MATTU UNIVERSITY

COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

STREAM INFORMATION TECHNOLOGY

TITLE

AFAAN OROMO MULTI LABEL NEWS TEXT CLASSIFICATION USING DEEP LEARNING APPROACH

MSc. THESIS

By:

ASCHALEW ALEMU

Main advisor: RAMETA MOSSISA (ASS/PRO)

Nov, 2023

MATTU, ETHIOPIA

MATTU UNIVERSITY

COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGYU

STREAM: INFORMATION TECHNOLOGY

TITLE: - AFAAN OROMO MULTI LABEL NEWS TEXT CLASSIFICATION USING DEEP LEARNING APPROACH

A Thesis Submitted as a Partial Fulfillment to the Requirements for the Award of the Degree of Master of Science in Information Technology

By: ASCHALEW ALEMU TESEMA

Main Advisor:- RAMETA MOSSISA(ASS/PRO)

Nov, 2023

MATTU, ETHIOPIA

ADVISORS' APPROVAL SHEET

This is to certify that the thesis prepared by **Mr/Ms Aschalew Alemu** entitled '**'Afaan Oromo Multi label News Text Classification**" submitted in partial fulfillment of the requirements for the degree of Master's/College of **Engineering And Technology with field of specialization** in **Information Technology** and as thesis research advisor, I hereby certify that I have read and evaluated this thesis prepared under my supervision. Therefore I recommend that the student has fulfilled the requirements and hence hereby can submit the thesis to the department.

RAMETA MOSSISA(ASS/PRO)	_____	_____
Name of main advisor	Signature	Date
_____	_____	_____
Name of co-advisor	Signature	Date

EXAMINERS APPROVAL PAGE

Title of the thesis **'Afaan Oromo Multi label News Text Classification '**

Name of the Student **Aschalew Alemu Tesema**

Approved by the examining committee members

Chairperson:	Signature	Date
External Examiner	Signature	Date
Internal Examiner	Signature	Date
Department head:	Signature	Date
College dean:	Signature	Date
SGS director:	Signature	Date

DECLARATION

I therefore certify that I, under the supervision of my advisor, prepared this thesis, which is titled "**Afaan Oromo Multi label News Text Classification**." Except as otherwise indicated in the text, all of the material presented here is original to me, and it hasn't been submitted in whole or in part for consideration towards any other degree or professional certification.

Name Student	Signature	Date
ASCHALEW ALEMU	_____	_____
Name of advisor	Signature	Date
RAMETA MOSSISA(ASS/PRO)	_____	_____
Name of Co-advisor	Signature	Date
_____	_____	_____

Abstract

The development of the internet has made Afaan Oromo's writings widely available both offline and online. Automatic text classification is an invertible response to the continuously expanding amount of information resources available. The practice of classifying a text or document into predetermined categories is called text categorization, or text classification (TC). It is recommended that this study employ the deep learning technique with word embedding on Afaan Oromo's multi label news text classifications. Since extracting feature values from news articles is difficult, this research suggests a deep learning strategy for news text categorization. The word order information, which is essential for classifying news texts, was not taken into consideration by the earlier approaches, which classified the text data by using a bag of words to represent the words in the text. Despite their low time complexity, the earlier models do not adequately account for the context and any semantic links between text words. Furthermore, as the number of characteristics and classes increased, the models' accuracy decreased. This thesis uses CNN, LSTM, BRNN, GRU and algorithms to perform deep learning approaches for Afaan Oromo multi-label news text categorization. To create a pre-trained word embedding model, utilize the Afaan Oromo news domain. Then, train our data using the CNN, LSTM, BRNN and GRU model create the classification process to recommend the best for the problem at hand. In this study, the models for the Afaan Oromo languages were constructed using newly gathered and annotated news datasets totaling nine thousand forty one (9041). After completing the following procedures, the Afaan Oromo News text documents are classified using those deep learning algorithms. Preprocessing, word embedding, deep network construction, output determination, model training, and classification. The semantics of the page are finished with word2vec, which uses neural network architecture to map similar words into a single vector. Consequently, the vector representations of words serve as the input for the deep network development component. Training, testing, and validation datasets are used to evaluate the model using accuracy and loss; finally, the performance of our models is compared. LSTM performs exceptionally well, scoring 98.71% accuracy and 98.71% precision, while CNN, BRNN and GRU score 94%, 94%, 96.40%, 97.48%, and 94.42%,92.4% respectively.

Keywords: Text classification, Deep learning, RNN, CNN, LSTM, GRU and word-embedding

Acknowledgements

First of all, I would like to thank the Almighty God for their support, help, and guidance in my overall life and the accomplishment of this work. My life's work rests entirely on him, thus I completed every task entrusting myself to his readiness and assistance.

Next, I want to sincerely thank my adviser Rameta Mosisa (Ass/Pro) for all of his help, advice, and positive remarks. I am very appreciative of my family, especially my wife Leilt Tsega, who has helped me get to this point. I hope she is doing well and appreciate all that she does for me.

My sincere thanks and admiration also go to Getacho Mamo (PHD) of Jimma University for his readiness to assist with research preparation and his guidance on how to perform better and accomplish more. Once more, Lalisa Tadese (Msc) of Jimma University, whose work on Afaan Oromo News Text classification on Single Labels has helped me succeed in my research, has my sincere gratitude.

In addition, I am really appreciative of my classmates' assistance and knowledge sharing in helping us complete the course and our final thesis.

Table of Content

Contents

List of Tables

List of figures

List of algorithms

List of equation

List of Acronyms and Abbreviations

AE	Auto Encoder
BOW	Bag of Words
CNN	Convolutional Neural Network
CBOW	Continuous Bag of Words
CPU	Central Processing Unit
DNN	Deep Neural Network
DL	Deep Learning
DBOW	Distributed Bag of Words
Doc2vec	Document to Vector
GPU	Graphical Processing Unit
GAN	Generative Adversarial Network
LIWC	Linguistic Inquiry and Word Count
LVQ	Learning Vector Quantization
LSTM	Long Short-Term Memory
ML	Machine learning
NLP	Natural Language Processing
RAM	Random Access Memory
RDF	Resource Description Framework
PCA	Principal Component Analysis
SG	Skip Gram
SVM	Support Vector Machine
SRNA	Semantic-aware recurrent deep Network Architecture
SRM	Structural Risk Minimization viii
TC	Text Classification
TF	Term Frequency
TF-IDF	Term Frequency Inverse-Document Frequency
BRNN	Bidirectional Recurrent Neural Network
GRU	Gated Recurrent Unit

CHAPTER ONE

1. INTRODUCTION

1.1. Background of the Study

The availability of enormous datasets and the growth of digital information have made text classification an increasingly crucial task in natural language processing in recent years. Text classification is crucial for news analysis since it allows for the automatic classification of news stories into a variety of categories, including politics, sports, business, entertainment, and more.

Historically, statistical models or rule-based systems have been used to do text classification in news analysis. However, there has been an increase in interest in using neural networks for this purpose with the development of deep learning. Classifications make life simpler. For instance, at a supermarket, if everything were arranged on the shelves at random, the experience of shopping would be unpleasant and time-consuming. Classification is used by the majority of financial organizations to determine credit ratings, and it is widely used in health to identify disorders.[1].

In another way Classification is the task of assigning observations to one of several predefined categories also called classes. It is a supervised learning (that means set of labels associated with each instance are already provided in the training data). An observation consists of a vector of attributes F, called features[2].

A vast amount of information is now connected to the internet and web technology. Additionally, there are efforts to digitize the paper-based text sources. There is an urgent need to organize the material available online given its constantly growing volume in order to extract relevant information from it. Therefore, automatic text classification becomes crucial for managing and organizing information. Grouping related objects or things together is the definition of classification. In supermarkets, similar items are grouped together on shelves, and there is a shelf for meat and a shelf for cleaning supplies.

Deep learning applications for news text classification include automatic news categorization, individualized news recommendations, and trend analysis. It also comes with a number of difficulties, including as handling skewed or noisy data, dealing with class imbalance, and guaranteeing the model's interpretability. Overall, applying deep learning to the classification of

news material is a fascinating field of study that has the potential to completely change how we interpret and consume news.

In this work, we focus on the Afaan Oromo multi label news text classification. Text classification can be done in two different ways: Manual and automatic classification. In the manual text classification, a human annotator interprets the content of text and categorizes it accordingly.

This method usually can provide quality results but it's time-consuming and expensive. While automatic text classification, or the task of automatically assigning semantic categories to natural language text, has become one of the key methods for organizing online information.

Since hand-coding classification rules is costly or even impractical, most modern approaches employ machine learning techniques to automatically learn text classifiers from examples [3].

There are many approaches to automatic text classification, which can be grouped into three different types of approaches: Rule-based approaches, Machine Learning based approaches and Hybrid approaches. Rule-based approaches classify text into organized groups by using a set of handcrafted linguistic rules. These rules instruct the system to use semantically relevant elements of a text to identify relevant categories based on its content. Each rule consists of an antecedent or pattern and a predicted category [4].

Machine learning based approaches mean ones where classification rules or equations are defined automatically using sample labeled documents.

This study was considering machine learning approach for Afaan Oromo Multi Label News Text Classification particularly neural networks deep learning Algorithm. Neural networks can be trained to solve problems which are difficult for conventional computers or human beings. Neural networks can be trained to perform complex functions in various fields of application including, pattern recognition, writer identification, text classification, speech recognition, computer vision and control systems [5]. Neural networks can be used for text classification tasks among others.

1.2. Statement of the Problem

According to the results of the experimentation was done SMO algorithm tends to be the best classifier for Afaan Oromo multi label news items as the number of classes in a category increases. Kamal employs decision tree, Naive Bayes, Bayes Network and support vector machine classifiers

[6]. According to his experiments both Bayes Net and J48 classifiers have better performance than the other two classifiers. Among Bayes Net and J48 classifiers, BayesNet classifier requires less time to build the classification model. So, in that study the BayesNet classification model selected as the Afaan Oromo multi label news text categorization system.

Both researchers Abera and Kamal employs single-label categorization (also called non overlapping categories).In this type of classification, each sample is represented by only one label. The great problem of single-label text categorization (TC) assigns only one predefined category to every invisible Natural language, in case when there are two or more than two categories in category space. Due to the overlapping nature of text with one another in category space, the categorization of every single document becomes impossible. Generally, it is impossible to categorize each document under a single label, because of the natural overlapping of the category spaces. Multi-label text classification for Afaan Oromo is not done until now. Also it's recommended by previous researchers.

Then, to solve the problem of single-label classification for Afaan Oromo text classification this work aimed at contributing in the area of Afaan Oromo text classification considering multi-label classification. So, in this research proposal, main aim is to classify each sample to zero or more labels according to predetermined classification. Also another aim of this study was see the performance of a neural network in deep learning approaches based on Afaan Oromo text news classification considering multi-label classification.

The chore of manually categorizing the ever-growing volume of news stories that are published online every day has grown to be difficult. Consequently, there is a rising need for automated news text categorization systems that can categorize news stories effectively and precisely.

Among the challenges in classifying news material are handling noisy data, correcting imbalanced classes, and selecting the best features for the classification model. News text can also be classified using deep learning models such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs), which are skilled at collecting the syntactic and semantic features of news items.

.

However, Afaan Oromo has received very little attention. One of the most well-liked and frequently accessed types of information online is news articles. For managing a huge number of news texts, automatic document classification is necessary among other things. [7].

Additionally, a huge amount of news texts are created and kept in Afaan Oromo. More precisely, OBN and FBC produce and store a large number of news documents. The news is provided in a wide variety of categories. It is difficult to manually classify such a huge number of categories. Therefore, more work in the field of text classification is needed from other academics in order to help Afaan Oromo, a language with limited technological resources. The most effective method or instrument for classifying Afaan Oromo text must be demonstrated through research. [8].

1.3. Research Questions

At the end of the study the following questions will be answers:

- Is it deep learning a viable method for developing Afaan Oromo multi-label text news classification?
- What part does the neural network approach using Deep Learning Algorithms play in the multi-label learning technique for classifying text news in Afaan Oromo?
- Is it possible to achieve higher classification accuracy with the Deep Learning method for multi-label classification?

1.4. Objectives of the Study

The general and specific objectives of this study are given below:

1.4.1. General Objective

The general objective of this research was to develop text classification for multi label Afaan Oromo using deep learning approach.

1.4.2. Specific Objectives

To achieve the general objective of the research are as follows;

- Review literatures on deep learning architectures and algorithms.
- To collect and compile a dataset for training and testing purposes.
- Develop a model based on deep learning for Afaan Oromo news text classification.
- Develop a prototype of the system
- Evaluate the performance of the model

1.5. Significance of the Study

The study has a number of important ramifications. First, it will make it possible for news organizations to automate the classification of news text, which can help them save time and money. The relationship between news substance, topic, and sentiment will also be better understood as a result, which is the second benefit. Third, it can assist increase the reliability of news recommendation systems, which will benefit users.

Classification is an activity that people perform across a variety of human endeavors. In circumstances of multi-label classification, this work will be useful for Afaan Oromo News text classification utilizing a deep learning strategy for OBN and FBC media. The performance of the Afaan Oromo Text News Classification needs to be improved, so this research was important for them researchers as well. Additionally, the purpose of this study proposal is to provide as a resource for scientists who are interested in multi-label categorization.

1.6. Scope and Limitation of the Study

1.6.1. Scope

The goal of this research is to determine whether a deep learning methodology could be used to automatically classify Afaan Oromo text news from Oromia Broadcasting Network (OBN) and FBC. Accidental news, education, technology, health, agriculture, politics, sports, and business are the eight categories taken into consideration.

1.6.2. Limitation

The only Afaan Oromo text news stories used in this study were those from the Oromia Broadcasting Network (OBN) and FBC publications that only contained a series of Afaan Oromo alphabets. No photos, tables, figures, or other visual aids were used.

CHAPTER TWO

2. Literature Review

Unstructured data remains a challenge in almost all data intensive application fields such as business, universities, research institutions, government funding agencies, and technology intensive companies.[9].

Classification is the process of assigning elements or units to classes (or types or groups) according to some criteria[8]. Automatic document classification becomes very important for information organization and storage because of the fast increasing amount of electronic text documents and the rapid growth of the World Wide Web. Text classification (TC) is regarded as decision making criteria which are very useful in content analysis and other such operation, but it holds if a particular text piece belongs to specific prescribed category [10].

"A survey on text classification: From traditional to deep learning techniques" by Weiwei Yang et al [11]. Provides a comprehensive review of the development of text classification techniques, from traditional methods such as Naive Bayes and Support Vector Machines (SVMs) to deep learning models such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). The paper also covers different types of text classification tasks, including sentiment analysis, topic classification, and spam filtering.

"Text classification algorithms: A survey" by Manisha Sharma et al [12]. Provides a detailed review of the different text classification algorithms and their performance in various text classification tasks. The paper compares traditional algorithms such as Naive Bayes and Decision Trees with modern techniques such as Random Forests and Gradient Boosting Machines (GBMs).

"A review of text classification algorithms" by Aruna Devi and N. B. Karunya [13]. Provides an overview of various text classification algorithms, including both traditional and deep learning approaches. The paper also discusses different text preprocessing techniques and feature selection methods that can be used to improve the performance of text classification models.

"Deep learning for text classification: A comprehensive review" by A. M. Rathore et al [14]. Provides an in-depth review of deep learning models used for text classification. The paper covers various types of deep learning models, including CNNs, RNNs, and Attention-based models. The

authors also discuss different approaches to a text representation, such as word embedding's and character-level embedding's.

"Recent advances in text classification: A review of techniques and applications" by S. S. Ahmad et al [15]. Provides a comprehensive review of recent advances in text classification techniques and their applications. The paper covers traditional and deep learning methods, as well as hybrid approaches that combine multiple techniques. The authors also discuss the challenges and future directions in the field of text classification.

"Text Classification and Naive Bayes" by Andrew McCallum: This paper provides an introduction to text classification and presents the Naive Bayes algorithm as a solution. The Naive Bayes algorithm is a probabilistic classifier that assumes independence between features. It has been widely used in text classification tasks[14].

"Support Vector Machines for Text Classification" by Thorsten Joachims: This paper presents the use of support vector machines (SVMs) for text classification. SVMs are a type of supervised learning algorithm that can be used for binary or multi-class classification tasks. The paper shows that SVMs can achieve high accuracy in text classification tasks[16].

"A Comparative Study on Sentiment Analysis for Social Media" by Wei Pang et al.: This paper presents a comparative study of different machine learning algorithms for sentiment analysis of social media data. The study compares the performance of SVMs, Naive Bayes, and other classifiers on a dataset of tweets. The paper concludes that SVMs outperform other classifiers in terms of accuracy[10].

"Convolutional Neural Networks for Sentence Classification" by Yoon Kim: This paper presents the use of convolutional neural networks (CNNs) for text classification. CNNs are a type of deep learning algorithm that can be used for text classification tasks. The paper shows that CNNs can achieve high accuracy in sentence classification tasks[17].

"Recurrent Neural Networks for Text Classification with Multi-Task Learning" by Xiang Zhang et al.: This paper presents the use of recurrent neural networks (RNNs) for text classification with multi-task learning. RNNs are a type of deep learning algorithm that can be used for sequential

data, such as text. The paper shows that multi-task learning can improve the performance of RNNs in text classification tasks[18].

Sentiment analysis: Sentiment analysis aims to classify the polarity of news articles, determining whether the text expresses a positive, negative, or neutral sentiment. A number of studies have explored different approaches to sentiment analysis, including rule-based systems, machine learning techniques, and deep learning models.

Topic modeling: Topic modeling involves identifying the main topics or themes covered in a set of news articles. This can be done using a range of techniques, including Latent Dirichlet Allocation (LDA) and Non-negative Matrix Factorization (NMF). A number of studies have explored the use of topic modeling for news text classification, including for the purposes of event detection and trend analysis.

Text categorization: Text categorization involves assigning news articles to pre-defined categories, such as sports, politics, and entertainment. This can be done using a range of approaches, including keyword-based methods, supervised machine learning, and unsupervised clustering. A number of studies have explored the effectiveness of different text categorization techniques for news text classification.

Named entity recognition: Named entity recognition involves identifying and categorizing named entities mentioned in news articles, such as people, organizations, and locations. This can be useful for a range of applications, including event detection and trend analysis. A number of studies have explored the use of named entity recognition for news text classification.

2.2. Text Classification

2.2.1. Concepts of Text Classification

Text classification is the process of classifying text into one or more groups in order to structure, filter, and organise it according to any number of criteria. For example, text classification is used in files, legal documents, medical studies, and even simple product reviews.
Companies are investing enormous sums of money in an effort to glean as many insights from data as they can since data is more crucial than ever.

It is crucial to find innovative ways to use text and document data because they are significantly more prevalent than other types of data. Data is naturally unstructured and abundant, therefore arranging it to comprehend it in understandable ways can significantly increase its worth. Relevant text can be automatically organized in a quicker and more economical manner by using Text Classification with Machine Learning.

Assigning a text to one or more predetermined classes or categories is the aim of text classification. A document, news story, search query, email, tweet, support ticket, customer review, user product evaluation, etc. could all be examples of text. Text classification is a Natural Language Processing (NLP) job that is frequently utilized in applications like sentiment analysis and the automatic subject classification of news articles. Text classifiers are now among the most crucial tools for tracking and organizing information due to the explosive proliferation of internet content. It is likely that new online documents may not fit into any of the previously defined or known divisions because fresh online text content is regularly produced by social media, blogs, and news stories to a new or unknown group.

Worku referring to Klein and Sebastian defined text categorization as a mapping of text documents to categories[7]. To clarify, if C= {c1, c2…cm} is a set of categories (classes) and D= {d1, d2,…,dn} is a set of documents, the purpose of text classification is assigning ci to dj (1<=i<=m and 1<=j<=n) a value of 0 if the document dj does not belong to ci; otherwise a value of 1.

Table 2. 1 Document to category matrix

	d1	---	d_j	---	d_n
c1	a_{11}	---	a_{1j}	---	a_{1n}
---	---	---	---	---	---
Ci	a_{i1}	---	a_{ij}	---	a_{in}
---	---	---	---	---	---
c_m	a_{m1}	---	a_{mj}	---	a_{mn}

In Table 2.1, d1…dn refers set of documents, c1…cm refers set of categories and a11…amn represent a value of 0 if the document does not belong to that category, otherwise a value of 1. Depending on the application, text classification may be either a single-label task or multi label task. Single-label task is assigning exactly one category to a document. And multi-label task is assigning one category or more categories for a given document.

A special kind of single-label classification is binary text classification, in which a document is going to be classified in either of the two available categories [19]. Based on Sebastian[19], text classification is a subjective task in the sense that two experts, human or artificial, may disagree on the decision of the category to be assigned for a document.

A news article could be filed under Education, Agriculture, Politics, Finance, Sport, or any other category, or even under neither, depending on the subjective judgment of the expert. Because of this, the meaning of a category is subjective. Text categorization can be divided into different categories using different criteria[19]. Depending on the application area, text categorization can be single-label or multi-label, on the other hand depending on the use of a text classifier text categorization can be document-pivoted, or category pivoted, further more based on the automation of the system text categorization can be hard or soft.

2.2.1.1. Single-Label versus Multi-label

Text Classification Traditional single-label classification is concerned with learning from a set of examples that are associated with a single label l from a set of disjoint labels L, $|L| > 1$. If $|L| = 2$, then the learning problem is called a binary classification problem (or filtering in the case of textual and web data), while if $|L| > 2$, then it is called a multi-class classification problem. However, it is impossible to categorize each document under a single label because of the nature of the text

overlapping each other in the category spaces. For example, the economics field often overlaps (relates) with the political science field.

In multi-label classification, the examples are associated with a set of labels Y ⊆ L. In the past, multi-label classification was mainly motivated by the tasks of text categorization and medical diagnosis. Text documents usually belong to more than one conceptual class. The multi-label text categorization assigns more than one predefined category to an "unseen" document and it is called as overlapping text categorization tasks because it is the task of assigning an object simultaneously to one or multiple category [20].

2.2.1.2. Category-Pivoted Versus Document-Pivoted Text Classification

There are two different ways of using a text classifier. Given dj ⊆ D, we might want to find all the ci ⊆ C under which it should be filed (document-pivoted categorization DPC); alternatively, given ci ⊆ C, we might want to find all the d j ⊆ D that should be filed under it (category-pivoted categorization CPC). DPC is thus suitable when documents become available at different moments in time, e.g., in filtering e-mail. CPC is instead suitable when

(i) a new category c |C|+1 may be added to an existing set C = {c1, . . . , c|C|} after a number of documents have already been classified under C, and

(ii) these documents need to be reconsidered for classification under c|C|+1

(iii) Document-pivoted categorization is used more often than Category-pivoted categorization, as the former situation is more common than the latter[19].

"Hard" Categorization versus "Soft" Classification

Hard categorization is when the classifier algorithm decides, yes or no, as to whether a document dj Є D belongs to a category ci Є C . This is the kind of decisions that are taken by autonomous text classifiers, that is, software systems that need to decide and act accordingly without human supervision. A different type of decision, sometimes referred to as a "soft" categorization decision is the one which consists of attributing a numeric score (e.g. between 0 and 1) to the pair (dj , ci) reflecting the degree of confidence of the classifier in the fact that dj belongs to ci . This allows, for instance, ranking a set of documents in terms of their estimated appropriateness for category ci, or ranking a set of categories in terms of their estimated appropriateness for dj. Such rankings

are often useful for non-autonomous, interactive classifiers, i.e. systems whose goal is to recommend a categorization decision to a human expert, who is responsible of taking the final decision. For instance, in a singlelabel text categorization task a human expert in charge of the final classification decision may profit from a system which pre-ranks the categories in terms of their estimated appropriateness to a given document dj [19].

2.3. Text Classification Approaches

Text classification can be done in two different ways: manual and automatic classification.

2.3.1. Manual Approach

In the manual classification, a human annotator interprets the content of text and categorizes it accordingly. This method usually can provide quality results but it's time-consuming and expensive. It's the assignment of one or more categories to documents by human experts. These experts have domain knowledge and familiar with the category structure being used. But, this problem can be improved by means of automatic text categorization.

2.3.2. Automatic Approaches

Automatic text categorization is the process of automatically classifying a set of documents into predefined categories. There are many approaches to automatic text classification, which can be grouped into four different types of systems.

2.3.3. Rule Based approach

Rule-based approaches classify text into organized groups by using a set of handcrafted linguistic rules. These rules instruct the system to use semantically relevant elements of a text to identify relevant categories based on its content. Each rule consists of an antecedent or pattern and a predicted category. Rule-based systems are human comprehensible and can be improved over time. But this approach has some disadvantages. For starters, these systems require deep knowledge of the domain. They are also time-consuming, since generating rules for a complex system can be quite challenging and usually requires a lot of analysis and testing. Rule-based systems are also difficult to maintain and don't scale well given that adding new rules can affect the results of the pre-existing rules.

2.3.4. Supervised approaches

In supervised training, both the inputs and the outputs are provided. Supervised learning as the name indicates the presence of a supervisor as a teacher. Basically supervised learning is a learning in which we teach or train the machine using data which is well labeled that means some data is already tagged with the correct answer. From the training, a model or classifier is constructed so that new unseen document will be classified based on the model constructed for each category.

2.3.5. Unsupervised approaches

Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. In this case, the machine's job is to categories unsorted data based on similarities, patterns, and differences without requiring any prior data training. In contrast to supervised learning, the absence of a teacher implies that the machine will not receive any instruction. The method, also called clustering; it may not found categories which are intuitive to humans.

2.3.6. Hybrid Approach

Hybrids systems combine a base classifier trained with machine learning and a rule-based system, which is used to further improve the results. These hybrid systems can be easily fine-tuned by adding specific rules for those conflicting tags that haven’t been correctly modeled by the base classifier.

2.4. Steps in Automatic

Text Classification With the aim of categorizing a given document into predefined categories, automatic text classification involves Preprocessing (like normalization, tokenization, stop word and number removal, stemming), feature selection and term weighting, learning classifiers and evaluating classifiers.

2.4.1. Pre-Processing

The first step of pre-processing which is used to presents the text documents into clear word format. The documents prepared for next step in text classification are represented by a great amount of features. Commonly the steps taken are.

Tokenization: A document is treated as a string, and then partitioned into a list of tokens.

Removing stop words: Stop words such as “the”, “a”, “and”, etc are frequently occurring, so the insignificant words need to be removed.

2.4.2. Feature Selection

In text processing, words of the text represent discrete, categorical features. We need a way to represent text data for machine learning algorithm. The mapping from textual data to real valued vectors is called feature selection. One of the simplest techniques to numerically represent text is Bag of Words.

Bag of Words (BOW): We make the list of unique words in the text corpus called vocabulary. Then we can represent each sentence or document as a vector with each word represented as 1 for present and 0 for absent from the vocabulary. Another representation can be count the number of times each word appears in a document. The most popular approach is using the Term Frequency-Inverse Document Frequency (TF-IDF) technique.

- **Term Frequency (TF)** = (Number of times term t appears in a document)/(Number of terms in the document)
- **Inverse Document Frequency (IDF)** =log(N/n), where, N is the number of documents and n is the number of documents a term t has appeared in. The IDF of a rare word is high, whereas the IDF of a frequent word is likely to be low. Thus having the effect of highlighting words that are distinct.

2.4.3. Text Classifier Learning

A text classifier for a category is automatically generated by a general inductive process (the learner) by observing the characteristics of a set of pre-classified documents, which dictates the characteristics that a new unseen document should have in order to belong to a certain category. So as to build classifiers for a category, there is a need to have a set of documents for which the category is known. In experimental text classification, it is customary to partition the set of text documents into training set and test set. The training set is the set of documents from which the learner builds the classifier and the test set is the set on which the effectiveness of the classifier is evaluated[19].

2.4.4. Text Classifier Evaluation

As for text search systems, the evaluation of document classifiers is typically conducted experimentally, rather than analytically. The reason is that, in order to evaluate a system analytically (e.g., proving that the system is correct and complete), we would need a formal specification of the problem that the system is trying to solve (e.g., with respect to what correctness and completeness are defined), and the central notion of TC (namely, that of membership of a document in a category) is, due to its subjective character, inherently non formalezeble. The experimental evaluation of a classifier usually measures its effectiveness (rather than its efficiency), that is, its ability to take the right classification decisions.

Table 2.2 is a contingency table [11] for which the formula of classifier accuracy is based.

Table 2. 2 Contingency Table for Computing Classifier Effectiveness

Category Ci		Expert Judgments	
		Class=Yes	Class=No
Classifier Judgments	Class=Yes	TP	FP
	Class=No	FN	TN

In Table 2.2, TP (True Positive) is the number of test documents correctly classified, FP (False Positives) is the number of test documents incorrectly classified, FN (False Negatives) is the number of test documents incorrectly not classified and TN (True Negatives) is the number of test documents correctly not classified. Hence, accuracy, A, is computed using

Formula 2.1

$$\boldsymbol{A = \frac{TP + TN}{TP + TN + FP + FN}}$$

Formula 2. 1 Accuracy of Text Classifier

2.5. Multi-Label Classification

Multi-label learning is concerned with learning from examples, where each example is associated with multiple labels. These multiple labels belong to a predefined set of labels. Assume that an instance x ∈X can be associated with a subset of labels y, which is referred to as the relevant set of labels for x., for which we use subset notation y ⊆ [L] or vector notation y ∈ {0, 1}L , as dictated by convenience. Assume Y is a given set of predefined binary labels Y = {λ1, . . . , λL }. For a given set of labelled examples D = {x1, x2 , . . . , xn} the goal of the learning process is to find a classifier h : X →Y, which maps an object x ∈ X to a set of its classification labels y ∈ Y, such that h(x)⊆ {λ1, . . . , λL} for all x in X. Multi-label classification exhibits several challenges not present in the binary case. The labels may be interdependent, so that the presence of a certain label affects the probability of other labels' presence. Thus, exploiting dependencies among the labels could be beneficial for the classifier's predictive performance.

2.5.1. Methods for Multi-label Classification

Multi-label classification methods can be grouped in two categories:

(1) Problem Transformation Methods.

(2) Algorithm Adaptation Methods.

2.5.1.1. Problem Transformation Methods

Problem transformation methods map the multi-label learning task into one or more single label learning tasks. Select Problem Transformation methods there exist many simple problem transformation methods that transform multi-label Dataset into single-label dataset so that existing single-label classifier can be applied to Multi label dataset. The Select Transformation method replaces the Label-Set of instance with one of its member. Depending on which one member is selected from it, there are several versions existing, such as, select-min (select least frequent label), Select-max (select most frequent label), select-random (randomly select any label). These methods are very simple but it loses some information. In this study we used this method (select transformation method) that is Select-max (select most frequent label)which maps the multi label problem into one single-label problem; In this study using textual data related to news, which were structured using the bag-of-words approach, was carried out, showing an improvement in the results obtained by our proposed multi-label classification method.

Table 2. 3 Original Multi label problem

Example	Attributes	Label set
1	X1	$\{\lambda_1 , \lambda_4\}$
2	X2	$\{\lambda_3 , \lambda_4\}$
3	X3	$\{\lambda_1\}$
4	X4	$\{\lambda_2 , \lambda_3, \lambda_4\}$

Table 2. 4 Transformed multi label problem

Ex.	**Label**
1	λ_4
2	λ_4
3	λ_1
4	λ_4

Max

Ex.	**Label**
1	λ_1
2	λ_3
3	λ_1
4	λ_2

Min

Ex.	**Label**
1	λ_1
2	λ_4
3	λ_1
4	λ_3

Random

Binary Relevance (BR) Classifier:- The baseline approach, called the binary relevance method extents to independently training one binary classifier for each label. In this method, a multi-label problem is converted into |L| number of binary SL (Single label) classification problems where L is a set of labels. Each of the binary classifiers votes separately to get the final result. This method of dividing the task into multiple binary tasks has something in common with the one-vs.-all (Ov A, or one-vs.-rest, Ov R) method for multiclass classification. Note though that it is not the same method in binary relevance we train one classifier for each label, not one classifier for each possible value for the label.

Label Power set (LP):-The most natural approach is the label power set method which generates a new class for every combination of labels and then solves the problem using multiclass classification approaches. The main drawback of this approach is the exponential growth in the number of classes, leading to several generated classes having very few labeled instances leading to over fitting. BR does not consider relationship between labels. This drawback of BR is overcome by LP, also called as LC (Label Cardinality).

Classifier Chain (CC):- Classifier chains method, which is based on the BR method, overcomes the disadvantages of BR and achieves higher predictive performance, but still retains important advantages of BR, most importantly low time complexity. CC offers a general problem transformation method that inherits the efficiency of BR and competes with the high accuracy of more computationally complex methods. The Classifier Chain approach like LP, also try to overcome the drawback of BR. Similar to BR, a ML problem is transformed into |L| number of SL problems where L denotes a set of labels and for each label Lj, a separate binary classifier Cj is designed. But the input for each classifier Cj is different. Like LP classifier, CC also needs selection of base classifier, uses J48 as base classifier by default.

1.5.1.2. Algorithm Adaptation Methods (AAM)

It extends and adapts the existing specific learning algorithm to directly handle the multi lable problem. It is an algorithm dependent method. Many methods belong to this category. Multi-Label Decision-Tree (ML-DT): This method is belonging from Decision Tree based method category. It is an adaptation of the well-known C4.5 Algorithm to handle multi-label data. The process is accomplished by allowing multiple labels in the leaves of the tree; the output of C4.5 is a decision tree which is constructed from top-down manner. The formula for calculating the entropy is modified for solving multi-label problems. The modified entropy sums all the entropies for each individual label. The Key property of ML-DT is its computational efficiency:

$$Entropy\ (D) = \sum_{i=1}^{q} -p_j\ log_2\ p_j - (1 - p_j) log_2(1 -\ p_j)$$

Equation1. 1 Entropy

Where D is the set of instances in the dataset and pj is the fraction of instances in D that belongs to the label j.

Multi-Label k Nearest Neighbors (ML-KNN): ML-KNN is a lazy learning approach and it is one of algorithm adaptation based method. It extends the traditional K Nearest Neighbor (KNN) method for multi-label classification. In addition to KNN, ML-KNN uses Bayesian reasoning approach. Firstly, k- nearest neighbors of a given test instance will be selected from the training set, then each label occurrence in the neighbor training set will be counted. Finally statistical analysis mechanism called Maximum Posteriori Principle (MAP) will be applied. Prior and posterior probability of a label will be estimated from the training set. This probability estimation will be used for the final prediction of the label set of a test instance[21]. In this Euclidean metric is used to measure distances between instances[18].

Support Vector Machine with Heterogeneous Feature Kernel (SVM-HF): This method exploits relationship among the classes. It enhances the basic purely text based SVM learner by augmenting the feature set with |C| extra features, one for each label in the dataset. The cyclic dependency between features and labels is resolved iteratively. Cosine similarity measure is used to calculate the similarity between two documents[18].

Ranking Support Vector Machine (Rank-SVM): It is a ranking approach for multi-label learning that is based on SVM. It is used to minimize the Ranking-loss. The main function they use is the average fraction of incorrectly ordered pairs of labels [21]. Multi-label Tree based Boosting methods: AdaBoost.MH & AdaBoost.MR: These two methods are based on Tree Based Boosting. AdaBoost.MH and AdaBoost.MR are two extension of AdaBoost for multi-label data. In this purpose of using concept of boosting is to find a highly accurate classification rule by combining many weak or base hypotheses, each of which may be moderately accurate. In this approach, the goal of the learning algorithm is to predict all of the correct labels. Thus, the learned classifier is evaluated in terms of its ability to predict approximation of the set of labels associated with the given document.

AdaBoost.MH is extended in to produce better human related classification rule. It is designed to minimize Hamming loss and Adaboost.MR is designed to find hypothesis which places the correct labels at the top of ranking.

Neural Network based: BP-MLL is an extension of the popular back-propagation algorithm for multi-label learning. It is neural network based method. The main modification is the introduction of a new error function that takes multiple labels into account. Given multi-label training set

2.5.2. Evaluating Multi-label Learners

Multi-label evaluation metrics that are widely used throughout the studies in the field are divided into two groups[22]: (1) Instance Based Metrics, (2) Label-Based Metrics. These two metrics indicate how well the algorithms perform.

2.5.2.1. Instance-Based Metrics

Instance-based metrics are evaluated for every instance and averaged over the whole dataset. Exact Match, Hamming Score, and Instance-Based {Accuracy, Precision, Recall, F1-Score} [22].

2.5.2.2. Label-Based Metrics

Label-based metrics are evaluated for every label and averaged over examples within each individual label. Macro and Micro-Averaged Precision, Recall and F1 Score [22]. In what follows, we show the measures used in this work to evaluate the multi-label classification methods considered.

Macro-precision:

$$\boldsymbol{Macro\_Precision} == \frac{\mathbf{1}}{\mathbf{Q}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{Q}} \frac{\boldsymbol{tpj}}{\boldsymbol{tpj + fnj}}$$

Equation1. 2 Macro_ Precision

where tpj and fpj are the number of true positives and false positives for the label λj considered as a binary class.

Macro-recall:

$$\boldsymbol{Macr\_recall} = \frac{\mathbf{1}}{\mathbf{Q}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{Q}} \frac{\boldsymbol{tpj}}{\boldsymbol{tpj + fnj}}$$

Equation1. 3 Macr_recall

where tpj and fpj are defined as for the macro-precision and fnj is the number of false negatives for the label λj considered as a binary class.

Macro-F1: is the harmonic mean between precision and recall, where the average is calculated per label and then averaged across all labels. If pj and rj are the precision and recall for all λj∈ h(xi) from λj∈*y*i, the macro-F1 is

$$\boldsymbol{Macro - F1 = \frac{1}{Q} \sum_{J=1}^{Q} \frac{2xpjxrj}{pj + rj}}$$

Equation1. 4 Macro F1

Micro-precision (precision averaged over all the example/label pairs) is defined as:

$$\boldsymbol{Micro\_precision = \frac{\sum_{j=1}^{Q} tpj}{\sum_{j=1}^{Q} tpj + \sum_{j=1}^{Q} fpj}}$$

Equation1. 5 Micro_ Precision

where tpj, fpj are defined as for macro-precision. Micro-recall (recall averaged over all the example/label pairs) is defined as

$$\boldsymbol{Micro\_recall = \frac{\sum_{j=1}^{Q} tpj}{\sum_{j=1}^{Q} tpj + \sum_{j=1}^{Q} fnj}}$$

Where tpj and fnj are defined as for macro-recall

Equation1. 6 Micro recall

Micro-F1: is the harmonic mean between micro-precision and micro-recall.Micro-f1 is defined as:

$$\boldsymbol{Micro_{f1} = \frac{2 * \mathbf{micro_{precision}} * \mathbf{micro\_recall}}{micro_{precision} + micro\_recall}}$$

Equation1. 7 Micro F1

2.6. Neural Network

A neural network has at least two physical components, namely, the processing elements and the connections between them. The processing elements are called neurons, and the connections between the neurons are known as links. Every links has a weight parameter associated with it. Each neuron receives stimulus from the neighboring neurons connected to it, process the information, and produces an output. Neurons that receive stimuli from outside the network are called input neurons. Neurons whose outputs are used externally are called output neurons. Neurons that receive stimuli from other neurons and whose output is a stimulus for other neurons in the neural network are known as hidden neurons[23].

2.6.1. Deep Neural Networks

Neural networks can be created from at least three layers of neurons: The input layer, the hidden layer(s) and the output layer. The hidden layer — or layers — in between consist of many neurons, with connections between the layers. As the neural network "learns" the data, the weights, or strength, of the connections between these neurons are "fine-tuned," allowing the network to come up with accurate predictions. When a neural network has many layers, it's called a deep neural network, and the process of training and using deep neural networks is called deep learning, Deep neural networks generally refer to particularly complex neural networks. These have more layers (as many as 1,000) and — typically — more neurons per layer. With more layers and more neurons, networks can handle increasingly complex tasks; but that means they take longer to train. Because GPUs are optimized for working with matrices and neural networks are based on linear algebra, the availability of powerful GPUs has made building deep neural networks feasible. In a "classic" neural network, information is transmitted in a single direction through a network, where each layer is fully connected to its neighbors, from the input to the output layers. However, there are two other types of neural networks that are particularly well-suited for certain problems: convolutional neural networks (CNNs) and recurrent neural networks (RNNs)[24].

2.6.1.1. Convolutional Neural Networks

Convolutional neural networks (CNNs) are frequently used for the tasks of image recognition and classification. For example, suppose that you have a set of photographs and you want to determine whether a cat is present in each image. CNNs process images from the ground up. Neurons that are located earlier in the network are responsible for examining small windows of pixels and detecting simple, small features such as edges and corners. These outputs are then fed into neurons in the intermediate layers, which look for larger features such as whiskers, noses, and ears. This second set of outputs is used to make a final judgment about whether the image contains a cat[17].

CNNs are so revolutionary because they take the task of localized feature extraction out of the hands of human beings. Prior to using CNNs, researchers would often have to manually decide which characteristics of the image were most important for detecting a cat. However, neural networks can build up these feature representations automatically, determining for themselves which parts of the image are the most meaningful.

2.6.1.2. Recurrent Neural Networks

Whereas CNNs are well-suited for working with image data, recurrent neural networks (RNNs) are a strong choice for building up sequential representations of data over time: tasks such as handwriting recognition and voice recognition. Just as you can't detect a cat looking at a single pixel, you can't recognize text or speech looking at a single letter or syllable. To comprehend natural language, you need to understand not only the current letter or syllable but also its context. RNNs are capable of "remembering" the network's past outputs and using these results as inputs to later computations. By including loops as part of the network model, information from previous steps can persist over time, helping the network make smarter decisions. Long short term memory (LSTM) units, gated recurrent units (GRUs) and bidirectional neural network (BRNN) can be added to an RNN to allow it to remember important details and forget the irrelevant ones[25].

2.6.2. Biological Neuron

A neuron has a cell body, a branching input structure: the dendrite, and a branching output structure: the axon, which connects to dendrites via synapses. Electro-chemical signals are propagated from the dendrite input, through the cell body, and down the axon to other neurons [26].

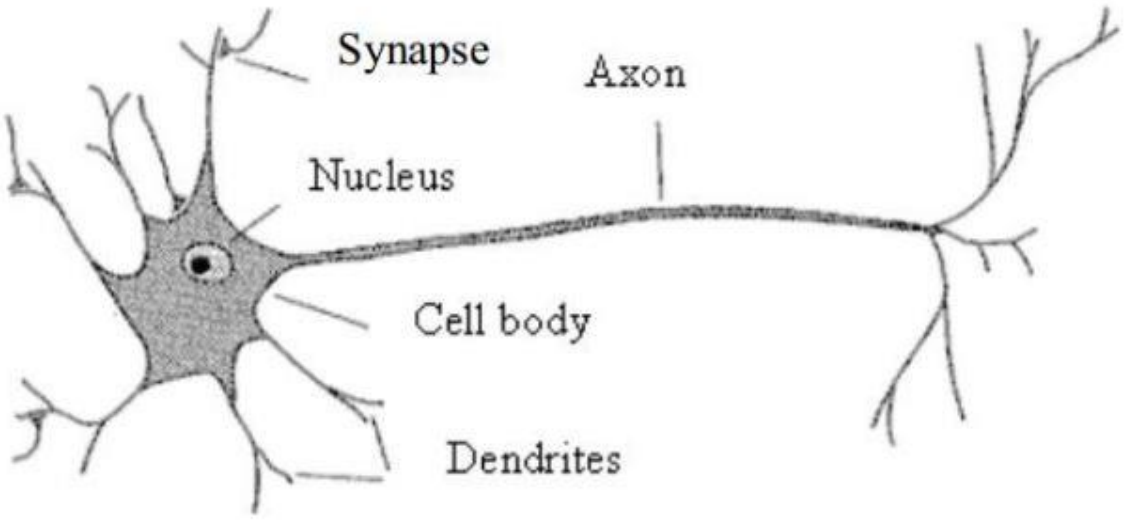


Figure 2. 1 Structure of Biological Neuron

A neuron fires, produces output, if its input signal exceeds a certain amount (the threshold), in a short period of time. Synapses vary in strength. Good connections allowing a large signal and slight connections allow only a weak signal. Each neuron produces only one output signal. The output signal is transmitted through the neurons outgoing connection. The outgoing connection splits into a number of branches. The outgoing branches terminate at the incoming connections of other neurons [26].

2.6.3. Artificial Neural Networks (ANN)

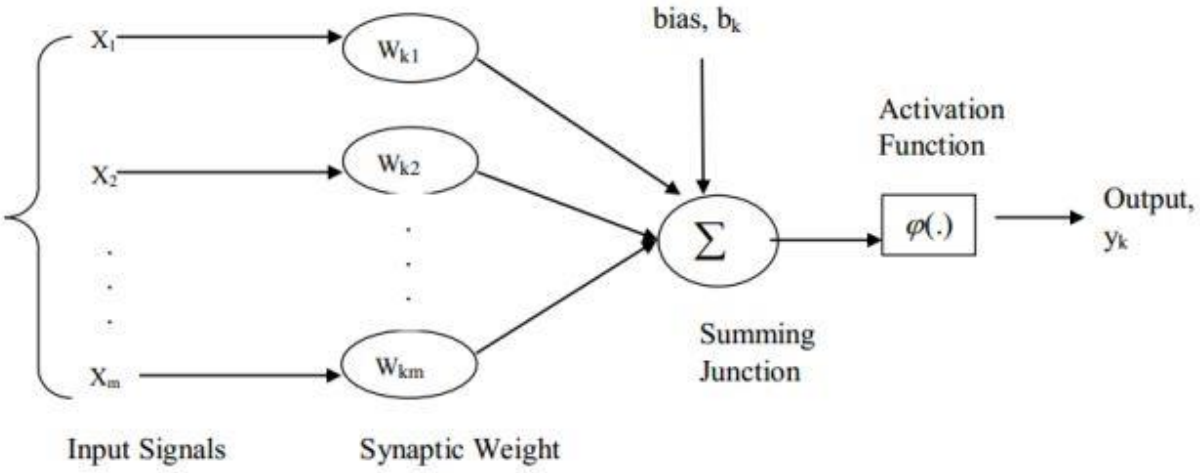
An Artificial Neural Network (ANN) is an information processing model that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. Two important things make neural networks similar to the brain. The first one is, it acquires knowledge from the environment through a learning process using the network. The other one is, the acquired knowledge is stored as weights, which shows interneuron connection strength[27].

2.6.4. Model of Artificial Neuron

Neuron is information processing element that is fundamental to the operation of the network. There are three constituents of neural networks[27].

1. Synapses: are connecting links, which have weights that shows its strength. Weights represented as wkj refers k neuron with j weight.
2. Adder: that sum input signals from sending neuron/s to receiving signals.
3. Activation Function (Squashing Function): for limiting the amplitude of neuron output.

Figure 2. 2 Illustrates the details of neuron model



In Figure 2.2, x1, x2,…,xm refer to input signals; wk1, wk2, …, wkm are synaptic weights of neuron k; bias bk has the effect of, increasing: for positive values or lowering: for negative values, the net input of the activation function.

2.6.5. Architecture of Artificial Neural Network

The architecture of NN can be of single layer or multilayer. In a single layer Neural Network, only one input layer and one output layer is there, while in multilayer neural network, there can be one or more hidden layer. An artificial neuron is an abstraction of biological neurons and the basic unit in an ANN. The Artificial Neuron receives one or more inputs and sums them to produce an output. Usually the sums of each node are weighted, and the sum is passed through a function known as an activation or transfer function. The intension of using neural networks is to model natural processing of the human brain. The brain is highly complex, nonlinear and parallel computing system. It has the capability to organize its constituent neurons, information processing elements, to perform computations like pattern recognition, perception, motor control, etc.; faster than the digital computers with such tasks[7].Architecturally, an artificial neural network is modeled using layers of artificial neurons, or computational units able to receive input and apply an activation function along with a threshold to determine if messages are passed along. In a simple model, the first layer is the input layer, followed by one hidden layer, and lastly by an output layer. Each layer can contain one or more neurons.

2.6.6. Advantages of Using Neural Networks

Neural networks are based on these properties, which include self-organization, parallel design, noise tolerance, and generalisation. Sonali B. enumerates the following advantages of neural networks:

- Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.
- Self-Organisation: An ANN can create its own organisation or representation of the information it receives during learning time.
- Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
- Pattern recognition is a powerful technique for harnessing the information in the data and generalizing about it. Neural nets learn to recognize the patterns which exist in the data set.
- The system is developed through learning rather than programming. Neural nets teach themselves the patterns in the data freeing the analyst for more interesting work.
- Neural networks are flexible in a changing environment. Although neural networks may take some time to learn an unexpected extreme change they are excellent at adapting to constantly changing information.
- Neural networks can build informative models whenever conventional approaches fail. Because neural networks can handle very complex interactions they can easily model data which is too difficult to model with traditional approaches such as inferential statistics or programming logic.
- Performance of neural networks is at least as good as classical statistical modeling, and better on most problems. The neural networks build models that are more reflective of the structure of the data in significantly less time.

2.6.7. Back Propagation Algorithm

The back propagation method is the most well-known and basic neural network learning algorithm. The Back-Propagation learning method is among the most significant advancements in neural networks. The algorithm's associated network is referred to as the Back-Propagation network (BPN). This network is used with multilayer feed-forwarding networks (input, output, and hidden), which are made up of processing components with continuous activation functions that are differentiable. This approach provides a procedure for altering the weights in a BPN to accurately identify the given input patterns for a specified set of training input-output pairs.

The gradient-descent approach is the fundamental idea behind this weight updating algorithm. Error propagates back to the concealed unit using this technique. The objective of training a neural network is to attain equilibrium between the network's responsiveness and its capacity to provide rational answers to inputs that are comparable, but not identical, to the training set.Therefore, once the input variables are selected, the previously mentioned functions regarding the network design could be systematized in the following steps:

- Selecting an initial configuration, which typically consists of one hidden layer with the number of neurons equal to half the sum of the input and output variables.
- Implementing a certain number of experiments with each configuration while retaining the best network, taking the criterion as mistake of selective data set. It is necessary to conduct a sufficient number of experiments with each network configuration to overcome the configurations that are finding local minimums, and preferably practice of resampling.
- In each experiment, if the network does not show satisfactory performance (under-learning), it is necessary to try adding more neurons in the hidden layer of the network. If this does not help, it is necessary to try to add a new hidden layer.
- If the function of selection error begins to increase (over-learning) it is necessary to try to reconfigure the network by subtracting the individual neurons from the hidden layer or even the entire hidden layer in complex network constructions.
- Once you determine effective network configuration, it is necessary to rearrange the existing data in new sets, and generate a new network starting from the initial configuration through previous training.

The BPN algorithm is produced in two phases here. In the first phase forward signal propagation occurs in the network. In the second phase the error terms are fed back to all other input units. In this case they are the feature vectors. Now the algorithm provided below:

1) Phase 1: Propagation: Each propagation involves the following steps:
    - Forward propagation of a training pattern's input through the neural network in order to generate the propagation's output activations.
    - Backward propagation of the propagation's output activations through the neural network using the training pattern target in order to generate the deltas of all output and hidden neurons.
2) Phase 2: Weight update: For each weight-synapse follow the following steps:
    - Multiply its output delta and input activation to get the gradient of the weight.
    - Subtract a ratio (percentage) of the gradient from the weight.

2.6.7.1. Text Classification Using Back Propagation Algorithm

This approach uses a non-linear feed-forward neural network that was trained using the backpropagation learning rule to classify text. In other words, we use text to classify text using supervised learning. The use of ANS in text classification makes sense. ANS offers a superior answer for problems that cannot be solved sequentially or using sequential algorithms. It helps with nontrivial mapping functions and complicated pattern recognition.

2.7. The Oromo language

The Oromo language, also known as Afaan Oromo, or "Oromo mouth," is a member of the Cushitic branch of the Afroasiatic language phylum, which is thought to be divided into six primary branches or families. The Cushitic branch is further subdivided into four groups: East, Central, South, and North. One of the Lowland languages of the East Cushitic group is Afaan Oromo. After Hausa and Arabic, Gene Gragg (1982) asserts that Afaan Oromo is likely the third most commonly spoken Afro-Asian language worldwide.

Currently, afaan Oromo is the working language of the federal government and is the official language of the largest regional state in Ethiopia, Oromia. It is also used as a working language in offices and for all non-language subjects in junior-secondary schools. In terms of writing, Afan Oromo has been using the "Qubee" (an alphabet based on Latin) as their official script since 1991.

The main inspiration for this writing system came from the fact that its letters accurately depict the vowels and consonants of a language.

2.7.1. Consonant and Vowel Phonemes

Like English language, Afaan Oromo has vowels and consonants. Afaan Oromo vowels are represented by the five basic letters such as a, e, i, o, u and five long vowels, indicated in the orthography by doubling the five vowel letters (aa, ee, ii, oo, uu). The difference in length of vowels results in change of meaning. These vowels pronounced in sharp and clear fashion which means, each and every word is pronounced strongly.

Example

Lafa- ground

Laafaa- soft

Table 2. 5 Vowels (Dubbiistoota) (Source: Kula et al[24]

Vowels			
	Front	**Center**	**Back**
Close	**i/I/,ii/i:/**		**u/U/,uu/u:/**
Mid	**e/E/,ee/e:/**		
Open		**a/A/**	

There are 28 consonants in ‘Qubee Afaan Oromo’ including digraphs. They are listed in table 2.6 with their sounds. Geminating (doubling a consonant) is also significant in Afaan Oromo because consonantal length can distinguish words from one another. Most Afaan Oromo consonants can be geminated except ‘h’ and digraphs.

Table 2. 6 Afaan Oromo consonants (Source: Kula et al[24])

Consonants						
		Bilabial/Labiodental	Alveolar Retroflex	Palato-alveolar Palata	Velar	Glottal
Stops and affricates	Voiceless	(p)	T	ch/t∫/	k	`/?/
	Voiced	b	D	j/d_3/	g	
	Ejective	ph/p`/	x/t`/	c/t ∫/	q/k`/	
	Implosive		dh/d/			
Fricatives	Voiceless	f	s	sh/∫/		h
	Voiced	(v)	(z)			
Nasals		m	N	Ny/n/		
Approximants		w	L	y/j/		
Rhotic			R			

2.7.2. Morphology

Morphology is the study of morphemes and their arrangements in forming words. Morphemes are the minimal meaningful units which may constitute words or parts of words [8].The two broad classes of morphemes are stems and affixes (prefix, infix and suffix). The stem is the main morpheme of the word, supplying the main meaning whereas affixes are used to add additional meaning to words[8].

Like in a number of other African and Ethiopian languages, Afaan Oromo has a very complex and rich morphology. Unlike languages like English, gender, number, definiteness, prepositions and others information are attached to Afaan Oromo Nouns, adjectives, verbs and etc that resulted in complex morphology of the language . For example, from the noun Kitaaba (book), the following words are generated through affixation: Kitaabilee (books), Kitaabicha (the book), Kitaaba ishee (her book), Kitaaba isaa (his book), Kitaabarraa (from book), Kitaabattii (the book), etc.

2.7.3. Over view of Afaan Oromo Stemmer

In the Afaan Oromo most of the grammatical information is conveyed through suffixes attached to the root or stem of words. Almost all Afaan Oromo nouns in a given text have person, number, gender and possession markers, which are concatenated and affixed to a stem form. According to [28] Afaan Oromo suffixes categorize into three basic groups: derivational, inflectional, and attached suffixes. Afaan Oromo attached suffixes are particles or postpositions like -arra, -bira, -

irra, -itti and -dha while inflectional suffixes comprises the most frequent and dominant suffixes such as –n, -lee, -een, -icha, -tu, -oo, - oota and -wwan. Oromo derivational suffixes such as -achuu, -eenyaa, -inaand -ummaa are often used for formation of a new words in the language following the stems or base forms of Afaan Oromo words. As an example, the Afaan Oromo stem or root man- can take the following different affixes forms: manoota (man-+oota), manneen (man- + een), manawwan (man- + wwan). Based on our current linguistic analysis and observations of Afaan Oromo syntax and morphological features, the most common order/sequence of the above major three Afaan Oromo suffixes (within a given word) is: .

2.8. Related Works

Different researches have been done for categorizing text documents in well-organized manner [6].Most of the text categorization had applied unsupervised and supervised learning methods independently. The commonly used techniques include k-means, repeated bisection, nearest neighbor classifiers, Bayesian classifiers, decision trees, and support vector machine. On the other hand, text categorization that combines text classification and clustering is an emerging topic of text categorization.

2.8.1. Local Languages

As to the knowledge of the researcher, four researches have been done on automatic Afaan Oromo text classification in case of single label classification by Abera[8], Kemal,Naol and Etana. But until now multi label text classification is not done for all local language(Afaan Oromo, Amharic and etc).

Abera[8] employed four automatic classifiers Sequential Minimal Optimization (SMO) algorithm from Support Vector Machines, NaiveBayesMultiNominal (NBM) from Bayesian Classifiers, J48 algorithm from the Decision trees and K-Nearest Neighbor (KNN) from the Lazy Learners for the Afaan Oromo news items classification. According to Abera [8]both SMO and NBM classifiers have best accuracy over the others, whereas the others J48 and KNN classifiers have the average accuracy on each category. The best result (accuracy) from both the SMO and BayesMultiNominal classifiers is obtained when the number of instance documents is approximately equal in the classes, and it was for the four categories of news items. The lower accuracy seen is for J48 on category of 7 classes and on category of 11 classes. According to his experiment, SMO has better accuracy over the other classifiers for the Afaan Oromo news items classification.

Kemal used four machine learning techniques to categorize the Afaan Oromo news texts those are Bayes Network, Naïve Bayes, J48 and SMO classifiers. According to Kemal from four classifiers both Bayes Net and J48 classifiers have better performance than the other two classifiers. He selects the BayesNet classification model as the Afaan Oromo news text categorization system due to its effectiveness and efficiency. Table 2.4 summarizes the results obtained by previous research works.

Naol employ J48, NaïveBayes, BayesNet, and SMO classifier algorithms were implemented for training text classification model depending on 8 main classes of documents. Among those classifications algorithms, J48 algorithm shows higher performance 94.3755% and hence it was utilized for Afan Oromo nonfiction text categorization.

Etana among the automatic classifiers he used Naïve Bayes (NB) and bayes networking. He employ apriori algorithm for generating frequent item in a given text document. The performance of the classification is analyzed to measure the accuracy of the classifiers in categorizing the Afaan Oromo news documents in to specified categories. The best result obtained by bayes networking Classifier is 97.15% and Naïve Bayes (NB) is 95.666% on nine categories data. According to his research bayes networking Classifier is more relevant for categorizing Afaan Oromo news document.

Table 2. 7 Previous Research Works on Automatic Afaan Oromo Text Classification

Name	Catagories	Methodes	Accurancy	
Abera	11	SMO	4-category data	95.82%
			7-categoy data	93.70%
			11-categoy data	91.49%
		NBM	4-category data	96.58%
			7-category data	87.62%
			11-categoy data	87.43%
		J48	4-category data	86.69%
			7-category data	79.69%
			11-categoy data	82.05%
		KNN	4-category data	91.25%
			7-category data	81.87%
			11-categoy data	85.27%
Kemal	6	Bayes Network,	97.77%	
		Naïve Bayes	93. 66%	
		J48	96.58%	
		SMO	84.93%	
Naol	8	J48	94.3755%	
		NaïveBayes	80.7889%	
		BayesNet	92.038%,	
		SMO	92.9876%	
Etana	9	Bayes Networking	97.15%	
		Naïve Bayes	95.666%	
Lelisa[29]	1	CNN	98.4%	
		LSTM	95%	
		BiLSTM	97.28%	

CHAPTER THREE

3. MATERIALS AND METHODS

3.1.Introduction

In this chapter we are going to discuss the algorithms used, the Afaan Oromo news text classification prototype, normalization, tokenization, stop word and number removal, document representation, feature selection or feature transformation, learning classifier and evaluating classifier of Afaan Oromo news texts preprocessing tasks.

The dataset used in this study was made up of several text documents that were gathered from the data source. Stop words, punctuation, and any other extraneous material was eliminated as part of the pre-processing and cleaning of these papers. Additionally, we'll make sure the dataset's distribution of classes is balanced.

3.2.Development Tools

Several open-source libraries, including NumPy, Pandas, Scikit-Learn, NLTK, Seaborn, and Matplotlib, were used along with the Python programming language by me. For feature extraction, we'll additionally employ a pre-trained word embedding model like GloVe or Doc2Vec.

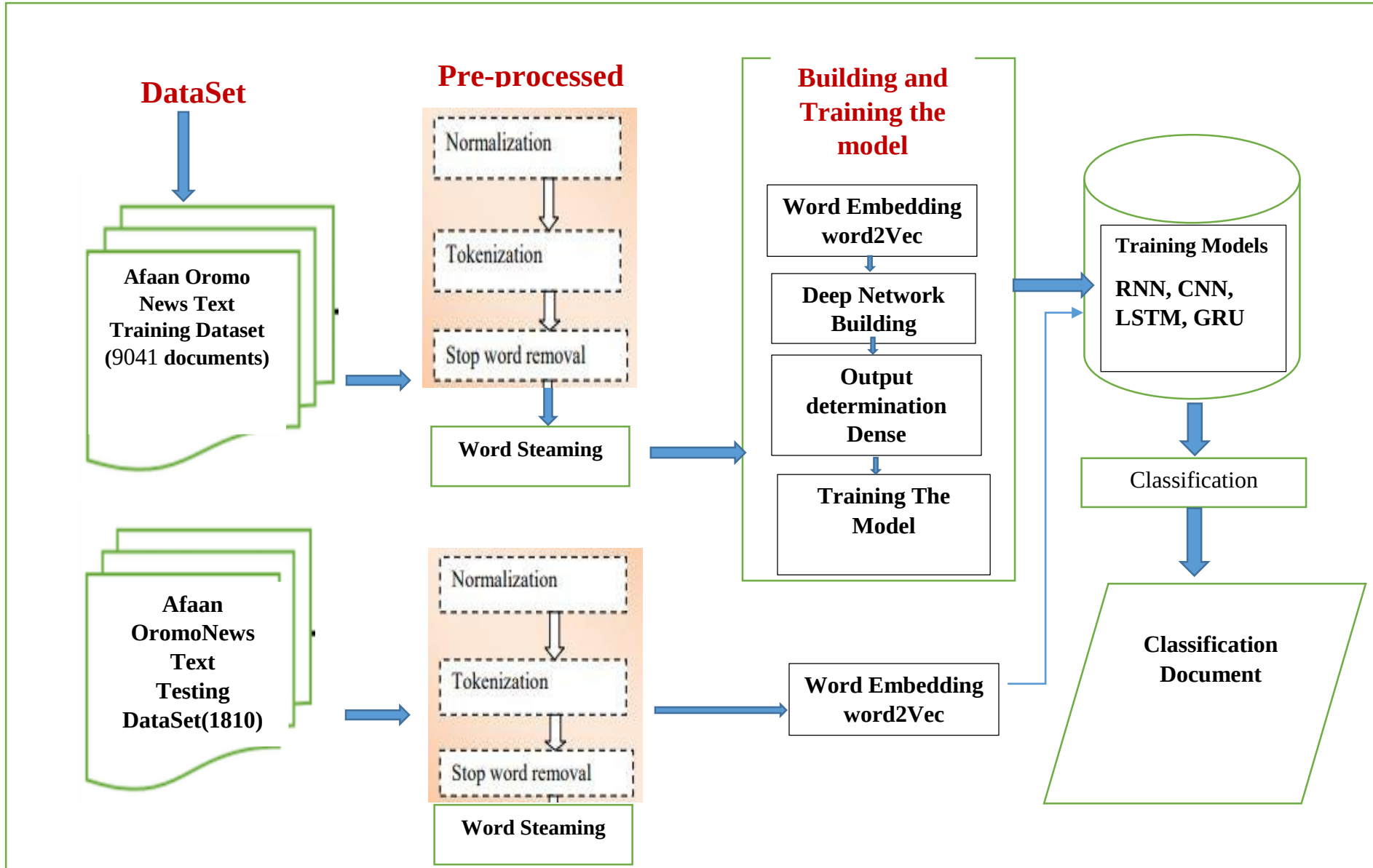
3.3. Methods

The study employed a deep learning approach to classify text documents into pre-defined categories.

3.4.Architecture of Afaan Oromo News Text Classification

The Afaan Oromo text news classifier's architecture is made up of seven fundamental parts. define the issue, gather data, pre-process, select a model building, Develop the model, Review the model, Finally, deploy the model after making necessary adjustments. The model is fed news texts written in Afaan Oromo. Preprocessing is the procedure used to get data ready for testing. The text is then transformed into numerical vectors after undergoing preprocessing operations as normalization, tokenization, punctuation, stop words, number removal, stemming, lemmatization, feature extraction, and document representation. Considering that training and test datasets are created from these preprocessed datasets in the form of matrices or comma-separated values (CSV), they are utilized for training and testing, respectively. The training data are used to create a model (classifier) is constructed. The test data are used to evaluate the model. Classes are assigned for news texts that were not seen during training as a result of the assessment. Finally, a judgment is reached based on the accuracy of the test results.

Figure 3. 1 Architecture of Automatic Afaan Oromo News Text Classification



3.5.Training and testing dataset

Deep learning model requires large amount of data for training, testing and validation. The data that we collect are divided in to training testing and validation and all data are pass through preprocessing phase for training to be an input for the embedding component. Those data are collected from seven domains. As shown in table 3.1, The corpus is divided into training and testing data using split of 20% for training and 80% are for training and validation

Table 3. 1 the number of news items in each category and the total number of news items

No	Class Name	Training	Testing
1	Balaa (Accident)	**1230**	246
2	Barnoota (Education)	950	190
3	Teknoolojii (Technology)	1100	220
4	Fayyaa (Health)	950	190
5	Siyaasa (Politics)	1020	204
6	Qonnaa (Agriculture)	1250	250
7	Spoortii (Sport)	1350	270
8	Biizinasii (Business)	1200	240
	Total	9041	1810

3.5.1. Preprocessing Oromo News Dataset

Preprocessing is the process of converting a document's textual information into a logical interpretation that classification algorithms can understand. Tokenization, stop word and number removal, stemming, document representation, and CSV (comma-separated values) production are tasks that are completed for the preprocessing of Afaan Oromo news. Following the completion of the preprocesses, the classifier is built using Python and the Recurrent Neural Networks (RNNs) learning technique. Finally, the model is assessed in light of the precise results. The techniques used to prepare the data for the classification task are covered in the following sections.

3.5.2. Normalization

Text normalization is a pre-processing operation used to enhance the text's quality and prepare it for machine processing. Case normalization, tokenization and stop word removal, Parts-of-Speech (POS) labeling, and stemming are the four key phases in text normalization.

```
start
Input: Text document corpus(F)and list of corresponding
normalizers(N), homophones(n) characters
Output: Normalized text document
    split words from file(F) in character level
    Find homophones (n) from F
      if n is in F
        replace it by the corresponding normalizer(N)
      else
        navigate through all words in F
     save F in new file as New file
stop
```

Algorithm 3. 1 Algorithms for Normalization

3.5.3. Tokenization

While linguistic preprocessing works with creating equivalence classes of tokens, which are the set of terms that are indexed, tokenization is the process of breaking character streams into tokens. In this work, tokenization is also utilized to tokenize documents and detach specific characters, including punctuation marks[30].The elimination of punctuation, numbers, and symbols is a part of the tokenization process. Typically, punctuation is utilized to meet a language's grammatical requirements. Python programming can manage the tokenization process by utilizing white space as a separator. However, there are situations when punctuation marks are placed right after a word, in which case the alignment tool will interpret them differently. Consequently, removing punctuation marks is also part of tokenization.

```
start
    Input: Normalized text document corpus(F)
    Output: Tokenized and numerically represented corpus
    split sentence's and phrases of file (F) into tokens
    assign a number to each unique token
stop
```

Algorithm 3. 2 Algorithms For Tokenization

3.5.4. Stop Word Removal

Words that appear too frequently in the collection's documents are poor discriminators. A word that appears in 80% of the collection's texts, according to Abera [7], is useless as an index term. Conjunctions, articles, and prepositions are excellent additions to a list of stop words. Common grammar-related stop words in Afaan Oromo include the following: ture, kan, kaan, kana, fi, akka, Hin, ni, irraa, itti, etc. These words are not helpful for identifying documents. According to the reporters of OBN and FBC, there are other stop words that are specific to the news, such as himaniiru, ibsaniiru, jedhaniiru, jedhan, and ibsameera. These stop words are used for elaboration and are common to all news.

```
start
    Input: Text document corpus(F) and list of stop words(S)
    Output: Text documents free from stop words
    find stop words (s) from file(F)
      if S is in F
        remove S from the file F
      else
        navigate through all words in F
     save   F in new file as new file
stop
```

Algorithm 3. 3 Algorithm For Stop Word Remove

3.5.5. Word embedding

Word embedding or word vector is an approach with which we represent documents and words. It is defined as a numeric vector input that allows words with similar meanings to have the same representation. It can approximate meaning and represent a word in a lower dimensional space. These can be trained much faster than the hand-built models that use graph embedding's like WordNet.

Term frequency-inverse document frequency (TF-IDF)

Term frequency-inverse document frequency is the machine learning algorithm that is used for **word embedding for text**. It comprises two metrics, namely term frequency (TF) and inverse document frequency (IDF). This algorithm works on a statistical measure of finding word relevance in the text that can be in the form of a single document or various documents that are referred to as corpus.

The term frequency (TF) score measures the frequency of words in a particular document. In simple words, it means that the occurrence of words is counted in the documents. The inverse document frequency or the IDF score measures the rarity of the words in the text. It is given more importance over the term frequency score because even though the TF score gives more weightage to frequently occurring words, the IDF score focuses on rarely used words in the corpus that may hold significant information.

Simpler machine learning and natural language processing problems including information retrieval, stop word removal, keyword extraction, and basic text analysis can all be solved with the help of the TF-IDF algorithm. It does not, however, effectively convey the semantic meaning of words in a sequence.

Now let's understand it further with an example. We will see how vectorization is done in TF-IDF.

$Tf\text{-}idf_{i,j}$ = Term $Frequency_{i,j}$ X Inverse Document $Frequency_{i}$

Where

$$\text{Term Frequency}_{i,j} = \frac{\text{Term i frequincy in document j}}{\text{Total no of Term in document j}}$$

$$\text{Inverse Document Frequency}_i = \log\left(\frac{\text{Total Documents}}{\text{No. of documents containing term i}}\right)$$

3.5.6. Deep Network Modeling

After the data are preprocessed and the embedding component is built the next component is building a deep network for training the data. As we describe in the literature review, there are different deep network architectures that are used for different applications of NLP such as RNN, CNN and so. RNN is a sequential model in the field of artificial intelligence and neural network. It processes input data that are represented in a sparse vector sequentially. But, while RNN process inputs sequentially, if the process is held on a large amount of data with a complex and large sequence it may lose important information, and the problems of long-term dependency is happened [45].

3.5.7. Output Determination

The final output of the model is determined by the dense layer which is a fully connected layer that accepts the deep representation of the input data and transforms to the final output class using an activation function Simpler machine learning and natural language processing problems including information retrieval, stop word removal, keyword extraction, and basic text analysis can all be solved with the help of the TF-IDF algorithm. It does not, however, effectively convey the semantic meaning of words in a sequence.

. As a result, we may use the SoftMax activation function to generate the final output that produces values in the range of 0 -to- 1. Thus, it is used as the final layer in classification models.

3.5.8. Training the model

The next process after the output is determined the model is trained using the training data and evaluated using the validation data which is split from the training data using validation split. To train the model the number of epochs and the amount of training data which is trained at a time (batch size) is determined. Training the model using all of the training data at a time is impossible and it need to subdivided to predefined number of batch size. The training capability of the model

is evaluated at each batch size and it is evaluated at each epoch. Finally, the trained model is saved and tested by the testing dataset.

3.6. Classifier

After the model is summarized and fit with the data to be evaluated with the training, testing, and validation data set separately by using the metric and the loss function. When the model is evaluated, the metrics are accuracy and loss; training accuracy, training loss, validation accuracy, validation loss, and testing accuracy. Thus, the higher the accuracy and the lower the loss indicates the better the model performs The process of classification accomplished on the trained model using the test data that pass through the preprocessing phase and embedded with the same dimension of the training and validation data are embedded. After the model is tested the summery of the classification is organized. The classification process also accomplished with external datasets that are not labeled into measure how much the model learns the data and capable of predicting external unknown domain text documents to their appropriate class. Thus, the final output of the testing result is documents with its corresponding class or classified document.

The process of building a classifier using a training dataset that can forecast the test dataset is known as classifier learning. The deep learning approach, CNN, LSTM, BRNN and GRU networks, has been utilized with the Python programming language for the goal of building classifiers.

When working on machine learning models, developers and data scientists frequently use Python as their preferred programming language. The reasons Python is so popular in the sector include its straightforward syntax, sizable community, and the scientific computer friendliness of its mathematical libraries. Because we need our matrix multiplication to be quick, we use the Python package numpy. Import data into Python and build networks for this study. The information has been ready.

Python can read the CSV (comma-delimited) file format in which the data has been prepared. Before it can be applied to classifying text, the CNN and recurrent neural network classifier needs to be trained. The Long Short-Term Memory (LSTM),GRU and BIRRN networks use

unsupervised learning to train the recurrent neural network classifier. A set of training documents and a description of the pre-defined categories to which the training documents belong are needed to train a neural network.

3.6.1. Convolutional Neural Network (CNN)

Convolutional neural networks (CNNs) are a class of deep learning algorithms that are especially well-suited for image recognition and processing tasks. Their architecture, which is modelled after the visual processing of the human brain, consists of convolutional layers, pooling layers, and fully connected layers. CNNs are particularly good at recognising hierarchical patterns and spatial dependencies within images.

Use 1-D convolutional layers that convolve over the input's time dimension to categories text data using convolutions.

3.6.2. Long short-term memory networks (LSTM)

Recurrent neural networks (RNNs) are a kind of neural networks that have the ability to learn long-term dependencies, particularly in sequence prediction tasks. With the exception of single data items like text, LSTM can comprehend the complete sequence of data because it contains feedback links. Applications for this include machine translation and speech recognition. An exceptional type of RNN that performs exceptionally well on a wide range of tasks is the LSTM.

The central role of an LSTM model is held by a memory cell known as a ‘cell state’ that maintains its state over time. The cell state is the horizontal line that runs through the top of the below diagram. It can be visualized as a conveyor belt through which information just flows, unchanged[31].

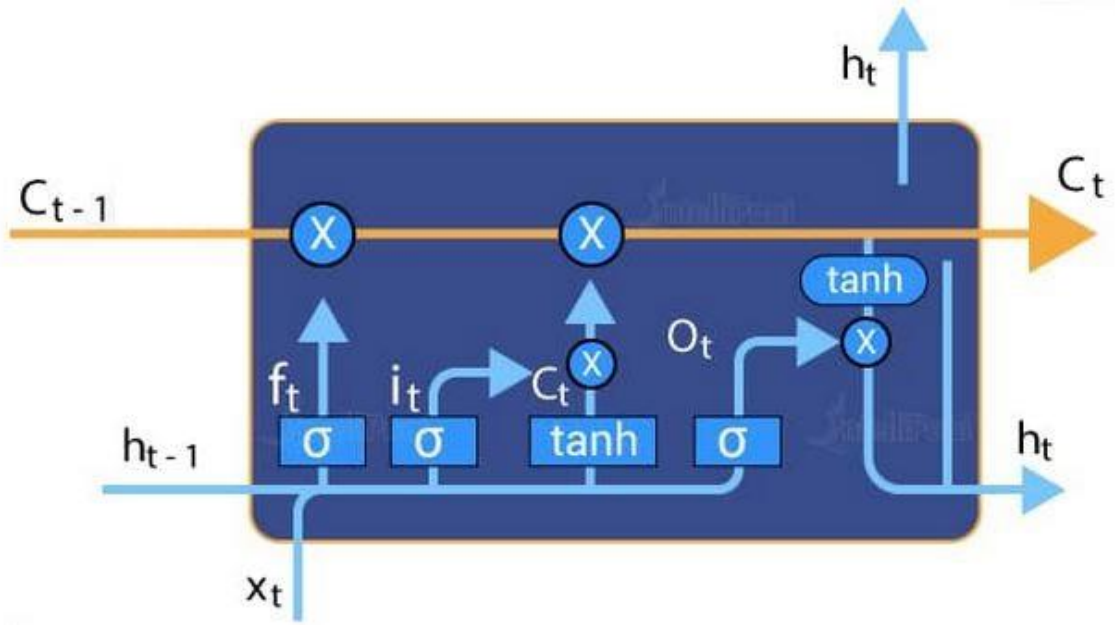


Figure 3. 2 LSTM Modeling sources:(m.tech in AI and ML,2023)

Information can be added to or removed from the cell state in LSTM and is regulated by gates. These gates optionally let the information flow in and out of the cell. It contains a point wise multiplication operation and a sigmoid neural net layer that assist the mechanism.

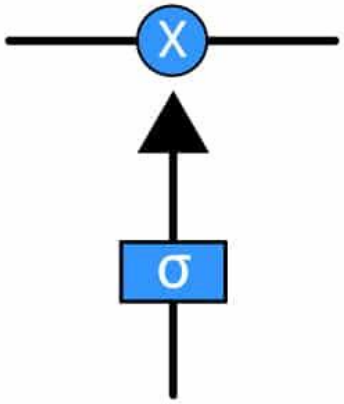


Figure 3. 3 Cell state in LSTM and is regulated by gates sources:(m.tech in AI and ML,2023)

The sigmoid layer gives out numbers between zero and one, where zero means ‘nothing should be let through’, and one means ‘everything should be let through’.

To normalise data, we employ a sigmoid function, and we calculate the error rate using its derivative. We will keep trying and modifying till our error rate is sufficiently low. Apply the

sigmoid function f(x)=1/(1+e-x) to get the study's output value, where x is the total amount of input. The sigmoid function's derivative:

Let us say sigmoid function is y=1/1+e-x then the derivative of the function is here:

dy/dx=(-1).(1+e-x) -2 .(-1).e-x = e-x /(1+e-x) 2 = 1+e-x /(1+e-x) 2 – 1/(1+e-x) 2 = 1/(1+e-x) – 1/(1+e-x) 2 , above sigmoid function assigned to y, means y =1/(1+e-x), so dy/dx= y+y2

= y(1-y)

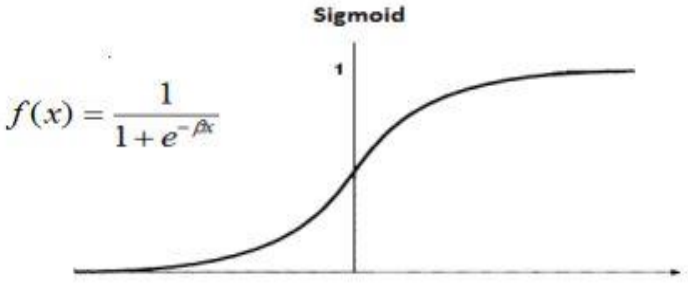


Figure 3. 4 Sigmoid function

3.6.3. Bidirectional recurrent neural networks(BRNN)

Data sequences are processed by a bidirectional recurrent neural network (BRNN), which has forward and backward layers of hidden nodes. The forward layer predicts the next output in a manner akin to that of the RNN, which saves the prior input in the hidden state. In the meantime, the backward layer updates the current hidden state by using the future hidden state as well as the current input. By incorporating both levels, the BRNN may better anticipate outcomes by taking historical and prospective situations into account

3.6.4. Gated recurrent units (GRU)

Selective memory retention is made possible using an RNN called a gated recurrent unit (GRU). The hidden layer of the model, which has the ability to add and remove information from memory, receives an update and forgets the gate

3.6.5. Classification Output and performance evaluation

The evaluation measures for single-label are usually different than for multi-label. In single label classification we use simple metrics such as precision, recall, accuracy, etc, In multi label classification, the prediction may be right, wrong or partially right, because, in the case documents belong to two or more classes, the prediction can hit them all (classes), any of them or just a few of them. Several measures have been proposed to evaluate the multi-label classifiers (Nayak, 2014). They are divided into example-based and label-based evaluation measures [Nayak, 2014]. The first are determined by averaging the discrepancies between the real and expected labels over all test cases. Subsequently, the review procedure is broken down into individual label evaluations, and the average of all labels is determined. Two example-based evaluation metrics are Hamming Loss and Classification Accuracy. Label-based evaluation metrics that are binary in nature, like recall, accuracy, and precision, can be employed.

These metrics are often averaged using two different methods: macro and micro. For the class-label, let tpj, fpj, and fnj stand for true-positives, false-positives, and false-negatives. We employed a label-based evaluation tool in this investigation. Macro-precision: where the numbers tpj and fpj represent the true positives and false positives for the label λ j, which is thought of as a binary class.

Prototype

```
11/11 [==============================] - 5s 435ms/step - loss: 0.2892 - acc: 0.9572 - val_loss: 0.2135 - val_acc: 0.9489
Epoch 13/20
11/11 [==============================] - 4s 392ms/step - loss: 0.1688 - acc: 0.9772 - val_loss: 0.1768 - val_acc: 0.9659
Epoch 14/20
11/11 [==============================] - 6s 512ms/step - loss: 0.1388 - acc: 0.9843 - val_loss: 0.1782 - val_acc: 0.9659
Epoch 15/20
11/11 [==============================] - 6s 535ms/step - loss: 0.1161 - acc: 0.9843 - val_loss: 0.0964 - val_acc: 0.9886
Epoch 16/20
11/11 [==============================] - 5s 417ms/step - loss: 0.0936 - acc: 0.9914 - val_loss: 0.0731 - val_acc: 0.9886
Epoch 17/20
11/11 [==============================] - 4s 361ms/step - loss: 0.0759 - acc: 0.9900 - val_loss: 0.0769 - val_acc: 0.9659
Epoch 18/20
11/11 [==============================] - 6s 519ms/step - loss: 0.0652 - acc: 0.9886 - val_loss: 0.0791 - val_acc: 0.9659
Epoch 19/20
11/11 [==============================] - 5s 413ms/step - loss: 0.0691 - acc: 0.9872 - val_loss: 0.0528 - val_acc: 0.9886
Epoch 20/20
11/11 [==============================] - 5s 471ms/step - loss: 0.0771 - acc: 0.9843 - val_loss: 0.0378 - val_acc: 0.9886
```

```
[138]: accr = model.evaluate(X_test,y_test)
       print('Test set\n  Loss: {:0.3f}\n  Accuracy: {:0.3f}'.format(accr[0],accr[1]))
```

```
10/10 [==============================] - 0s 34ms/step - loss: 0.0685 - acc: 0.9727
Test set
  Loss: 0.068
  Accuracy: 0.973
```

Algorithm 3. 4 Proto Type Model Training and Validation Accuracy

CHAPTER FOUR

4. EXPERIMENT AND EVALUATION

4.1.Introduction

First off the entire work's methodologies, experiments, and evaluation techniques are covered in full in this chapter. In order to conduct experiments, the instruments and programming languages are explained, along with the evaluation metrics and procedures and the evaluation's outcomes in relation to the metrics or methodologies employed for evaluation and the data used for experimentation. The experimental conditions such as the computer's capacity, operating system, RAM, processor, etc, are explained, and the protocols and outcomes are recorded. As stated in the literature review and associated work, the performance and actual results of this study are also contrasted with those of other researchers in the field. The comparison can be explained using the accuracy values, The comparison may be described through the values of the accuracy, the amount of data used and the approaches they used concerning their final result of performance measure.

4.2. Experimental procedure

This portion of the chapter provides a detailed description, along with some examples and graphical or tabular representations, of the actions and techniques used to develop and evaluate the Afan Oromo text news classification utilizing a deep learning methodology.

4.2.1. Dataset collection

To conduct the experimentation and produce a better outcome and a well-performing model for the categorization of Afan Oromo texts, a substantial amount of data is needed. By their very nature, deep learning methods work best with large datasets and demand a lot of data. Text news classification documents in Afan Oromo were gathered from several sources, including the online news of Oromia Broadcasting Network (OBN) and Fana Broadcasting (FBC) and various online newspapers and magazines. For the purpose of this study, eight categories are taken into consideration.

The eight classes have been selected based on random sampling, which are Balaa, Barnoota, Technology, Fayyaa, Siyaasa, Qonnaa, Ispoortii and Biizinasii. The number of news items in each category and the total number of news items are shown in Table 4.1.

Table 4. 2 the number of news items in each category and the total number of news items

No	Class Name	Number Of News
1	Balaa (Accident)	1230
2	Barnoota (Education)	950
3	Teknoolojii (Technology)	1100
4	Fayyaa (Health)	950
5	Siyaasa (Politics)	1011
6	Qonnaa (Agriculture)	1250
7	Spoortii (Sport)	1350
8	Biizinasii (Business)	1200
	Total	9041

4.2.2. Tools and programming language

A. Tools

There are different development tools of deep learning for natural language processing that is open source and easily adapted. Thus, the anaconda is a free and open-source distribution of R and python programming languages. It is popular because it brings many of the tools used in data science and deep learning with just one install, so it's great for having a short and simple setup. To conduct this work Keras, Tensor Flow, and Genism are used as a tool. Keras is a high-level neural network, written in Python and capable of running on top of Tensor Flow and it enables easy and fast prototyping, supports CNN and RNN and it runs seamlessly on CPU and GPU. TensorFlow is an end to end open-source platform for deep learning and it builds and trains deep learning models by using the high-level Keras API. Keras and TensorFlow are used for preprocessing and the construction of the model for training and validation. Genism is a library for document similarity analysis and it is a production-ready tool that you can trust with several crucial components of NLP applications. The categorization component of this work is done by the genism

46 library for developing the word embedding that analysis the similarity of words with in-text document. Word2vec is developed for categorization support for the model.

B. Programing Language

Python is an object-oriented, an interpreter, and high-level programming language with dynamic text classification. Python programming is used for the experimentation of the Afaan Oromo Text News classification using deep learning approaches using Jupiter notebook editor.

C. Experimental setup

Deep learning experimentations require high processor and GPU supported computing. In this study, we used a computer with a memory capacity of 8 GB RAM, 2.41 GHz processor, and 64-bit Windows 10 operating system.

4.2.3. Text data cleaning or preprocessing

After data are collected and programming languages and tools are selected the next step for the experimentation of Afan Oromo news text classification using deep learning approaches is text data preprocessing. The preprocessing tasks are adopted as it is described in chapter three, the collected data are processed to make it ready for training. During preprocessing 363 stop words (words which don't have any contribution for the context constriction of documents) are removed and words that have different representations but having the same meaning are normalized into canonical form. Deep learning algorithms and architectures use numerical representation as an input for the model to be trained. Thus, the text data are tokenized and converted into a numerical representation which is called tokenization. Those tokens are padded into the uniform representation of matric since the input of the deep learning model is represented in uniform matric. During padding, the maximum number of words from the entire document is selected and the other documents are padded to the same dimension with it. The variation of the number of words in each text file affects the sparseness of the padding structure. While padding to make the small number of word documents uniform with the maximum on, the zero (0) padded is added. But the vector may be sparser, the larger the zero-padded element of the vector may lead the model less performed. Sparse vectors must be converted to dense vector or the sparseness of the data must be reduced by adjusting the pad size during padding. The numerically represented and padded tokens

converted into a vector representation, thus the representation of tokens into vectors is embedding components of the model.

4.2.4. Word Embedding

The embedding component is the representation of the words to feature learning in which each word or phrase from the vocabulary is mapped to 300 dimensions of vector. It improves the performance of text classification based on the deep neural network by considering the catagorization of the document. word2vec is one of the embedding representations in the Text classification and neural network field of study.

Table 4. 3 parameters needed for training word2vec

No	Parameters to train word2vec	Values Of parameter	Descriptions of the parameter	Remark
1	Windows	10	Maximum skip length window between words	
2	Embedding dimension	130	Size of word vector in which each word is represented	
3	Learning rate	0.05	Learning rate for SGD estimation	
4	Number of epochs	50	Number of training epochs	
5	Number of threads	12	Number of training thread	
6	SG	1	Choosing Skip-gram model	
7	Iteration	20	The training iteration	
8	Minimum frequency	3	This may discard words appear minimum frequency in the document	
9	Embedding directory	“E/MyWordVector”	The directory in which the word2vec is saved	

4.2.5. Model construction using algorithms (CNN, LSTM, GRU and BIRNN)

After the data are preprocessed and the embedding component is built the next component is the algorithms (CNN, LSTM, GRU, and BIRNN) layer that accepts the vector representation of the data as an input to learn features. The learning capability of those algorithms are depending on the performance of the embedding layer, the amount of input data, and the number of neurons used for training. The algorithms (CNN, LSTM, GRU, and BIRNN) layer of the model contains the number of neurons, the dropout, and the recurrent-dropout as a parameter and the model may contain one or multiple algorithms layer that depends on the problem that we solve and the data that we have for training.

Most of the time single layer deep learning Algorithms is used for simple and linear problems, whereas, multiple layer Algorithms is used for nonlinear and convex problems. We are doing our experimentation both in single and multiple layer deep learning algorithms and finally, we select the one that performs well. The dropout component has a value between 0 and 1, which is a fraction of the units to drop for the linear transformation of the input data; we use 0.2 or 20% dropout for the experimentation. Whereas the recurrent-dropout is a fraction of values between 0 and 1 to drop for the linear transformation of the recurrent state; we use 0.2 or 20% recurrent dropout for the experimentation.

Dense layer

The dense layer is a fully connected layer that accepts the deep representation of the input data and transforms into the final output class using an activation function. The dense layer of the model contains two basic parameters, the number of class as a dimension of the output vector and the activation function. We use the SoftMax activation function to generate the final output that lays values in the range of 0 -to- 1. We have also eight (8) class such as, Accident, politics, business, Sport, Technology, Health, Education and Agriculture, in which the output layer maps the result in to 8 dimensions of vector. Therefore, it is used as the final layer in the text classification models.

Compiling the model

After the output layers are defined the model needs to be compiled. To compile the model, the loss function, the optimizer, and the metric that validate the model need to be specified. The 50 optimizer controls the learning rate. The learning rate determines the computational of the optimal weights for the model to be calculated. We used the sparse categorical cross-entropy as a loss function since we have 8 classes to be predicted after compiling the model. The metric is used for plotting the result after the history of the model is saved and we used accuracy as a metric for our model. The metrics contain training loss, training accuracy, validation loss, validation accuracy, and testing accuracy.

Summarizing the model

After the model is compiled, it needs to be summarized to visualize and generate each layer of the model with their corresponding output shape and number of parameters. It generates the total parameter in the model by summarizing from each layer and identifies the number of the trainable and non-trainable parameter from the total parameter.

```
In [125]: print(model.summary())
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 130, 128)          1024000

 spatial_dropout1d_1 (Spati  (None, 130, 128)          0
 alDropout1D)

 lstm_1 (LSTM)               (None, 64)                49408

 dense_1 (Dense)             (None, 8)                 520

=================================================================
Total params: 1073928 (4.10 MB)
Trainable params: 1073928 (4.10 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
None
```

Algorithm 4. 1 Summary of the model with S *1*

Model fitting

After the model is summarized it should be fit with the data that are prepared as a training set with their corresponding label or category. The fitting model component includes the training set, the label or the annotation of training data (the class it belongs), the number of epochs (the number of times the model goes over the entire dataset), and the batch size (how may training examples are processed at a time), as well as the validation split (how much of the training data is taken each epoch to measure how well the model generalizes on unseen data), need to be specified. We used validation split as 20% of our training set and 20 and 30 alternative numbers of epochs with 32 batch sizes.

```
In [126]: history = model.fit(X_train, y_train, epochs=epochs, batch_size=batch_size,validation_split=0.2,callbacks=[EarlyStopping(monit
```

Algorithm 4. 2 Model Fit

Evaluating the Model

After the model is fitted with the data, its performance should up to be evaluated with the training, testing, and validation dataset separately using the metric that is defined in the model fitting component. Those metrics are accuracy and loss; training accuracy, training loss, validation accuracy, validation loss, and testing accuracy. Thus, the higher the accuracy and the lower the loss indicates the better the models performs. Using the loss and the accuracy values of the model we adjust the parameters which are the batch size, the embedding dimensions, and the number of neurons in the embedding, number of algorithms (CNN, LSTM, GRU and BIRNN) layers, and the value of dropout of the models to get a better-performed models.

```
In [138]: accr = model.evaluate(X_test,y_test)
          print('Test set\n  Loss: {:0.3f}\n  Accuracy: {:0.3f}'.format(accr[0],accr[1]))

          10/10 [==============================] - 0s 34ms/step - loss: 0.0685 - acc: 0.9727
          Test set
            Loss: 0.068
            Accuracy: 0.973
```

Algorithm 4. 3 Model Evaluation

Testing and Classifications

The final and most important step is to classify the data using the model we trained after it has been constructed, fitted with data, and assessed using assessment metrics. The model must be loaded and saved in order to be classified. As a result, the model is saved by employing the save technique. Model () with h5 extensions, and then load utilizing models to conduct testing.load_model () from the Keras module. Two primary methods are used to assess the model's performance: the test data and external data, to which the model's class or domain is unknown.

Testing accuracy is thus produced when we employ the testing data. Just 0% of the testing data is categorized correctly. All preprocessing processes completed for the training and testing data must be passed when testing the model with randomly selected unknown class or category data. Following preprocessing, the model matches the content of the provided document to each class, calculating the likelihood that the given document belongs to each class. Ultimately, the class with the highest probability is projected to be the one to which the given document belongs.

The probability of belonging to a particular class is stored based on its index, and the prediction process is carried out by storing the labels or classes in a single one-dimensional array that has the names of the classes in the appropriate index along with the index number appended for each class during labeling.

Table 4. 4 Test result using external data

No	Doc-No	Probability of being in class in %								Actual class	Predicted class
		Politics	Accident	Health	Sport	Agriculture	Business	Technology	Education		
1	D1	0.07	1.6	6.5	99.3	0.039	1.08	5.5	0.08	Health	Sport
2	D2	0.03	0.038	0.07	0.74	0.0036	99.9	8.8	1.4	Business	Business
3	D3	1.3	0.10	79.38	20.4	0.26	0.175	9.6	0.25	Health	Health
4	D4	10.5	1.4	52.6	32.15	0.5	6.4	89.7	0.89	Sport	Sport
5	D5	97.2	0.15	22.1	0.47	0.7	3.39	1.08	1.6	Business	Business
6	D6	98.6	0.07	0.47	4.57	3.1	9.92	0.41	3.2	Sport	Sport
7	D7	99.3	0.0019	0.0017	0.028	1.73	97.6	2.10	1.2	Sport	Politics
8	D8	23.1	0.0019	0.19	0.12	1.63	3.3	0.4	0.5	Sport	Sport

4.3.Evaluation

Various assessment metrics are used to assess the system's or the model's performance. One of the key evaluation metrics for various deep learning applications is accuracy. It is the extent to which a system, models, or specification's output complies with the right value or a standard. The algorithms (CNN, LSTM, GRU and BIRNN**)** structure is used to assess our algorithms text categorization models. The preprocessing component processes datasets for training, testing, and validation. As a result, in addition to the training, testing, and validation datasets, the model is trained using those datasets, and its accuracy and performance are assessed. The assessment of the algorithms (CNN, LSTM, GRU and BIRNN) structure's nature is based on the quantity of neurons, the number of epochs, as well as the quantity of built-in hidden layers with constant batch size. Therefore, for single-layer and three-layer algorithms (CNN, LSTM, GRU and BIRNN) with arbitrary 50 and 100 epoch, we employ 50 and 100 neurons, respectively.

A total of 9041training, 20% testing, and 10% validation datasets are used in the assessment procedure. As a result, the models learns 32 training data points at a time and computes the training accuracy. The models evaluates the training data at every learning rate in which the data is learned once at a time (batch size = 32). However, validation accuracy and loss assess how well the models trained during each period.

Table 4. 5 Evaluation result for training, testing, and validation dataset

NO	No of layers	No of neurons	No of epochs	Validation accuracy	Training loss	Validation loss
1	1	50	50	97.30	0.68	0.42
2	1	100	50	86.53	0.32	0.45
3	3	50	50	76.3	0.54	0.62
4	3	100	50	76.3	0.65	0.70
5	1	50	100	84.4	0.23	0.49
6	1	100	100	85.71	0.15	0.47
7	3	50	100	82.3	0.42	0.55
8	3	50	100	83.4	0.45	0.42

Table 5.7, demonstrates the, validation accuracy, validation loss, and testing accuracy of the model trained on epoch (50,100), number of neurons (50,100) separately. For each experimentation the performance of the model is differ based on those parameters, from those the model with 100 number of neurons, 100 epochs on a single layer performs better which is testing accuracy LSTM performs exceptionally well, scoring 98.71% accuracy and 98.71% precision, while CNN, BRNN and GRU score 94%, 94%, 96.40%, 97.48%, and 94.42%,92.4% respectively.

Discussion

Text documents from 9041 Afaan Oromo News accomplish the suggested models while comparing those deep learning algorithms (CNN, LSTM, GRU and BIRNN). Fifteen percent (10%) are used for validation, seventy percent (70%) are for training, and fifteen percent (20%) are used for testing and model training. The quantity of training data affects the model's performance. Better results are obtained with a large amount of training data, while the model's performance decreases with a limited number of data. All of the training data cannot be supplied to the models at once during training; to get around this issue, batch size is defined to divide the data into smaller size divisions, which are then passed to the model one at a time, with the weight of the neural network passing to the next node is updated at the end of each step.

The other factor that affects the performance of the models is the number of epochs and neuron, the number of epochs is the number of times the models trained using the entire dataset and at each epoch, the entire dataset is passed forward and backward through the neural network once. Learning repeatedly increases the capability of the models to learn the feature and content of the data. But, training a models with a large number of times may lead to over fitting, the over fitting problems need to be optimized to the normal learning state. Thus, dropout optimizes the over fitting through deactivating arbitrarily some amount on nodes from the given node or neurons.

The number of neurons affects the performance of the models. It is more important than the number of layers. Table 4.3 clearly shows its effect on the performance of the models, using less neuron on the layer may lead to loss some basic information's because its learning capability determined on the neuron it contains. There is a great change of accuracy and loss when using 50 and 100 neurons for training, for fewer neurons there is an increase in the loss (the measure of how well the models generalized to unseen records) and a decrease the accuracy for both the training, testing and validation phase. When we use 100 neurons maximum of LSTM performs scoring 98.71% accuracy while CNN, BRNN and GRU score 94%, 96.40%, and 94.42% and testing accuracy using 100 epochs LSTM 98.71% CNN, BRNN and GRU score 94%, 97.48%, and 92.4% are the result of the models after training.

The number of layers affect the performance of the model. Multiple layer network enables the network to more eager to recognize the certain aspect of the input data. Most of the time the number of layers in deep learning model determined by the nature of the problem; multiple layer networks are needed for more complex, non-linear, and convex problems whereas single layer networks are used for linear and simple problems. In our experimentation, we use a three-layer, eight layers and single-layer those algorithms network to analyze the change in it.

As shown in Table 4.3 there is a change in the testing accuracy, validation accuracy, and validation loss. We have got better performance when we use a single-layer network since the problem that we solve is not too complex and convex. It learns the content of the data and translates it to a single output; it has many to one relationship having a set of input documents, a sequence of words and the corresponding output is a single class or label. Out of the four deep learning methods, LSTM performs best with a testing accuracy of 98.71 when using a single-layer network, and 97.30 when using a three-layer network.

Generally speaking, our models are built using LSTM, an enhanced RNN variant that excels at learning long-term dependencies, particularly in sequence prediction tasks. With the exception of individual data points like words, LSTM can comprehend the complete sequence of data because it includes feedback links.

CHAPTER FIVE

5. Conclusion and Recommendation

5.1. Conclusion

The goal of this research project was to investigate the methods for classifying Afan Oromo news texts using deep learning. The fundamental components of the models for classifying text documents are outlined in detail throughout this investigation. The document preprocessing (stop-word removal, normalization, tokenization, and padding), model building (word embedding, deep network, output determination, and model compiling), summarization and fitting, evaluation and testing, and evaluation and testing modules comprise the designing component. These are the fundamental modules in designing and executing the entire work.

Less feature engineering is needed when using deep learning in the neural network domain, where the word embedding component is neural-based feature extraction from unstructured text data in vector representation with a certain level of similarity between tokens across the dataset. Therefore, the word classification is computed using the document's vector representation.

Word2vec is a tool for vector word representation that produces categorization similarity between words. We employed word2vec, a Skip-Gram based representation of the target word's categorization from the provided sentences utilizing the terms in its surrounding context. The deep network, which transfers the features of the data to the output layer in order to produce the expected output of the model, is the main component of this study. We trained and constructed the Afan Oromo news text classification models using an enhanced algorithms (CNN, LSTM, GRU and BIRNN). A sequential model called LSTM addresses some of the drawbacks of CNN,GRU and BIRNN is used to process time sequence data. Its memory, known as the cell state, lessens the issue of long-term dependency and vanishing gradient. Since LSTM is a sequential process, the cell state is utilized to retain the contents of the previous hidden state and the supplied processed input data to determine the next hidden state. It has a gate that is used to decide which data should be retained, which should be ignored, and which should receive extra consideration when training a cell state.

Generally speaking, we employ the LSTM technique to construct Afaan Oromo news text categorization after training and testing all algorithms (CNN, LSTM, GRU, and BIRNN) using Python tools. This is dependent on the results I obtained. Unstructured Afaan Oromo news text documents are used in the model's development. The total outcome demonstrates that as the number of neurons and epochs rises, the model performs better overall. Accuracy and loss are used to gauge the model's performance during the training, validation, and testing phases. According to the final result, the LSTM models reach testing accuracy of 98.71 and validation accuracy of 97.30, whereas other models achieve less than LSTM models validation accuracy utilizing.

Contribution to the study

The study's primary contributions are summed up as:

- ✓ Using neural word embedding technology, a general model for Afaan Oromo news text categorization is developed.
- ✓ The thesis advances the use of word embedding in text data analysis for news document feature extraction.
- ✓ By creating a general word2vec that is utilized for various NLP applications, the thesis makes a contribution.

5.3. Future work and recommendation

In this study, we have made an effort to investigate how to construct an Afaan Oromo news text classification model using a deep learning methodology. Nonetheless, several initiatives may be taken in the field of text categorization to improve results through the use of various feature extraction and learning techniques. The primary recommendations we would want to make for further work that needs to be done are:

- ✓ To train the word2vec model and obtain a satisfactory classification representation of a text document, rich or substantial amounts of data are required.
- ✓ Utilizing LSTM hybridization in conjunction with other deep learning architectures to reap the benefits of alternative models.
- ✓ The training rate of the model is influenced by the machine's performance, hence it is advised to train the model on a high-processing computer with a lot of neurons and epochs.
- ✓ Lemmatization can be used to decrease the dimensionality of the data.

Reference

[1] P. Gavras, “Ratings game,” *Financ. Dev.*, vol. 49, no. 1, pp. 34–37, 2012.

[2] C. X. Guo *et al.*, “Large-scale cooperative 3D visual-inertial mapping in a Manhattan world,” *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2016-June, pp. 1071–1078, 2016, doi: 10.1109/ICRA.2016.7487238.

[3] Z. Xinhua, “Building Maximum Entropy Text Classifier Using Semi-supervised Learning Supervisor,” no. October, 2004.

[4] M. Rashighi and J. E. Harris, “乳鼠心肌提取 HHS Public Access,” *Physiol. Behav.*, vol. 176, no. 3, pp. 139–148, 2017, doi: 10.1053/j.gastro.2016.08.014.CagY.

[5] D. Howard and B. Mark, “Neural Network Toolbox User’s Guide,” *The MathWorks*, p. 846, 2004.

[6] A. B. Mohammed, “Decision Tree , Naïve Bayes and Support Vector Machine Applying on Social Media Usage in NYC / Comparative Analysis Decision Tree , Naïve Bayes and Support Vector Machine Applying on Social Media Usage in NYC / Comparative Analysis,” *Tikrit J. Pure Sci.*, vol. 22, no. 9, pp. 94–99, 2017.

[7] W. Kelemework, “Automatic Amharic text news classification: Aneural networks approach,” *J. Sci. Technol*, vol. 6, no. 2, pp. 127–137, 2013.

[8] ABERA DIRIBA GEMECHU, “School of Graduate Studies Informatics Faculty Departement of Information Science Automatic Classification of Afaan Oromo News Text : the Case of Radio Fana By Automatic Classification of Afaan Oromo News Text :,” 2009.

[9] Mu’adzah, T. L. Ahmad, and A. N. Kusumawati, “L Iterature R Eview,” *J. Bisnis Digit. dan Sist. Inf.*, vol. 1, no. 1, pp. 1–11, 2020.

[10] A. Gasparetto, M. Marcuzzo, A. Zangari, and A. Albarelli, “ Survey on Text Classification Algorithms: From Text to Predictions,” *Inf.*, vol. 13, no. 2, pp. 1–39, 2022, doi: 10.3390/info13020083.

[11] Q. Li *et al.*, “A Survey on Text Classification: From Shallow to Deep Learning,” *ACM Trans. Intell. Syst. Technol.*, vol. 37, no. 4, 2020, [Online]. Available: http://arxiv.org/abs/2008.00364

[12] S. Wasi and S. Yousuf, “Parallel comparison of sentiment analysis techniques A survey,” *Ijcse.Net*, vol. 7, no. 02, pp. 63–68, 2018, [Online]. Available: http://www.ijcse.net/docs/IJCSE18-07-02-022.pdf

[13] G. R. Ashisha, S. T. George, K. M. Sagayam, and S. Pramanik, “Analysis of Diabetes disease using Machine Learning Techniques: A Review,” pp. 1–17, 2022, [Online]. Available: https://doi.org/10.21203/rs.3.rs-1572946/v1

[14] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, “Deep Learning Based Text Classification: A Comprehensive Review,” vol. 1, no. 1, pp. 1–43, 2020, [Online]. Available: http://arxiv.org/abs/2004.03705

[15] W. H. Bangyal *et al.*, “Detection of Fake News Text Classification on COVID-19 Using Deep Learning Approaches,” *Comput. Math. Methods Med.*, vol. 2021, 2021, doi: 10.1155/2021/5514220.

[16] T. Joachims, “Text categorization with support vector machines: Learning with many relevant features,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 1398, pp. 137–142, 1998, doi: 10.1007/s13928716.

[17] H. Fang, C. Lu, F. Hong, W. Jiang, and T. Wang, “Convolutional Neural Network for Heartbeat Classification,” *2021 15th IEEE Int. Conf. Electron. Meas. Instruments, ICEMI 2021*, pp. 253–258, 2021, doi: 10.1109/ICEMI52946.2021.9679581.

[18] Y. Zhang, S. Garg, Y. Meng, X. Chen, and J. Han, “MotifClass: Weakly supervised text classification with higher-order metadata information,” *WSDM 2022 - Proc. 15th ACM Int. Conf. Web Search Data Min.*, pp. 1357–1367, 2022, doi: 10.1145/3488560.3498384.

[19] F. Sebastiani, “Text categorization,” *Encycl. Database Technol. Appl.*, no. Ml, pp. 683–687, 2005, doi: 10.4018/978-1-59140-560-3.ch112.

[20] S.-J. Lee and Y. Jiang, “Multilabel Text Categorization Based on Fuzzy Relevance Clustering,” *Fuzzy Syst. IEEE Trans.*, vol. 22, pp. 1457–1471, Dec. 2014, doi:

10.1109/TFUZZ.2013.2294355.

[21] K. Chen, R. Li, Y. Dou, Z. Liang, and Q. Lv, *Ranking Support Vector Machine with Kernel Approximation*, vol. 2017. 2017.

[22] M. Zhang and Z. Zhou, “A Review on Multi-Label Learning Algorithms,” pp. 1–43.

[23] D. Endalie and G. Haile, “Automated Amharic News Categorization Using Deep Learning Models,” *Comput. Intell. Neurosci.*, vol. 2021, 2021, doi: 10.1155/2021/3774607.

[24] O. M. Singh, “Nepali Multi-Class Text Classification,” 2018.

[25] P. Liu, X. Qiu, and H. Xuanjing, “Recurrent neural network for text classification with multi-task learning,” *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 2016-Janua, pp. 2873–2879, 2016.

[26] D. Siganos, “Contents :,” vol. 4, pp. 1–13, 2011.

[27] T. Edition, *Neural Networks and*.

[28] E. Kula-eversole, E. Nagoshi, Y. Shang, J. Rodriguez, R. Allada, and M. Rosbash, “Surprising gene expression patterns within and between PDF-containing circadian neurons in Drosophila,” vol. 107, no. 30, pp. 13497–13502, 2010, doi: 10.1073/pnas.1002081107.

[29] L. Tadesa, G. Mamo, and A. Asseffa, *Afaan Oromo News Text Classification Using A Deep Learning Approach*. 2022.

[30] G. G. Eggi, “Afaan Oromo Text Retrieval System,” *MSc, Addis Ababa Univ.*, no. June, pp. 23–25, 2012.

[31] L. Architecture, “Introduction to Long Short Term Memory (LSTM)”.

Appendix

Appendix one: List Afaan Oromo Stop Word

"himaniiru", "ibsaniiru", "jedhaniiru", "jedhan","Aanee","aga", "achi", "achuma", "adda", "addaatti", "afoo", "agarsiisoo", "akkasumas", "al ala", "alatti","alla", "akka", "akkam", "akkamii", "akkamiitu", "akkana", "akkataa", "akkataan", "akkas", "akkasitti", "akkuma", "akksumas","amma", "ammaa", "ammo", "ala", "alatti", "alla", "an", "ana", "anee", "ani", "asham", "asitti", "attam", "ati", "awu", "bira", "biraa", "biratti", "biro", "biroon", "biros", "biyyam", "booda", "booddee","bukkee", "cinaa", "abalatees","dhaa", "dhaan", "dudduuba","dugda", "duuba", "dura", "duraa", "eega", "eegana", "eegasii", "ennaa", "eenuu", "eenyu", "eennu", "eenyufaa", "eenyuun", "eennuun", "eenyuuf", "eennuuf", "eenyufaa", "eenyufaadhaan","eenyufaaf", "eenyufaarraa", "eenyufatti", "eenyurra", "eennurraa", "eenyurraa", "eennurraa", "eenyurratti", "eennuratti", "eenyuuf", "eenyuree", "eennuree", "eenyutti", "eennutti", "eenyum", "eessa", "ega", "eessatti", "eessarraa", "eessaaf", "eessaan", "erga", "ergii", "f", "faa", "faallaa", "fagaatee", "faati", "faatidhaa", "faatiidhaa", "fakkaatanitti","fakkaatti", "fakkaatu", "fakkaattu", "fakkeenyaa", "fakkeenyaaf", "fedhetuu", "fi", "fk" "fkf", "fkn" "fknf", "fuuldura", "fundura", "fullee", "fuullee", "gaa", "gad", "gadi", "gaditti", "gahaa", "gajjallaa", "gala", "galan", "galani", "galuu", "gama", "gamma", "gar", "gara", "garam", "garamiin", "garamitti", "garana", "garas", "gararraa", "gararree","garii", "garjalee", "garuu", "giddu", "gidduu", "godhe", "gootee", "gubbaa", "haa", "haga", "hagam", "hamma", "hammam", "hammamiif", "hammamiin", "hammamtu","hanga","henna", "himantu", "hin", "hinjira", "hinjiru", "hinjirtu", "hinjirtan", "hinjiru", "hoggaa", "hoo", "hunda", "hunduma", "idda", "iddoo", "if", "illee", "immoo",

"inni", "irra", "irraa", "irraan", "irratti", "irrattii", "isa", "isaa", "isaaf", "isaan", "isaanii", "isaaniif", "isaaniitiin","isaaniis", "isaanniis", "isaaniirratti", "isarraa", "isaanirraa", "isatti", "ishe", "ishee", "isee", "ishii", "ishiif", "isiidhaa", "ishiidhaa", "isii", "isiin", "ishiifaa", "ishiin", "isin""isinii", "isiniif", "isiniin", "isiniis", "isinirraa", "ittaanee", "itti", "ittillee", "ittumallee", "ittiin", "ituu", "ituullee","jala", "jara", "jechaan", "jechoota", "jechuu", "jechuun", "jedhu", "jedhus", "jira", "jiraadha",

"jiraadhe", "jiraanne", "jiraate", "jiraattee", "jiraatta", "jiraatti", "jiraanna", "jiraatan","jiraattan", "jiran", "jiranii", "jiranirra", "jiranu", "jirre", "jirta", "jirti", "jirra", "jirtan", "jirtanu", "jirtu", "jiru", "kaa", "ka'e", "ka'en", "kam", "kamfaa", "kami", "kamidha", "kamifaadha", "kamiin", "kamiinu", "kamiinuu", "kamiif", "kaminiyyuu", "kamirraa", "kamitti", "kamttuu", "kamitu","kamuma", "kamuu", "kamirrattu", "kamiyyuu", "kan", "kana", "kanaa", "kanaan", "kanaaf", "kanaafuu", "kanaafi", "kanaafiis", "kanaatti", "kanarra", "kanarratti","kanisaanii", "kanishii", "kankee", "kankeeny", "kankoo", "kanneen", "kanneeni", "kanarraa", "karaa", "kee", "keenna", "keenya", "keessa", "keessan", "keessatti","keeysa", "kiyya", "kkf", "koo", "kun", "qunnama", "kuni", "kunis", "kunoo", "kunneen", "kanneeni", "laata", "lafa", "lama", "maal", "maalfaa", "maali", "maalif","maaliin", "maalirraa", "maalirratti", "maaliree", "maalittuu", "maaltu", "maaluma", "maalumaaf", "malee", "manna", "maqaa", "mee", "meeqa", "meeqaan", "meeqaaf",

"meeqarraa", "meeqatti", "meeqatu", "meerre", "meerreree", "miti", "moo", "na", "naa", "naaf", "naan", "nan", "naannoo", "natti", "nu", "nuhi", "nu'i","nurraa", "nuu", "nuuf", "nuun", "nuti", "nuyi", "obboo", "odoo", "of", "ofii", "firraa", "ofiif", "ofiin", "oftiin", "ofitti", "ofuma", "ofumaa", "ofumaaf"

Appendix Two: Training result with 20 neuron and 20 epochs

```
None

In [126… history = model.fit(X_train, y_train, epochs=epochs, batch_size=batch_size,validation_sp

Epoch 1/20
11/11 [==============================] - 13s 554ms/step - loss: 2.0437 - acc: 0.2924 - v
al_loss: 1.9895 - val_acc: 0.5114
Epoch 2/20
11/11 [==============================] - 5s 401ms/step - loss: 1.8175 - acc: 0.4180 - va
l_loss: 1.7236 - val_acc: 0.2841
Epoch 3/20
11/11 [==============================] - 5s 416ms/step - loss: 1.5609 - acc: 0.3495 - va
l_loss: 1.6149 - val_acc: 0.2841
Epoch 4/20
11/11 [==============================] - 5s 443ms/step - loss: 1.4502 - acc: 0.4280 - va
l_loss: 1.4525 - val_acc: 0.5398
Epoch 5/20
11/11 [==============================] - 5s 440ms/step - loss: 1.3690 - acc: 0.5649 - va
l_loss: 1.2795 - val_acc: 0.7159
Epoch 6/20
11/11 [==============================] - 5s 442ms/step - loss: 1.1856 - acc: 0.7004 - va
l_loss: 0.9351 - val_acc: 0.8409
Epoch 7/20
11/11 [==============================] - 4s 394ms/step - loss: 0.8501 - acc: 0.8060 - va
l_loss: 0.6510 - val_acc: 0.8352
Epoch 8/20
11/11 [==============================] - 5s 500ms/step - loss: 0.5172 - acc: 0.8873 - va
l_loss: 0.5897 - val_acc: 0.7898
Epoch 9/20
11/11 [==============================] - 5s 420ms/step - loss: 0.3777 - acc: 0.9230 - va
l_loss: 0.3195 - val_acc: 0.9261
Epoch 10/20
11/11 [==============================] - 5s 491ms/step - loss: 0.3481 - acc: 0.9387 - va
l_loss: 0.2688 - val_acc: 0.9489
Epoch 11/20
11/11 [==============================] - 5s 415ms/step - loss: 0.2739 - acc: 0.9586 - va
l_loss: 0.2142 - val_acc: 0.9716
```

```
Epoch 14/20
11/11 [==============================] - 6s 512ms/step - loss: 0.1388 - acc: 0.9843 - va
l_loss: 0.1702 - val_acc: 0.9659
```

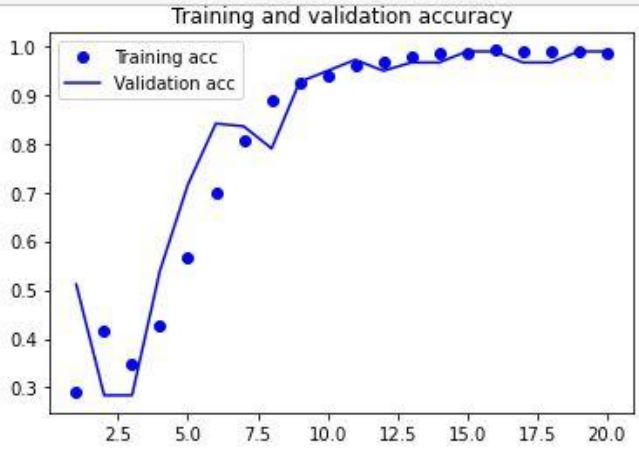
```
Epoch 15/20
11/11 [==============================] - 6s 535ms/step - loss: 0.1161 - acc: 0.9843 - va
l_loss: 0.0964 - val_acc: 0.9886
Epoch 16/20
11/11 [==============================] - 5s 417ms/step - loss: 0.0936 - acc: 0.9914 - va
l_loss: 0.0731 - val_acc: 0.9886
Epoch 17/20
11/11 [==============================] - 4s 361ms/step - loss: 0.0759 - acc: 0.9900 - va
l_loss: 0.0769 - val_acc: 0.9659
Epoch 18/20
11/11 [==============================] - 6s 519ms/step - loss: 0.0652 - acc: 0.9886 - va
l_loss: 0.0791 - val_acc: 0.9659
Epoch 19/20
11/11 [==============================] - 5s 413ms/step - loss: 0.0691 - acc: 0.9872 - va
l_loss: 0.0528 - val_acc: 0.9886
Epoch 20/20
11/11 [==============================] - 5s 471ms/step - loss: 0.0771 - acc: 0.9843 - va
l_loss: 0.0378 - val_acc: 0.9886
```

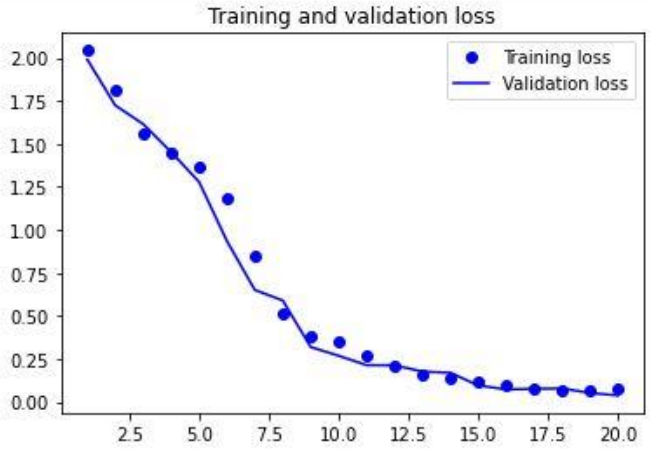
```
accr = model.evaluate(X_test,y_test)
print('Test set\n  Loss: {:0.3f}\n  Accuracy: {:0.3f}'.format(accr[0],accr[1]))
```

```
10/10 [==============================] - 0s 34ms/step - loss: 0.0685 - acc: 0.9727
Test set
  Loss: 0.068
  Accuracy: 0.973
```

In [128…

```
import matplotlib.pyplot as plt

acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(1, len(acc) + 1)

plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()

plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()

plt.show()
```





In [130…
```
txt = ["Qeenxeen caalmaan mul'achuuf utuu hintaanee waloon miidhaguuf yaadaa fi hubannoo
seq = tokenizer.texts_to_sequences(txt)
padded = pad_sequences(seq, maxlen=max_len)
pred = model.predict(padded)
labels = ['Technology','Business','Politics','Health','Sport','Accident','Agriculture','
print(pred, labels[np.argmax(pred)])
```

```
1/1 [==============================] - 0s 60ms/step
[[0.00716844 0.12632425 0.7320775  0.00847637 0.02937466 0.00700949
  0.06167845 0.02789081]] Politics
```