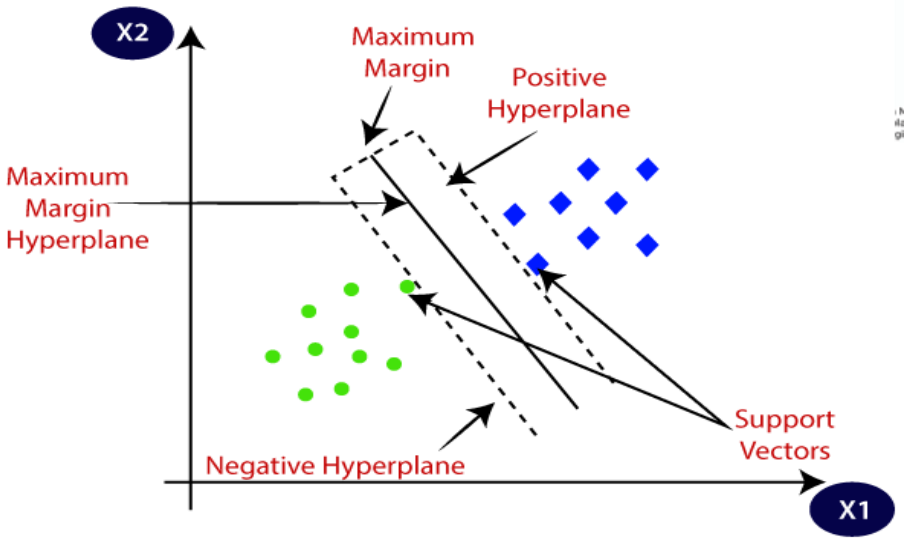


Figure 2: SVM model architecture

2.3.2. Deep Learning Approaches

The deep learning approach is a type of machine learning on an artificial neural network that can learn features of the text by the model itself. Deep learning approaches can handle context as compared with classical machine learning (Kamath et al., 2018). For this study, we experimented with Convolutional Neural Network (CNN), Long Short Term Memory (LSTM), and Bidirectional Long Short Term Memory (BiLSTM).

Convolutional Neural Network (CNN)

Convolutional Neural Network is a deep learning algorithm that performs well for different NLP tasks. Forward neural networks in the form of CNNs have a layer of neurons for convolutional operations. The visual nerve's function served as inspiration for CNN's design. Convolutional kernels, commonly referred to as filters, are used by neurons to react to input from the activations of neighboring neurons. Moving the convolution kernel across the complete range of values is the process of convolution. The convolution operation in this example represents the multiplication of the convolution kernel and input values (Tavcar et al., 2013). The input data is on the left, the filter is in the middle, and the convolution output is on the right in the figure below. Figure 3 shows the CNN convolutional process (source: (Maslej-Krešňáková et al., 2020)).

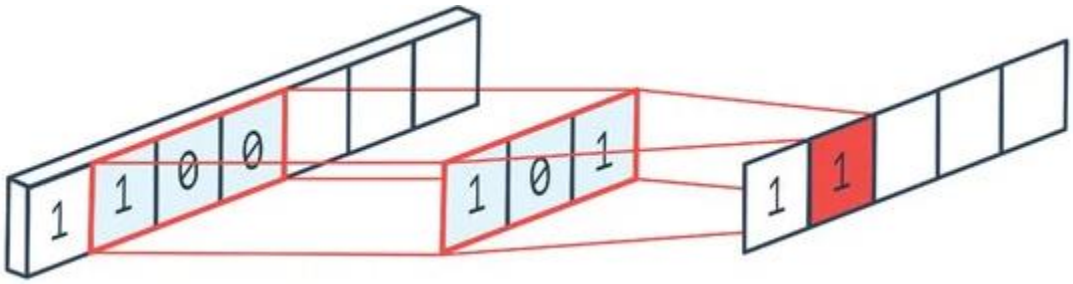


Figure 3: 1D CNN Convolutional Process

Pooling layers are used in convolutional networks to decrease the number of outputs, simplify computation, and prevent over-fitting. The sampling layers are typically inserted right after the convolution layers since duplicate data is produced as the convolution kernels cycle through the different inputs. To get rid of extra data, employ the pooling layers. For NLP jobs that require key phrases and local contexts for decision-making, CNN will be a good option.

Long text inputs on CNN won't be good. But an experiment is needed to demonstrate it. Figure 4 below shows the CNN sample architecture (source: (Phung & Rhee, 2019)).

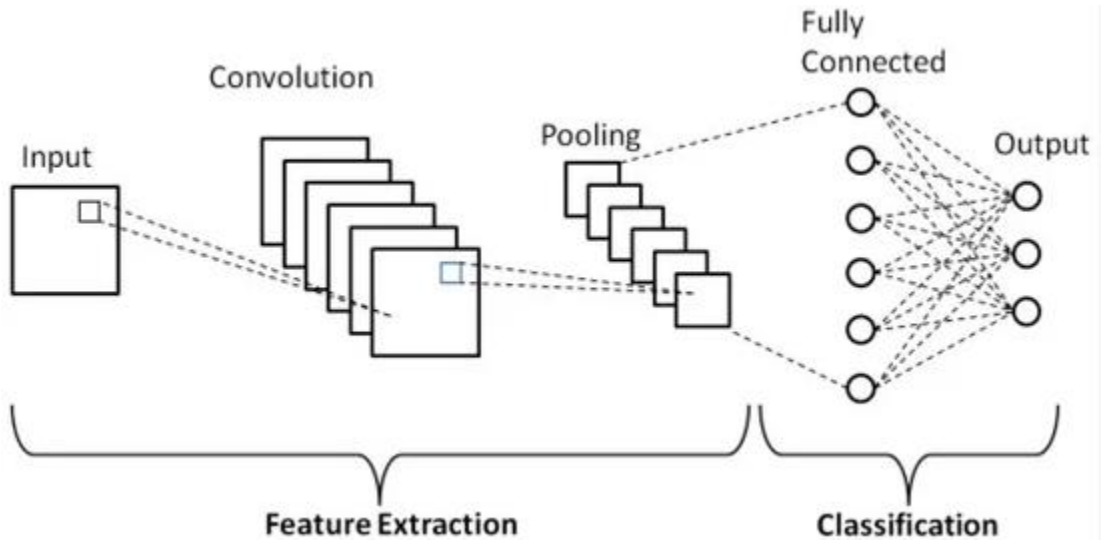


Figure 4: CNN algorithm sample Architecture

Long Short Term Memory (LSTM)

The Long Short-Term Memory (LSTM) is a sort of recurrent neural network that includes a memory for storing long-term data. The LSTM is well suited to dealing with the vanishing gradient problem due to its more sophisticated structure. When the LSTM is used in any sequential activity, long-term information and context are preserved (Kamath et al., 2018). Figure 5 (source:(Graves, 2012)) below shows a normal recurrent neural network and the vanishing gradient problem, which causes context to be lost, as well as how information and context are preserved in LSTM (Graves, 2012). We can comprehend the fundamental distinction between LSTM and ordinary recurrent networks.

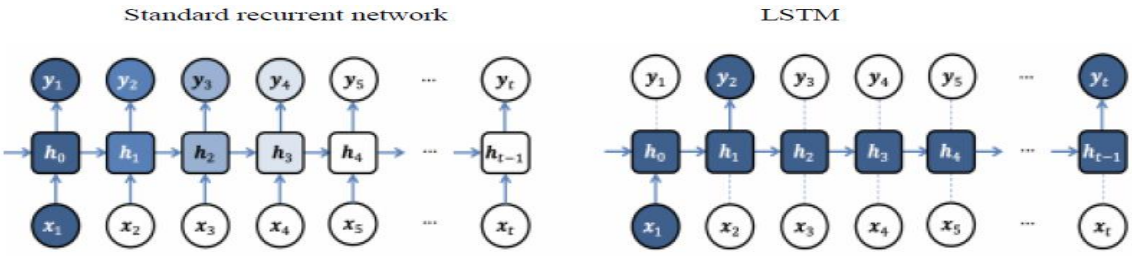


Figure 5: Comparison of standard recurrent network and LSTM

The LSTM network differs from other types of neural networks in that it is composed of layers of linked memory blocks rather than entangled neurons. The block has gateways that manage its state, functionality, and information flow. Gateways can determine whether data in a series is important and needs to be retained. Long-range context preservation is best handled by LSTMs, however, only forward context is captured (one-directional). The LSTM uses four memory block elements to carry out the following tasks (Tavcar et al., 2013).

- **Input gate:** is used to keep track of the data being entered into the memory block.
- **Cell gate:** is used to store long-term information.
- **Forget gate:** is utilized to decide which information should be kept and which should be eliminated.
- **Output gate:** based on the input and memory unit, utilized to decide what operation to do on the output.

Bidirectional Long Short-Term Memory

A subset of LSTM networks is Bidirectional Long Short-Term Memory (Bi-LSTM) networks. Bi-LSTMs are made up of two layers that are hidden from view. The input sequence is processed forward by the first hidden layer. However, the sequence is processed backward by the second hidden layer (Kamath et al., 2018).

The output layer may access the past and future backgrounds of each point in the series to these hidden layers. The LSTM, as well as its bidirectional variations, proved to be incredibly beneficial. They may learn when and how to forget particular pieces of knowledge, as well as when and why not to use certain gateways in their architecture.

Deep Learning model Hyper Parameters

The variables known as Hyperparameter control the network's structure, such as the number of hidden units, as well as how the network is trained (E.g. Learning Rate). Before training (before weight and bias optimization), Hyperparameters are specified. Hyperparameters are crucial since they directly affect how the training model behaves and have a significant impact on how well the model performs. The most common hyper parameters are presented below:

- **Hidden Layers:** these are the layers between the input and output layers. Using regularization techniques, several hidden units within a layer may improve accuracy. If the number of units is reduced fitting may occur and if the number of hidden layers is too large overfitting will occur. Hence an average number of hidden layers are better.
- **Dropout:** is a regularization technique that helps to avoid overfitting by increasing validation accuracy and thus improves generalizing power of the model to be developed.
- **Activation function:** these are functions that are employed in neural networks to decide whether or not a neuron can fire by computing the weighted sum of input and biases. The activation function can be either linear or non-linear, depending on the purpose it serves (Nwankpa et al., 2018). Sigmoid, Hyperbolic Tangent Function (Tanh), Softmax, Softsig, and Rectified Linear Units (ReLU, Softplu) are some of the activation functions. Among the aforementioned activation functions, sigmoid is commonly used for binary classification models.
- **Learning rate:** this is the rate at which network parameters are updated. A low learning rate slowed the learning process. A larger learning rate accelerates the learning process (Nwankpa et al., 2018).
- **Number of epochs:** the number of times the entire training data is presented to the network during training.
- **Batch size:** is the number of subsamples given to the network after which parameter updates occur most of the time batch size can be the power of 2.

2.4. Related Works

The study conducted for the English language (Salton et al., 2016) used the classical machine learning approach and Skip-Thought Vectors (sent2vec) to generate distributed representations of features predictive of idiom token classification. Their study employed a set of sentences from the British National Corpus, which included 53 distinct Verb Noun Constructions and the sentences were encoded in three ways: uni-skip, bi-skip, and comb-skip. To evaluate the results, four expressions on various models and distributed representations were used.

The analysis only used Verb Noun Construction (VNC) expressions to distinguish idioms from literals. On the contrary, they used only the sentence containing the target phrase as input, which makes them less reliant on a potentially inaccurate or incomplete model of discourse context.

(Verma & Vuppuluri, 2015) their study was focused on the semantics of idiom phrases so that the properties of individual terms in a phrase differ from the properties of the phrase as a whole. For the experiment, the researchers used three data sets: englishclub.com, Oxford Dictionary of Idioms, and Verb Noun Construction (VNC) corpus. The achievement of their study was determined using a union and intersection methodology. As a result of this research, a model that recognizes idiomatic expression using dictionary-based type was developed.

The study was carried out by using the Lesk word sense disambiguation algorithm to detect idiomatic phrases automatically from dictionaries (Muzny & Zettlemoyer, 2013). The analysis included three lexical features and five graph-based features. The study concentrated on extracting English language phrases from web data and the results were evaluated using the Wiktionary default rule and the Lesk word sense disambiguation algorithm by considering the meanings of the word become unclear, it becomes an idiom, however not all ambiguous words are idioms.

Their model was restricted to dictionary terms and employed a pattern of word matching technique, with all ambiguous words ruled out as idioms.

(Peng & Feldman, 2016) proposed two approaches for representing the context of idioms and literals by computing the inner product of context word vectors with a vector that represents a goal expression. The researchers used 2984 VNC tokens from BNC as well as a list of VNC tokens classified as literal, idioms, or unknown. The investigation was limited to the occurrence and frequency of terms in context.

A study was conducted to classify phrases as literal or idioms depending on the distribution of terms (Peng & Feldman, 2016). From their study assumption, a literal expression distribution differs from an idiomatic expression distribution.

The analysis used a covariance matrix and word vectors obtained from the Word2Vec tool to represent the distribution of terms in vector space. The researchers conducted their study using data from the National Science Foundation. The output was evaluated using 12 datasets for 20 runs in the study. The study looked at the frequency of occurrence of a word to determine whether it was an idiom or a literal word. They used a small dataset to test and train the model.

(Salton et al., 2017) stated that they used four different models to overcome the limitations of the state-of-the-art model for Verb Noun Idiomatic Combination (VNIC) type idiom identification. In their study, they have proposed a probabilistic approach, Smoothed Probabilities, Interpolated Back-off Probabilities, and Normalized Google Distance. They used 319 idioms and 319 literals from the British National corpus to train the idiom identification model and 95 idioms and 95 literals from the British National corpus to test the model. According to their findings, feeding the fixedness metrics to an SVM improves the F1 score on the same VNIC type identification task and they achieved 85% of the F1 score.

(Fenta, 2021) designed an idiom classification model for the Amharic language at the phrase level using a supervised machine learning approach by taking a total of 1000 idioms and literal words. As a dataset, they prepared phrase-level idioms and literals. They used word2vec word embedding with the K-Nearest Neighbor algorithm and achieved 97.5% accuracy.

2.5. Summary of Related Works

There is not enough work done before for Amharic idiomatic expression identification. We began working on an idiom recognition model for the Amharic language after examining the negative impact of idioms in several NLP studies. In foreign languages, the negative effects of idioms on NLP research have been stated. The rich nature of idioms in every language is what distinguishes them.

When we set out to conduct this study, we evaluated idiom detection methods utilizing several languages to recommend the best methodology for our study.

In the mentioned linked work, the researchers employed their methodology, approach, and research results. Researchers used a VNC corpus of grammatical part of speech tag sequences to distinguish idioms such as “lose head”, “lose face”, and “create a scene” but the tag sequence of Amharic idioms followed NN, NV, and VN. As a result, extracting a common feature based on a part of speech tag sequence, such as that of English idioms, is challenging to apply to Amharic text since it uses different tag sequence formats.

When we look at the nature of idiomatic expressions, it is tough because Amharic idioms are either pure or semi-pure expressions, thus when the idiom is semi-pure, the Amharic language has both the idiomatic and literal meanings of the expressions. Researchers employed the term frequency approach of the VNC part of the speech tag corpus to represent phrases, taking into account word ambiguity. Researchers employed ambiguity of words, rule-based approaches, and pattern-matching approaches, however not all ambiguous words are idioms, and the other approaches are static. To avoid the detrimental influence of NLP applications and the gaps found in pieces of literature, we suggested a deep learning approach for Amharic idiomatic expression identification.

CHAPTER THREE

3. Designing Amharic Idioms Identification Model

3.1. Introduction

We employed an experimental research design methodology for the proposed work which enables us to manipulate different Hyperparameters and experimentation setups to study their effect on the proposed model. Using this research methodology different experimental setups were implemented and evaluated their effect on the proposed research work. So that, we can evaluate and understand the effects of various variables on the research procedures which stand to develop a solution (Maxion, 2009). Different variables are modified in this research process to see how they affect other variables. Datasets, experimental settings, and model Hyperparameters are the variables in question. These variables are changed in the proposed work to examine their impact and discover the best-fit setup. In general, the experimental research technique has aided us in measuring and analyzing the impact of the above variables on our research.

To build the proposed work, a deep learning approach is employed. The proposed work passed through the following phases: dataset acquisition, dataset annotation, dataset preprocessing, model development, classification, identification of IE, the semantics of IE, and evaluation. The first task of this work is collecting sentences with Amharic idiomatic expressions and sentences without Amharic idiomatic expressions. Once the dataset is collected, then the datasets are annotated by domain experts using the annotation guidelines to train and test the model. After the completion of the annotation subtask, the dataset is preprocessed by designing text preprocessing (normalization, stop word removal, and punctuation mark and number removal) algorithms. The process of converting collected data into a format that can be used to train and test the model is called text preprocessing (Agarwal, 2015).

Since machines can’t understand text data directly as we humans do, it is a must to apply different word representation techniques to change the data into a numeric vector. After word representation, the model is trained, tested, and evaluate the performance. We have experimented with CNN, LSTM, BiLSTM, SVM, and KNN.

3.2. Development tools

For experimentation, we used the python programming language to accomplish the preprocessing tasks as well as, for model training, and model testing. Since python programming language has the capability of user-friendly as well as high-level and general purpose programming language. Anaconda Jupiter notebook python framework is used as a code editor with different Python packages. Keras deep learning libraries are used for backend computations. Different python packages like Numpy, Pandas, and matplotlib are employed for analyzing and tabulating the developed model.

Python: is a high-level scripting language, interpreted, interactive, and object-oriented. It is intended to be a very comprehensible language. It has a rarer syntactical arrangement than other languages. The reason that we selected python programming language throughout the implementation is because of its characteristics as a user-friendly and general-purpose powerful programming language (Sharma et al., 2020).

Keras: is an application programming interface (API) for high-level neural networks that are built on Python. Rapid deep neural network-based model development is the goal of this open-source neural network library. In this study, we built Keras on top of Tensor Flow. A user-friendly deep learning library with modularity, extensibility, and ease of use is called Keras. Quick prototyping and flawless execution on both CPU and GPU are made possible by using Keras for deep learning. We have utilized the Keras deep learning library while taking the aforementioned qualities into account.

Scikit-learn (Sklearn): is the most usable and robust machine learning library. It uses a Python consistency interface to give a set of efficient tools for machine learning and statistical modeling, such as classification, regression, clustering, and dimensionality reduction. Numpy, Scipy, and Matplotlib are the foundations of this package, which is mostly written in Python.

Numpy: is a python package that stands for “Numerical Python” or “Numeric Python” which allows doing quick mathematical computations on arrays and matrices. So in this study, we use the Numpy python library for array-based numeric representation of our text dataset.

Pandas: is like Numpy which is one of the most popular python libraries. It offers high-performance assemblies and file examination tools that are simple to use by providing an in-memory 2d table object called a Data frame. It provides objects for multi-dimensional arrays like spreadsheets complete with column and row labels. In our study, we use this library to load and operate our dataset.

Matplotlib: is a visual charting library for python programming language. For the model training phase, it is used to plot different results and display various graphs.

3.3. Dataset Collection

Dataset is a crucial resource for research which takes a lot of time to collect and organize. The dataset for the proposed work is collected from fictions books (“Fiker eske mekabir”, “yeburka zemeta”, “lejinet temelso aymetam”, “chilancel”, “sememen”, “yeasimba fiker”, “megbat ena mewutat”, “yaltenore lijinet”, “alwoledem”, “piassa”, “gungun”, “gemena”,”hiyaw fiker”, “arest situlegn”, “yetekolefebet kulf”, “merkogna”, “jobiraw”, “lijenet”, “yetinbit ketero”, “sewena essest”, “dertogada”, “berkrketa”, “afincho”, “jobiraw”, “keadims bashager”, “tikusat”), educational books (amharic grade 9 - 12), historical books (“atse tewodros”, “kedamawi hailesilasie niguse negest hiwotena yeethiopia ermeja”), and bible. We have used 700 idiomatic expression terms which are collected from Amharic idiom books (“Dagnachew Worku with Amsalu Aklilu”, “Debebe HaileGiorgis Engida”) to collect sentences that are employed to trigger the idiomatic expression. After that, the Amharic language experts check whether the collected sentences contain an idiomatic expression or not. The dataset for the Amharic Idiomatic Expression Identification task can be prepared at a document level, paragraph level, or sentence level. For the proposed identification work we used a sentence-level dataset. Since it is the base for the paragraph as well as the document-level dataset.

3.4. Dataset Annotation

Once the dataset is collected, the next task is annotating those sentences by preparing annotation guidelines. Since we are dealing with idiomatic expression identification the Amharic language experts have requested to label those sentences as Idioms and Non Idioms. To avoid bias and unfairness we have prepared the annotation guidelines which are attached in (Appendix A). These annotation guidelines are reviewed by Bahir Dar

university linguistics masters students and the annotation was done by three students. After the labeling is done, classifying the dataset as idiom and non-idiom is going through. Figure 6 shows the annotation process of a given sentence.

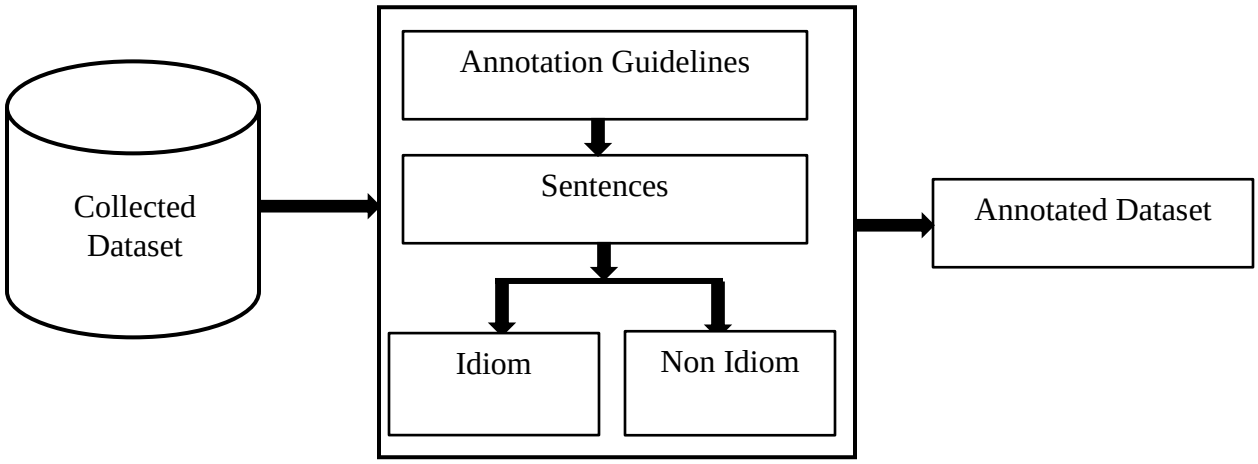


Figure 6: Annotation process

The annotation guidelines hold information about, which sentence is categorized as Idiom and Non-Idiom. In general, the guidelines show that if the text contains an idiomatic expression term it is labeled as Idiom and if the text does not contain an idiomatic expression term inside the sentence labeled it as Non-Idiom.

3.5. Proposed model architecture

The proposed model architecture contains four components that are preprocessing, word representation, model development, and model testing. First, the preprocessing component is done. After that, the preprocessed dataset is converted to numeric vectors using the word embedding technique to make the dataset ready for the proposed model development. Next, we built and train the proposed idiomatic expression classification and identification model. In this component, CNN, LSTM, and BiLSTM algorithms were trained to study their prediction performance for the proposed work. After all the developed model was evaluated using the aforementioned performance measure metrics discussed in section 3.6. The proposed model architecture is presented in Figure 7. As we can see from figure 7 classifying the sentences as Idiom and Non-Idiom is done first to ignore Non-Idiom sentences since it is not used longer for our work. The main focus of this work is classifying the given sentence as Idioms or Non-Idiom. If the text is

categorized as Idioms, the identification of idiomatic expression terms and giving their hidden meaning was done.

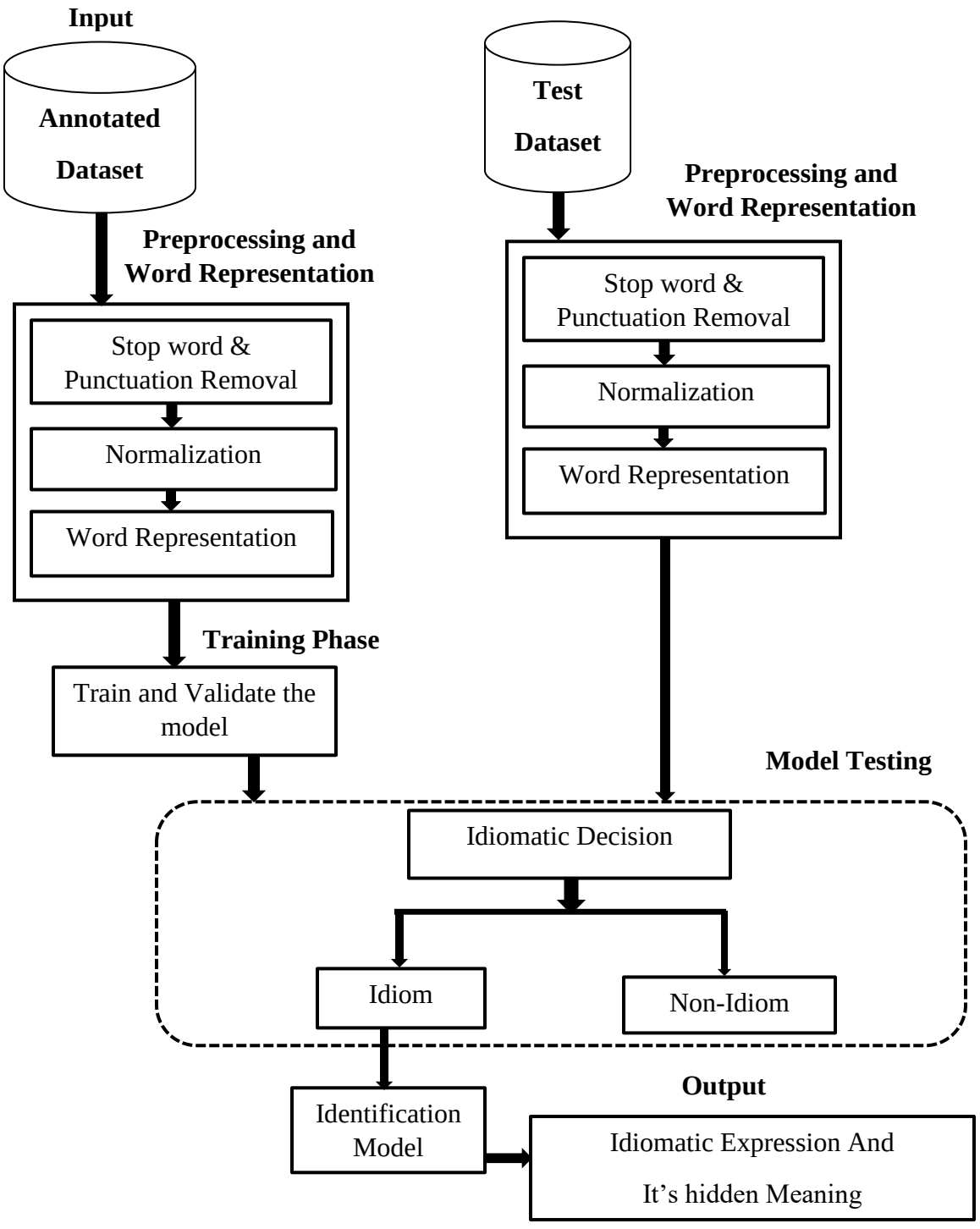


Figure 7: Proposed Model Architecture

3.5.1. Text preprocessing

The next step after the dataset collection is the preparation of the dataset to a form that is suitable for the model training and testing. When we are working with textual data preprocessing usually involves different series of tasks like text cleaning, formatting, and creation of its representation which is used in the model training. Text preprocessing is the major component for different NLP tasks. The documents, paragraphs, sentences, words, or characters identified at this stage are the fundamental text units that should be passed to further processing stages like idiomatic expression identification, text classification, information extraction, etc.

Text preprocessing is the task of numerous activities because the datasets may contain different formats like numbers, and special characters and the most common words which are unnecessary terms like prepositions, articles, and pronouns can be eliminated through these collections of preprocessing subtasks (Ly et al., 2020).

Once the dataset is collected and annotated/labeled, the text needs to be preprocessed by designing text preprocessing algorithms. Punctuation marks and number removal is the leading step that is used to eliminate punctuation marks and numbers. The normalization sub-task is processed to normalize the alphabets having different representations but the same sound and interpretation into a common representation. Since different representations for the same texts make the model represent those texts differently it reduces the model's performance.

Punctuation mark and number removal

In this step, the punctuation marks and numbers are removed since it is not useful for our work. The pseudo-code for punctuation and number removal is presented in Figure 8.

We removed Amharic language punctuation marks (፨, !, ፤, ፣, ”,“,-), numbers ('፩','፪','፫','፬','፭','፮','፯', '6', '7', '8', '9'), brackets((),[],{}) etc.

Input: sentence dataset

Start

Read our sentence dataset

Load list of numbers

Load list of punctuations

While all sentences in the dataset accessed:

Access each sentences

While all terms are accessed from the sentences:

Check that whether punctuation exists or not

If it exists remove it

Check that number exists or not

If it exists remove it

End

Output: text without punctuations and numbers

Figure 8: Algorithm for Punctuation, and Number removal

Stop word removal

Stop words are a list of common terms that don't carry meanings by themselves and it is a language-specific words. In the Amharic language words such as "ሁሉ", "ሆኖም", "ማለት", "እንጂ", and "ሁሉም" are some of the stop word lists which should be removed from our dataset in our case as well as for other similar NLP applications to save computational time to process them. Removing stop words is one of the preprocessing sub-tasks and allows us to focus on words that we need by eliminating the most common terms from the texts (Ly et al., 2020). Each language has its stop word lists that can be adjusted according to the problem and having these we have prepared our own Amharic stop word lists by compiling from a (Tewodros, 2003) paper done on the Amharic language. From the prepared stop word lists, we have implemented an algorithm (see Figure 9) which removes the available stop words from the text by searching.

Input: sentence dataset

Start

Read our sentence dataset

Load list of stop words

Change loaded stop word text into list

While all sentences in the dataset accessed:

Access each sentences

While all terms are accessed from the sentences:

Access each term

Check that if the accessed term is in stop word list or not

If it exists remove it

End

Output: stop word free text

Figure 9: Algorithm for Stop word Removal

Normalization

There are two notions of character normalization in Amharic NLP research. In the current trend normalization is a common preprocessing task in most NLP research; one group argues that normalization should not be done on Amharic characters or alphabets as they have distinct meanings. The other group argues that characters with the same phoneme or sound did not have any distinct meanings so we need to use one representative alphabet for all phones so the characters need to be normalized. (a single word may be written in different representations of an Amharic character sound so that normalization is needed to remove the redundancy of words that have the same meaning). Having this truth, we have done normalization to remove the redundancy of the same words in a different representation. So that it reduces the impact on the proposed model. In the Amharic language, some characters that have the same sound are (**ሀ፤ሃ፤ሐ፤ሓ፤ኀ፤ኃ፤ኻ**), normalized as "**ሀ**", (**ዐ፤ዓ፤አ፤ኣ**), normalized as "**አ**" (**ጸ፤ፀ**), normalized as "**ፀ**",(**ሰ፤ሠ**) normalized as "**ሰ**" presented in Table 1. So that using the normalization algorithm (see Figure 10) all varieties of such characters having the same sound change into one form. Let us see some words with the same sound which have the same meaning.

Table 1: Normalization

English words	**Spelling variation Amharic words**	**Normalized Amharic word**
Sun	**ጸሀይ/ፀሀይ**	**ፀሀይ**
Stomach	**ሆድ/ሖድ**	**ሆድ**
World	**አለም/ዓለም/ኣለም**	**አለም**
Sky	**ሰማይ/ሠማይ**	**ሰማይ**
Eye	**አይን/ዐይን/ዓይን**	**አይን**

Input: sentence dataset

Start

Read our sentence dataset

Load list of similar sound Amharic alphabets

While all sentences in the dataset accessed:

Access each sentences

While all terms and alphabets are checked:

Check the availability of alphabet varity, but same sound

If it exists, change it to common representation

Otherwise preserve it

End

Output: Normalized list of terms

Figure 10: Algorithm for Normalization

3.5.2. Word Representation

In this phase, text features extraction techniques that work well for idiomatic expression identification were determined. Once the text dataset is preprocessed, the dataset should be converted into vector representation to make it understandable by the model and to build the machine learning or deep learning model. The tests remaining after text preprocessing need to be presented in numeric value using word embedding techniques bag of words(BOW), term frequency-inverse document frequency(TFIDF), glove(Glove), Fast Text, Word2Vec, a continuous bag of words (CBOW), Keras embedding are some of word embedding techniques (Sharma et al., 2020).

Even if the first two BOW and TFIDIF cannot capture semantics or the relation of words we used TFIDF for the machine learning approach. Word2vec, Glove, and Fast Text can capture the semantics and the relation of words. Word2vec and Glove cannot handle unseen words easily but the fast text does. On the other hand, Word2vec models see the local context. However, Glove and Fast Text models use a global context. Whereas, Keras provides an embedding layer for neural networks that can be used to compute the embedding of text datasets. The input data must be integer encoded, with each word represented by a different number. Starting with random weights, the embedding layer learns an embedding for each phrase in the training dataset (Shim et al., 2018). So from those embedding techniques, we used Keras embedding for deep learning and TFIDF for the machine learning approach as well as word2vec for both ML and DL identification with meaning part. TFIDF word representation sees the frequency of a term across documents to see the importance of that term (Shim et al., 2018). It is considered by the number of words that occur in a document divided by the entire number of words in a document. Idiomatic expression identification is a classification and identification task. So for experimentation, we used CNN, LSTM, and BiLSTM deep learning approach as well as SVM, and KNN machine learning approach. Then the performance result is compared according to different measurement metrics.

3.5.3. Model Development

In this phase, the actual work is idiomatic expression classification based identification, and giving their corresponding meaning is done by train and testing the model. First, the classification goes through by taking the input sentence and classifying it as Idioms or Non-Idioms classes. Then if the input sentence is categorized as Idioms, the identification of idiomatic expression terms and giving their corresponding meaning is done. There are three approaches dictionary-based, classical machine learning and deep learning approaches are avail for our idiomatic expression identification model. From those approaches, we have carried out experiments using deep learning (CNN, LSTM, and BiLSTM) algorithms and machine learning (SVM, and KNN) to develop the model.

Due to its seq2seq modeling and long-term memory characteristic, LSTM is good at handling context. The limitation of LSTM is that it does not preserve both the forward and backward context of words. This is due to LSTM's neural network's forward-only nature. The other disadvantage is that LSTM and Bi-LSTM require a lot of training time, whereas, CNN needs a little (Maslej-Krešňáková et al., 2020). For comprehending word context, Bi-LSTM outperforms LSTM and CNN. This is because Bi-LSTM contains two distinct hidden layers. Forward input sequence processing is done at the first hidden layer. On the opposing side, the second hidden layer processes the sequence backward. The ability of Bi-LSTM's two hidden layers makes it more effective at accurately capturing the context of words in input phrases. However, we asserted that the performance of the aforementioned techniques may vary depending on the dataset, input text size, and language issues. To approve experimentation on our dataset and experimentation setups, deep learning algorithms CNN, LSTM, and Bi-LSTM were attempted for selecting the best-fit algorithms for the proposed idiomatic expression identification research work. The experimentation result of those algorithms was presented in the result and discussion section.

3.5.4. Proposed Bi-LSTM model architecture

After the dataset processing is finished, as shown in Figure 11, the word embedding layer is triggered to determine the word representation of the datasets. The Bi-LSTM layer for context representation and classification receives its input from the dataset's generated word representation (embedding). The forward LSTMs compute the forward context of the words. On the other hand, the backward LSTMs compute the context of the backward. Following concatenation, the features computed by the forward and backward LSTM are fed to the following layer for additional computation. In addition to this, we also eliminated certain neural nodes using a dropout layer to prevent model overfitting. The activation function transforms the weighted input from the dropout layer that has been transformed and added into node or output activation for that input to choose the neural network output, it is employed. The last layer, the dense layer accomplishes an idiomatic decision by accepting the output of the activation function as input.

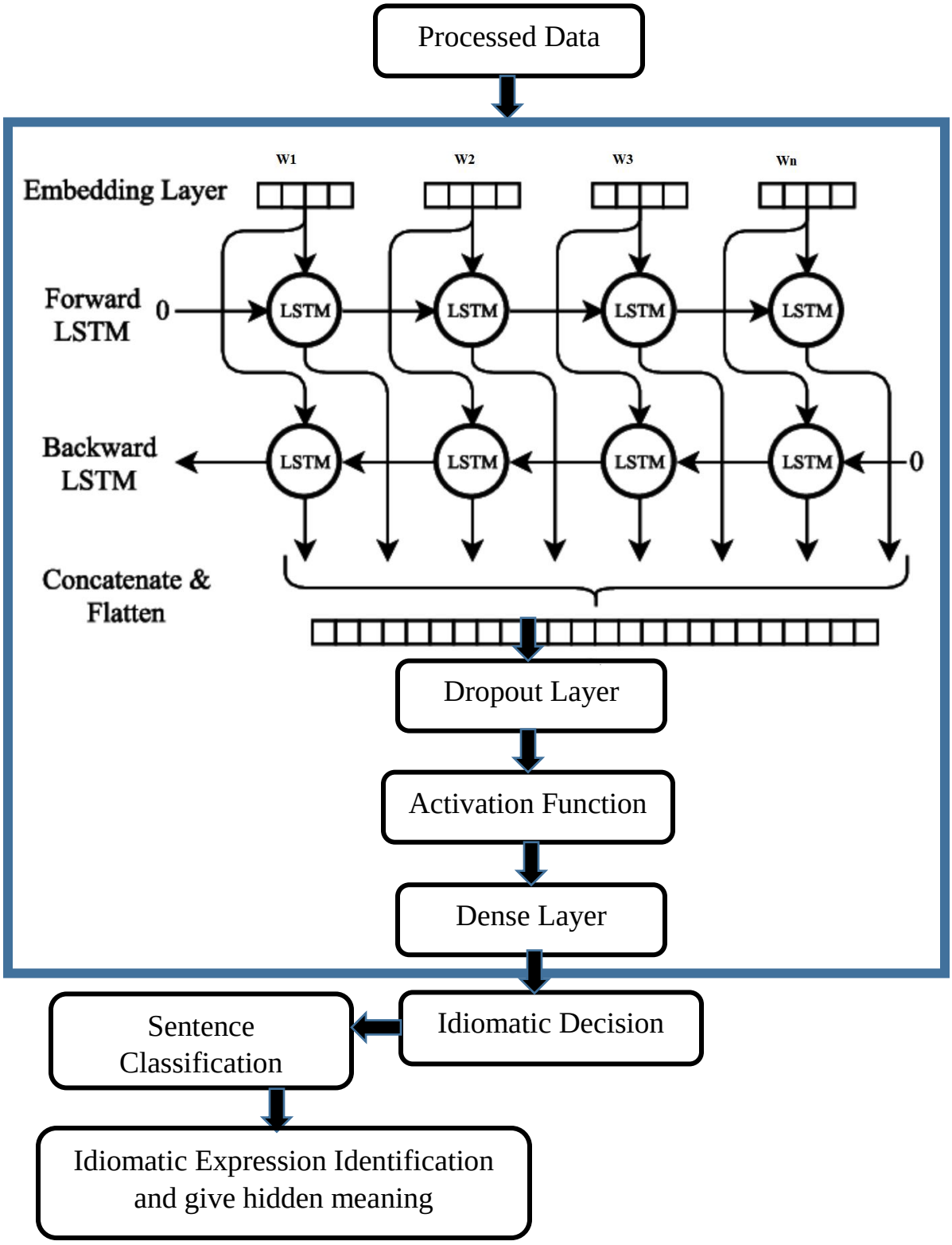


Figure 11: Proposed Bi-LSTM model architecture

3.6. Model Performance Evaluation

The last phase of this investigation work is evaluating the performance of the developed model. In this work, the developed model is evaluated using evaluation metrics Precision, Recall, F-score, and accuracy. Since those evaluation metrics are good measures (Hossin M et al., 2011) for textual data by identifying and analyzing the correct and misclassified data from the dataset. Recall measures the proportion of idiomatic/non-idiomatic expressions that are correctly predicted. Precision measures the proportion of predicted idiomatic/non-idiomatic expressions. The F-score measure is used to analyze the balanced performance measures of recall and precision. Accuracy is also used to measure the general effectiveness of the proposed model. The aforementioned performance indicator can be generated using categorization results, which are frequently reported in a Confusion matrix format (Ting, 2017).

Confusion matrix

A confusion matrix is a table that shows how well a categorization method performs and the performance of a grouping algorithm is shown and summarized using a confusion matrix. The confusion matrix represents counts from actual and predicted values using four outcomes:

- **True Positive (TP):** shows the amount of correctly identified idiomatic expressions as idiomatic.
- **True Negative (TN):** shows the amount of correctly identified non-idiomatic expressions as non-idiomatic.
- **False Positive (FP):** shows the amount of misclassified Non-idiomatic expressions as idiomatic.
- **False Negative (FN):** shows the amount of misclassified idiomatic expressions as Non-idiomatic.

Table 2: Confusion Matrix for Idiomatic Expression Identification

		Actual value	
Predicted Value		Idioms	Non-Idioms
	Idioms	TP	FP
	Non-Idioms	FN	TN

Precision is the proportion of the number of positive examples classified over all the examples classified.

$$\text{Precision (P)} = \frac{TP}{TP+FP}$$

Recall is the proportion of the number of positive examples classified over all the positive examples.

$$\text{Recall (R)} = \frac{TP}{TP + FN}$$

F1- measure is the normalized value of the two measurement metrics precision and recall performance measures.

$$\text{F1-measure} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} * \text{Recall}}$$

The simplest evaluation metric is accuracy so the overall effectiveness of the algorithm is calculated by dividing the correct labeling against all detection.

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

3.7. Summary

In this chapter, we presented the research methodology followed, development tools, data collection, text preprocessing operations, word representations applied, and the architecture of the proposed work. We followed an Experimental research methodology and used a python programming language development environment. The dataset is collected from Amharic Idiomatic books, fiction books, educational books, and historical books. After the dataset was collected, it was annotated by the annotators using different annotation guidelines. After the annotation, designed preprocessed algorithm for stop word and punctuation mark removal, and normalization is applied. Keras embedding, word2vec, and TFIDF have been experimented with as word representations. CNN, LSTM, and BiLSTM deep learning algorithms as well as SVM, and KNN machine learning algorithms were attempted for Amharic Idiomatic expression identification and giving its corresponding hidden meaning task. Finally, the performance measure is taken place using recall, precision, f1-measure, and accuracy measurement metrics.

CHAPTER FOUR

4. Result and Discussion

4.1. Introduction

In this chapter, we have presented the dataset used, the algorithm we have experimented with, the results of the experiments done, and the performance measure for the proposed model. For experimentation purposes, we have collected a total of 4800 idiomatic and non-idiomatic Amharic sentences for classification based identification. Since there is a shortage of getting more data for our proposed work. Among these datasets 2400 sentences are Idioms and 2400 sentences are Non Idioms to ensure the data balance. We used the following algorithms: SVM, KNN, CNN, LSTM, and BiLSTM to build the model.

4.2. Experimentation Setups

4.2.1. Dataset Description and Distribution

For model development, we used 4800 Amharic sentences for idiomatic expression identification. Those datasets are divided into training, validation, and testing as summarized in table 3. Sentences with a maximum length of 23 words and a minimum length of 5 words are included in the dataset. The datasets that we get appear at a minimum of five words and a maximum of twenty-three words. Since machine learning or deep learning needs the same size input we used a technique called padding, to ensure that all the input sequence data is having the same length as the input points.

Table 3: Dataset Splitting Summary

Idiomatic Expression Identification Dataset				
	Training	Validation	Testing	Total
Idiomatic	1920	240	240	2400
Non-Idiomatic	1920	240	240	2400
Total	3840	480	480	4800

The dataset is split into training, validation, and testing which is presented in (Table 3) using 60:20:20, and 80:10:10 splitting ratios for the classification.

However, the second splitting ratio of 80:10:10 makes our model perform well. So, this ratio has been selected for our model splitting ratio. When we experiment using Bi-LSTM, it achieves an accuracy of 99%, and 97% for 80:10:10, and 60, 20, and 20 respectively. As much as possible the training ratio should be high and the testing, as well as validation ratio, should be small for a small dataset. Comparably, our dataset is small and that is why, the smallest splitting ratio (80:10:10) works well, as we have checked by experiment.

4.2.2. Environment and Hyperparameter Setups

We set up environmental and Hyperparameter setups for our experimentation (training and testing). We used an HP laptop with an Intel(R) Core(TM) i3-4000U CPU@ 2.40GHz 2.40 GHz processor and 4GB RAM for training and testing. To evaluate the selected algorithms CNN, LSTM, and BiLSTM for idiomatic expression identification. Keras embedding and Word2vec are used as word representation for the deep learning approach whereas, TFIDF word representation is used for the machine learning approach to represent the terms in vector form. We have used the major hyperparameter shown in Table 4. The training accuracy, loss accuracy, and testing accuracy result of the experiment were documented in Appendix B.

Table 4: Hyperparameter Setups

No	**Hyperparameter**		**Experiment 1**	**Experiment 2**	**Experiment 3**
1	Batch Size	Training	128	64	32
		Validation	64	32	16
		Testing	64	32	16
2	Learning Rate		0.01	0.0001	0.1
3	Dropout		0.5	0.2	0.4
4	Activation Function		Tanh	Sigmoid	ReLU
5	Epoch		10	20	15
6	Optimizer		Nadam	Adam	SGD
7	Sequence Length		23	23	23
8	Embedding Dimension		128	200	64

For experimentation, the above Hyperparameters are selected after trying different setups (Experiment 1, Experiment 2, and Experiment 3) which perform well. We have developed the model using the aforementioned Hyperparameter setups and the result that we are presented is based on those setups. The Hyperparameter setup with batch size 64, 32, and 32 for training, validation, and testing respectively, learning rate 0.0001, dropout 0.2, and epoch 20 performs better. We discovered that the aforementioned Hyperparameter value allows us to improve training and prediction/testing performance through experimentation. Too small a batch size takes a long training time which creates noise for our model and large batches lead the model to have poor generalizing power which creates a lower model performance. Therefore the intermediate batch size 64/32/32 makes the proposed model perform well by eliminating the problems of small and large batch sizes.

A high or low learning rate value forces the loss function not to be reduced to the optimal range. As a result the learning rate 0.0001 value works well for our work by minimizing the loss and increasing the accuracy. The importance of using the dropout layer with a specified value is used to control the model overfitting. Using minimum dropout leads to occur underfitting and maximum dropout leads to overfitting. So for this work, we used a 0.2 dropout value to eliminate the overfitting problem by increasing the accuracy and by reducing the loss. A large epoch value makes the model overlearn whereas; a low epoch number makes the model not learn enough which reduced the model's performance.

4.2.3. Hyperparameter Setups for Machine Learning

We have used the major Hyperparameter (kernel = linear) for SVM algorithm, (n_neighbors=50) for KNN and (max_features=100, random_state=100) for both experimentations SVM, and KNN. We have used a grid search cross-validation to select the best hyper-parameters. We have developed the model using the aforementioned hyper-parameter setups and the result that we are presented is based on those setups. We discovered that the aforementioned hyper-parameter value allows us to improve training and prediction/testing performance through experimentation.

The hyper-parameter (kernel= linear) makes the SVM works relatively well by creating a clear margin separation between classes. There is no probabilistic justification for the

classification because the support vector classifier works by placing data points above and below the classifying hyperplane. So that, to draw a clear margin between the classes and get a better result we used the linear kernel as we have checked by the experiment.

4.2.4. Model configuration

After we finished the required experimentation setups, we built the proposed model. Keras's deep learning API is used to develop the proposed model. As depicted in figure 12, it has five hidden layers embedding layer, the Bi-LSTM layer which is the hidden layer, flatten layer, the dropout layer, and the dense layer are confined in the model configuration. The model holds 10.04 million parameters and among these parameters all are trainable. Finally, by fitting the model development dataset, the model was trained and validated. In the CNN model configuration there are seven hidden layers which are the embedding layer, conv1D layer, Max pooling layer, two dense layers for input, flatten layer, and dense for the output layer. LSTM model configuration has three hidden layers which are the embedding layer, LSTM layer, and dense layer.

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_4 (Embedding)     (None, 23, 200)           10000000

 bidirectional_1 (Bidirectio  (None, 50)               45200
 nal)

 flatten_3 (Flatten)         (None, 50)                0

 dropout_1 (Dropout)         (None, 50)                0

 dense_8 (Dense)             (None, 2)                 102

=================================================================
Total params: 10,045,302
Trainable params: 10,045,302
Non-trainable params: 0
_________________________________________________________________
```

Figure 12: BiLSTM Model Configuration

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 embedding_1 (Embedding)      (None, 23, 200)           10000000

 conv1d_1 (Conv1D)            (None, 22, 100)           40100

 max_pooling1d_1 (MaxPooling  (None, 11, 100)           0
 1D)

 dense_3 (Dense)              (None, 11, 64)            6464

 dense_4 (Dense)              (None, 11, 32)            2080

 flatten_1 (Flatten)          (None, 352)               0

 dense_5 (Dense)              (None, 2)                 706

=================================================================
Total params: 10,049,350
Trainable params: 10,049,350
Non-trainable params: 0
_________________________________________________________________
```

Figure 13: CNN Model Configuration

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 embedding_2 (Embedding)      (None, 23, 200)           10000000

 lstm (LSTM)                  (None, 100)               120400

 dense_6 (Dense)              (None, 2)                 202

=================================================================
Total params: 10,120,602
Trainable params: 10,120,602
Non-trainable params: 0
_________________________________________________________________
```

Figure 14: LSTM Model Configuration

4.3. Experimentation Result of Idiomatic Expression Identification Model for Machine Learning Approaches

In this section, we have presented the training and testing performance of SVM, and KNN models for the classification of idiomatic expressions. As we have put in the Hyperparameter setups (section 4.2.3), we used the hyperparameters listed there to experiment. The performance results of the selected models are presented below.

a) Experimentation Result of SVM model for identification of idiomatic expression

The performance result of the SVM model according to the different hyperparameters is presented as follows.

Table 5: SVM model performance measure

Experiments		Precision (avg)	Recall (avg)	Fl-score (avg)	Accuracy
Experiment 1	Kernel: 'rbf'	86%	86%	86%	85.71%
Experiment 2	Kernel: 'sigmoid'	84%	86%	85%	84.01%
Experiment 3	Kernel: 'linear'	87%	86%	86%	86.39%

As described in table 5 support vector machine model achieves good performance in experiment three in terms of accuracy including precision, recall, and f1-score measurement metrics. However, SVM showed poor performance in experiment two. We understand that this is because of the hyperparameter tuning.

```
Confussion matrix of SVM
[[139  23]
 [ 17 115]]
              precision    recall  f1-score   support

           0       0.89      0.86      0.87       162
           1       0.83      0.87      0.85       132

    accuracy                           0.86       294
   macro avg       0.86      0.86      0.86       294
weighted avg       0.87      0.86      0.86       294

Model test accuracy
Accuracy: 86.39%
```

Figure 15: confusion matrix of SVM with experiment three

b) Experimentation Result of KNN model for identification of idiomatic expression

The K-nearest neighbor model performance measure is presented below.

Table 6: K-Nearest Neighbor Classifier performance measure

Experiments		Precision (avg)	Recall (avg)	Fl-score (avg)	Accuracy
Experiment 1	n_neighbors=50	86%	85%	85%	85.03%
Experiment 2	n_neighbors=90	81%	80%	80%	79.59%
Experiment 3	n_neighbors=100	81%	79%	79%	78.91%

As shown in table 6 in experiment one the K-Nearest Neighbor achieved good performance in accuracy additionally the performance measures precision, recall, and fl-score. However, in experiment three KNN shows poor performance. As we have seen by the experiment the performance variation appears because of the Hyperparameter tuning.

```
Confussion matrix of KNN
[[66 15]
 [ 7 59]]
              precision    recall  f1-score   support

           0       0.90      0.81      0.86        81
           1       0.80      0.89      0.84        66

    accuracy                           0.85       147
   macro avg       0.85      0.85      0.85       147
weighted avg       0.86      0.85      0.85       147

Model test accuracy
Accuracy: 85.03%
```

Figure 16: confusion matrix of KNN with experiment one

4.3.1. Comparison of machine learning models

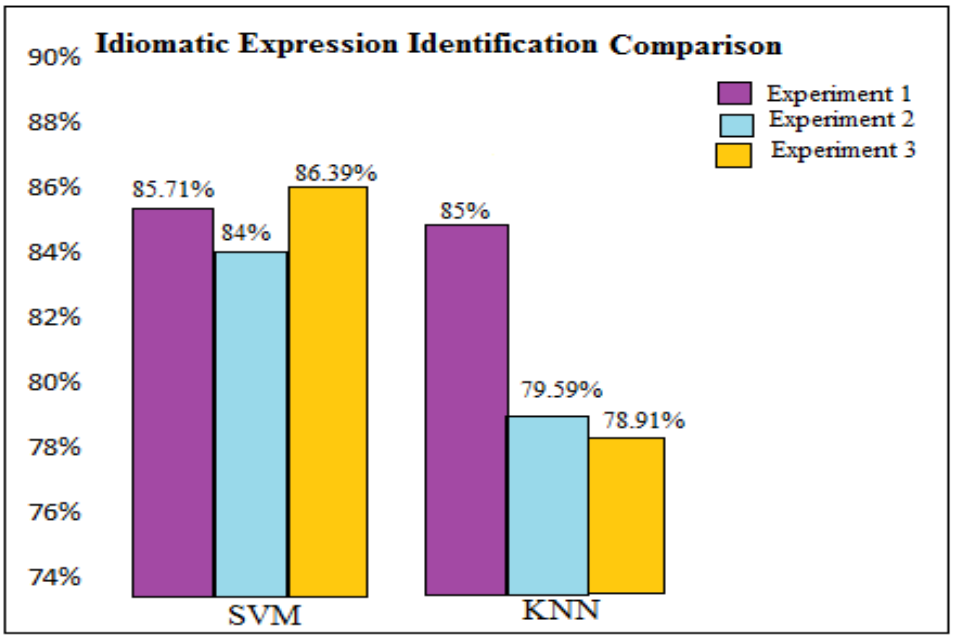


Figure 17: Comparison of SVM and KNN using three experiments

The above graph (figure 17) shows the comparison of two machine learning algorithms which are SVM and KNN. As we see in the graph support vector machine achieved a good performance than KNN algorithms. The reason behind it is to differentiate the data SVM uses threshold functions, not the number of features based on margin. This shows that SVM does not depend upon the presence of a number of features in data. In addition, because of the kernel function SVM can perform best in differentiating non-linear data from linear data by constructing the hyper-plane between them. K-nearest neighbor achieved a little bit lower performance result than the support vector machine. The lowest performance of KNN is due to the laziness of the learning algorithm. This means that it does not learn a discriminative function from the training data rather it memorizes the training dataset.

4.4. Experimentation Result of Idiomatic Expression Identification Model using DL

In this section, we have presented the training, validation, and testing performance of CNN, LSTM, and Bi-LSTM models for the classification of idiomatic expressions. We have evaluated the three models using the same Hyperparameter as we have put in the environment and Hyperparameter setups section; we used the second hyperparameter setup (experiment 2) which works well. The performance results of the selected models are presented below.

A) Experimentation Result of CNN model for identification of idiomatic expression

The training and validation loss was initially very substantial, as seen in figure 19. However, the loss also decreased as the number of epoch rounds rise. When we looked at the training and validation accuracy of CNN, it was 94 % and 93 %, respectively, as shown in Figure 18. These models' accuracy fell somewhere at the last after Bi-LSTM and LSTM. The performance result of the CNN model according to the different Hyperparameters is presented as follows in Figure 18 and Figure 19.

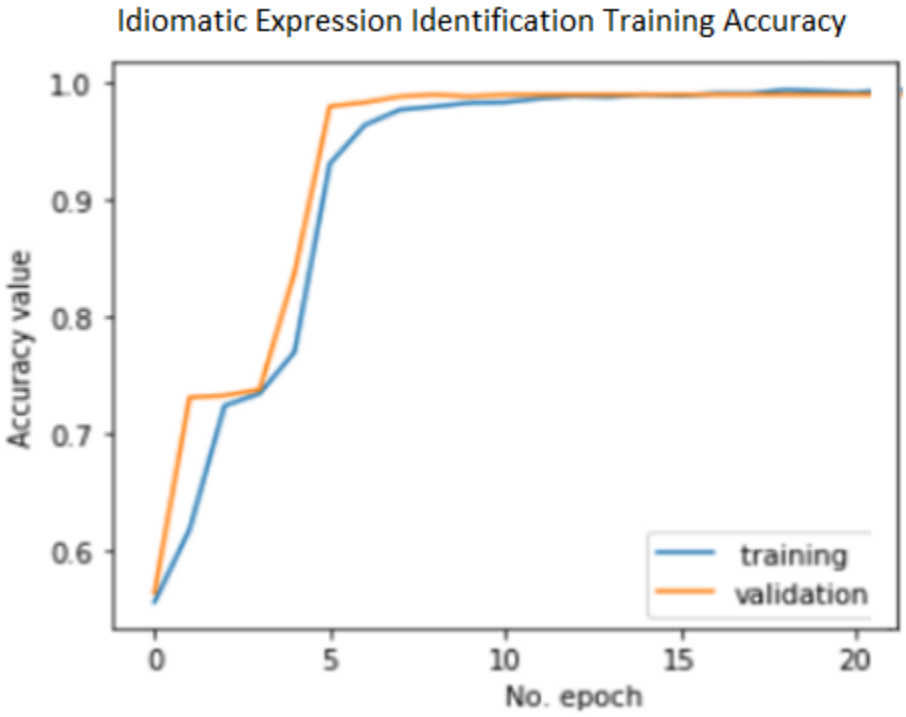


Figure 18: Training and Validation Accuracy of CNN model on experiment two

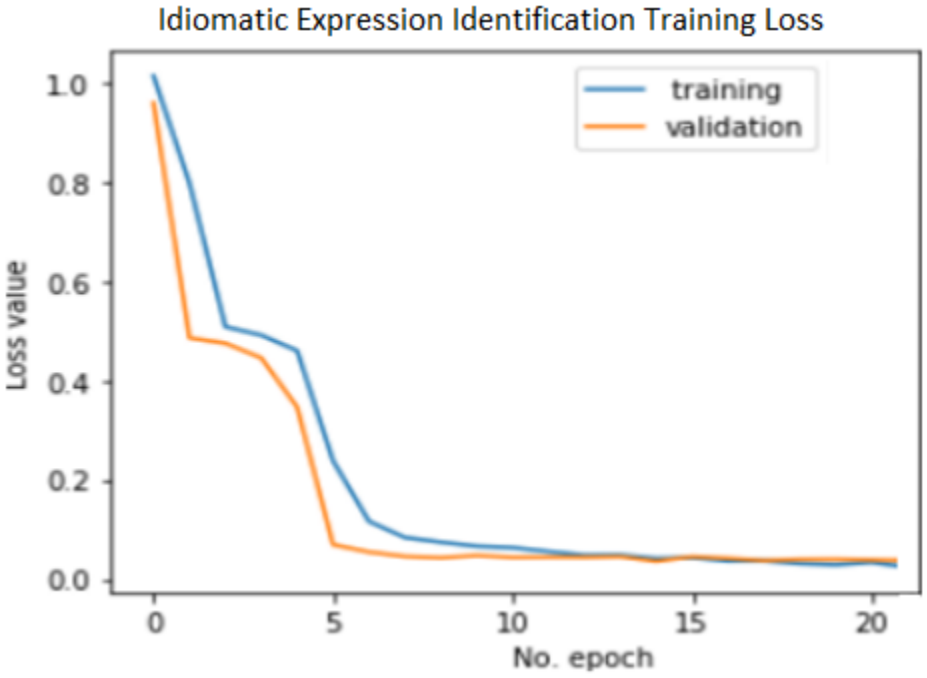


Figure 19: Training and Validation Loss of CNN model on experiment two

As described in table 7 CNN model achieved better performance in terms of test accuracy 94% including precision 94%, recall 93%, and f1 score of 93.5% on experiment two because of the Hyperparameter tuning using the test set data.

Table 7: CNN model performance for three experiments

Experiments	Precision (avg)	Recall (avg)	Fl-score (avg)	Accuracy
Experiment 1	86%	85%	85%	85.03%
Experiment 2	94%	93%	93.5%	94%
Experiment 3	86%	84%	85%	86%

B) Experimentation Result of LSTM model for identification of idiomatic expression

In Figure 21, we see that the training and validation loss was very high. However, as the epoch round increases the training and validation loss decrease. At some epochs the training and validation loss was 69% and 67% respectively however, for the last epochs 17 to 20 it was reduced to 30% and 32%. The training and validation accuracy also increased from 50% and 51% to 97% and 96% respectively as we can see from Figure 20 and Figure 21.

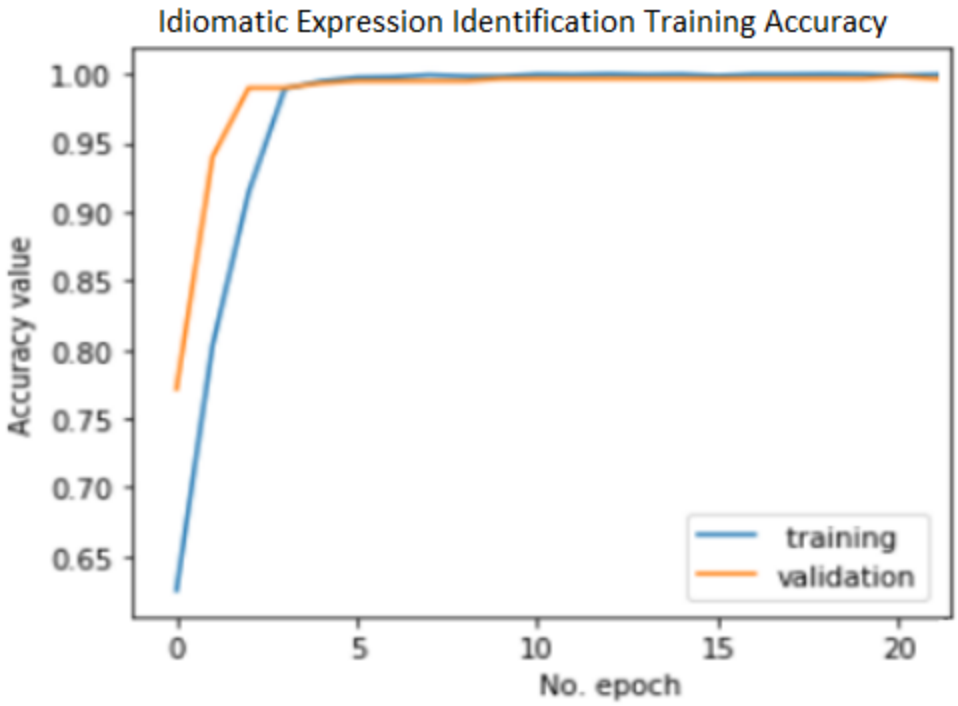


Figure 20: Training and Validaiton Acccuracy of LSTM Model on experiment two

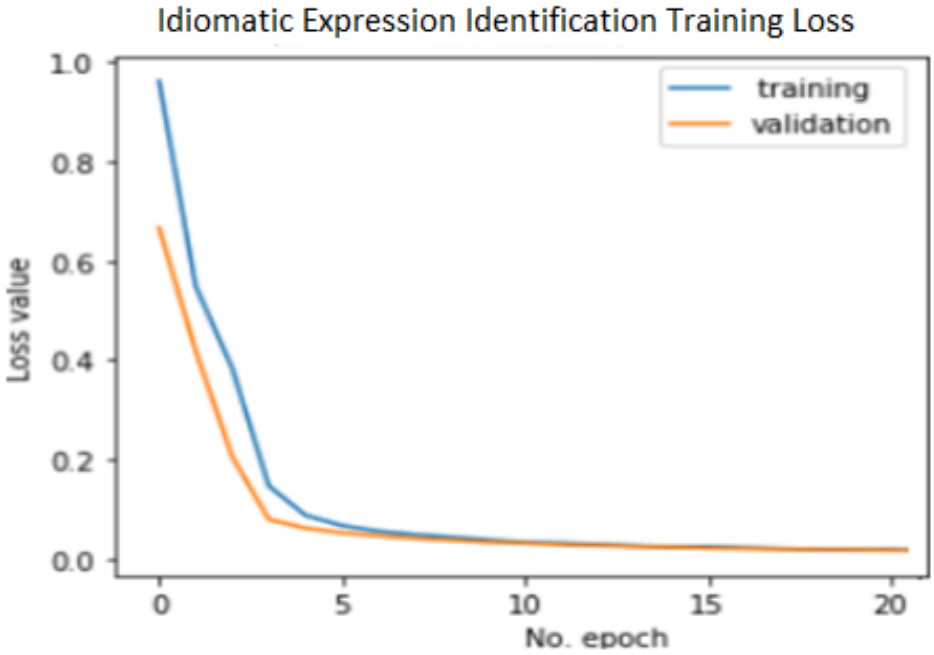


Figure 21: Training and Validation Loss of LSTM Model on experiment two

Table 8 shows the results of the three experiments conducted using the LSTM algorithm. Alike CNN, LSTM also obtained an accuracy of 95% in the second experiment. This depicts that the Hyperparameter tuning improves the results of the model.

Table 8: LSTM model performance for three experiments

Experiments	Precision (avg)	Recall (avg)	Fl-score (avg)	Accuracy
Experiment 1	89%	87%	88%	89%
Experiment 2	95%	93%	94%	95%
Experiment 3	85%	84%	84.5%	85%

As we can see from above table 6 it is clear that LSTM achieves good performance in terms of accuracy 95% as well as precision 95%, recall 93%, and f1-score 94% in experiment two. However, the LSTM model performs poorly in experiment three even compared to the CNN algorithm. We understand that the Hyperparameter tuning makes a clear difference.

C) Experimentation Result of Bi-LSTM model for identification of idiomatic expression

As shown in Figure 22 and Figure 23 the training and validation loss of the Bi-LSTM model was 68% and 69% at the initial state respectively and at the end of the training it becomes 17% and 28%. According to the training and validation accuracy at the initial point, it was 53% and 51% respectively. However, in the end, it becomes 99% and 91%. 99.75% accuracy from the test set. When we compared the Bi-LSTM model training and validation accuracy with LSTM, Bi-LSTM outperforms LSTM.

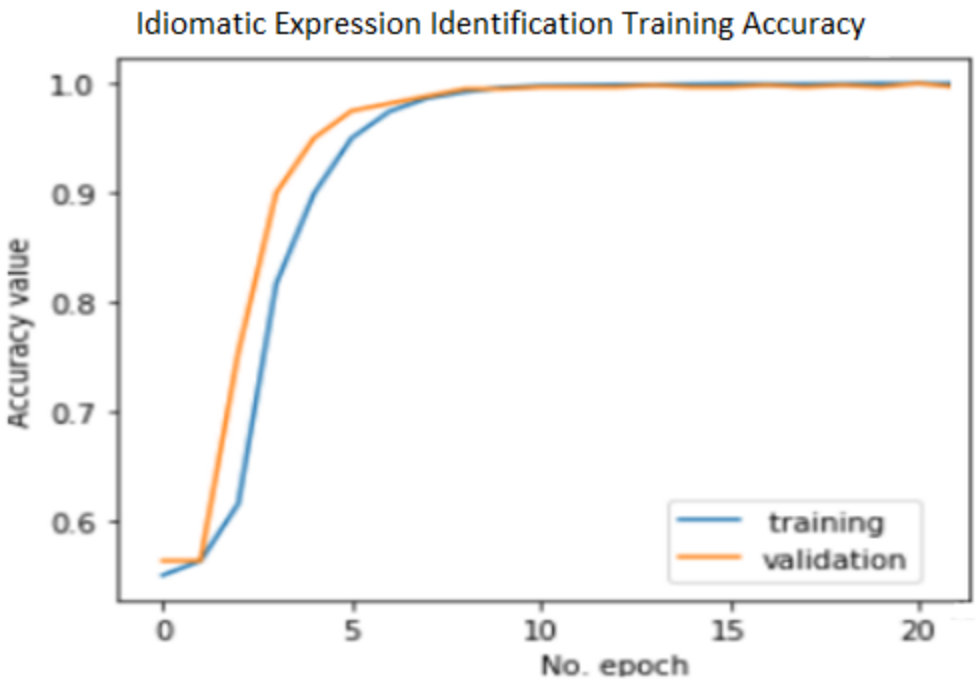


Figure 22: Training and Validation Accuracy of Bi-LSTM Model on experiment two

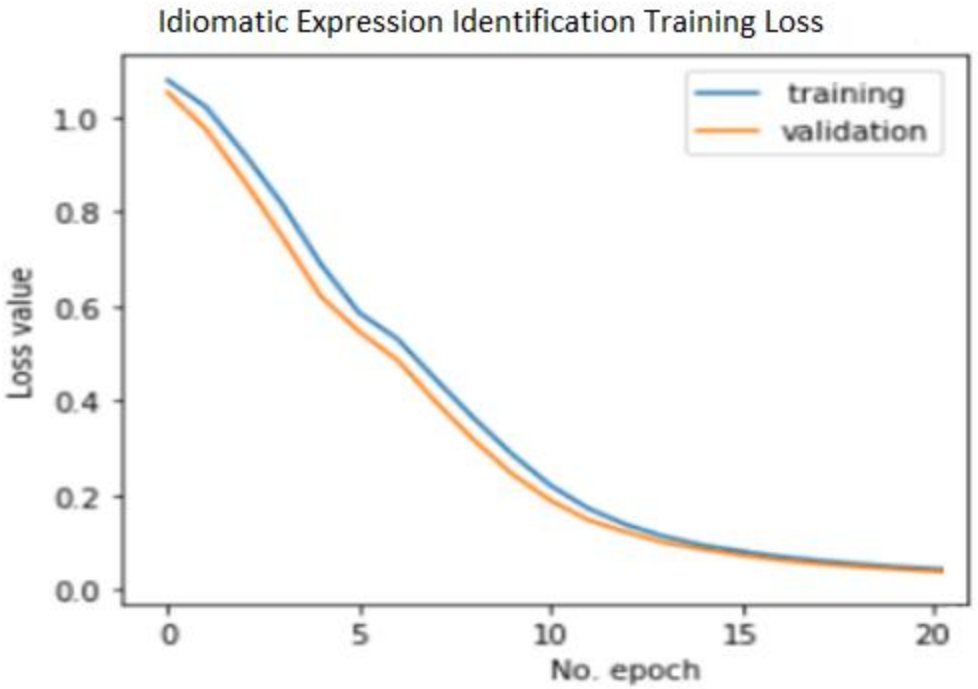


Figure 23: Training and Validation Loss of Bi-LSTM Model in experiment two

As shown in figure 22 and figure 23 the training and validation loss was high at the initial epochs, however, the loss decreased as the epoch increased. As the epoch increased the training and validation accuracy improved. The model learns the training dataset correctly as the learning iteration increases as shown in the above figures.

```
Epoch 1/20
31/31 [==============================] - 29s 594ms/step - loss: 0.6922 - accuracy: 0.5192 - val_loss: 0.6905 - val_accuracy: 0.5182
Epoch 2/20
31/31 [==============================] - 18s 574ms/step - loss: 0.6894 - accuracy: 0.5207 - val_loss: 0.6882 - val_accuracy: 0.5182
Epoch 3/20
31/31 [==============================] - 17s 535ms/step - loss: 0.6862 - accuracy: 0.5207 - val_loss: 0.6852 - val_accuracy: 0.5182
Epoch 4/20
31/31 [==============================] - 16s 517ms/step - loss: 0.6815 - accuracy: 0.5227 - val_loss: 0.6806 - val_accuracy: 0.5182
Epoch 5/20
31/31 [==============================] - 16s 519ms/step - loss: 0.6741 - accuracy: 0.5343 - val_loss: 0.6727 - val_accuracy: 0.5409
Epoch 6/20
31/31 [==============================] - 17s 542ms/step - loss: 0.6609 - accuracy: 0.5813 - val_loss: 0.6584 - val_accuracy: 0.5682
Epoch 7/20
31/31 [==============================] - 16s 524ms/step - loss: 0.6379 - accuracy: 0.6424 - val_loss: 0.6352 - val_accuracy: 0.6182
Epoch 8/20
31/31 [==============================] - 16s 529ms/step - loss: 0.6016 - accuracy: 0.7348 - val_loss: 0.6000 - val_accuracy: 0.7273
Epoch 9/20
31/31 [==============================] - 16s 516ms/step - loss: 0.5490 - accuracy: 0.8318 - val_loss: 0.5503 - val_accuracy: 0.8318
Epoch 10/20
31/31 [==============================] - 16s 510ms/step - loss: 0.4863 - accuracy: 0.9167 - val_loss: 0.4981 - val_accuracy: 0.9182
Epoch 11/20
31/31 [==============================] - 16s 531ms/step - loss: 0.4255 - accuracy: 0.9561 - val_loss: 0.4521 - val_accuracy: 0.9227
Epoch 12/20
31/31 [==============================] - 16s 518ms/step - loss: 0.3773 - accuracy: 0.9783 - val_loss: 0.4169 - val_accuracy: 0.9182
Epoch 13/20
31/31 [==============================] - 16s 518ms/step - loss: 0.3374 - accuracy: 0.9808 - val_loss: 0.3898 - val_accuracy: 0.9318
Epoch 14/20
31/31 [==============================] - 17s 543ms/step - loss: 0.3031 - accuracy: 0.9854 - val_loss: 0.3714 - val_accuracy: 0.9045
Epoch 15/20
31/31 [==============================] - 17s 562ms/step - loss: 0.2751 - accuracy: 0.9884 - val_loss: 0.3499 - val_accuracy: 0.9136
Epoch 16/20
31/31 [==============================] - 17s 539ms/step - loss: 0.2522 - accuracy: 0.9914 - val_loss: 0.3287 - val_accuracy: 0.9227
Epoch 17/20
31/31 [==============================] - 16s 517ms/step - loss: 0.2269 - accuracy: 0.9914 - val_loss: 0.3169 - val_accuracy: 0.9045
Epoch 18/20
31/31 [==============================] - 17s 556ms/step - loss: 0.2109 - accuracy: 0.9939 - val_loss: 0.3016 - val_accuracy: 0.9091
Epoch 19/20
31/31 [==============================] - 16s 528ms/step - loss: 0.1921 - accuracy: 0.9960 - val_loss: 0.2905 - val_accuracy: 0.9182
Epoch 20/20
31/31 [==============================] - 18s 578ms/step - loss: 0.1789 - accuracy: 0.9939 - val_loss: 0.2809 - val_accuracy: 0.9136
```

Figure 24: Selected Model Training and Validation Status

The Bi-LSTM model performance evaluation result is presented in Table 9 below. As is shown in figure 22 that Bi-LSTM achieved good performance in experiment two as compared to experiment one and experiment three. The low prediction performance comes from the Hyperparameter tuning done in experiments.

Table 9: Bi-LSTM model performance for three experiments

Experiments	Precision (avg)	Recall (avg)	Fl-score (avg)	Accuracy
Experiment 1	89%	87%	88%	89%
Experiment 2	99%	97%	98%	99%
Experiment 3	88%	88%	88%	88%

4.4.1. Comparison of the selected models

For idiomatic expression identification, we evaluated the selected models using different Hyperparameter tuning to enhance the model's prediction performance. The training time and prediction performance are the main evaluation methods for the result achieved. When we see the training time taken by the model developed at the training phase, 10 minutes, 7 minutes, and 5 minutes are taken by Bi-LSTM, LSTM, and CNN respectively. Bi-LSTM takes much more time for training as compared to LSTM and CNN. The LSTM model takes a smaller time than the Bi-LSTM model and a longer time than the CNN model. However, the CNN model takes a shorter training time than the two models which are Bi-LSTM and LSTM. Bi-LSTM performs better than LSTM and CNN. The prediction performance of the three modes is summarized below.

Based on previously learned features the model can predict idiomatic expressions accurately. The performance of the three prediction models has presented in Figure 20 in terms of accuracy. Among the three experimental setups, the Hyperparameter setup used in experiment 2 performs better than the other two Hyperparameter setups (experiment one and experiment three). The second experiment obtained a better result in all the algorithms implemented, Bi-LSTM (99%), LSTM (95%), and CNN (94%).

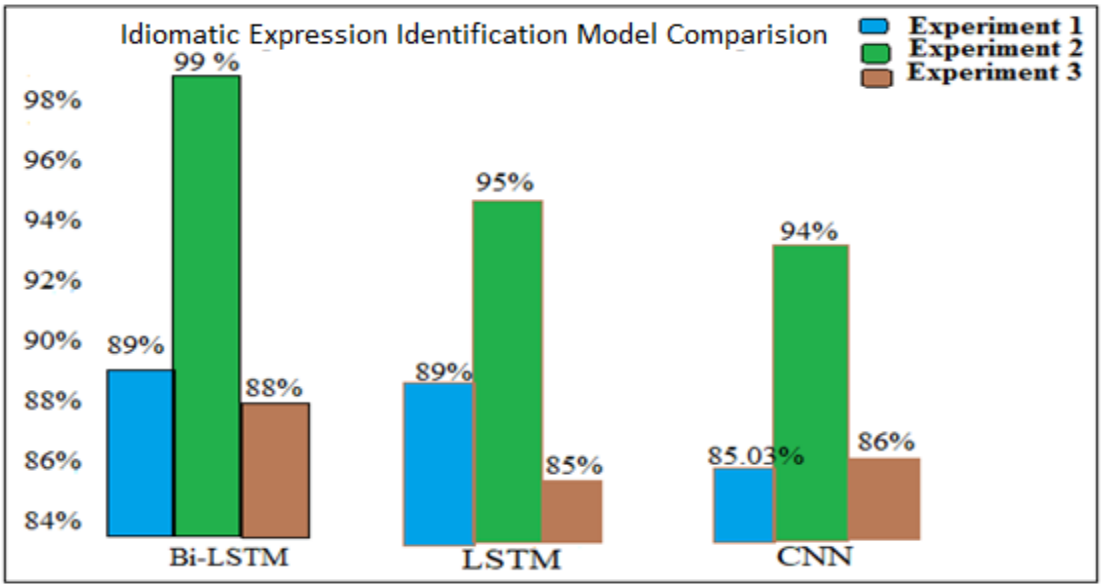


Figure 25: Idiomatic expression identification comparison using accuracy in three experiments

The above graph (figure 25) shows the comparison of three deep learning algorithms which are CNN, LSTM, and BI-LSTM. As it can be seen in the graph BI-LSTM achieved the highest prediction performance of the other two algorithms. The reason might be that Bi-LSTM works by context understanding than LSTM and CNN by using two hidden layers which are forward and backward. The variation showed epochs, batch size, and learning rate. An intermediate training batch size is preferable for a model building with good accuracy. Too small epoch and learning rate show high training and validation loss and small training and validation accuracy since the model does not get the chance to learn more. So, an average epoch value works better for this work.

4.5. Idiomatic Expression Identification from ML and DL

To identify the idiomatic expression terms from the given text once the trained sentences are categorized as Idiom and Non-Idiom using algorithms of SVM, KNN, CNN, LSTM, and BiLSTM. The idiom sentences go for the identification of existing idiomatic expression terms and give their hidden meaning process by developing a word2vec model which is presented in (Appendix E). If the given sentence contains an idiomatic expression term, it identifies that word, phrase, or clause and gives its corresponding hidden meaning according to the trained context.

The word2vec model, which employs a neural network architecture embedding layer, is developed for idiomatic expression identification. Python Gensim library was used to train the Amharic word2vec with various parameters like window size which is used to see the maximum distance between words around the target term. In our case Workers are set 2 to apply training parallelization to speed up. The vector size is set to 100 even if bigger size values can lead to a more accurate model it requires more training data. Min count value is used as 1 to see if the term appears once from a given text. The Sg parameter used as 0 to specify CBOW is taken and 1 to indicate skip-gram is used. In this embedding, it is trained using a sentences dataset. To build this model we used a total dataset of 4800 sentences and trained using Continuous Bag of Words (CBOW) which predicts the words that fall in between or predict the target word from the context. Whereas, in skip-gram, the neural network takes in a word to predict the surrounding words or predict the target words from context.

Results from the developed word2Vec Model

```
Enter the sentence
ለመሆኑ ይህ ኩክ_ማውጫ ጓደኛህ ጤነኛ ነው ትንሽ ምክር ቢጤ ያስፈልገዋል ባይ ነኝ::
Idiomatic Expression: ኩክ_ማውጫ The Meaning: ወራኛ፣ አሳባቂ

Enter the sentence
ዘንድሮ በነበረው የጥምቀት በአል አከባበር ላይ የተገኘው ህዝብ እልቆ_መሳፍርት አልነበረውም::
Idiomatic Expression: እልቆ_መሳፍርት The Meaning: ስፍር ቁጥር የሌለው

Enter the sentence
ታናሽ እህቱ ግን አሁንም በሕይወቱ ውስጥ መመላለሷን ቀጥላለች::
sentence ታናሽ እህቱ ግን አሁንም በሕይወቱ ውስጥ መመላለሷን ቀጥላለች:: has no Idiomatic Expression
```

Table 10: Comparison of the proposed model with CNN, SVM, LSTM, and KNN models

Models	**Accuracy**
CNN	94%
SVM	86%
LSTM	95%
KNN	85%
Proposed Model (BiLSTM)	99%

4.6. Discussion

In this work, we have experimented with three deep learning and two machine learning algorithms for the idiomatic expression identification model. We looked at the three models on different Hyperparameter setups to see how the Hyperparameter setups affected the results and which model perform the best. Those three models' performance was measured using recall, precision, f-measure, and accuracy.

In addition, we employed a confusion matrix to administer how many of the testing samples were correctly classified and how many were wrongly classified/misclassified for model error analysis.

We also observed how stopword removal, normalization, and punctuation mark removal affected the results. First, we tried to stop word removal and achieved an accuracy of 93%, normalization and achieved an accuracy of 92%, and punctuation mark removal achieved an accuracy of 94% separately and together achieved an accuracy of 97% for the BiLSTM model. We achieved 90% accuracy without using the preprocessing task. We also recognize that using all the aforementioned text preprocessing activities improves the proposed work's performance.

For the word representation technique, we have used Keras embedding, word2vec for DL, and TFIDF for ML. For the experimentation, we used conventional neural networks, Long Short Term Memory, Bidirectional Long Short Term Memory algorithms from the deep learning approach as well as Support vector machine, and K-nearest neighbor algorithms from machine learning. From those deep learning algorithms CNN, LSTM and BiLSTM, BiLSTM has got a large score than CNN and LSTM. The reason is BiLSTM model is more capable of context understanding than LSTM since it uses two hidden layers. The first hidden layer processes the sequence forward and the second hidden layer processes the sequence backward which makes the BiLSTM capable to capture the context of words in the input sentence (Kamath et al., 2018). From machine learning algorithms SVM achieved a better result than KNN because of its kernel and threshold function which gives the capability to differentiate the given data. In General, BiLSTM outperforms LSTM, CNN, SVM, and KNN since; it can understand the backward as well as the forward context of the given data. As we understand from our experimentation Hyperparameter tuning for three algorithms produces a change in the model’s prediction performance. To summarize, we have done the analysis and compared those models' prediction performances.

In the error analysis phase, we have seen that there is model overfitting. As we can deduce from our model network, overfitting results from two fundamental factors: the number of hidden layers and the dropout rate. We've tried three different methods to manage and resolve this issue. First, we attempt to correct the problem by reducing the number of hidden layers. Second, we've made an effort to expand the dataset by gathering more data. Thirdly, we added a dropout layer and once again we modified it by

raising the dropout value during model construction. The overfitting issue is resolved and the performance of the suggested model is improved with the use of the aforementioned strategies.

From the idiomatic expression identification task we have seen that for most of the experiments, for the idiomatic class precision is high, however, for the non-idiomatic class, the recall is large. This shows that most non-idiomatic samples from the testing dataset are correctly classified as non-idiomatic and the idiomatic class higher precision value shows that from the testing dataset most of the classified samples as idiomatic are real idioms. Generally, this shows that the model performed well. We tried to minimize the problems encountered by collecting a large dataset as much as possible to enhance the prediction/identification performance of the model.

To sum up, from the above Hyperparameter setup shown in Table 4 experiment two performs better. Among the three algorithms Bi-LSTM (99%), LSTM (95%), and CNN (94%). Bi-LSTM performs better than the other two algorithms CNN and LSTM for idiomatic identification. Bi-LSTM is a strong tool for modeling sequential dependencies between words and phrases in both directions of the sequence; thus it performs better than CNN and LSTM. The reason behind this is that Bi-LSTM is made up of two hidden layers. The first LSTM layer processes the sequence forward and the second LSTM layer on the other side processes the sequence backward (reversion flow as seen in Figure 12).

When we look at CNN's performance, we observed that it did well for sentences with few words. For lengthy sentences, it is unable to deliver the intended results. Because most of the sentences in this dataset are long, with a maximum of 23 words, CNN does not perform well overall for the suggested model. Additionally, unlike CNN, unidirectional LSTM handles the context of words in long sentences while analyzing performance. However, unlike Bi-LSTM, LSTM is unable to capture word context in both the forward and backward directions. In conclusion, Bi-LSTM beats CNN and LSTM due to the shortcomings of CNN and LSTM that have been discussed and looked into during experimentation.

CHAPTER FIVE

5. Conclusion and Future Work

5.1. Conclusion

In this work, we have experimented with five algorithms SVM, KNN, CNN, LSTM, and BiLSTM by Hyperparameter tuning for idiomatic expression identification model for Amharic text. stopword removal, normalization, and punctuation mark removal preprocessing techniques are used since those tasks are beneficial for the proposed work. Word representation Keras embedding layer and word2vec for DL and TFIDF for ML are used to implement and test the selected algorithms. CNN takes the shorter training time, and LSTM and Bi-LSTM take the second and third respectively in terms of the model training time which is depicted in (section 4.4.1). The Bi-LSTM model performs well for idiomatic expression identification because of its forward and backward context handling features. Bi-LSTM achieved a prediction performance of 99%. Bi-LSTM improves a prediction performance of 4% and 5% for idiomatic expression identification as compared with LSTM and CNN respectively. CNN, SVM, LSTM, KNN, and BiLSTM achieved accuracy of 94%, 86%, 95%, 85%, and 99% respectively. As we have seen the results of all experimented algorithms BiLSTM outperforms SVM, KNN, CNN, and LSTM.

5.2. Contribution of the study

In this work, we have developed an Amharic idiomatic expression identification model using a deep learning approach. The key contributions of this work are:

- We have collected Amharic idiomatic expression datasets which can be an initial point for other researchers to study further. We have prepared 4800 sentences which are annotated by experts. Therefore, the datasets that are collected for this work can be used as an initial point for further investigation.
- We have developed a deep learning model which can classify and identify the idiomatic expressions from Amharic text. The model proposed for other languages cannot apply to the Amharic language due to morphology, grammar,

and semantics difference, so we can conclude that the proposed model is our contribution.

- This study is used to reduce the problems that happened to the existence of idiomatic expressions for different NLP applications like machine translation, semantic analysis, and sentiment analysis.

5.3. Future Work

In this study, we have developed an idiomatic expression identification model for Amharic texts. In addition, the model provides the hidden meaning of the idiom words, phrases, or clauses. However, this work does not address the nature of idiom words or phrases as pure or semi-pure. Therefore, it is one of the open research areas for further investigation. The datasets collected for this work may prone to spelling and grammar errors which might mislead the model in understanding the accurate context. Besides, spelling and grammar errors were not considered thus it needs integration. Finally, although a good amount of datasets is prepared in this study, we believe that increasing the dataset will enhance the performance of the model. Since the deep learning approach needs a huge amount of data to provide an accurate result.

References

Agarwal, V. (2015). Research on Data Preprocessing and Categorization Technique for Smartphone Review Analysis. *International Journal of Computer Applications*, *131*(4), 30–36. https://doi.org/10.5120/ijca2015907309

Ankit, & Saleena, N. (2018). An Ensemble Classification System for Twitter Sentiment Analysis. *Procedia Computer Science*, *132*(Iccids), 937–946. https://doi.org/10.1016/j.procs.2018.05.109

Ayu, I. G., & Dian, A. (2021). *The Analysis of Idiomatic Expression in Bidirectional Texts of Children Stories " Ceritarakyat Bali " a nd " t he Little Mermaid ": an Introduction of Idiomaticex Pression to Children*. *4*(1), 124–136.

Bouamor, H., Max, A., & Vilnat, A. (2011). Monolingual alignment by edit rate computation on sentential paraphrase pairs. *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, *2*, 395–400.

Bzdok, D., Krzywinski, M., & Altman, N. (2018). Points of significance: Machine learning: Supervised methods. *Nature Methods*, *15*(1), 5–6. https://doi.org/10.1038/nmeth.4551

Cacciari, C., & Glucksberg, S. (1991). Chapter 9 Understanding Idiomatic Expressions: The Contribution of Word Meanings. *Advances in Psychology*, *77*(C), 217–240. https://doi.org/10.1016/S0166-4115(08)61535-6

Callison-Burch, C., Koehn, P., Monz, C., & Zaidan, O. F. (2011). Findings of the 2011 Workshop on Statistical Machine Translation. *WMT 2011 - 6thWorkshop on Statistical Machine Translation, Proceedings of the Workshop*, 22–64. https://doi.org/10.3115/1626431.1626433

Cohen, K. B. (2013). Biomedical Natural Language Processing and Text Mining. In *Methods in Biomedical Informatics: A Pragmatic Approach* (Error). Elsevier Inc. https://doi.org/10.1016/B978-0-12-401678-1.00006-3

Dd, U.-, & Cft, U. C. (1989). *tlJD '89* 1*.

Edouard, A. (2017). *Event detection and analysis on short text messages*. Université Côte D'Azur.

Farhadloo, M., & Rolland, E. (2016). Fundamentals of sentiment analysis and its applications. *Studies in Computational Intelligence*, *639*, 1–24. https://doi.org/10.1007/978-3-319-30319-2_1

Fellbaum, C., Geyken, A., Herold, A., Koerner, F., & Neumann, G. (2006). Corpus-based studies of german idioms and light verbs. *International Journal of Lexicography*, *19*(4), 349–360. https://doi.org/10.1093/ijl/ecl031

Fenta, A. (2021). *AUTOMATIC IDIOM RECOGNITION MODEL FOR AMHARIC LANGUAGE*.

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 (Switzerland)*, *12*(1), 1–9. https://doi.org/10.3390/info12010020

Graves, A. (2012). Supervised sequence labelling. In *Supervised sequence labelling with recurrent neural networks* (pp. 5–13). Springer.

Hashimoto, C., Sato, S., & Utsuro, T. (2006). Japanese idiom recognition: Drawing a line between literal and idiomatic meanings. *COLING/ACL 2006 - 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Main Conference Poster Sessions*, *July*, 353–360. https://doi.org/10.3115/1273073.1273119

Hillert, D. G. (2004). Spared access to idiomatic and literal meanings: A single-case approach. *Brain and Language*, *89*(1), 207–215. https://doi.org/10.1016/S0093-934X(03)00384-5

Hinkel, E. (2017). Teaching Idiomatic Expressions and Phrases: Insights and Techniques. *Iranian Journal of Language Teaching Research*, *5*(3), 45–59. http://ijltr.urmia.ac.ir

Hordofa, B. A. (2020). *Event Extraction and Representation Model from News Articles*. *16*(3), 1–8.

Hossin M, Sulaiman MN, Mustpaha N, & Rahmat RW. (2011). Improving Accuracy Metric With Precision and Recall Metrics for Optimizing Stochastic Classifier. *Proc. of the Third Int. Conf. on Computing and Informatics, ICOCI*, *8*(052), 105–110. http://www.icoci.cms.net.my/proceedings/2011/papers/105.pdf

Javatpoint. (2018). Support Vector Machine (SVM) Algorithm - Javatpoint. In *JavaTpoint*. https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

Kamath, C. N. N., Bukhari, S. S., & Dengel, A. R. (2018). Comparative Study between Traditional Machine Learning and Deep Learning Approaches for Text Classification. *Proceedings of the ACM Symposium on Document Engineering 2018*.

Kumar, S., Behera, P., & Jha, G. N. (2017). A classification-based approach to the identification of Multiword Expressions (MWEs) in Magahi Applying SVM. *Procedia Computer Science*, *112*, 594–603. https://doi.org/10.1016/j.procs.2017.08.059

Ly, A., Uthayasooriyar, B., & Wang, T. (2020). A survey on natural language processing (nlp) and applications in insurance. *ArXiv Preprint ArXiv:2010.00462*.

Mantyla, K. (2004). Idioms and Language Users: The Effect of the Characteristics of Idioms on Their Recognition and Interpretation by Native and Nonnative Speakers of English. In *Dissertation Abstracts International, C: Worldwide*. http://search.proquest.com/docview/85637231?accountid=8330%5Cnhttp://library.anu.edu.au:4550/resserv?genre=dissertations+&+theses&issn=&title=Idioms+and+Language+Users:+The+Effect+of+the+Characteristics+of+Idioms+on+Their+Recognition+and+Interpretation+by

Maslej-Krešňáková, V., Sarnovský, M., Butka, P., & Machová, K. (2020). Comparison of deep learning models and various text pre-processing techniques for the toxic comments classification. *Applied Sciences*, *10*(23), 8631.

Maxion, P. R. A. (2009). *Experimental Methods for Computer Science*. 7695. https://doi.org/10.1109/LADC.2009.29

Muzny, G., & Zettlemoyer, L. (2013). Automatic idiom identification in Wiktionary. *EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, October*, 1417–1421.

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. *ArXiv Preprint ArXiv:1811.03378*.

Owens, J. (2016). The lexical nature of idioms. *Language Sciences*, *57*, 49–69. https://doi.org/10.1016/j.langsci.2016.05.004

Peng, J., & Feldman, A. (2016). Experiments in idiom recognition. *COLING 2016 - 26th International Conference on Computational Linguistics, Proceedings of COLING 2016: Technical Papers*, 2752–2761.

Phung, V. H., & Rhee, E. J. (2019). A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets. *Applied Sciences*, *9*(21), 4500.

Popovíc, M., Ney, H., De Gispert, A., Mariño, J. B., Gupta, D., Federico, M., Lambert, P., & Banchs, R. (2006). Morpho-syntactic information for automatic error analysis of statistical machine translation output. *HLT-NAACL 2006 - Statistical Machine Translation, Proceedings of the Workshop*, *June 2006*, 1–6. https://doi.org/10.3115/1654650.1654652

Salton, G. D., Ross, R. J., & Kelleher, J. D. (2016). Idiom token classification using sentential distributed semantics. *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, *1*, 194–204. https://doi.org/10.18653/v1/p16-1019

Salton, G. D., Ross, R. J., & Kelleher, J. D. (2017). Idiom type identification with smoothed lexical features and a maximum margin classifier. *International Conference Recent Advances in Natural Language Processing, RANLP*, *2017-Septe*(2009), 642–651. https://doi.org/10.26615/978-954-452-049-6-083

Sharma, A., Khan, F., Sharma, D., Gupta, S., & Student, F. Y. (2020). Python: The

Programming Language of Future. *International Journal of Innovative Research in Technology*, *6*(12), 115–118. http://ijirt.org/master/publishedpaper/IJIRT149340_PAPER.pdf

Shim, H., Shin, N., Stern, A., Aharon, S., Binyamin, T., Karmi, A., Rotem, D., Etgar, L., Porath, D., Pradhan, B., Kumar, G. S., Sain, S., Dalui, A., Ghorai, U. K., Pradhan, S. K., Acharya, S., Quan, L. N., Rand, B. P., Friend, R. H., … Gmbh, Z. (2018). No 主観的健康感を中心とした在宅高齢者における 健康関連指標に関する共分散構造分析Title. *Advanced Optical Materials*, *10*(1), 1–9. https://doi.org/10.1103/PhysRevB.101.089902%0Ahttp://dx.doi.org/10.1016/j.nantod.2015.04.009%0Ahttp://dx.doi.org/10.1038/s41467-018-05514-9%0Ahttp://dx.doi.org/10.1038/s41467-019-13856-1%0Ahttp://dx.doi.org/10.1038/s41467-020-14365-2%0Ahttp://dx.doi.org/1

Sikdar, U. K., & Gambäck, B. (2018). Named entity recognition for amharic using stack-based deep learning. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10761 LNCS*, 276–287. https://doi.org/10.1007/978-3-319-77113-7_22

Sprenger, S. A., Levelt, W. J. M., & Kempen, G. (2006). Lexical access during the production of idiomatic phrases. *Journal of Memory and Language*, *54*(2), 161–184. https://doi.org/10.1016/j.jml.2005.11.001

Tavcar, R., Dedic, J., Bokal, D., & Zemva, A. (2013). Transforming the lstm training algorithm for efficient fpga-based adaptive control of nonlinear dynamic systems. *Informacije Midem-Journal of Microelectronics Electronic Components and Materials*, *43*(2), 131–138.

Tewodros, H. G. (2003). Amharic Text Retrieval : An Experiment Using Latent Semantic Indexing (LSI) With Singular Value Decomposition (SVD). *Source*, *July*. http://localhost/xmlui/handle/123456789/14764

Ting, K. M. (2017). Confusion Matrix. *Encyclopedia of Machine Learning and Data Mining*, *October*, 260–260. https://doi.org/10.1007/978-1-4899-7687-1_50

Verma, R., & Vuppuluri, V. (2015). A new approach for idiom identification using meanings and the web. *International Conference Recent Advances in Natural Language Processing, RANLP*, *2015-Janua*, 681–687.

Vilar, D., Xu, J., D'Haro, L. F., & Ney, H. (2006). Error analysis of statistical machine translation output. *Proceedings of the 5th International Conference on Language Resources and Evaluation, LREC 2006*, *May*, 697–702.

Yimam, S. M., Alemayehu, H. M., Ayele, A., & Biemann, C. (2021). *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

Zaid, M. A. (2019). An analysis of idiomatic expression used by characters in Hotel Transylvania movies. *Faculty of Education and Teacher Training State Islamic University of Sultan Maulana Hasanuddin Banten*. http://repository.uinbanten.ac.id/4081/1/AN ANALYSIS OF IDIOMATIC EXPRESSION USED BY CHARACTERS IN HOTEL TRANSYLVANIA MOVIES.pdf

Appendix A

Dataset Annotation Guidelines

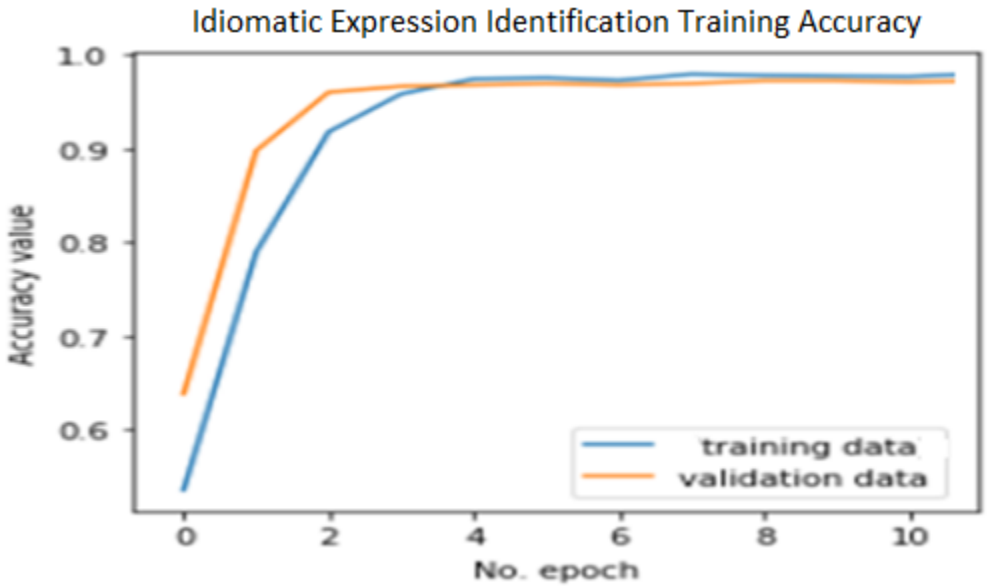
Cases	Label	Idiomatic Expression term	Meaning
If the given sentence contains an idiomatic expression.	Idiom		
If the sentence doesn't contains any idiomatic expression.	Non Idiom		

Annotation Hints

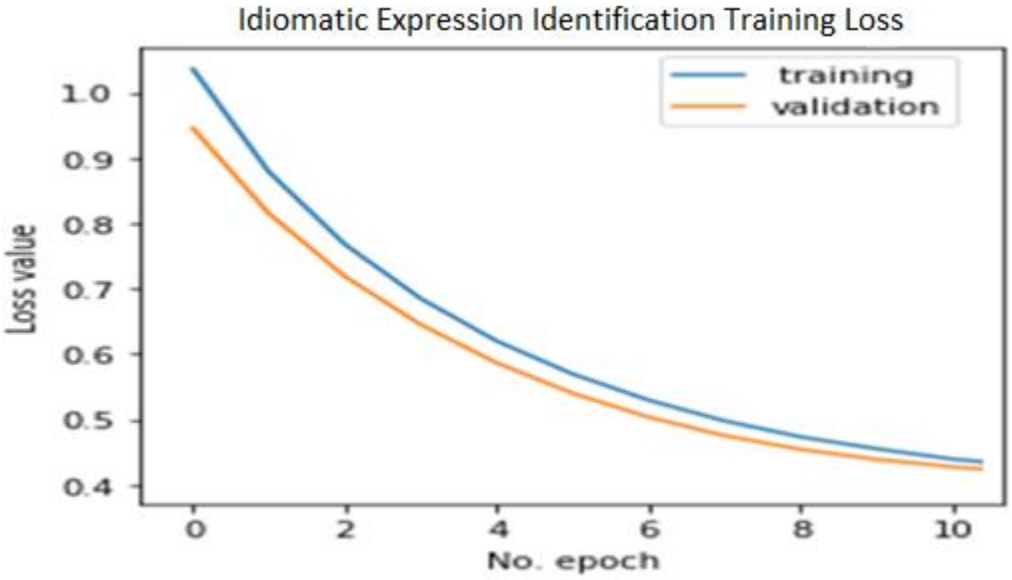
- Please read carefully each sentence which is found in excel file.
- If there may be a typing error please start by correcting it.
- Check that the sentences satisfy which cases among the two cases listed above.
- Once you identify the case the sentence belongs to put the label (Idiom/ Non Idiom)
- If you see a sentence with an idiomatic expression, identify the idiomatic expression term that you get and put that term in idiomatic expression term column and its corresponding hidden meaning in the meaning column.

Appendix B

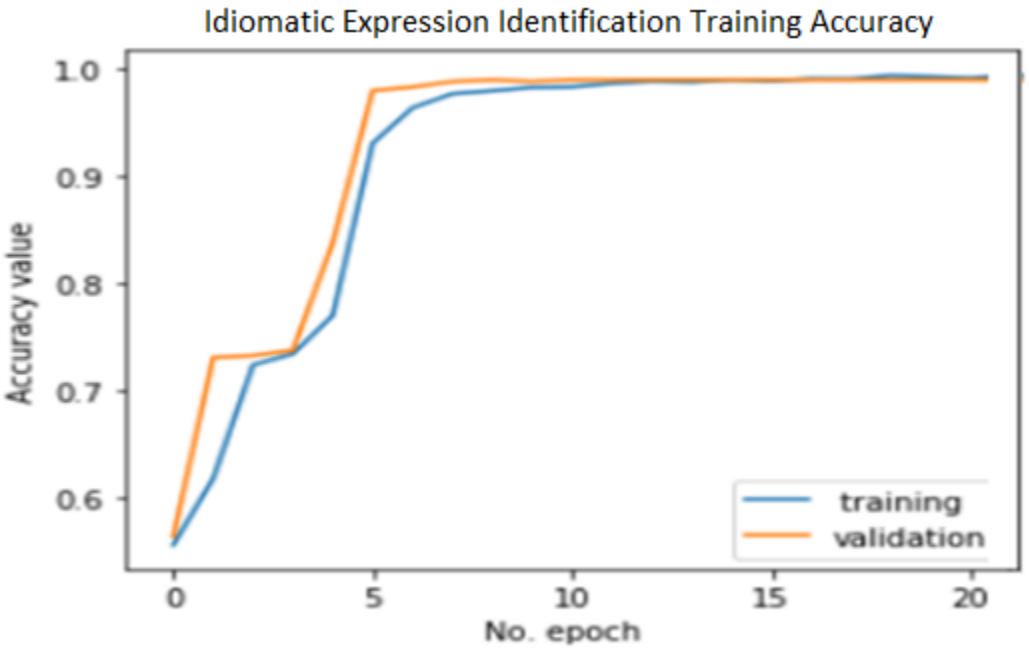
Training and Validation accuracy and loss on three experimental setups for CNN, LSTM, and Bi-LSTM



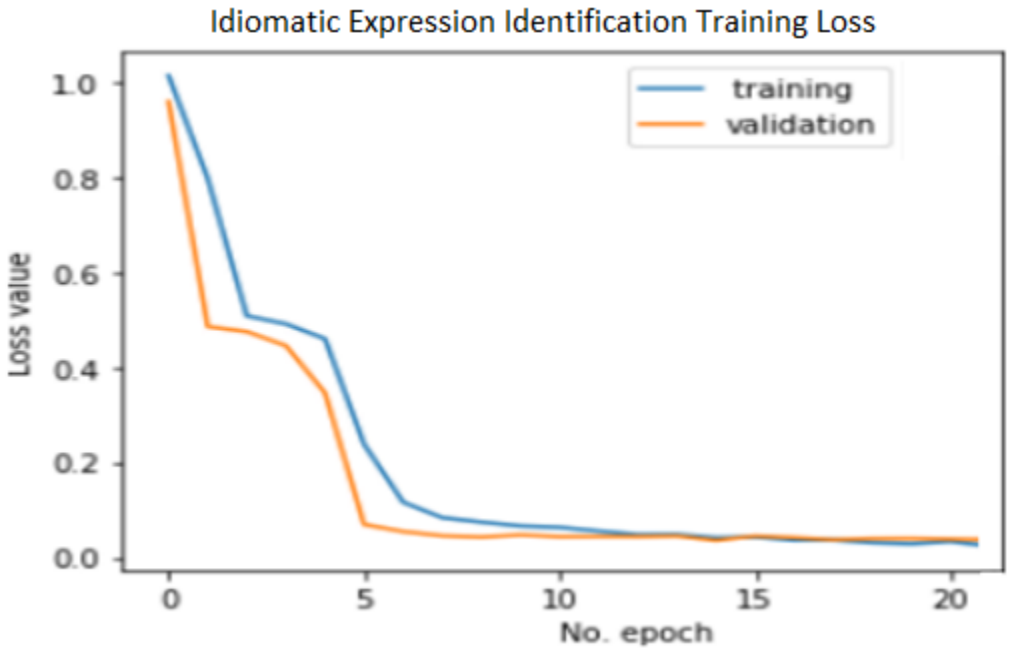
Training and Validation Accuracy from on CNN Model on setup 1



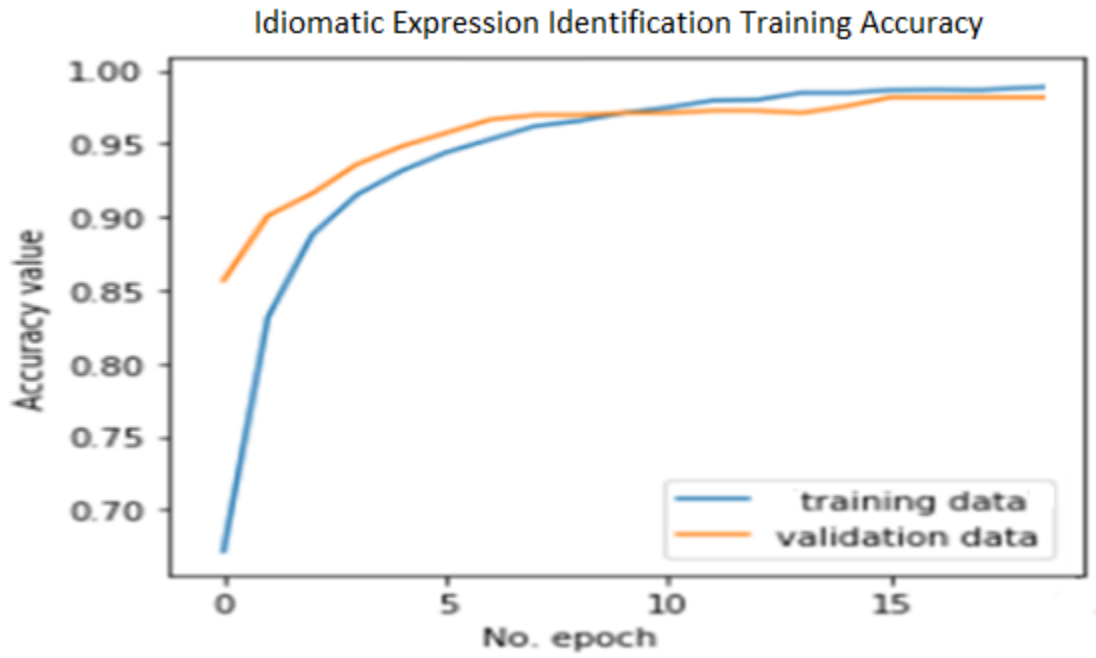
Training and Validation Loss from CNN Model on setup 1



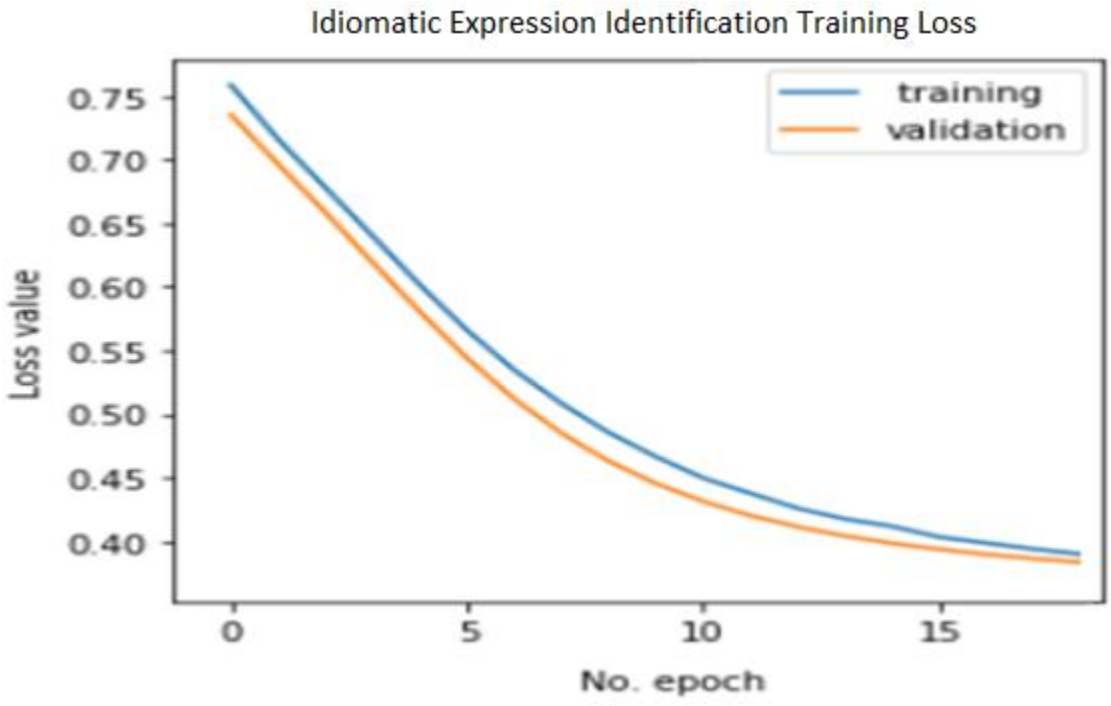
Training and Validation Accuracy from CNN Model on Setup 2



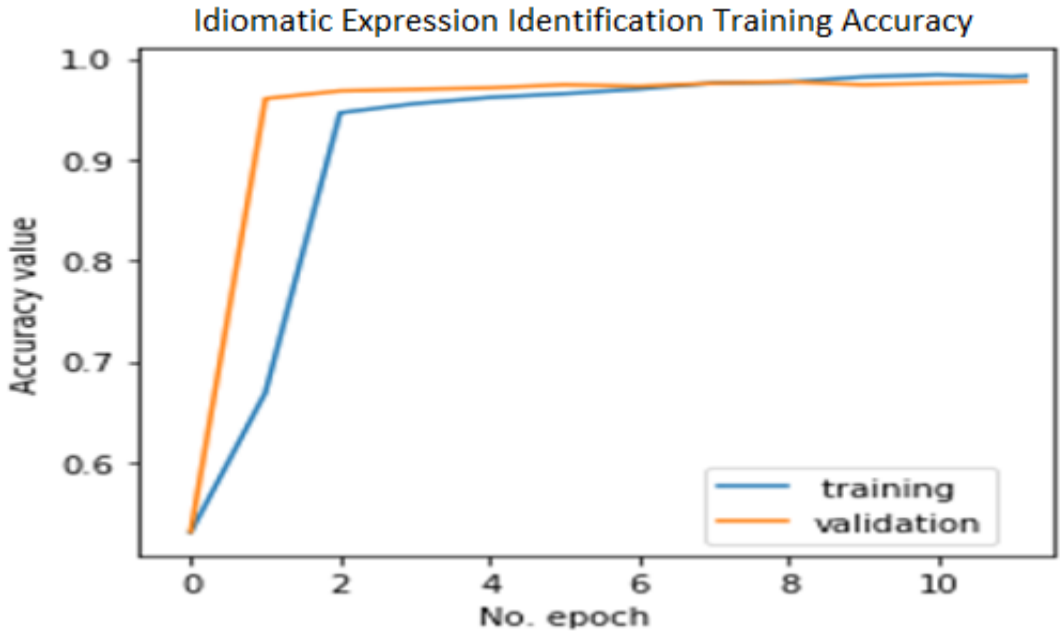
Training and Validation Loss from CNN Model on Setup 2



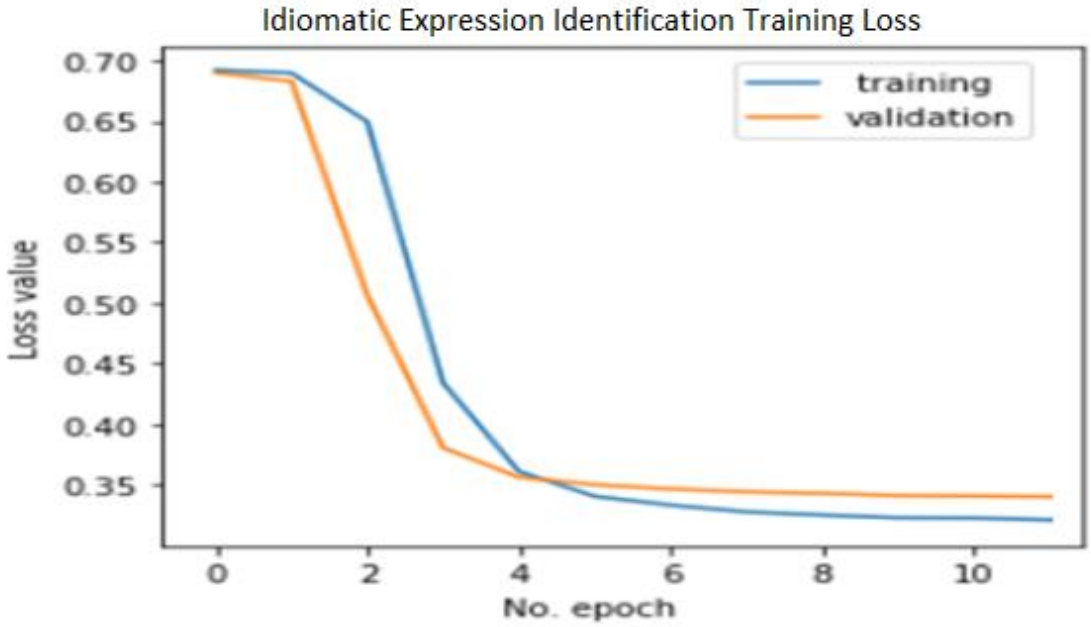
Training and Validation Accuracy from CNN Model on Setup 3



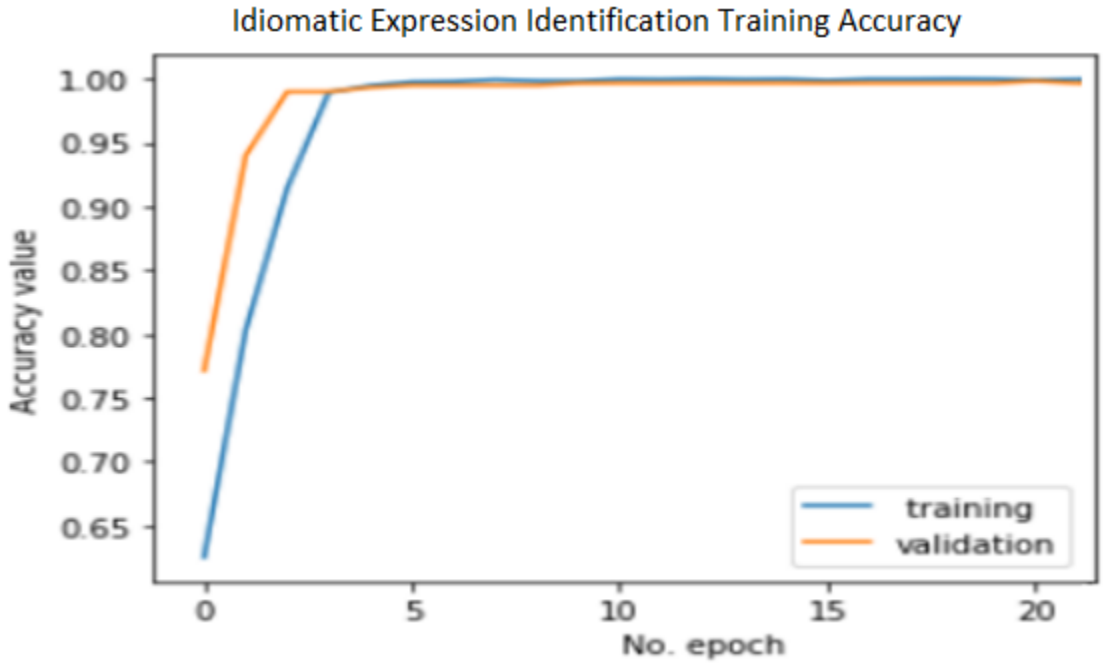
Training and Validation Loss from CNN model on Setup 3



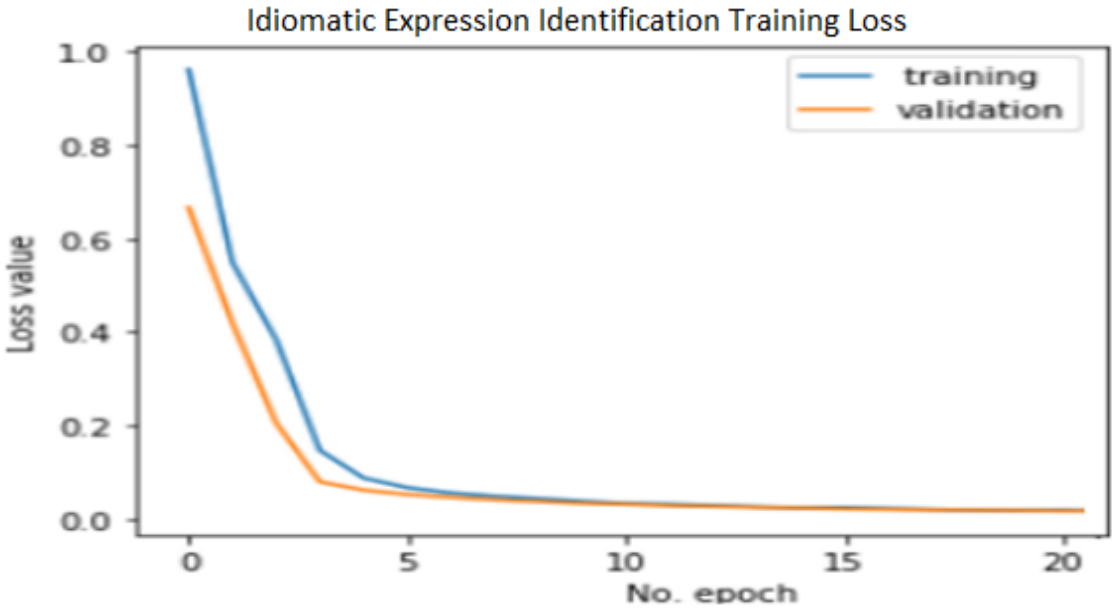
Training and Validation Accuracy from LSTM Model on Setup 1



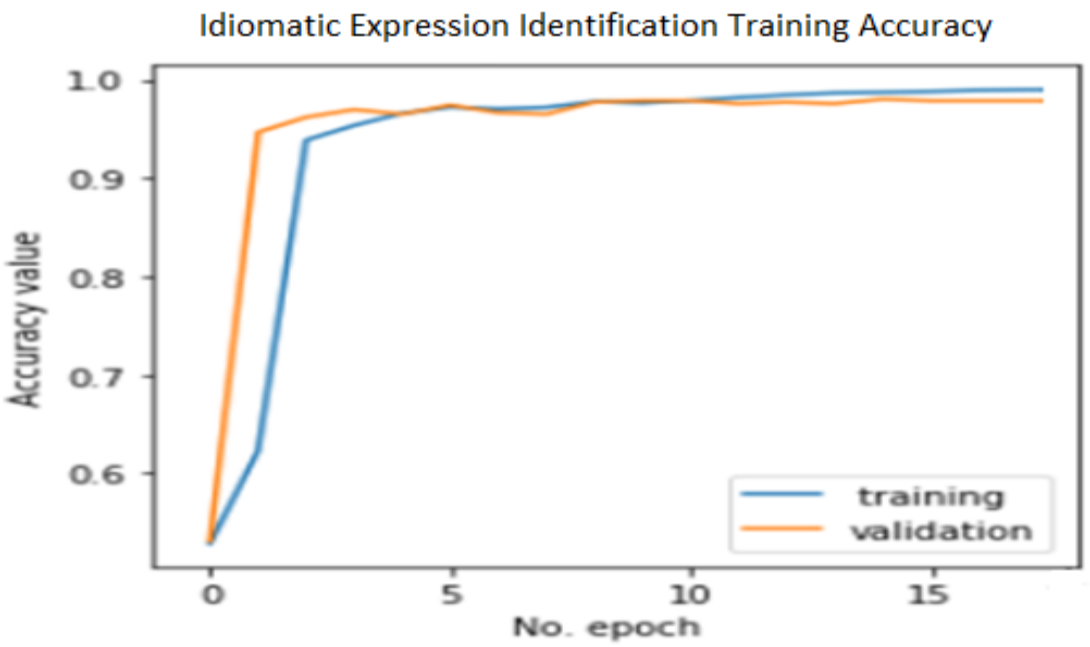
Training and Validation Loss from LSTM Model on Setup 1



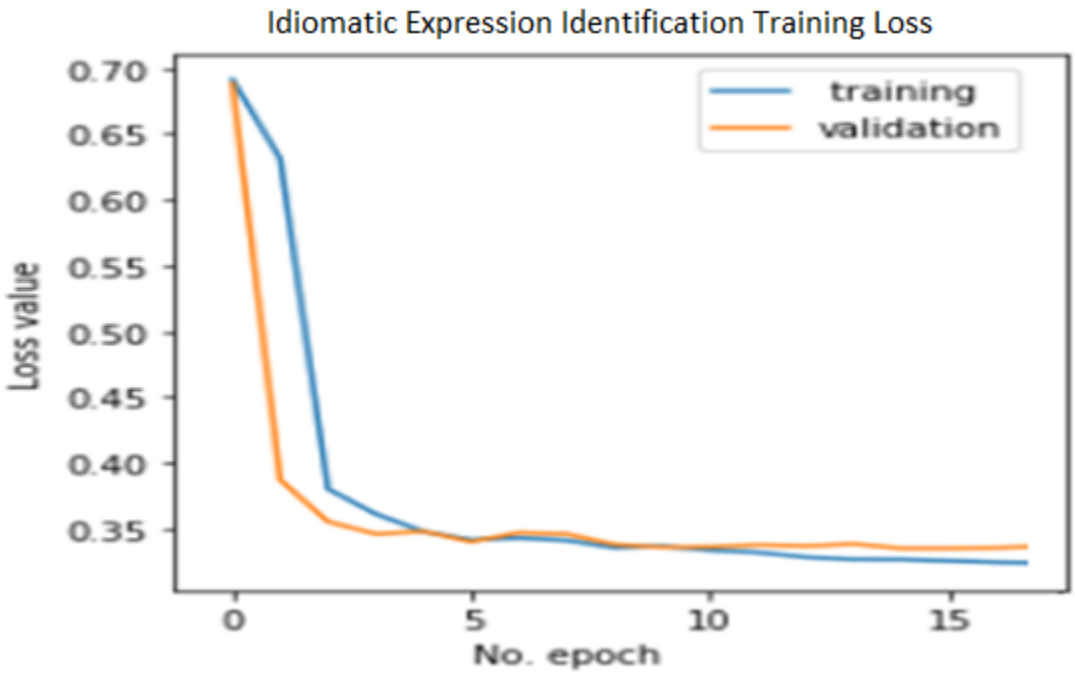
Training and Validation Accuracy from LSTM Model on Setup 2



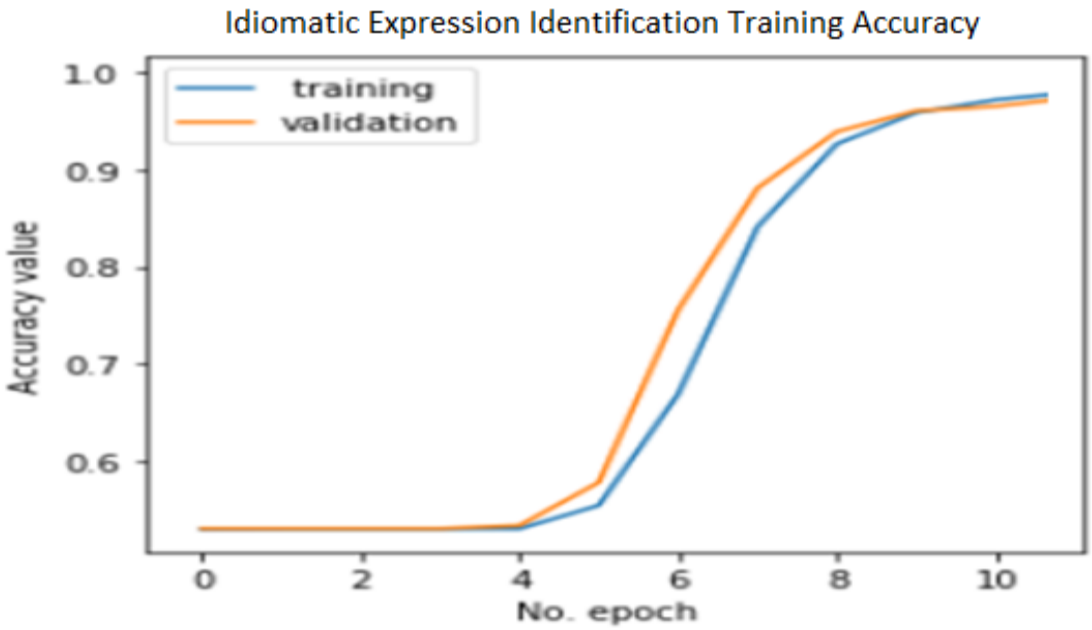
Training and Validation Loss from LSTM Model on Setup 2



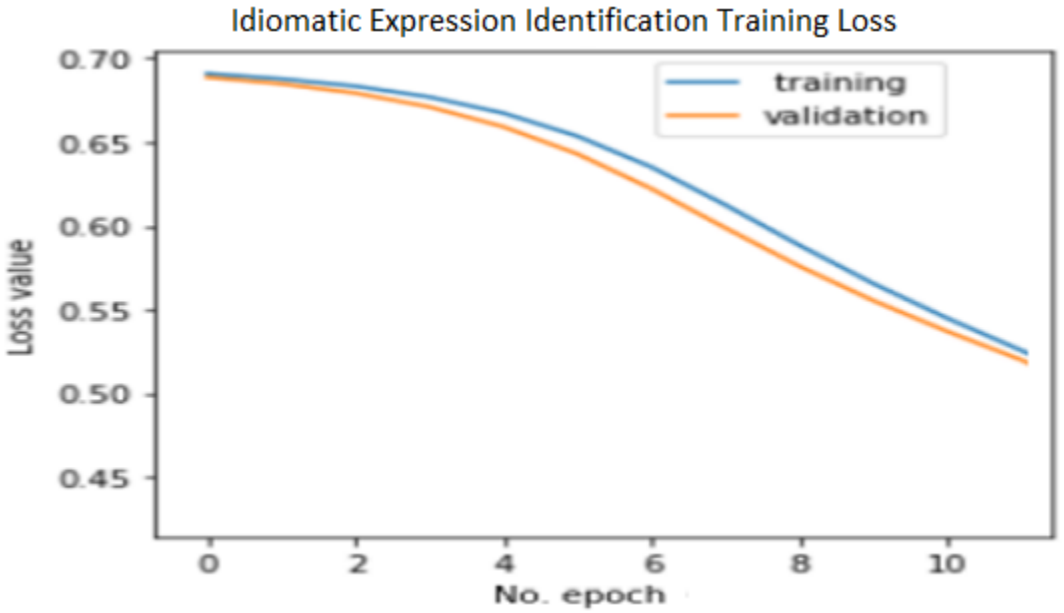
Training and Validation Accuracy from LSTM Model on Setup 3



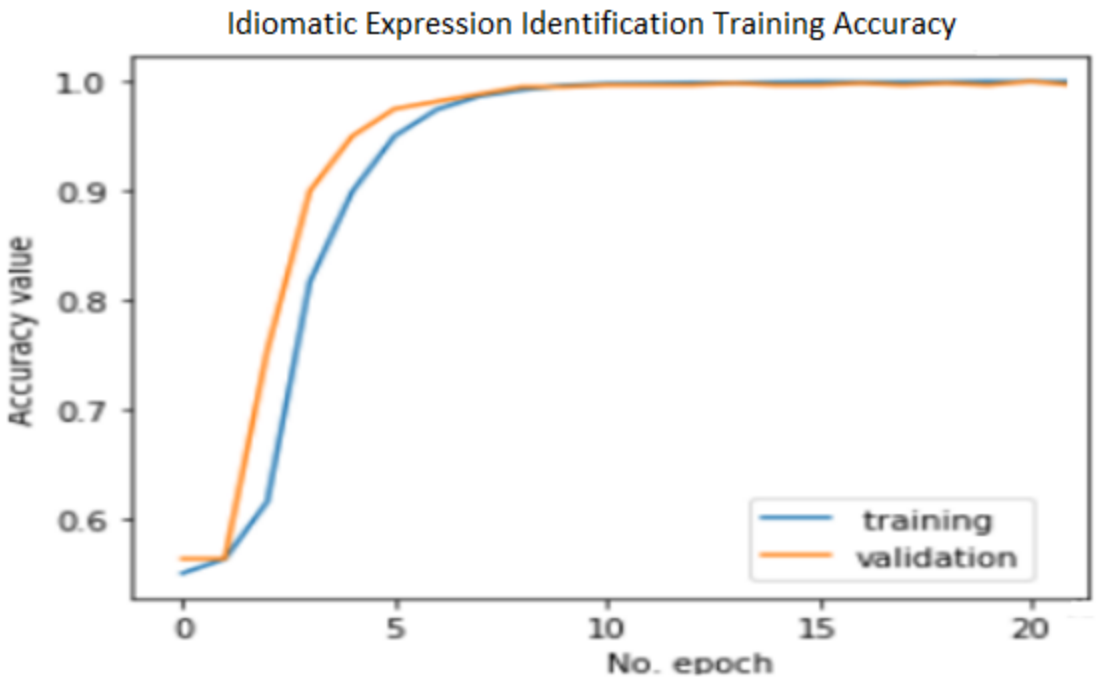
Training and Validation Loss from LSTM Model on Setup 3



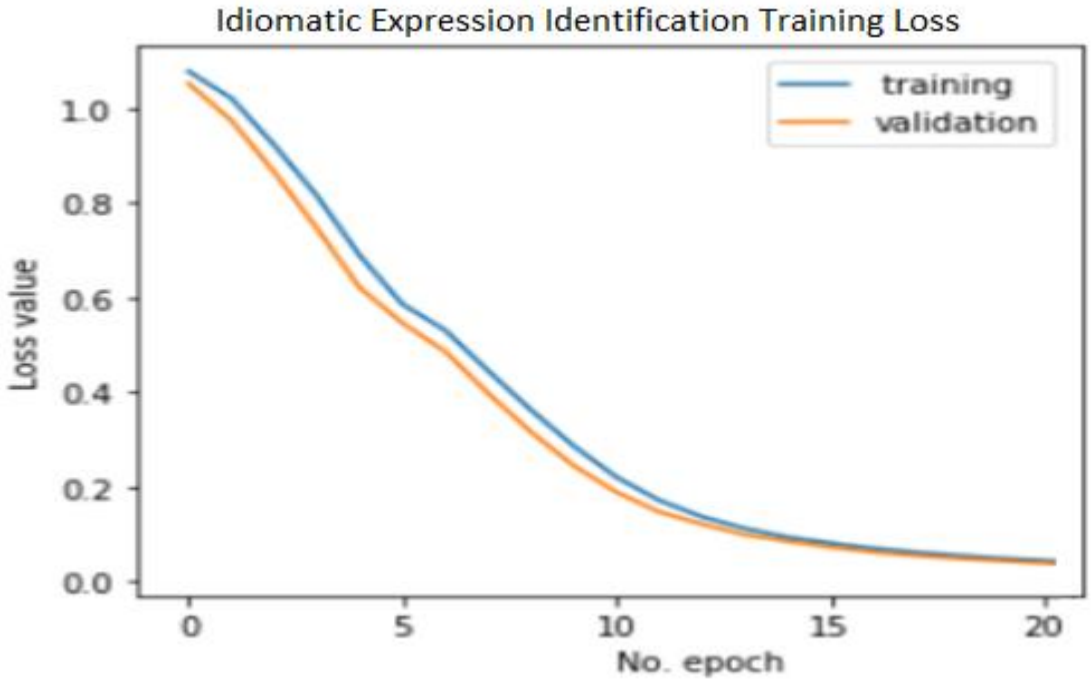
Training and Validation Accuracy from Bi-LSTM Model on Setup 1



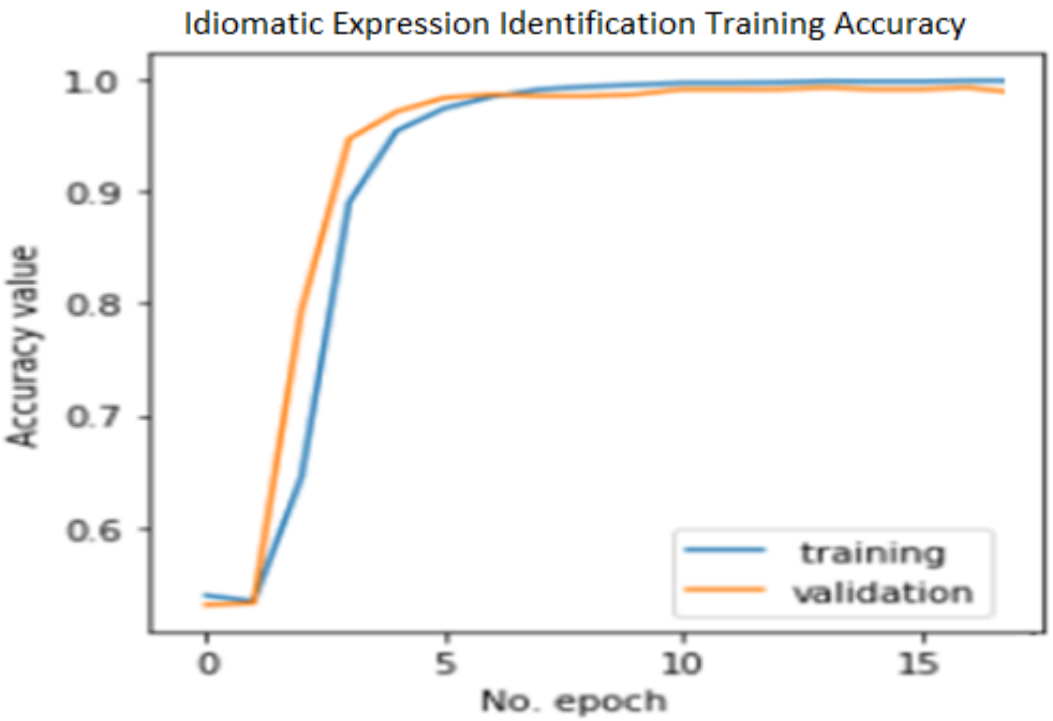
Training and Validation Loss from Bi-LSTM model on Setup 1



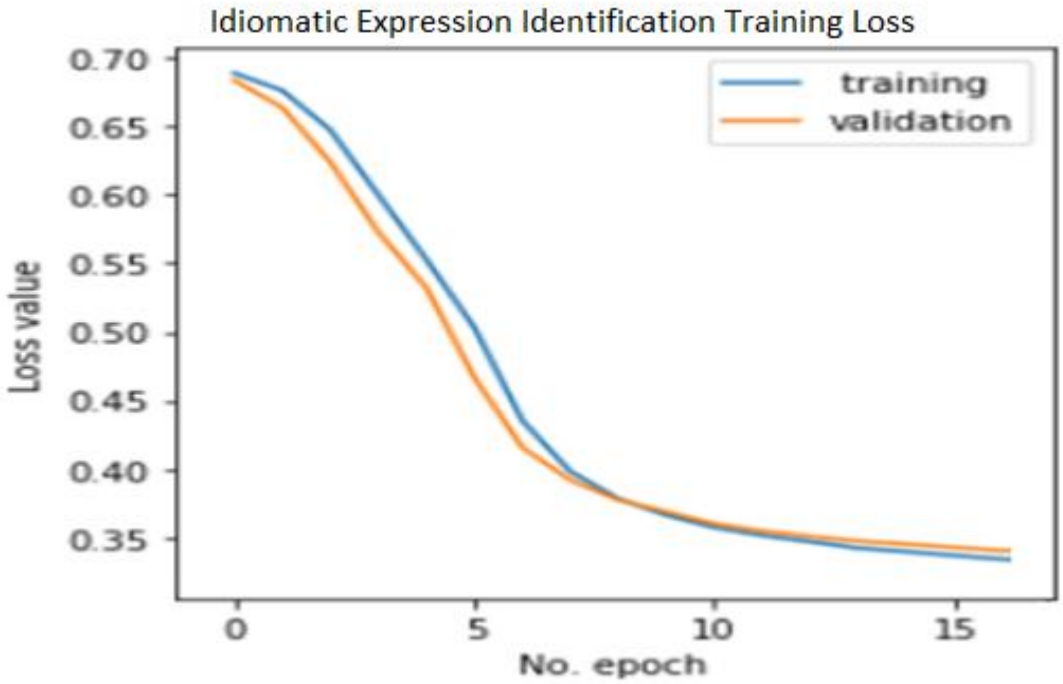
Training and Validation Accuracy from Bi-LSTM Model on Setup 2



Training and Validation Loss from Bi-LSTM Model on Setup 2



Training and Validation Accuracy from Bi-LSTM Model on Setup 3



Training and Validation Loss from Bi-LSTM Model on Setup 3

Appendix C

Sample Dataset

	Sentences	Label
2	በሉ ቶሎ ጨርሱና ተነሱ ይች እሳት አፍ የመጣች እንደሆነ አትለቃችሁም።	1
3	ምን ትነግረኛለህ ተወው በቃ ይዜን እሳት የላሰ ልጅ እማ አውቀው የለ እንዴ።	1
4	እንጀራ በሬው የሆነ ሰው ሽህ አመት ቢለፋ ካለው ላይ ለተቸገሩት ካሳካፈለ መቼም አያልፍለትም።	1
5	አትጠንብ እኮ የመተዳደሪያ ሙያዋ እንግዳ ቁርበት ምንም በሰዎች ዘንድ የተጠላ ቢሆንም እሷ እንጀራዋ ነው።	1
6	ጋሽ ደለለ አንድም ቀን አስቦ ማውራት አያውቅም ሁልጊዜ እንግዳ ደራሽ የሆነ ሀሳብ ነው የሚያልቀው።	1
7	በአሁኑ ጊዜስ በየአገሩ ያለው አስተዳደር በአጠቃላይ እውር ልብ ነው።	1
8	ካሳየ ባለፈው ሳምንት በስብሰባው ላይ የተናገረው ንግግር እዝነ ህሊና ውስጥ የሚቀመጥ ንግግር ነው።	1
9	ከዚህ ሽክላ ቤት ጀርባ ያለው ግቢ ከፌት ካለው ይልቅ ሰፋ ያለ ነው።	0
10	ዙሪያውን የሚታዩት እንደ ዋናው ቤት በሽክላ የተሠሩ ሰርሺስ ክፍሎች አጥሩን ታከው የተገነቡ ናቸው።	0
11	እንድ ሰርሺስ ክፍል ፌት ለፌት አንዲት ሌት በጎማ የተሰራ ጥቁር ጓንት አድርጋ የህጻናቱን ልብስ ታጥባለት።	0
12	አጠንብ ደግሞ ሌላ ሌት በእግሮቿ መሃል ያስቀመጠቻትን የአንዲት ህጻን ፀጉር ካባ ትሰራለች።	0
13	ይህን ሀሞተ ቢስ ወዲያ በለው።	1
14	ወይም በታሪኩ ውስጥ ካላቸው ሚና እንጻር ለእውነተኛ ስማቸው ቢጠሩ ቀር ሊላቸው ይችላል ተብለው የተገመቱ ሰዎች ስምም ተቀይሯል።	0
15	ደራሲው እውነታውን ለማገን ስሚጋብዝ የምልህ እስትታል::	0
16	የነራዬውን ስለት ከሩቅ እንዳየ ሀሞቱ ፈሰሰ።	1
17	በምዕመናን መካከል ሀራ ጥቃ ተቀባይነት የለውም።	1
18	ግቢው ከፌትም ከኋላም በሊሚንቶ ሊሽ የተሸፈነ ነው።	0
19	ሀር ጥለት ለአሽንክታብ ማጥለቂያ ማለፌያ ነው።	1
20	በምክንያት እመን እንጂ ሀሳበ ግትር አትሁን።	1
21	የእሱ ሀብተ ስባራ ነት እኔንም ጭምር ጎድቶኛል።	1
22	በአመለዘውድ ስም የተቀመጡት ደግሞ ከደብዳቤዎቿ የተቀነጨቡ ናቸው::	0
23	በምዕራፉ መጀመሪያ ላይ የተናገሩት ተቀምጦ በታሪኩ ውስጥ ስማቸው ያልተጠቀስ ሰዎች ሰቅጋፍ ማንነታቸው ተገልጿል።	0
24	ልጅቷ ሀብተ ስንኩል በመሆኗ እንጂ ይዜኔ የስንት ጨቅላዎች እናት በሆነች ነበር።	1
25	በሶስተኛው ክፍል ግን ከቃለመጠይቁ መረጃ በተጨማሪ የእመለዘውድን ደብዳቤዎች አካትቻለሁ።	0
26	በአንድ ደብዳቤ ውስጥ ያሉ የህዳግ ምልክት የተደረገባቸው ።	0
27	መግለጫ የሚፈልጉ ስሞች ወይም ጉዳዮች የመጽሃፉ አካል ሆነው ሰቀሳሉ እንዲነበሰ በሚል የተብራሩት እገዱ በታች ሳይሆን ከደብዳቤው በታች ነው።	0
28	ከዚህ ሰፈር ውስጥ እንደ ቃሊም ያለ ሀብተ ነቢ የለም።	1
29	ትዕዛዝ ኣቀላጥፎ ለመፈፀም የተፈጠረ እስኪመስል ድረስ ለሌሎች ኣብረውት እየሰሩ ለሚያሰሩት አንደት አርስ ነው።	1

Appendix D

Training Code for CNN, LSTM, and BiLSTM Model

CNN Model

In [25]:
```
# create the model
max_features =50000
embedding_dim =200
sequence_length =23
modelc = tf.keras.Sequential()
modelc.add(tf.keras.layers.Embedding(max_features,embedding_dim,input_length=sequence_length))
modelc.add(tf.keras.layers.Conv1D(filters=100, kernel_size=2, activation='relu'))
modelc.add(tf.keras.layers.MaxPooling1D(pool_size=2))
modelc.add(tf.keras.layers.Dense(64, activation='sigmoid'))
modelc.add(tf.keras.layers.Dense(32, activation='tanh'))
modelc.add(tf.keras.layers.Flatten())
modelc.add(tf.keras.layers.Dense(2, activation='sigmoid'))
modelc.compile(loss=tf.keras.losses.BinaryCrossentropy(),optimizer=tf.keras.optimizers.Adam(1e-4),metrics=['accuracy'])
print(modelc.summary())
```

LSTM Model

In [32]:
```
# create the model
max_features =50000
embedding_dim =200
sequence_length =23
modell = tf.keras.Sequential()
modell.add(tf.keras.layers.Embedding(max_features,embedding_dim,input_length=sequence_length))
modell.add(tf.keras.layers.LSTM(100,dropout=0.35,recurrent_dropout=0.2))
modell.add(tf.keras.layers.Dense(2, activation='sigmoid'))
modell.compile(loss=tf.keras.losses.BinaryCrossentropy(from_logits=True),optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001)
print(modell.summary())
```

BiLSTM Model

In [36]:
```
# create the model
max_features =50000
embedding_dim =200
sequence_length =23
modelBl = tf.keras.Sequential()
modelBl.add(tf.keras.layers.Embedding(max_features,embedding_dim,input_length=sequence_length))
modelBl.add(tf.keras.layers.Bidirectional(LSTM(25,dropout=0.35,recurrent_dropout=0.2)))
modelBl.add(tf.keras.layers.Flatten())
modelBl.add(tf.keras.layers.Dropout(0.2))
modelBl.add(tf.keras.layers.Dense(2, activation='sigmoid'))
modelBl.compile(loss=tf.keras.losses.BinaryCrossentropy(from_logits=True),optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001
print(modelBl.summary())
len(modelBl.layers)
```

Appendix E

Word2vec Model Development

In [41]:
```
w2vembeding = w2vword.sentences.apply(gensim.utils.simple_preprocess)
embedding_model = gensim.models.Word2Vec(
    sg=0,
    window=1,
    min_count=1,
    workers=2,
    vector_size=100
)
embedding_model.build_vocab(w2vembeding, progress_per=1000)
embedding_model.train(w2vembeding, total_examples=embedding_model.corpus_count, epochs=20)
embedding_model.wv.save_word2vec_format('save.bin', binary=True)
print(w2vembeding)
```

Idiomatic Expression Identification Code

In [44]:
```
for x in w2vword['Idiom']:
sent=""
i=0
for word in sents.split():
        if det.__contains__(word):
            for words in sents.split():
                if words==word:
                    sent+=words+" "
                else:
                    sent+=words+" "
            print("Idiomatic Expression : "+word+ "The Meaning:"+str(w2vword.Meaning[i])
```