MULTI-LABEL AND SINGLE-LABEL TEXT CLASSIFICATION USING STANDARD MACHINE LEARNING ALGORITHMS AND PRE-TRAINED BERT TRANSFORMER

HUDA ALFIGI

JANUARY 2023

ÇANKAYA UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING

MASTER'S THESIS

INFORMATION TECHNOLOGIES

MULTI-LABEL AND SINGLE-LABEL TEXT CLASSIFICATION USING STANDARD MACHINE LEARNING ALGORITHMS AND PRE-TRAINED BERT TRANSFORMER

HUDA ALFIGI

JANUARY 2023

ABSTRACT

MULTI-LABEL AND SINGLE-LABEL TEXT CLASSIFICATION USING STANDARD MACHINE LEARNING ALGORITHMS AND PRE-TRAINED BERT TRANSFORMER

ALFIGI, Huda

Master of Science in Information Technologies

Supervisor: Doç. Dr. ABDUL KADIR GORUR

January 2023, 86 pages

Natural language processing (NLP) research has received a great deal of attention in recent times, because of the increasing availability of digital documents and the resulting need to access them in various ways. The explosion of digital text data demonstrates the need to develop diverse text processing and classification techniques. The most essential and vital challenge in NLP is text classification. It was proposed for this purpose to classify documents and texts into pre-determined categories based on their contents, and it has since become one of the most popular methods of implementing machine learning. The machine learning (ML) paradigm is one where a generic inductive approach learns to create a privately classified text using a set of classified texts and the features of the classes of interests. Furthermore, discovering the relevant information can help improve information retrieval efficiencies while reducing the overload of information. Traditional models typically require artificial methods for obtaining good sample attributes before classifying them using standard machine learning algorithms. Therefore, feature extraction restricts the method's effectiveness significantly. On the other hand, deep learning differs from typical models, which are getting more attention because they incorporate feature extraction into the model building approach by performing a series of nonlinear

transformations that assist in transferring feature representations to outputs. Furthermore, deep learning algorithms avoid the need for experts to define rules and attributes, instead automatically providing high-level semantic representations for texts. Therefore, in these studies, we explore the capabilities of contextually embedding derived from pre-trained models like BERT, and make use of multi-label classification of text documents in a huge English news dataset, in addition to some traditional machine learning methods to be applied in a small English news dataset. Finally, another version of BERT, Arabic BERT, explores sentiment polarity toward extracted aspects in an Arabic hotel review dataset.

Keywords: Multi-label Classification, Machine Learning, Arabic Sentiment Analysis, Deep Learning, BERT, Single-label Classification.

ÖZ

STANDART MAKINE ÖĞRENIMI ALGORITMALARI VE ÖNCEDEN EĞITILMIŞ BERT TRANSFORMER KULLANARAK ÇOK ETIKETLI VE TEK ETIKETLI METIN SINIFLANDIRMA

ALFIGI, Huda

Bilgi Teknolojileri Yüksek Lisans

Danışman: Dr. Abdül Kadir GÖRÜR

Ocak 2023, 86 Sayfa

Doğal dil işleme (DDİ) araştırmaları, dijital belgelerin artan kullanılabilirliği ve bunlara çeşitli şekillerde erişme ihtiyacı nedeniyle son zamanlarda büyük ilgi görmüştür. Dijital metin verilerindeki patlama, çeşitli metin işleme ve sınıflandırma tekniklerinin geliştirilmesi ihtiyacını ortaya koymaktadır. DDİ'deki en temel ve hayati zorluk metin sınıflandırmasıdır. Bu amaçla, belgeleri ve metinleri içeriklerine göre önceden belirlenmiş kategorilere ayırmak için önerilmiştir ve o zamandan beri makine öğrenimini uygulamanın en popüler yöntemlerinden biri haline gelmiştir. Makine öğrenimi (MÖ) yaklaşımı, genel bir tümevarım yaklaşımının bir dizi sınıflandırılmış metin ve ilgi sınıflarının özelliklerini kullanarak özel olarak sınıflandırılmış bir metin oluşturmayı öğrendiği bir yöntemdir. Ayrıca, ilgili bilgilerin keşfedilmesi, fazla bilgi yükünü azaltırken bilgi alma verimliliğini artırmaya yardımcı olabilir. Geleneksel modeller, standart makine öğrenimi algoritmalarını kullanarak sınıflandırmadan önce iyi örnek nitelikleri elde etmek için genellikle yapay yöntemler gerektirir. Bu nedenle, özellik çıkarma yöntemin etkinliğini önemli ölçüde kısıtlar. Öte yandan, derin öğrenme, özellik temsillerinin çıktılara aktarılmasına yardımcı olan bir dizi doğrusal olmayan dönüşüm gerçekleştirerek özellik çıkarma işlemini model oluşturma yaklaşımına dahil ettiği için daha fazla ilgi gören tipik modellerden farklıdır. Ayrıca,

derin öğrenme algoritmaları, uzmanların kuralları ve öznitelikleri tanımlama ihtiyacını ortadan kaldırır, bunun yerine metinler için otomatik olarak üst düzey anlamsal temsiller sağlar. Bu nedenle, bu çalışmalarda, BERT gibi önceden eğitilmiş modellerden elde edilen bağlamsal gömme yeteneklerini keşfediyoruz ve küçük bir İngilizce haber veri kümesinde uygulanacak bazı geleneksel makine öğrenimi yöntemlerine ek olarak, büyük bir İngilizce haber veri kümesindeki metin belgelerinin çok etiketli sınıflandırmasından yararlanıyoruz. Son olarak, BERT'in bir başka versiyonu olan Arapça BERT, Arapça bir otel incelemesi veri kümesinden çıkarılan yönlere yönelik duygu eğlimini araştırmaktadır.

Anahtar Kelimeler: Çok Etiketli Sınıflandırma, Makine Öğrenmesi, Arapça Duygu Analizi, Derin Öğrenme, BERT, Tek Etiketli Sınıflandırma.

ACKNOWLEDGEMENT

All praise is due to Allah Who guided me to this. I could not truly have been led aright if Allah had not guided me.

Firstly, I would like to express my sincere appreciation and gratitude to Assist. Prof. Dr. Abdul Kadir GÖRÜR and Assist. Prof. Dr. Roya Choupani respectively for the excellent guidance and for providing me with an excellent atmosphere to conduct this research. Above all, thanks to them for teaching me how to become a researcher. My special gratitude also goes to the rest of the thesis committee Assist. Prof. Dr. Murat SARAN and committee Assist. Prof. Dr. Erdal ERDAL for the encouragement and insightful comments.

Many thanks to Dr. Khaled Jady for dedicating many hours to my assistance, continuous moral support and recommendations for improving this thesis.

Finally, I most express my profound, sincere gratitude to my parents, my family, and especially my husband for his unparalleled love, encouragement, help and continuous support both financially and emotionally throughout my years of study and the process of this thesis. This attainment would not have been possible without him. I dedicate this accomplishment to him.

TABLE OF CONTENTS

STAMENT OF NONPLAGIARISM III
ABSTRACT IV
ÖZ VI
ACKNOWLEDGEMENT VIII
LIST OF TABLES XIII
LIST OF FIGURES XIV
LIST OF ABBREVIATIONS XV
CHAPTER I INTRODUCTION 1
1.1 MOTIVATIONS 1
1.2 TEXT CATEGORIZATION METHODS 1
1.2.1 Manual Categorization 2
1.2.2 Rule Based Categorization 2
1.2.3 Automatic Text Categorization 3
1.3 TEXT CATEGORIZATION APPLICATION 3
1.3.1 Web-Page Classification 4
1.3.2 Spam Filtering 4
1.3.3 Word Sense Disambiguation 4
1.3.4 Sentiment Analysis 4
1.3.5 Recommended System 4
1.3.6 Documents Organization 4
1.3.7 Text Filtering 5
1.3.8 Documents Summarization 5
1.4 OBJECTIVES OF THE STUDY. 5
CHAPTER II TEXT CLASSIFICATION 6
2.1 TEXT CLASSIFICATION DEFINITION 6
2.2 SINGLE-LABEL TEXT CATEGORIZATION 6
2.2.1 Binary Classification 6

2.2.2 Multi-Class Classification 7
2.3 MULTI-LABEL TEXT CLASSIFICATION 7
CHAPTER III LITRATURE REVIEW 8
3.1 ENGLISH TEXT CLASSIFICATION 8
3.2 ARABIC TEXT CLASSIFICATION 11
CHAPTER IV TEXT CLASSIFICATION TECHNIQUES 13
4.1 SINGLE-LABEL CLASSIFICATION TECHNIQUES 13
4.1.1 Rocchio Classifier 13
4.1.2 Boosting And Bagging 14
4.1.2.1 Boosting Classifier 14
4.1.2.2 Bagging Classifier 15
4.1.3 K-Nearest Neighbor 15
4.2 MULTI-LABEL CLASSIFICATION TECHNIQUES 16
4.2.1 Problem Transformation Method 16
4.2.1.1 Label Power Set 17
4.2.1.2 Binary Relevance 17
4.2.1.3 Copy Transformation 19
4.2.2 Algorithm Adaptation Method 19
4.2.3 Deep Learning Method 20
4.2.4 Deep Neural Network (DNN) 21
4.2.4.1 Recurrent Neural Network (RNN) 22
4.2.4.2 Long Short-Term Memory (LSTM) 22
4.2.4.3 Gated Recurrent Unit (GRU) 23
4.2.4.4 Convolutional Neural Network (CNN) 23
CHAPTER V METHODOLOGY 24
5.1 TEXT CLASSIFICATION PROCESS USING STANDARD MACHINE LEARNING ALGORITHMS 24
5.1.1 Dataset 25
5.1.2 Data Preparation and Preprocessing 26
5.1.2.1 Text Preprocessing 26
5.1.2.1.1 Tokenization 26
5.1.2.1.2 Stop Words 26
5.1.2.1.3 Capitalization 26

5.1.2.1.4 Noise Removal 27
5.1.2.1.5 Stemming 27
5.1.2.1.6 Lemmatization 27
5.1.2.2 Weighted Words 27
5.1.2.2.1 Bag Of Words (BOW) 27
5.1.2.2.2 Term Frequency-Inverse Document Frequency (TF-IDF) 28
5.1.2.3 Dimensionality Reduction 29
5.1.2.3.1 Feature Selection 29
5.1.2.3.1.1 Document Frequency 30
5.1.2.3.2 Feature Extraction 30
5.1.2.3.2.1 Principle Component Analysis 30
5.1.3 Classification Algorithms 31
5.1.3.1 Support Vector Machine (SVM) 31
5.1.3.2 Naïve Bayas (NB) 32
5.1.3.3 Decision Tree (DT) 33
5.1.3.4 Random Forest (RF) 34
5.1.4 Advantages and Disadvantages of the Classification Algorithms 34
5.1.5 Model Construction 36
5.2 TEXT CLASSIFICATION PROCESS BY USING PRE-TRAINED BERT MODELS 37
RCV1-v2 Dataset 37
HARD Dataset 37
ABSA Dataset 38
5.2.1 Datasets 38
5.2.1.1 RCV1-v2 Dataset 38
5.2.1.2 HARD Dataset 38
5.2.1.3 ABSA Dataset 38
5.2.1.4 Data Preparation and Preprocessing 39
5.2.2 Pretrained Deep Learning Models for Text Classification 40
5.2.2.1 Transformers 40
5.2.2.2 BERT 41
5.2.2.3 Arabic Bert 42
5.2.3 Design Of Downstream Model 42
5.2.3.1 Model Parameters 42

5.2.3.1.1 Transfer Learning 43
5.2.3.2 Activation Function: Softmax and Sigmoid 43
5.2.3.3 Adam Optimizer 44
5.3 EVALUATION MATRICES 44
5.3.1 Accuracy 44
5.3.2 Micro Avraged-F1 45
5.3.3 Macro Avraged-F1 46
CHAPTER VI EXPERIMENTS 47
6.1 EXPERIMENTAL SETUP 47
6.2 EXPERIMENTAL RESULTS 47
CHAPTER VII CONCLUSIONS AND FUTURE WORK 53
REFERENCES 55
APPENDICES 66
APPENDIX 1 DATASET SAMPLES 66

LIST OF TABLES

Table 4.1: Example of Multi-label Data Set 16
Table 4.2: Transformed Data Set using LP Method..... 17
Table 4.3: Constructed Binary Relevance Data (a)…….18
Table 4.4: Constructed Binary Relevance Data (b)…….18
Table 4.5: Constructed Binary Relevance Data (c)…….…18
Table 4.6: Constructed Binary Relevance Data (d)…….18
Table 4.7: Transformed Data Set using Copy Transformation……19
Table 5.1: Advantages and Disadvantages of Support Vector Machines (SVM)……34
Table 5.2: Advantages and Disadvantages of Naïve Base (NB)…….35
Table 5.3: Advantages and Disadvantages of Decision Tree (DT)…….35
Table 5.4: Advantages and Disadvantages of Random Forest (RF)……36
Table 6.1: Experiments System Specification Settings in Jupyter Notebook and Colab……...47
Table 6.2: Experimental Results of Standard Machine Learning Models…..……48
Table 6.3: Experimental Results of BERT Transformer Models……..…….49
Table 6.4: Experimental Results of the same Datasets in the Literatures……49
Table 6.5: The Best Experimental Results of our Experiments and the Literatures…50
Table 1.1: Reuters-21578 Dataset Sample…….……67
Table 1.2: RCV-v2 Dataset Sample…….……..68
Table 1.3: HARD Dataset Sample……..……69
Table 1.4: ABSA Dataset Sample……...……70

LIST OF FIGURES

Figure 1.1: Manual Text Categorization Process…….……2
Figure 1.2: Rule Based Text Categorization Process……3
Figure 1.3: Automatic Text Classification Process……..3
Figure 4.1: The Boosting Classifier Architecture……...15
Figure 4.2: The Bagging Classifier Architecture……....15
Figure 4.3: Illustrate the Architecture of a Typical DNN……22
Figure4.4: Convolutional Neural Network (CNN) Architecture for Text Classification……...23
Figure 5.1: Step by Step Text Classification Process in Reuters-21578 Dataset. ….25
Figure 5.2: This figure shows the linear and non-linear Support Vector Machine (SVM)…….32
Figure 5.3: Random Forest Architecture……....34
Figure 5.4: Step by Step Text Classification Process in RCV1-v2 Dataset…….37
Figure 5.5: Step by Step Text Classification Process in HARD Dataset……...37
Figure 5.6. Step by Step of prediction ABSA Dataset by using transferred learned model……...38

LIST OF ABBREVIATIONS

ABBREVIATIONS

DM	: Data Mining
NLP	: Natural Language Processing
ML	: Machine Learning
DL	: Deep Learning
KE	: Knowledge Engineering
TC	: Text Classification
NB	: Naïve Base
DT	: Data Mining
SVM	: Support Vector Machine
NNet	: Neural Network
KNN	: K-Nearest Neighbor
LLSF	: Linear Least-squares Fit
RNN	: Recurrent Neural Network
CNN	: Convolutional Neural Network
RCNN	: Recurrent Convolutional Neural Network
GCN	: Graph Convolutional Network
TF-IDF	: Term Frequency-Invers Document Frequency
ULMFiT	: Universal Language Model Fine-tuning
LM	: Language Model
PAG	: Passive Aggressive
LR	: Logistic Regression
ABT	: AdaBoost
ABSA	: Aspect-based Sentiment Analysis
RF	: Random Forest
Perceptron	: PRN

GRU	: Gated Recurrent Units
CRF	: Conditional Random Field
IAN	: Interactive Attention Network
LP	: Label Power Set
BR	: Binary Relevance
MLC	: Multi-label Classification
MMAC	: Multi-label Associative Classification
BPMLL	: Back-Propagation Multi-Label Learning
MLkNN	: Multi-label k-nearest neighbor
IBLR	: Instance Based Learning by Logistic Regression
LSTM	: Long Short-Term Memory
DNN	: Deep Neural Network
DR	: Dimensionality Reduction
DF	: Document Frequency
PCA	: Principal Component Analysis
BoW	: Bag Of Word

CHAPTER I
INTRODUCTION

1.1 MOTIVATIONS

Because of the variety of numerous sources from which vast volumes of digital data are derived, such as internet data, websites, social networks, e-mails, internet resources, and many others, text mining research has gotten a lot of interest in recent years, because texts are a tremendous resource of information. However, due to its unstructured format, extracting knowledge and insights from the text may be complex and time-consuming [1]. Text mining applications involve text categorization, text summarization, text document filtering, question-answering systems, and opinion analysis classification, also known as sentiment analysis or emotional analysis [2].

machine learning (ML), natural language processing (NLP) and Data mining(DM) techniques work together to identify patterns in various types of text documents from various categories and perform task execution in an automatic manner [2]. Deep learning algorithms recently beat previous machine learning methods to produce state-of-the-art results in different domains, including natural language processing, image classification, and face recognition, among others [3]. Deep learning algorithms have been successful in this domain due to their deep structural models, which are suggested for extracting features and creating representations on massive datasets due to their high adaptive and nonlinear feature extraction abilities [4].

1.2 TEXT CATEGORIZATION METHODS

Text categorization, also described as topic classification or text classification. It is a typical NLP task that tries to set labels or classes to input text such as sentences, queries, essays, and documents [1]. For instance, news items could be sorted into the text classes to which they belong, e-mails can be checked for spam, feelings or opinions from product evaluations can be analyzed, and so on [5]. Text classification can be performed in several ways, as in the following:

1.2.1 Manual Categorization

Manual categorization entails a professional analyst analyzing the textual documents and assigning them to the appropriate text classes. In general, this kind of structure allows for a lot more flexibility and reliable retrieval of information, but it is quite time-consuming and expensive [6][7][1]. Fig.1.1 provides an illustration of the manual text categorization process's technique.

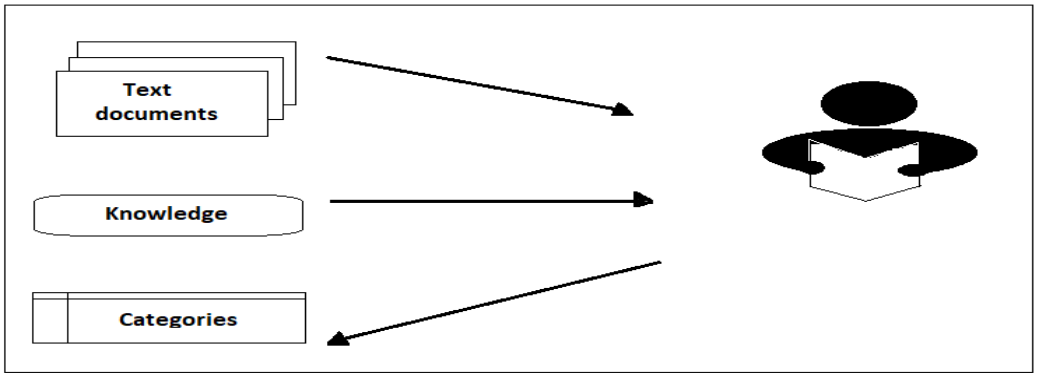


Figure 1.1: Manual Text Categorization Process

1.2.2 Rule Based Categorization

Methods based on rules require deep domain knowledge to classify text into multiple groups using a set of pre-defined rules [1]. In the 1980s, the most common method for developing automated document classifiers was to manually construct an expert system able to generating TC choices using knowledge engineering (KE) approaches.

If <DNF formula > then <category>

A DNF ("disjunctive normal form") formula is a disjunction of conjunctive conditions; if the document matches the formula, that is meets at minimum one of the clauses, it is classified as <category> [8]. The disadvantage of this strategy was the high cost of human power necessary for designing and maintaining the rule set, i.e., modifying the rule set as a consequence of future class additions or deletions, or changes in the meaning of existing classes [2]. Fig 1.2 illustrates rule-based classification process.

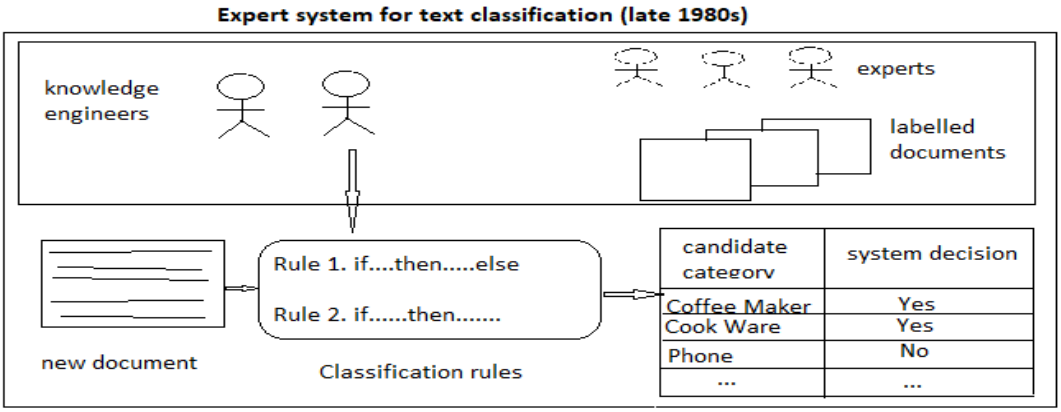


Figure 1.2: Rule Based Text Categorization Process

1.2.3 Automatic Text Categorization

The rule-based method has been replaced by supervised machine learning-based ways for categorizing texts based on information observed. A machine learning model discovers intrinsic correlations between texts and their classes using pre-labeled instances as training data [1]. This method provides a number of benefits over expert systems. To begin with, a greater level of automation is introduced: the engineer must create an automatic builder of text classifiers (the learner) rather than just building a text classifier. The learner may then can be utilized to generate a range of different classifiers for a variety of domains and applications after it has been developed; all that is required is to provide it with the proper sets of training texts [2]. The mechanism of the automatic text categorization approach is demonstrated in Fig. 1.3.

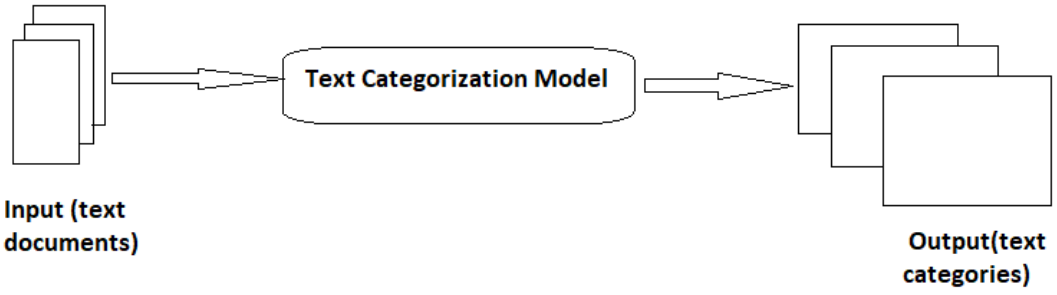


Figure 1.3: Automatic Text Classification Process

1.3 TEXT CATEGORIZATION APPLICATION

There has been great interest in TC due to numerous applications that we can benefit from it, of which we will briefly cover the most relevant ones here:

1.3.1 Web-Page Classification

Lately, TC has attracted a lot of attention for automatically categorizing online pages or blogs within the structured catalogs offered by big Internet portals. Instead of querying an entire web browser, a searcher might find it simpler to go first through the structure of categories and then narrow the search to a specific category of concern when web pages are cataloged in this fashion. Organizing web pages in an automatic manner has great benefits, while manual classification of a big enough fraction of the Web is impossible [9][10][11][8].

1.3.2 Spam Filtering

TC is an active area of research in which TC algorithms are used to classify personal e-mail into two categories [LEGITIMATE and SPAM] in order to protect users from unwelcome mass mailings[12][13].

1.3.3 Word Sense Disambiguation

It is the process of giving the words in a document the right meaning (sense). Regarding the topic of this document, TC might simplify the disambiguation task [14][15].

1.3.4 Sentiment Analysis

This is the process of analyzing people's views in text data (such as product evaluations, movie reviews, or tweets) and extracting their polarity and perspective, whether positive or negative [1][16].

1.3.5 Recommended System

Content-based recommender systems make suggestions based on the item's description and the user's priority profile [17].

1.3.6 Documents Organization

In general, TC approaches could be used to handle a variety of different document organization and archiving challenges, whether for the personalized organization or the structuring of a corporation’s document base [18].

1.3.7 Text Filtering

The process of categorizing a flow of received documents is sent asynchronously from an information generator to an information client [19] . A newsfeed is a common example in which the process is that the news organization and the client is the newspaper. In this situation, the filtering system must prevent the customer from receiving materials that he or she is unlikely to be interested in [20].

1.3.8 Documents Summarization

The document's summary may contain terms and phrases that are not present in the original texts, multi-document summary is required due to continuous growth of online content, As a result, many scholars concentrate on this issue of extracting key elements from a document via text categorization [21: 1][22].

1.4 OBJECTIVES OF THE STUDY.

(a) Exploring the modeling capabilities of contextual embeddings from pre-trained language models by using two versions of Bert model.

(b) A comparison of the performance of standard machine learning algorithms and pre-trained deep learning models across various dataset sizes.

(c) Exploring the performance between two text classification tasks:

   I. Single-label classification.

   II. Multi-label classification.

(d) Classification of unknown input using saved trained model.

CHAPTER II
TEXT CLASSIFICATION

2.1 TEXT CLASSIFICATION DEFINITION

Text categorization is the process of giving a boolean expression to each combination $(d_j, c_i) \in D.C$, where D represents a set of documents, $D = \{x_1, x_{2...}, x_n\}$ where x_i denotes a sample data (i.e., a document, a textual segment) containing s sentences, each of which has $_{ws}$ words with $_{lw}$ letters, and $C = \{c1,..., c|C|\}$ denotes a collection of predetermined classes [23]. A choice to file (d_j) under (c_i) is indicated by a value of 'T' set to (dj, ci), whereas a decision not to file (d_j) under (c_i) is indicated by a value of 'F' (c_i). In more technical terms, the purpose is to estimate the unknown target function $\Phi^{\check{}} : D \times C \rightarrow \{T, F\}$ using a function $\Phi : D \times C \rightarrow \{T, F\}$ known as the classifier (aka hypothesis, or model, or rule) [8][24]. Furthermore, text categorization is categorized into different types: single-label and multi-label text categorization. In this section, we will discuss and detail these types.

2.2 SINGLE-LABEL TEXT CATEGORIZATION

For a given number k, each $d_j \in D$ should have precisely k (or $\leq$ k, or $\geq$ k) components of C. The situation in which every element d_j should always be assigned to precisely one category is called single-label classification. Single label classification (also known as non-overlapping categories) is divided into two cases of classification: binary classification and multi-class classification.

2.2.1 Binary Classification

In the situation of text and web-based data, binary classification is often known as filtering. It is typical instance of single-label classification is interested in learning from a collection of instances referred to a single label l from a set of disassemble labels L, |L| > 1. If |L| equals 2 [25].

To put it another way, each ($d_j \in D$) have to classify into whether category ci or its complementary ci.

2.2.2 Multi-Class Classification

The concept of multi-class task is similar to binary classification in that each text instance is assigned to a single label, whereas (if $|L| > 2$) the collection of labels includes more than two labels or categories [25].

2.3 MULTI-LABEL TEXT CLASSIFICATION

Multi-label text categorization (also known as overlapping categories) indicates to the task of categorizing (or labeling) a text document into one or more categories [26]. Text categorization and multi-label text categorization are commonly used in conflicts. For instance, in medical diagnostics, a patient could have diabetes and stomach cancer both of them at the exact time[25]; music categorization [27]; semantic scene classification [28]; and email spam detection [29]. An image may relate to even more than one conceptual category at the same time in semantic scene categorization, such as beaches and sunsets. Similarly, a song might be classified as belonging to more than one style in music classification.

The objective is to learn a function h: $Rm \rightarrow 2Y$ from the training sample $D = \{(x_i, y_i) | 1 \leq i \leq N\}$, where N represents the entire amount of training examples, x_i represents the $i\text{-th}$ instance input in Rm and $y_i \subset Y$ represents a subgroup of the label space Y [26].

CHAPTER III
LITRATURE REVIEW

3.1 ENGLISH TEXT CLASSIFICATION

In [30], they proposed a comparison between two inductive learning algorithms, a Bayesian classifier(NB) and a decision tree(DT) learning algorithm, with different sized feature sets selected by information gain. They have applied their experiment to different size datasets. They have used the old version (v1) of Reuters Newswire that consisted of 21,450 documents and 135 categories, and the second one is the MUC-3 dataset that contains 1,500 docs sourced from the US Foreign Broadcast Information Service (FBIS). Both algorithms have achieved good results on the Reuters dataset but much less well on the MUC-3 data. They discovered that feature selection was essential and that a preliminary filtering of features using a global evaluation measure could help the contextual feature selection method utilized by decision trees.

Another study that has applied supervised machine learning techniques was introduced by [31], which were the Support Vector Machines (SVM), a neural network (NNet) approach, the Naive Bayes (NB) classifier, the k-Nearest Neighbor (kNN) classier, and the Linear Least-squares Fit (LLSF) mapping. They compared the accuracy of different classifiers in terms of category frequencies, and how much the training data provided influenced the success of each approach.

In this experiment, the authors [32] used Reuters 21578, which is the same version that we used in our experiment. The main reason for this experiment is to compare the accuracy of the SVM algorithm with the rest of the four common methods in text classification: Naive Bayes (NB), the C4.5 decision tree and distance weighted k-nearest neighbor method. The experiment showed that SVM outperforms other methods with the ability to perform well in high dimensionality and no need for feature selection. Additionally, the SVM does not demand any parameter tuning, so it can find good parameter settings automatically. [33] suggested a new deep learning

Algorithm, XML-CNN, to handle multi-label co-occurrence patterns in a way that optimizes the purpose and structure of a neural network architecture by using a binary cross-entropy loss that is better suited for multi-label issues, a dynamic max pooling scheme that collects broader and deeper information from various areas of the document, and a hidden bottleneck layer for improved document representations in addition to decreasing model complexity. Additional study has been done by using deep learning approaches, which have been introduced in [34]. They have suggested a Recurrent Convolutional Neural Network (RCNN) and applied it to the issue of text classification to handle the limitations in the CNN and RNN models, which are able to capture contextual information by using a convolutional neural network to build the representation of text and the recurrent structure. Moreover, they have pre-trained the word embedding with default parameters in word2vec by applying the Skip-gram model. [26] To address the issue of multi-label text categorization, convolutional neural networks (CNN) and recurrent neural networks (RNN) were combined. This study[35] introduces HDLTex, a novel method for classifying documents into hierarchical categories by combining various deep learning techniques. The accuracy levels obtained by the combinations of RNN at the higher level and DNN or CNN at the lower level were significantly greater than those attained by traditional methods like naive Bayes or SVM.

In this paper, [36] has proposed a unique text categorization technique called Text Graph Convolutional Networks (Text GCN). For the entire corpus, they generated a non - homogenous word document graph and converted the problem of document classification into one of node classification. They created edges between nodes based on word co-occurrence across the whole corpora (word-word edges) as well as word occurrence in text documents (document-word edges). The weights of the edges that contain the semantic correlation of words between documents node and a word node and between two-word nodes are calculated by the terms frequency-inverse document frequency (TF-IDF) and point-wise mutual information (PMI) respectively. They feed the text graph into a simple two-layer GCN after it has been constructed. Computer vision has considerably benefited from transfer learning, but current NLP methods still call for task-specific adaptations and training from scratch. As a result, [37] have proposed Universal Language Model Fine-tuning (ULMFiT) by which pretraining a language model (LM) on a huge global corpora and offering unique strategies to hold onto prior knowledge allows for powerful learning across a

variety of tasks and preventing catastrophic forgetting during fine-tuning, including slanted triangular learning rates, discriminative fine-tuning and gradual unfreezing. Recent advances in text classification using neural network-based techniques have proved significant. Models that use attention mechanisms [38], like pretrained transformers such as Bert [39], are one of the self-attention models based on huge corpora and multi-task pre-training that have demonstrated the ability to capture the contextual information present in a sentence or document. Bert has been utilized in wide range in TC since it is more effective than other models [40][41][42].

Today, transfer learning is of utmost significance in the field of research. By using several transformer versions, researchers are working hard to increase accuracy in every study. As in this study [43], several transfer learning models have been applied to make a comparative assessment of their performance at a high level, including BERT-large, BERT-base, RoBERTa-base, XLM-RoBERTa-base, DistilBERT, ALBERT-base-v2, and BART-large, for binary text classification on the COVID-19 fake news dataset and the COVID-19 English tweet dataset, all of which are taken from reputable repositories. In this research [44], the VGCN-BERT model has been proposed, which combines BERT's functionality with a VGCN to integrate a vocabulary graph embedding module with BERT. Through the many BERT layers, local information and global information can interact, affect one another, and collaborate to create a final classification representation. The VGCN-BERT strategy outperformed BERT and GCN alone in these experiments on various text classification datasets, and it also achieved higher effectiveness than that reported in other publications. In this study [45], HTrans—a hierarchical transfer learning-based method for training binary classifiers for classifying classes—was suggested. In order to categorize categories at lower levels of the taxonomy, this method makes use of model parameters that were previously learned at higher levels of the taxonomy. This study on the RCV1 dataset demonstrated the benefits of employing pre-trained parameters of the model from higher level categories for classifiers of categories with fewer training samples. Additionally, it was demonstrated that binary classifiers perform far better than multi-label models. Finally, by employing the optimum class weights learned during the training of the binary classifiers, efficiency was shown over the state-of-the-art multi-label model.

3.2 ARABIC TEXT CLASSIFICATION

Because of the lack of Arabic datasets, there is not a huge amount of literature that is based on Arabic texts. What's more, in this section, we have covered some papers that could be related to our experiment. The first study was conducted on the HARD dataset for sentiment analysis, introduced by [46], in which they made this dataset reachable by the research community in the Arabic language. Additionally, they have done three different experiments by selecting different six traditional algorithms that are commonly used in sentiment analysis, such as: Logistic Regression (LR), Random Forest (RF), Passive Aggressive (PAG), Support Vector Machine (SVM), Perceptron (PRN) and AdaBoost (ABT), with which they obtained very different results with different techniques of feature extraction and varying dataset versions.

In [47], proposed an Aspect-based Sentiment Analysis (ABSA). It investigated the modeling abilities of contextually word embeddings from pre-trained deep learning models, such as BERT, and made use of sentence that is pair input on the Arabic ABSA task. To handle the aspect sentiment polarity classification challenge, an Arabic BERT model has been used with a linear classification layer. The results showed that it surpassed state-of-the-art efforts and was resistant to overfitting. Another study has introduced the ABSA dataset by [48] which has been performed via a type of RNN (GRU), with two GRU models constructed, as follows: (a) A deep learning architecture based on a cutting-edge model that extracts the major opinionated characteristics (i.e., (OTE)) from representations that include both terms and characters using a bidirectional GRU, CNN, and CRF hybrid (BGRU-CNN-CRF). (b) An IAN based on bidirectional GRU (IAN-BGRU) is used to determine sentiment polarity toward retrieved characteristics from the data.

In [49] they presented the Arabic BERT by building convolutional neural networks with a pre-trained BERT model and a collection of pre-trained Bert Arabic models. They used a basic Arabic BERT model (bert-base-arabic) to create vector representations by integrating the output of the last four hidden layers of a base-sized pre-trained BERT. These embeddings were fed into five distinct convolutional filters in simultaneously. In this study [50], An Arabic language representation model was developed in this study to advance the state-of-the-art in a number of Arabic NLU challenges. Building on the BERT concept, which stacked bidirectional transformer encoders, they developed ARABERT. The BERT base architecture, which has 12

encoder blocks, 768 hidden dimensions, 12 attention heads, 512 maximum sequence length, and a total of about 110 million parameters, was employed. In order to better adapt the model to the Arabic language, they also added additional preprocessing before the model's pretraining.

Deep neural network techniques have recently been used to solve numerous text classification issues, particularly in the classification of English-language texts. One of the most well-known models is the convolutional neural network (CNN). CNN is not often used in the classification of Arabic text, nevertheless. In addition, parameter setting issues prevented the latest investigations from achieving superior classification performance. A new hybrid learning algorithm for Arabic text was created to get around this restriction. [51] This study suggests classifying Arabic text using convolutional neural networks based on genetic algorithms. The parameters of CNN are optimized using a genetic algorithm. Using two sizing datasets, the suggested model is evaluated and contrasted with leading-edge research. The outcomes demonstrated a 4–5% enhancement in classification performance. In this work [52] The classification of Arabic text using many classes and a single label is presented using a deep learning model. As a result, a novel strategy employing the mutual information feature is suggested to get the words ready for categorization in deep learning models using hybrid models. Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) make up this model's architecture, which results in C-LSTM for text classification. The network is composed of a CNN layer, an LSTM layer, a dropout layer, a dense layer, and a global max pooling layer.

CHAPTER IV
TEXT CLASSIFICATION TECHNIQUES

Text categorization is the process of extracting information from raw text data and estimating text data categories based on that information. Text classification models have been presented in earlier years in a variety of forms. The two most crucial models used in two distinct text categorization tasks will be discussed in this chapter: Both single-label and multiple-label classifications.

4.1 SINGLE-LABEL CLASSIFICATION TECHNIQUES

Traditional approaches improve text classification speed and accuracy while increasing their range of applications. The raw input text must first go through preprocessing in order to train traditional models before being fed into the classifier in accordance with chosen characteristics. Here, we go into great detail about a few typical classifiers for single-label classification. First, we discuss standard text classification techniques such as Rocchio classification. Following that, we cover ensemble-based learning algorithms like bagging and boosting, primarily used for text analysis and query learning strategies. In most data mining domains, logistic regression (LR), one of the most basic classification strategies. Non-parametric approaches like k-nearest neighbor have also been investigated and used in classification challenges (KNN). Furthermore, there are other models that can be used in single-label classification, naïve base (NB), random forest (RF), such as support vector machine (SVM) and decision tree (DT), which we will discuss in the next chapter, in addition to the advanced deep learning models, which we will address in this section.

4.1.1 Rocchio Classifier

In [53] in 1971 as a method for querying full-text databases using relevance feedback. Since then, this approach has been used for text and document classification and improved by other scholars [54]. TF-IDF scores are used for every informative

term rather than of Boolean features in this type of classification technique. The Rocchio classifier generates a prototype vector for each label using training documents. This prototype is a weighted average of the vectors in the training content that correspond to a particular category. Then, it places every test text document in the category with the highest similarity to each prototype vector. A centroid of a class c is calculated using the average vector:

$$\underset{\mu}{\rightarrow}(\mathrm{C}) = \frac{1}{|\mathrm{Dc}|}\sum_{d\in \mathrm{Dc}} \overrightarrow{v_d} \tag{4.1}$$

$\underset{vd}{\rightarrow}$ is the weighted vector representation of document d and Dc being the collection of documents in D that correspond to class c. The estimated label for document d is the one that has the shortest distance measure between the centroid and the document:

$$c^* = \arg\min \|\overrightarrow{\mu_c} - \overrightarrow{v_d}\| \tag{4.2}$$

4.1.2 Boosting And Bagging

Voting classification methods like boosting and bagging have been developed and optimized for the classification of textual data sets and documents. Bagging does not consider the performance of the prior classifier, whereas boosting flexibly alters the spread of the training dataset based on the effectiveness of the prior approach.

4.1.2.1 Boosting Classifier

In 1990, R.E. Schapire introduced the boosting algorithm as a way to improve the outcomes of a poor training system.[55]. The improvement of this method was made by [56]. A number of weak classifiers are combined in this ensemble modeling technique in an attempt to generate a strong classifier. It is performed by developing a model by employing weak models in sequence. First, a model is constructed by utilizing the train data collection. The second one is then constructed to resolve the defects in the preceding model. In this approach, they are added until whether the whole training dataset is estimated correctly or the maximum number of approaches has been added. Finally, it creates a strong learner by merging the discoveries of the weak learner and enhancing the model's predictive capacity. Boosting prioritizes cases

that were improperly classified or had more errors as a result of previous weak rules. Figure 4.1 shows how a boosting strategy for 2D datasets works.

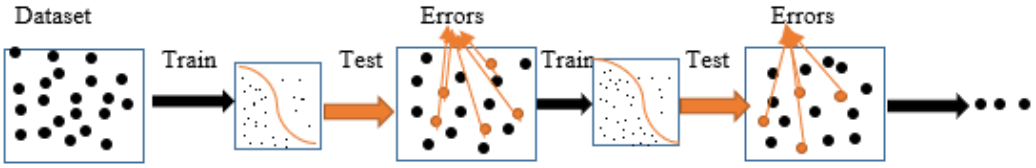


Figure 4.1: The Boosting Classifier Architecture.

4.1.2.2 Bagging Classifier

L. Breiman presented the bagging algorithm [57] as a vote classifier technique in 1996. The algorithm was built by using various random samples. Random samples produced a homogeneous sample from the training data collection. When the N bootstrap samples B1, B2, BN are generated, N classifiers (C) are provided, each of which is built from a distinct bootstrap sample Bi. The class that is predicted more frequently by its sub-classifiers is the output of our classifier C, which is composed of or created from C1, C2, and CN. A straightforward bagging technique that trains N models is shown in Figure 4.2.

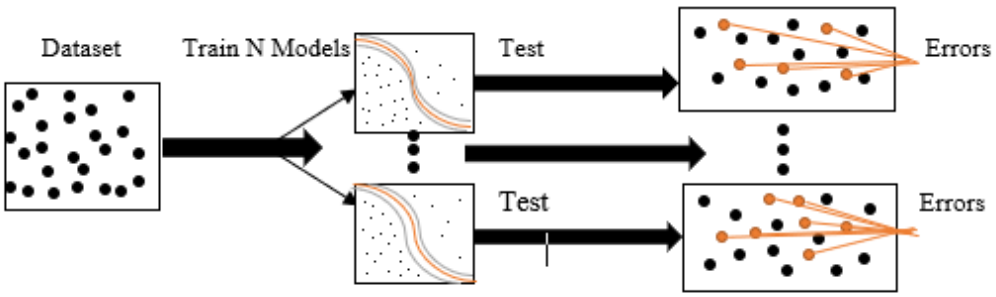


Figure 4.2: The Bagging Classifier Architecture.

4.1.3 K-Nearest Neighbor

The k-nearest Neighbors (KNN) algorithm is a non-parametric classification method. Numerous research fields employ this technique for text categorization applications. [58]. By giving a testing dataset x, the KNN method selects the k documents in the train data that are closest to the testing document x. It then ranks the category choices based on the K neighbors' classification. The category score of the of

the neighboring documents could be determined by how similar x and each neighboring document are. It is possible for multiple KNN documents to fall into the exact class; in this instance, the sum of the scores would represent the class k's matching score to the test document x. The algorithm sets the candidate in the class that received the greatest score from the testing document x after sorting the score values. The KNN architecture is shown in the figure, which was created using a 2D data set for simplicity. The KNN's decision rule is:

$$\mathcal{F}(x) = arg\ \max_{j} S(x, Cj) \tag{4.3}$$

$$= \sum_{di \in KNN} sim(x, di)\ y(di, Cj)$$

Where S is the score value relative to $S(x, C_j)$, the candidate i's score is relative to class j's score, and the output of f(x) is a label for the testing set document.

4.2 MULTI-LABEL CLASSIFICATION TECHNIQUES

For single-label classification tasks, most classical learning methods are designed. As a result, several techniques in the literature divide the multi-label problem into many single-label problems, allowing single-label algorithms to be employed.

In this section, we will divide the methods that handle multi-label classification into three main categories: problem transformation methods, algorithm adaptation methods, and lastly, deep learning approaches that could outperform all the previous standard machine learning methods and obtained state-of-the-art results.

4.2.1 Problem Transformation Method

Methods for converting a multi-label classification issue into regression problems or single-label classification, each with a rich literature of learning approaches [25]. There are some algorithms under this method that were introduced to handle this problem, such as binary relevance, label power sets, and copy method transformation, which we are going to discuss individually.

We will utilize the dataset in Table 4.1 to demonstrate these strategies. It is made up of four samples of four categories: sports, religion, science, and politics.

Table 4.1: Example of Multi-label Data Set

EX.	Sports	Religion	Science	Politics
1	X			X
2			X	X
3	X			
4		X	X	

4.2.1.1 Label Power Set

A label powerset (LP) is a type of conversion that generates one label for each subset of labels in the multi-label training dataset. As a result, the new set of labels matches the previous set's powerset. A single-label classification method may handle the modified data set after this transformation procedure. This classifier could then be used to give one of the new labels to new samples, which could then be mapped back to the original label [59]. Table 4. illustrates the outcome of applying this LP approach to modify the data set from Table 4.1.

A label powerset is only suggested for datasets with a few labels since the powerset combinations are limited to 2L, where L is how much of various labels are in the dataset. The generated powerset data tends to become sparse for datasets with a huge number of labels, making it more difficult for the classifier to work [5].

Table 4.2: Transformed Data Set using LP Method

EX.	Sports	(Sports ∧ Politics)	(Science ∧ Politics)	(Science ∧ Religion)
1		X		
2			X	
3	X			
4				X

4.2.1.2 Binary Relevance

The most widely used problem transformation approach (BR) trains |L| binary classifiers H_l: $X \rightarrow \{l, \neg l\}$, one for each of the label's l in L. It converts the original dataset into |L| datasets labeled as l if the original example's labels included l and as ¬l otherwise. It's the same approach they use to solve a single-label multiclass problem [60].

Because each classifier can only predict one label, the BR classification outcome is the sum of the labels that each classifier positively predicts. Binary relevance has some drawbacks, such as linear complexity with regard to the number of labels since it trains a separate binary model for each individual label. [5] .

In addition, including the fact that it ignores any label correlations and treats labels as mutually exclusive, which is incorrect when dealing with labels, MLC is an issue [61]. Table 4.3, Table 4.4, Table 4.5, and Table 4.6 show the four datasets that are constructed by the BR method when applied to the dataset in Table 4.1.

Table 4.3: Constructed Binary Relevance Data (a)

EX.	Sports	¬Sports
1	X	
2		X
3	X	
4		X

Table 4.4: Constructed Binary Relevance Data (b)

EX.	Politics	¬Politics
1	X	
2	X	
3		X
4		X

Table 4.5: Constructed Binary Relevance Data (c)

EX.	Religion	¬Religion
1	X	
2	X	
3		X
4		X

Table 4.6: Constructed Binary Relevance Data(d)

EX.	Science	¬Science
1	X	

Table 4.6: Continued

2		X
3	X	
4		X

4.2.1.3 Copy Transformation

Another different type of transformation problem entails duplicating each multi-label sample n times, where n represents the number of labels allocated to that sample. After that, each duplicated instance is given a single label from the original collection. Whether using weights or not [25][62]. The copy-weight transformation is a variant of the copy transformation that assigns a weight 1 / n to each replicated instance depend on how much labels exist in the original instance [5][63]. This strategy does not lose any information, but it ignores significant label correlations and may slow the learning process by raising the number of single-label occurrences in the dataset [61]. Table 4.7 shows the outcome of applying this strategy to modify the dataset in Table 4.1.

Table 4.7: Transformed Data Set using Copy Transformation

EX.	Class
1	Sports
1	Politics
2	Science
2	Politics
3	Sports
4	Religion
4	Science

4.2.2 Algorithm Adaptation Method

So many researchers have modified and improved many traditional algorithms to solve the problem of MLC because of their superior efficiency in tackling single-label classification problems. One of these algorithms is the ML-C4.5 decision tree, which was adapted to handle multi-label classification by [64]. Two modifications were made: the first modification enabled leaves to have multiple labels, and the second modification was to modify the definition of entropy to acquire enough information to establish which classes the particular pattern belonged to. Another algorithm is Multiclass Multi-label Associative Classification (MMAC) [65] is an

method based on the Associative Classification (AC) principles. First, it uses the copy transformation approach to convert the multi-label dataset into a single-label dataset. Then, using if–then rules, it works on training a single-label associative classifier to predict a single label. Finally, it combines the predictions of rules with the same predecessor to generate a rule with many labels in the rule's consequence.

Moreover, there is another adaptation [66] : a multi-label adaptation of the kNN lazy learning method. Actually, this strategy is based on the BR paradigm. Essentially, ML kNN applies the kNN algorithm to every label l separately: it selects the k nearest instances to the testing samples and labels those that are labeled with at least l as positive, while the rest are labeled as negative. The use of prior probabilities distinguishes this technique from utilizing the original kNN algorithm on the altered issue with BR. In addition, as an output, ML-kNN may generate a ranking of the labels.

The Rank SVM algorithm for multi-label classification has been adapted by [67]. It's a linear model that attempts to keep a large margin while minimizing a cost function. Ranking loss is the loss function that they use, which is defined as an average proportion of inaccurately ordered pairs of classes. However, as previously stated, a ranking algorithm has the drawback of not producing a collection of labels. Additionally, Back-Propagation Multi-Label Learning (BPMLL) is a multi-label learning neural network method proposed by [68]. It's based on the well-known back-propagation method. The major alteration to the method is the adoption of a new error function that accounts for multiple labels. The new function was created to capture the properties of multi-label learning, namely that labels related to a specific instance should be rated greater than labels not related to another instance.

Other algorithms considered as adaptations are Adaboost.MH and Adaboost.MR [69], the Naive Bayes algorithm was proposed with a multi-label adaptation in [70] and the Instance Based Learning by Logistic Regression (IBLR) [71] adaptation combines the k-NN algorithm's instance-based learning principle with logistic regression. In order to improve categorization, it additionally examines the labels of neighboring examples as characteristics.

4.2.3 Deep Learning Method

Deep learning is achieving tremendous advances in addressing long-standing difficulties in the artificial intelligence area, including a wide range of NLP tasks. Deep learning structures are built from networks of units. They have input units that

represent words, output units that reflect the class or classes of concern, and weights on the connections joining units that show dependent relationships. The input units for classifying a testing document dj are fed the word weights wkj; the activation of the units is passed via the network, and the score provided by the output unit(s) establishes the classification result (s).Backpropagation is a common technique of training deep learning models, in which the weights of the words in a train document are fed into the initial units, then if a classified incorrectly happens, the error is "backpropagated" to adjust the network's weights and omit or reduce the error. Moreover, there is a collection of the most common deep learning models, each of which will be described briefly below.

4.2.4 Deep Neural Network (DNN)

Deep neural networks (DNNs) are expected to be trained by connecting layers in a hidden section so that every layer only gets connections from the layer before it and only provides connections to the layer after it [35]. The link between the input feature space and the DNN's initial hidden units is the input. The input layer might be generated using word embedding, TF-IDF, or another technique. he output layer for multi-class classification is equal to the number of classes, or one for binary classification, as illustrated in Figure 4.3 the architecture of a typical DNN. DNN is implemented as a discriminative trained approach that employs a typical back-propagation technique with sigmoid (Equation (4.4)) or ReLU (Equation (4.5)) as the most well-known activation function. A Softmax function (Equation (4.6)) should be used as the output unit for multi-class categorization.

$$f(x) = \frac{1}{1+e^{-x}} \in (0,1) \tag{4.4}$$

$$f(x) = \max(0, x) \tag{4.5}$$

$$\sigma(z)j = \frac{e^{zj}}{\sum_{k=1}^{k} e^{-zk}} \tag{4.6}$$

Where ∀ j ∈ {1, . . ., K}

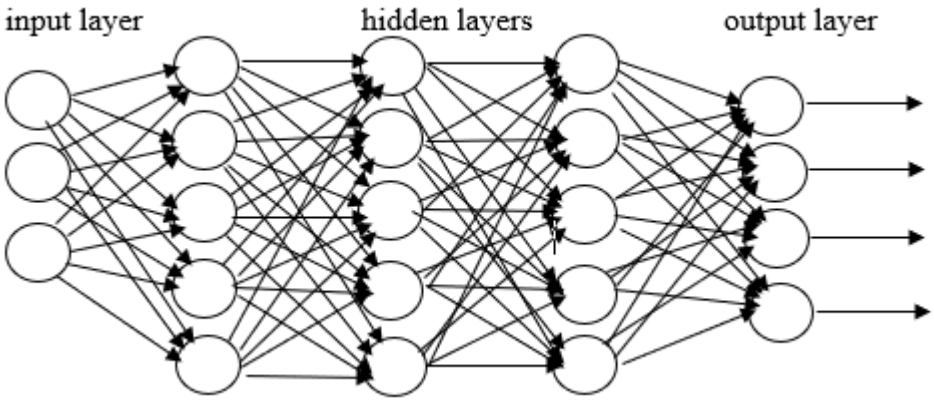


Figure 4.3: Illustrate the Architecture of a Typical DNN

4.2.4.1 Recurrent Neural Network (RNN)

The recurrent neural network (RNN) is another type of artificial neural design that is extensively utilized in text and classification studies [72][73 : 1-2]. RNN offers the preceding dataset in a sequence more weight. As a consequence, this strategy is effective for classifying string, text, and sequence data. In a more advanced manner, an RNN investigates the information of preceding nodes, allowing for the best semantic representation of a dataset's structure, which obviously contains three layers: the input unit which includes word embedding, feature extraction, hidden units, and finally the output unit. The RNN gets an input at each timestep, modifies its hidden state, and provides a prediction. For text categorization, RNNs are often used instead of LSTM or GRU because RNN models are ineffective and frequently outperform feed-forward neural networks. When the gradient descent technique's error is propagated back through the network, the RNN is sensitive to the issues of vanishing gradient and exploding gradient.

4.2.4.2 Long Short-Term Memory (LSTM)

LSTM was introduced by [74], and many research experts have since added to it [75]. Compared to the standard RNN, LSTM is a specific subset of RNN which tackles these issues by retaining long-term dependencies much effectively. When it comes to handling the vanishing gradient problem, LSTM is extremely beneficial by incorporating a memory cell to retain information over unlimited time periods and contains three gates (input gate, the output gate and forget gate) to manage the stream of information to and from the cell. LSTM employs multiple gates to precisely limit

the amount of information that is allowed into each node state, while having a chain-like structure comparable to an RNN.

4.2.4.3 Gated Recurrent Unit (GRU)

GRUs are an RNN gating technique proposed by [76] and [77]. The GRU architecture is a simple version of the LSTM. A GRU, on the other hand, is distinguished from an LSTM by its two gates and lack of internal memory. Furthermore, a second non-linearity is not used.

4.2.4.4 Convolutional Neural Network (CNN)

A convolutional neural network (CNN) is a deep learning architecture is used to classify documents in hierarchical order [34][78]. [3][79] CNN has two special layers in its algorithm for image processing. First is, convolution layer. An image vector is convolved with a collection of kernels of size d*d. These convolution layers are referred to as feature maps, and they can be layered to offer many filters on the input. Second, CNNs use a max pooling layer to decrease the amount of information from one layer to the following in the network, which lowering computational complexity. To transport the pooled output from stacked feature maps to the following layer, the mappings are flattened as one column. Additionally, both the weights and the feature detector filters are modified while back-propagating stage of a convolutional neural network. In Figure 4.4 the CNN architecture for text classification contains word embedding as input layer, 1D pooling layer, 1D convolutional layers, fully connected layers, and finally an output layer (in a multicategory problem, calculating the probability of each class).

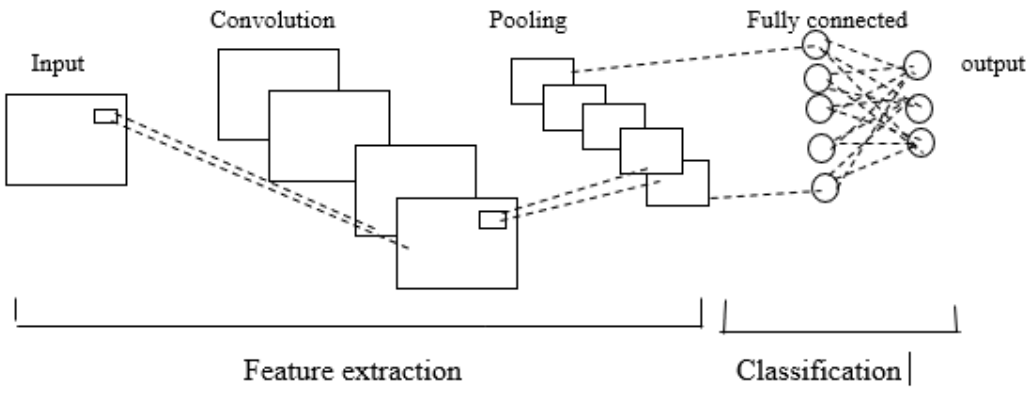


Figure 4.4: Convolutional Neural Network (CNN) Architecture for Text Classification.

CHAPTER V
METHODOLOGY

In the research community, machine learning algorithms are frequently employed to solve numerous difficulties in NLP, particularly in text classification. The benefits of this approach over knowledge engineering are high effectiveness and eas of handling in different domains with different classification issues, especially in single-label classification and, more recently, multi-label text classification, which have been widely applied to real-life problems.

On the other hand, deep learning has had far better success in earlier studies for diverse problems in natural language understanding. Because it involves very little hand engineering, it can readily take advantage of changes in the volume of available computing and data. In addition, the development of new deep neural network learning techniques and designs will speed up this process.

In this study, we have applied some traditional machine learning algorithms to handle classification problems on a small and old version of the Reuters dataset, which is Reuters-21578, and adapted recent deep learning architectures to a huge and last version of the Reuters dataset, which is RCV1-v2, in addition to the Arabic HARD dataset. In this chapter, we will describe each process, including first describing the dataset for each. Second, we will cover the experiments step by step, including the techniques that handled and manipulated the data, the descriptions of the algorithms that we have used in each dataset, and the model building that we have done for each dataset. Finally, the evaluation metrics that we have determined to assess the experiment results in our study.

5.1 TEXT CLASSIFICATION PROCESS USING STANDARD MACHINE LEARNING ALGORITHMS

The TC procedure is divided into two phases: testing and training. The key components of these stages that were used in this study are represented in Figure 5.1

- **First,** a collection of labeled documents is offered in the training phase.
- **Second,** Documents are first preprocessed to remove unnecessary and disruptive words.
- **Third,** to develop a classification model, these features must be weighted before being submitted to the classifier.
- **Fourth,** due to the high dimensionality of these data, some Dimensionality Reduction (DR) approaches were used to extract discriminant information from our dataset.
- **Finally,** the models that we used to train our dataset and do some prediction and evaluation phases.

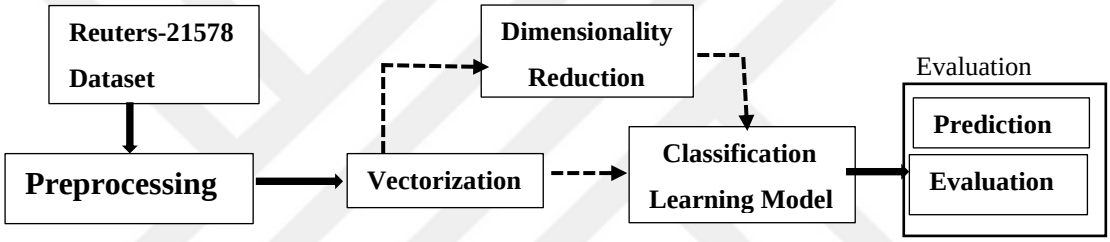


Figure 5.1: Step by Step Text Classification Process in Reuters-21578 Dataset.

5.1.1 Dataset

Reuters-21578: In 1987, the Reuters newswire was used to gather the documents in this collection. This dataset used to be a favorite among researchers who studied text categorization starting in 1996. (Now superseded by RCV1). It has been modified from this Reuters-22173 preview version and currently has 21,578 documents. There are 7,769 training texts, 90 training classes, and 3,019 testing texts in the Modified Apte ("ModApte") Split, this is the split that we have considered in our experiment[1].

[1] See Appendix-1 for sample Dataset

5.1.2 Data Preparation and Preprocessing

5.1.2.1 Text Preprocessing

Preprocessing and feature extraction are critical phases for text classification applications. At this stage, we describe methods that we have used in preparing the dataset for the classification process by cleaning texts, which remove implied noise and unneeded words such as stop words, and enable meaningful featurization. Due to this, noise and unneeded attributes can affect the performance of many algorithms, particularly probabilistic and statistical learning algorithms. The following are the techniques that have been used in this phase:

5.1.2.1.1 Tokenization

Tokenization is a processing technique that partitions a text sequence into tokens, which are sentences, words, symbols, or even other significant chunks. The primary purpose of this stage is the exploration of the words in a sentence. This is the primary purpose of this stage, as in the following statement.

he decided to sleep for another four, after sleeping for four hours.

The tokens in this case are as follows:

{“he” “decided” “to” “sleep” “for” “another” “four” “After” “sleeping” “for” “four” “hours}.

5.1.2.1.2 Stop Words

Many textual and document categorization phrases, such as "about," "a,” “above," "across," "afterwards," "again," "after," and so on, are not essential enough to be used in classification algorithms. The most typical way to handle these phrases is to exclude them from the data.

5.1.2.1.3 Capitalization

To construct a sentence, textual and document datasets have a variety of capitals. Distinguishing capitalization might be challenging when categorizing huge documents because they include multiple phrases. When dealing with inconsistent

capitalization, the most effective trick is to lower case all the letters. Which is All text and document words are combined into a single feature space in this approach.

5.1.2.1.4 Noise Removal

Many unnecessary characters, such as special characters and punctuation, can be found in most textual data sets. Although essential special characters and punctuation are required for human comprehension of texts, they can harm categorization algorithms.

5.1.2.1.5 Stemming

One word in NLP can have multiple unique variants, all of which have the same semantic information. Stemming is a mechanism for merging diverse word variants into a unified subspace. Text stemming changes words using linguistic processes such as affixation to produce alternative word forms. For instance, the stem of the word "playing" is "play."

5.1.2.1.6 Lemmatization

Lemmatization is a natural language processing approach that exchanges a word's suffix with another or removes the suffix entirely to reveal the basic word structure (lemma).

5.1.2.2 Weighted Words

Text datasets are typical examples of unstructured data. A classifier or a strategy for developing a classifier cannot directly interpret texts. These unstructured texts patterns must be turned into a structured feature set before using any mathematical modeling as part of a classifier. In our investigation, we used the widely known feature extraction methods Term Frequency-Inverse Document Frequency (TF-IDF), and Term Frequency (TF). Each document is converted into a vector including the frequency of the words in that document in all weight words techniques.

5.1.2.2.1Bag Of Words (BOW)

The bag-of-words model is the standard type of weighted word feature extraction. It is a condensed and simple representation of a text document from specific portions of the text, based on characteristics such as word frequency, where each word

is assigned a number that reflects the quantity of times that word occurs over the entire corpus and may later be used to determine the key features of the documents. The BoW method results in the creation of a list of words. Because they do not form sentences, grammar, appearance order, and semantic links between words are ignored in the collection and construction of words in a matrix. Here is an example of BoW we have picketed from the dataset we worked with:

"island telephone share split approved island telephone said previously announced twoforone common share split approved shareholders annual meeting"

Bag-of-Words (BoW) {" island", "telephone", "share", "split", "approved", "said", "previously", "announced", "twoforone", "common", "shareholders", "annual", "meeting"}

Bag-of-Feature (BoF) = {2,2,2,2,2,1,1,1,1,1,1,1,1}

5.1.2.2.2Term Frequency-Inverse Document Frequency (TF-IDF)

In [80] suggested Inverse Document Frequency (IDF) as a technique to be applied with word frequency to decrease the impact of common terms in the dataset. In a given dataset D, a word t_i and a doc $d_j \in D$, we represent the quantity of times a t_i appears in a d_j by the notation tf_{ij}. This is referred to as the term or word frequency that we have discussed above. The inverse document frequency for a term t_i is described as:

$$W(d,t) = TF(d,t) \times \log \frac{D}{df(t)} \tag{5.1}$$

where D denotes the total number of documents in the collection and df(t) is the number of documents containing the word. idf_i is equal to zero if the term t_i appears in each document in the dataset. The higher the idf_i score, the less documents the term t_i occurs in.

$Tfi_j * idf_i$ is the definition of the measurement known as the term frequency-inverse document frequency (tf-idf). It is a way of measuring how significant a term t_i is in a particular document d_j. It is a term frequency measure that gives terms that are less frequent in the corpus a higher weight. The significance of extremely common phrases will thus decrease, which may be a desirable aspect.

5.1.2.3 Dimensionality Reduction

DR is one of the biggest obstacles facing TC research nowadays. Data preprocessing operations can be slowed down by memory complexity and high time demands because text or document data collections frequently contain numerous unique terms. Using straightforward, low-cost algorithms is a common approach to this issue. These inexpensive methods, however, do not always perform as well as anticipated in specific data sets [81]. Many academics choose to use dimensionality reduction to decrease the memory complexity and time of their applications in order to avoid performance deterioration.

DR is also helpful because it helps minimize overfitting, which is the tendency for classifiers to perform well when reclassifying the training data but poorly when categorizing new data. By using DR, the overfitting could be removed even if we used a small training data set. However, there is a risk that by eliminating terms, important information about the meaning of the texts will be lost. Furthermore, DR approaches assist in removing obtrusive and unnecessary words. In addition to this, care must be taken during the reduction phase to ensure maximum cost-effectiveness [8] .There are two approaches to DR that are addressed by completely different strategies; we will discuss them individually in the following two parts.

5.1.2.3.1 Feature Selection

Term selection, sometimes referred to as term space reduction (TSR), aims to choose the terms from the initial set that are more crucial and produce the maximum effectiveness when utilized for the classification process [82]. The selection process is carried out by adapting either the filter approach or the wrapper approach. In the wrapper approach, a new word collection is created by adding or omitting words from an initial word set. A classifier is built and evaluated on a validation dataset using a newly generated term set. The word set with the greatest efficacy is picked up [83][8]. Unlike the filter strategy, which prioritizes variable ranking methods for variable selection by order. Ranking techniques are utilized because they are straightforward and have a proven track record in actual applications. The variables are scored using an appropriate ranking criterion, and variables below the threshold are eliminated. Ranking strategies are filter strategies since they are used prior to classification to clear out the less important variables [84][83]. In TC, a number of feature scoring techniques have been applied. What's more, we have applied in our experiment the simplest and

most effective one, which is document frequency, which we will discuss in the rest of this section.

5.1.2.3.1.1 Document Frequency

Document frequency is a straightforward and efficient global TSR function that ensures only the features that appear in the greatest number of documents are kept in use. For each unique word in the training corpus, we computed the document frequency, and we eliminated from the feature set any terms whose document frequency fell under a threshold limit. This appears to suggest that the words that occur most commonly in the corpus are the most valuable for TC, or, to put it another way, that infrequent keywords are either uninformative for category prediction or unimportant to overall performance. In both scenarios, eliminating unusual terms reduces the feature space's dimensionality [85][8]. Many publishers use the strategy of document frequency, eliminating any phrases that appear in less than x training examples (where x represents the threshold) and the commonly used threshold ranges from 1 to 3.

5.1.2.3.2Feature Extraction

Term extraction aims to produce a collection of "synthetic" terms that are as effective as possible from the original set. The reason for utilizing synthetic words is that the original word set might not have the best dimensionality for representing document content because of the widespread issues with polysemy, homonymy, and synonymy. Additionally, The new representation's dimensions can be thought of as a linear combination of the old ones [8]. Many methods have been adapted to TC in order to perform DR using feature extraction. Among of them is the most important and effective method that have been used by researches which is PCA that we will discuss in the rest of this section.

5.1.2.3.2.1 Principle Component Analysis

The most common method for multivariate analysis that can be used as a preparing technique to minimize the dimensions of a data before applying learning algorithm to it is principal component analysis (PCA)[86]. PCA reduces dimensionality by embedding data into a linear subspace with fewer dimensions and determining a subspace in which the data basically fits. To "preserve as much variety

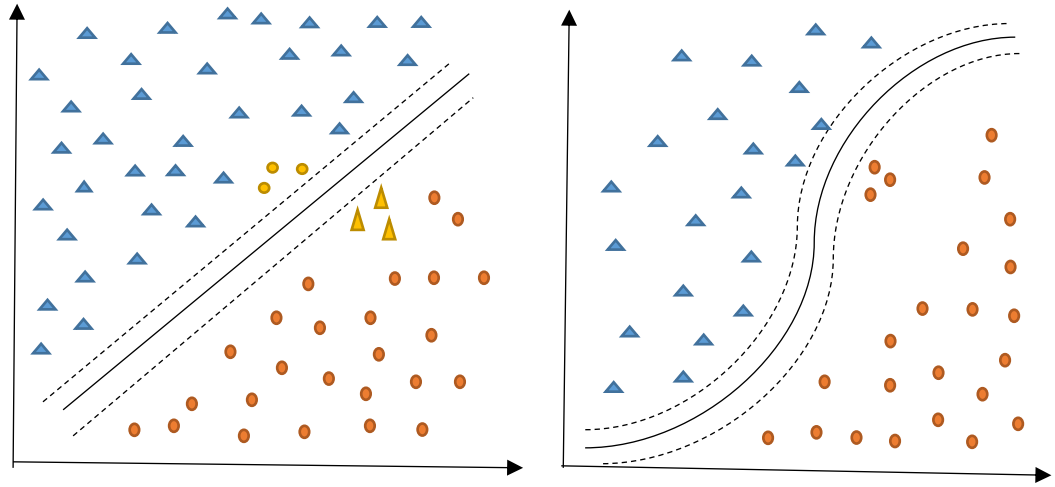
as possible," this entails identifying new variables that are uncorrelated and optimizing variance[87][88]. PCA is a useful tool for reducing noise, redundancy, word ambiguity, and dataset ambiguity, as well as for avoiding the over-fitting issue[89].

5.1.3 Classification Algorithms

Choosing the best classifier is the most vital phase of the process for text classification. In this section we address the traditional machine learning techniques that are applied to classify the Reuters-21578 dataset, including the most common approaches: Naïve Bayes (NB), Decision Tree (DT), Support Vector Machine (SVM) and Random Forest (RF).

5.1.3.1 Support Vector Machine (SVM)

SVM was originally proposed by [90]. SVM was designed for binary classification. SVM is a regression and classification prediction tool that employs machine learning theory to maximize estimated accuracy while automatically preventing over-fitting. When the input is linear, SVM basically generates a hyper-plane that generates a boundary between the various types of data in two dimensions. SVM is used to plot each dataset element in an N-dimensional space, where N referred to the total number of characteristics in entire corpora. The best hyperplane to partition the data can be identified by maximizing the margins between the data points (support points) in the training dataset. As a result, one benefit of increasing the margin is better empirical performance. Another defense of this is that even if we place the boundary slightly incorrectly, the likelihood of misclassification is decreased. Additional benefits would include improved classification and avoiding local minima. [91] . A hyperplane is used to divide the data, and a kernel approach is used to extend this to non-linear boundaries, according to the goals of SVM. The input data is non-linearly mapped to a high-dimensional space using this kernel. As a result, the new mapping can be separated linearly. This mapping is produced by the kernel by transforming the input data into a feature space that permits the construction of a similarity metric using the dot product. The Figure 5.2 shows the non-linear and linear classifier which is used for $2-$ dimension datasets. On the other hand, many scholars work on multi-class issues using this common strategy [92], which is the way that we have applied the SVM to our news dataset.



A) Linear Separation **B) Non-Linear Separation**

Figure 5.2: This figure shows the non-linear and linear Support Vector Machine (SVM) for a 2 Dimensions dataset.

5.1.3.2 Naïve Bayas (NB)

The Naive Bayes classifier has been a well-known probabilistic machine learning method for categorizing documents since the 1950s. The theoretical foundation of the Naive Bayes classifier is Thomas Bayles' Bayes theorem [93][94], which is described as follows.

$$p(y|x) = p(y)p(x|y)/P(x) \tag{5.2}$$

The fundamental difference between the theorem and naive bayes is that the former assumes that features are independent of one another and that there is no relationship between characteristics that belong to the same class. By using the details in sample data, Naive Bayes provides a method for calculating the likelihood function p(x|y) of each class y given an item x. Once we have these estimates, we may use them to categorize data. Due to this assumption, attribute-value data is entitled to the following equation:

$$p(x|y) = \prod_{i=1}^{n} p(x_i|y) \tag{5.3}$$

Where x_i is the value of the i th attribute in x, and n is the number of attributes.

$$p(x) = \prod_{i=1}^{k} p(C_i)p(x|C_i) \tag{5.4}$$

Where k is the number of classes and c_i is the i th class.

There are two different iterations of naive Bayes that are frequently used for text classification. [95] The multi-variate Bernoulli model, which employs naive Bayes and represents each document as a vector of binary variables, each of which indicates whether a given word is present or absent. However, only the words that are included in a document are taken into consideration while calculating its probability.

In contrast, the multinomial model that we have applied to news document classification in our experiment. In this kind of model, data on how frequently a term appears in a document is used. Every time a word appears in a document, it is treated as a separate event. These incidences are thought to be unconnected to one another. The probability of each word event given to the class is added to determine the chance of a document given to the class.

5.1.3.3 Decision Tree (DT)

A decision tree is a text and data mining categorization technique that was first proposed by [96] and developed by [97]. A decision tree (DT) text classification algorithm is a tree in which the central nodes are labels for terms, the leaves are labels for categories, and the branches branching off from them are labels for tests on the weight the term has in the test document. When a leaf node is reached, the label of this node is subsequently assigned to d_j This classifier categorizes a test document d_j by repeatedly checking the weights when the words labeling the internal nodes have in vector d_j.

A "divide and conquer" approach could be used to learn a DT for category ci. This approach involves first determining if all training samples have the same label (either c_i or $\bar{c_i}$; second, if not, choosing a term (t_k), partitioning Tr into classes of documents that have the same value for t_k; and finally, placing each class in a separate subtree. Recursively repeating the process on the subtrees until each leaf of the resulting tree has training samples allocated to the same category c_i results in the label for the leaf being determined. The most important stage is selecting the term t_k on which to run the partition; this decision is typically made using an information gain or entropy criterion.

5.1.3.4 Random Forest (RF)

The Random Forest technique is a tree-based method that increases predicted accuracy by bootstrapping aggregation and randomly predicting. This approach, which utilized t tree as parallel, was introduced by[98] in 1995. As shown in Figure 5.3 random forest is a combined approach of decision trees that created using a dataset that has been randomly divided and is dependent on the divide-and-conquer strategy. The single decision trees are created by using a feature selection indicator, such as information gain, the Gini index and the gain ratio for each feature. Each tree is dependent on a different random sample. Each tree in a classification problem casts a vote, and the most popular class is ultimately selected as the solution. In a regression case, the final result is assumed to be the mean of all tree outputs. Compared to other non-linear classification techniques, it is more straightforward and potent.

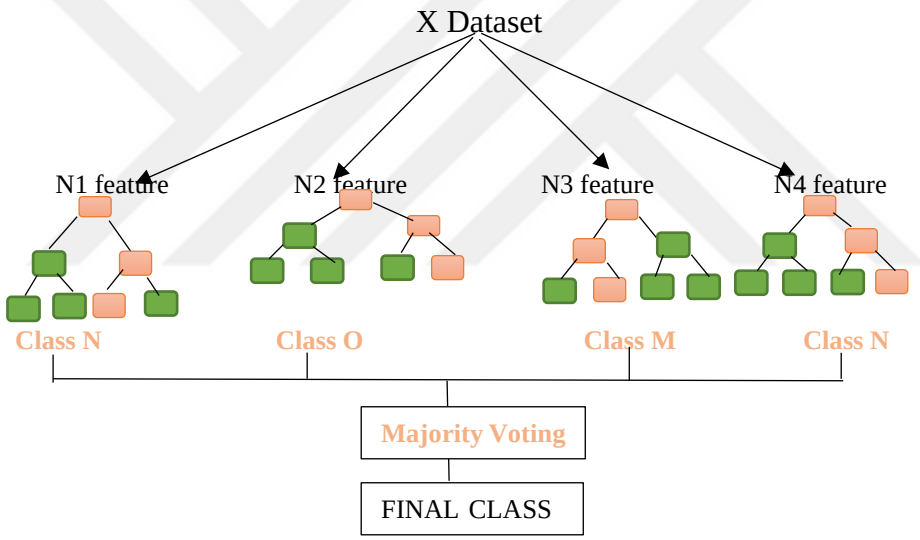


Figure 5.3: Random Forest Architecture.

5.1.4 Advantages and Disadvantages of the Classification Algorithms.

Table 5.1: Advantages and Disadvantages of Support Vector Machines (SVM).

Advantages	**Disadvantages**
• The kernel represents a non-linear modification that allows data to be linearly separable. • SVM effectively generalizes the new samples. • The SVM algorithm can generate a singular result when	• findings that are opaque due to a large number of dimensions. • SVM might not be able to display the companies' score as a parametric function based on financial ratios due to the possibility of very high dimensions.

Table 5.1 Continued

the optimality is rounded. This is a key distinction between SVM and neural networks, which generate many solutions based on local minima and are hence unreliable across samples. • Choosing the right kernel would lead to a focus on similarities between similar companies, simply because the more similar two or more companies are, the higher their worth	• Each company has a distinct weight through the use of a Gaussian kernel that is dependent on the variance between its own financial ratio and the support vector ratios of the existing sample data.

Table 5.2: Advantages and Disadvantages of Naïve Base (NB).

Advantages	**Disadvantages**
• It requires very little storage space for both training and classification and trains extremely quickly. • It is simple to implement and does not have many parameters, unlike Support Vector Machines and Neural Networks. • NB categorization is highly transparent and incorporates data from a variety of characteristics to make the final prediction. • NB is completely resistant to missing data because they are simply discarded while calculating probabilities and don't affect the outcome. • NB has gained a lot of interest from researchers since it is resistant to noise and irrelevant features.	• NBC strongly infers the shape of the distribution of the data. • NBC is similarly constrained by the lack of data; therefore, any potential value in feature space must have a likelihood value assessed by a frequentist. • The issue with NBC is that it does not perform well with data sets with unbalanced classes. • The naive Bayes model makes the rarely true assumption that all predictors (or characteristics) are independent. This reduces the algorithm's usefulness in a practical usage case.

Table 5.3: Advantages and Disadvantages of Decision Tree (DT).

Advantages	**Disadvantages**
• It is simple to implement. • DT creates models that are simple to understand.	• The decision tree is a highly efficient learning and prediction algorithm, but it is also quite sensitive to even the smallest

Table 5.3 Continued

Advantages	Disadvantages
• It can process both categorical and continuous data. • It can handle noise and properties with missing value.	changes in the data and is prone to overfitting. • Validation techniques and pruning can counteract these impacts, although this is a gray area. • This approach also has issues with extrapolating results outside of the sample.

Table 5.4: Advantages and Disadvantages of Random Forest (RF).

Advantages	Disadvantages
• It detects outliers and abnormalities in well-informed data. • It represents one of the most precise learning algorithms. It creates classifiers that are incredibly accurate for numerous datasets. • It provides an estimate of the crucial classification-related variables.	• In contrast to other techniques like deep learning, random forests are highly quick to train on text data sets, but once trained, they produce predictions quite slowly. • Reducing the number of trees in the forest is necessary to create a faster structure since more trees in the forest make the prediction step more time-consuming.

5.1.5 Model Construction

This section illustrates the structure of the traditional machine learning models selected for the experiment. In this study, we attempt to attain good results by using our dataset in two different ways and comparing the results with the literature. As we have described above, Reuters-21578 has 90 categories. First, we used the entire dataset with all the classes, and second, we used the top ten categories in the dataset, and we applied all the techniques that we have described earlier on both.

In our experiment, we primarily concentrated on making comparisons by utilizing multiple techniques during the stages of data preparation and preprocessing. For instance, when cleaning data, we used lemmatization and stemming. Additionally, we used two different word-weighting techniques, including lemmatization and stemming: term frequency-inverse document frequency (tf-idf) and bag of words (Bow). These methods have all been combined with dimensionality reduction

techniques using the two separate ways we stated earlier in this chapter: document frequency (df) and principal component analysis (PCA).

5.2 TEXT CLASSIFICATION PROCESS BY USING PRE-TRAINED BERT MODELS.

In this phase, we describe the structure of the second experiment that applies advanced deep learning approaches to datasets of different types of languages by using pre-trained Bert models with different versions that have recently achieved successful results and outperform multiple traditional models. We first describe the datasets; second, the pre-trained model that was selected; and finally, we explain and highlight the mechanisms and parameters used in training the selected model. The key components of this methodology that were used in this study are represented in Figure 5.4, Figure 5.5 and Figure 5.6.

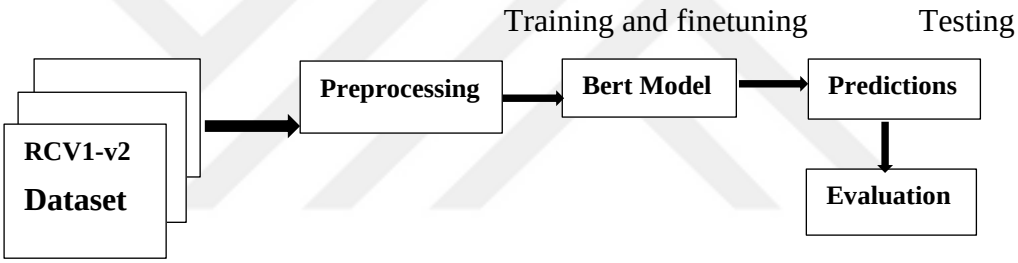


Figure 5.4: Step by Step Text Classification Process in RCV1-v2 Dataset.

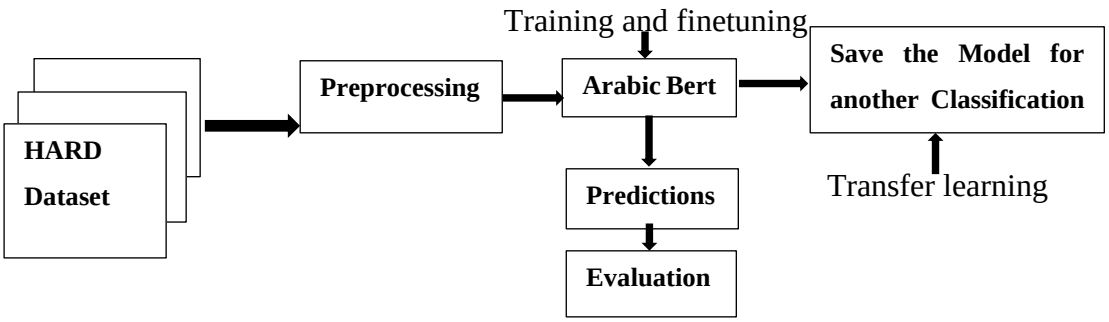


Figure 5.5: Step by Step Text Classification Process in HARD Dataset.

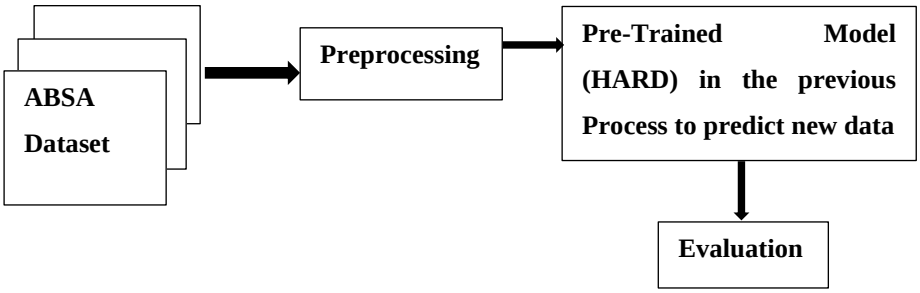


Figure 5.6. Step by Step of prediction ABSA Dataset by using transferred learned model.

5.2.1 Datasets

5.2.1.1 RCV1-v2 Dataset

RCV1-v2:[99] is a collection of more than 800,000 newswire articles that have been carefully categorized and recently made available for research by Reuters Ltd. The RCV1-v2 text classification test collection is created by modifying the problems in the raw RCV1 data. There is a train set (document IDs 2286 to 26150) and a test set (document IDs 26151 to 810596), just like in the LYRL2004 train/test split. We only take into account the 103 topic categories as labels[2].

5.2.1.2 HARD Dataset

The HARD dataset: [46] is a compilation of Arabic hotel reviews collected from the website Booking.com, which focuses on online accommodation reservations. The dataset originally contained 981143 reviews. Positive and negative opinions were included in each review. We combined the positive and negative opinions for each hotel into one review, yielding a total of 49,907 reviews. The dataset initially included reviews with both Arabic and Latin text. The reviews are decreased to 373772, which represents the entirety of the imbalanced HARD dataset after the reviews have been cleaned and the Latin content has been removed. The balanced subset dataset, however, includes 94052 reviews, with 47084 negative and 46968 positive reviews.

5.2.1.3 ABSA Dataset

After We ran our experiment on the HARD Arabic dataset and received excellent results, so we saved the model and used it to make some predictions on another Arabic hotel review dataset, which was proposed at SemEval-2016 in support

[2] See Appendix-1 for sample Dataset

of ABSA's multiple languages task, which involved work in eight languages and seven domains [100]. The dataset contains 4802 testing tuples and 19,226 training tuples. The dataset was annotated using the XML schema. The dataset is f sentence pair input which is made up of a collection of reviews, each of which has a set of sentences, each of which contains three tuples: OTE, aspect-category and aspect polarity. Because our model was trained on a dataset with only one sentence review and aspect polarity, we have pressed it to go with our model[3].

5.2.1.4 Data Preparation and Preprocessing

The RCV1-v2 dataset is unbalanced, and we have applied our experiment to the original data split in addition to splitting the training data part into validation and training. Additionally, the dataset has been cleaned and preprocessed, which is the same process that we have done in the previous model and dataset by removing unwanted characters, numbers, and punctuation in the whole dataset, so that would not affect our model.

However, the HARD data set is balanced, and we divided it into training, testing and validation subsets using the ratios of 70%, 20% and 10%, respectively. The Arabic language has a different preprocessing from English due to its complex and varied morphology. The performance of our classifiers could be seriously hindered by the unstructured format of reviews. So, we go through a number of steps in our preprocessing procedure in order to preserve sentiment hints. For instance, omitting punctuation marks before extracting emoticons can harm the effectiveness of the process. Preprocessing and normalizing the reviews is crucial in order to remove redundant and meaningless data. Here, we outline the preprocessing techniques we used in our study, such as word normalization, to make the text as uniform as feasible.

All Arabic punctuations, including the comma, semicolons, and question marks, are deleted during word normalization, along with the tokenization of dates and digits. Furthermore, the same terms that have different spellings must be normalized. This issue is considerably more common in Arabic because of Tashkil, which includes diacritical marks, Hamzah, and Maddah letters. As a result, the various iterations of the same thing must be combined into one. For instance, here are four typical Arabic spellings of the word "felt": احسست, أحسستِ, أحسستَ, أحسست.

[3] See Appendix-1 for sample Dataset

Additionally, we eliminate numerous occurrences of a single character, which are typically inserted by users to emphasize an idea. For instance, in English, if someone wanted to emphasis how icy the weather is, he can excessively redundant the letters of the emphasized syllable: verrrrrrrrrrrrrrry(جدددددددا). In contrast, it is especially important to avoid removing legitimate repeating characters because there are words that originally had repeated letters.

5.2.2 Pretrained Deep Learning Models for Text Classification

Pre-trained language models [101] considerably improve NLP tasks, such as text categorization, by efficiently learning global semantic representation. It typically uses unsupervised techniques to automatically mine semantic information before creating pre-training objectives so the machines could learn to recognize semantics. The Transformer is a unique pre-training task for the current PTMs, which are typically trained with broader scale-corpora or more powerful, or deeper architectures.

5.2.2.1 Transformers

Transformer is a well-known deep learning model that has been commonly used in a variety of areas, including speech recognition, computer vision and natural language processing (NLP). The Transformer was first introduced as a machine translation sequential model in [102]. According to the "attention is all you need" paper, the Transformer is the initial transduction model, using self-attention to generate representations of its input and output instead of using convolution or sequence-aligned RNNs.

The Transformer is a sequential model including an encoder, a decoder, and a core block with " a feed-forward network and an attention " repeated N times in each. The Transformer architecture can generally be applied in three main ways:

- Encoder-Decoder. Sequence-to-sequence modeling frequently employs the complete Transformer architecture (e.g., neural machine translation).
- Only an encoder. Only one such encoder is utilized, and the encoder's outputs are used to represent the input sequence. Typically, classification or sequence labeling issues are addressed using this.
- Only a decoder. The encoder-decoder cross-attention module is likewise deleted, leaving only the decoder in use. Common applications for this include language modeling and sequence generating.

5.2.2.2 BERT

Recent studies have demonstrated the value of pre-trained models on large corpora for text classification and other NLP applications. These models are also highly effective at world language representation by integrating a huge unlabeled data, thereby avoiding the need to train new models from scratch. The Bert-based model is one of the best pre-trained model types. Bert has been introduced by [39] and is built on a multi-layer bidirectional transformer. In contrast to earlier language representation models, Bert is trained in all layers by focusing on both right and left context. As opposed to earlier context-free models, which produced a single representation of the word embedding for each word in the lexicon, BERT takes into consideration the context for each instance of a given word, resulting in a contextualized embedding that is unique for each sentence.

The Bert structure is divided into two stages: fine-tuning and pre-training. During the pre-training procedure, the model is trained on unsupervised text data using a variety of pre-training tasks, which are masked LM and predictions of the next sentence. In the fine-tuning phase, the BERT model is initially set up with the parameters of pre-training stage, and all those parameters are adjusted using classified data from the downstream tasks. The architectural design of the BERT-base model includes an encoder with 768 hidden size, 12 self-attention heads and 12 transformer blocks. BERT produces a sequence representation from the input sequence that is up to 512 tokens. The sequence is made up of more than one segments, the first of which is [CLS], which represents the special classification embedding. [SEP] is the sequence's second token, used to demarcate segments.

BERT employs the first token's [CLS] final hidden state as a representation of the entire sequence for tasks involving text classification. To estimate the likelihood of label c, a simple SoftMax classifier is implemented on top of BERT.

$$\rho(c|h) = softmax(Wh) \tag{5.5}$$

Where W is the task-specific parameter matrix. We jointly tweak all the parameters from BERT and W by maximizing the likelihood of the proper label.

5.2.2.3 Arabic Bert

Because there was no pre-trained BERT version for Arabic [49], created four Arabic BERT language models from scratch and made them available to a broader audience. ArabicBERT is a family of BERT language models made up of four varied sizes models: base, large, mini and medium that were trained utilizing masked language modeling. Those different models were trained on the same data for 4 million steps using a vocabulary set of 32,000 words. They were created using a corpus made up of OSCAR data and recent Wikipedia data, totaling 8.2 billion words. Additionally, there is no uncased and cased version of the model for Arabic characters, and they do not have lower or upper case. The corpus is therefore not limited to just standard Arabic but also includes some dialectical (spoken) Arabic, which improved the performance of the models when using data from social media platforms.

5.2.3 Design Of Downstream Model

5.2.3.1 Model Parameters

While training the model, we have used an early stopping callback function to prevent it from overfitting and to enable us to specify the performance measure to observe by keeping track of the loss function on the validation data with a training epoch number that we have determined to be equal to 50. When the model's performance stops optimizing on a validation dataset rather than being run through all epochs, the training process is stopped. Early stopping, hence, prevents overfitting while also significantly reducing the training process duration by employing fewer epochs. Additionally, we repeated the training process multiple times, and each time we saved the best weights based on the validation loss and loaded them during prediction and in another training process to obtain high performance from the model.

Word embeddings from the initial token [CLS], which was the final hidden vector (C) of the BERT model, are passed into a simple linear layer with dropout layer to perform sentiment and news classification. The bias term b is added to the weight W (the trained parameters) matrices and the input h^l_t multiplications to create a linear transformation of their input features to output features. The probability of each category P is then calculated using the softmax or sigmoid function. Furthermore, it has been found that the BERT-Base model, which has 12 attention heads, 12 hidden layers and a hidden size of 768, performs best with fixed-length sequences. So, we chose 400, 100, and 60 tokens as the maximum length, using a straightforward

technique. The Adam optimizer was applied to fine-tune the model on our downstream task with a learning rate of 2e-5, a dropout rate of 0.3, and a batch size of 32,16, as recommended in the recent research and original paper.

5.2.3.1.1 Transfer Learning

Transfer learning is a deep learning (DL) research issue that focuses on preserving information learned from addressing one problem and using it to address a separate but connected challenge. That is what we have used in our study, as described in Figure 5.4, by using Arabic Bert to train the model that we have constructed with one linear layer on top of it by applying it to the Arabic hotel reviews dataset. After that, the model achieved superior results, so we saved it (transfer learning) and used it to predict another Arabic hotel review dataset without a training phase.

5.2.3.2 Activation Function: Softmax and Sigmoid

In neural networks, activation functions are widely employed. They perform a number of crucial tasks for a neural network. The main goal of using an activation function is to generate non-linearity in the neural network.

The sigmoid function is additionally known as the logistic function. Its non-linearity, continuous differentiability, and smooth S-shaped function make it the most popular activation function, translating each value of x into the range [0, 1]. The largest input is extremely closely mapped to 1, whereas the smallest input is very closely mapped to 0. Furthermore, because the sigmoid function is not symmetric around zero, all of the neurons' output values will have the same sign. By scaling the sigmoid function, this problem can be resolved. Usually, it can perform well if the numbers in the training data are all positive. The Equation 5.7 below demonstrates how the function can be mathematically represented.

$$f(x) = \frac{1}{(1 + e^{-x})} \tag{5.6}$$

The softmax function is a sigmoid function cocktail. Given that we are aware that sigmoid functions return values in the interval 0 to 1, we may think of these values as the likelihood of data points that belonging to a specific class. The softmax, in contrast to the sigmoid function, that is utilized for binary classification, can be

employed for multiclass classification issues. The function returns the probability for every data point across all distinct classifications. It can be expressed mathematically as Equation 5.8:

$$softmax(xi) = \frac{e^{x}i}{\sum_{j=1}^{k} e^{xj}} \tag{5.7}$$

When we create a network for multiple class classification, the output will contain the same number of neurons as the target classes. The output of a single neuron relies on the other output neurons when softmax is the activation function.

5.2.3.3 Adam Optimizer

In [103]The Adam optimizer was introduced and its name originally came from adaptive moment estimation. Adam is an effective stochastic optimization technique that uses only first-order gradients and little memory. The Adam optimizer was created to combine the benefits of two commonly mentioned optimizers: RMSProp, which performs well in non-stationary and on-line environments, and AdaGrad, which works well with sparse gradients. The benefits of the Adam Optimizer include: suitable for issues with plenty of data and parameters, it is computationally efficient, memory-light, and it is invariant to diagonal rescaling of gradients.

5.3 EVALUATION MATRICES

The two criteria that are most frequently used to evaluate text categorization models are accuracy and the F1 score. Later, with the existence of some specific tasks or the growing difficulty of classification tasks, the assessment metrics are enhanced. When evaluating the performance of multi-label text classification, for instance, evaluation metrics like Macro-F1 and Micro-F1 are employed, which are different from the evaluated matrices that are used in single-label and multi-class classification. The number of category labels used in multi-label text classification matrices can vary, and the text is split into many categories.

5.3.1 Accuracy

Accuracy is one of the major measures that are used to evaluate the efficacy of a classification model based on a confusion matrix. It can be measured as the

percentage of right estimates out of all estimates [104]. False positive, true positive, false negative, and true negative are denoted by FP, TP, FN and TN respectively. Equation 5.9 below defines the classification "accuracy".

$$Accuracy = \frac{(Tp+TN)}{(TP+FP+TN+FN)} \tag{5.8}$$

5.3.2 Micro Avraged-F1

As mentioned at the beginning of this section, the F-measure or F-score is a well-known and commonly used tool for evaluating experiments in the text classification domain. Regardless of the class label or mistake rate, f-measure is a critical parameter for imbalanced test sets. For example, the majority of test samples have a class label. F1 is the harmonic mean of Recall and Precision. The greatest value of an F-score is 1.0, which represents excellent recall and precision, while the minimum value is 0 if either recall or precision is zero. Precision, Recall, and F1 as defined Equation (5.10), Equation (5.11), Equation (5.12) respectively.

$$\text{Precision} = \frac{TP}{(TP + FP)} \tag{5.9}$$

$$Recall = \frac{TP}{(TP + FN)} \tag{5.10}$$

$$F1 = 2 * \frac{precision*recall}{(precsion+recall)} \tag{5.11}$$

We calculated two averaged multi-label versions of each measure: the micro-average and the macro-average. We chose the micro-F1 as our primary metric since it gives the weights equally to each document as a result, and it is regarded as a per document average, so it is influenced by the performance of frequent categories. Because the Micro-F1 score sets an equal weights to each observation, when the classes are imbalanced, the classes with more observations have a greater influence on the final result [105 : 142-150]. As a result, the final score hides the performance of the minority classes while exaggerating the majority. Micro-F1 takes into consideration the quantity of categories, making it suited for unbalanced data. Micro-

F1 is described as Equation (5.13), micro averaged precision in Equation (5.14) and micro averaged recall in Equation (5.15):

$$Micro\ avrage - f1 = \frac{2Pt*Rt}{P+R} \tag{5.12}$$

Where:

$$\text{Micro average precision } = \frac{\sum t\epsilon s\ TPt}{\sum t\epsilon s\ TPt+FPt} \tag{5.13}$$

$$\text{Micro average recall } = \frac{\sum t\epsilon s\ TPt}{\sum t\epsilon s\ TPt+FNt} \tag{5.14}$$

5.3.3 Macro Avraged-F1

The macro-average score [105: 142-150] distributes the equivalent weights to each category without taking frequency into account, making it a per-category average. This means that a majority and a minority will both contribute equally. Formally, Macro-F1 is defined as the following Equation (5.16), and the Equations (5.17) (5.18) describe Macro average precision and Macro average recall respectively, where S denotes a set of labels.

$$\text{Macro average} - \text{F1} = \frac{1}{S}\sum_{t\epsilon S}\frac{2Pt\times Rt}{Pt+Rt} \tag{5.15}$$

Where:

$$\text{Macro average precision } = \frac{TPt}{TPt+FPt} \tag{5.16}$$

$$\text{Macro average recall } = \frac{TPt}{TPt+FNt} \tag{5.17}$$

MacroF1 computes the average F1 of all labels and disregards the quantity of data that handles each label equally. As a result, it is easily influenced by labels with great recall and precision. When there are a large or extremely large number of categories.

CHAPTER VI
EXPERIMENTS

This section contains the essential configuration settings as well as the results of the experiments for this thesis.

6.1 EXPERİMENTAL SETUP

In our experiment, we used Jupyter Notebook through Anaconda to set up our environment, which is supported by the hugging face libraries that were used to construct and train the networks. Also, the scikit-learn package was utilized to develop and train the standard algorithms. Therefore, the experiments were carried out on a GPU to accelerate the training process as well as a CPU, which ran on the Python-3 programming language. Furthermore, we used the Google-Colab Cloud Service to run some standard machine learning models with dimensionality reduction techniques. As show in table 5.1 experiments system specification settings in jupyter notebook and colab. The code and experiments for these studies can be found in a GitHub repository[4].

Table 6.1: Experiments System Specification Settings in Jupyter Notebook and Colab

	SPECIFICATION
GPU	NVIDIA GeForce GTX 1660 SUPER
CPU	11th Gen Intel(R) Core(TM) i7-11700KF @ 3.60GHz 3.60 GHz
RAM	12.68 GB Available
Disk	78.17 GB Available

6.2 EXPERİMENTAL RESULTS

In these experiments, we test our models using the macro-F1 measure, the micro-F1 measure, and the accuracy metrics discussed in Section 5.3. We also contrast and compare our results with literature that we described in Section 3.

[4] https://github.com/HudaAlfigi/Text-Classification

Table 6.2: Experimental Results of Standard Machine Learning Models

Dataset	Feature Extraction	Dimension Reduction	Classifier	Micro-F1	Macro-F1	Accuracy
Reuters-21578	TF-IDF	DF =3	RF DT SVM(OVR) SVM(LP) NB	0.783 0.747 0.856 0.844 0.771	0.193 0.335 0.407 0.438 0.150	0.711 0.721 0.792 0.825 0.688
Reuters-21578	TF-IDF & Stemming	DF =3	RF DT SVM(OVR) SVM(LP) NB	0.779 0.755 0.859 0.851 0.771	0.185 0.322 0.397 0.460 0.144	0.708 0.727 0.800 0.837 0.687
Reuters21578-Top_10Cat	TF-IDF & Lemmatization	DF = 3	RF DT SVM(OVR) SVM(LP) NB	0.839 0.825 0.904 0.898 0.798	0.132 0.199 0.232 0.240 0.073	0.787 0.802 0.866 0.878 0.739
Reuters-21578	TF-IDF & Lemmatization	DF =3	RF DT SVM(OVR) SVM(LP) NB	0.748 0.765 0.868 0.865 0.656	0.166 0.352 0.470 0.533 0.049	0.671 0.739 0.813 0.847 0.572
Reuters-21578	BoW	DF =3	RF DT SVM(OVR) SVM(LP) NB	0.784 0.758 0.816 0.794 0.430	0.199 0.339 0.348 0.381 0.188	0.712 0.730 0.740 0.792 0.544
Reuters-21578	BoW & stemming	DF =3	RF DT SVM(OVR) SVM(LP) NB	0.776 0.754 0.821 0.793 0.446	0.189 0.319 0.347 0.385 0.196	0.702 0.732 0.745 0.789 0.546
Reuters-21578	BoW & Lemmatization	DF =3	RF DT SVM(OVR) SVM(LP) NB	0.779 0.758 0.819 0.799 0.437	0.192 0.328 0.351 0.397 0.191	0.706 0.731 0.743 0.794 0.545
Reuters-21578	TF-IDF	PCA=0.95	RF DT SVM(OVR) SVM(LP)	0.560 0.697 0.864 0.861	0.048 0.236 0.444 0.532	0.472 0.692 0.806 0.843

Table 6.2 Continued

Reuters-21578	TF-IDF & Stemming	PCA=0.95	RF DT SVM(OVR) SVM(LP)	0.553 0.690 0.866 0.865	0.052 0.243 0.461 0.539	0.461 0.674 0.812 0.845
Reuters-21578	TF-IDF& Lemmatization	PCA=0.95	RF DT SVM(OVR) SVM(LP)	0.565 0.687 0.866 0.862	0.046 0.242 0.464 0.528	0.477 0.678 0.810 0.844

Table 6.3: Experimental Results of BERT Transformer Models.

Dataset	**Classifier**	**Micro-F1**	**Macro-F1**	**Accuracy**
Reuters-21578	BERT	0.96	0.88	0.84
RCV1-v2	BERT	0.86	0.57	0.616
Arabic-HARD	BERT	0.949	0.95	0.949
Arabic-ABSA	BERT	0.910	0.910	0.910

Table 6.4: Experimental Results of the same Datasets in the Literatures.

Dataset	**Classifier**	**Micro-F1**	**Macro-F1**	**Accuracy**	**F1-Measure**
Reuters-21578[31]	SVM KNN LSF NNet NB	0.8599 0.8567 0.8498 0.8287 0.7956	0.5251 0.5242 0.5008 0.3765 0.3886	-- -- -- -- --	-- -- -- -- --
Reuters-21578[32]	NB ROCCHIO C4.5 K-NN SVM	0.720 0.799 0.794 0.823 0.864	-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
Reuters-21578[26]	BR CC ML-kNN ML-HARAM CNN-RNN	0.878 0.879 0.595 0.762 0.855	0.385 0.395 0.239 0.237 0.322	-- -- -- -- --	-- -- -- -- --
RCV1-v2[26]	BR CC ML-kNN ML-HARAM CNN-RNN	0.853 0.847 -- -- 0.849	0.687 0.693 -- -- 0.712	-- -- -- -- --	-- -- -- -- --
Arabic-HARD [46]	LGR Passive - Aggressive SVM Perceptron	-- -- -- -- --	-- -- -- -- --	-- -- -- -- --	0.94 0.92 0.94 0.92

Table 6.4 Continued

	RF AdaBoost	-- --	-- --	0.94 0.92	
Arabic-ABSA [47]	BERT-Linear-single BERT-Linear-pair	-- -- -- --	-- -- -- --	85.93 89.51	-- -- -- --

Table 6.5: The Best Experimental Results of our Experiments and the Literatures**.**

		Our Experimental Results			**Literatures Results**			
Dataset	**Classifiers**	**Mi-F1**	**MaF1**	**Acc**	**Mi-F1**	**MaF1**	**Acc**	**F1Score**
Reuters-21578[31]	SVM KNN LSF NNET NB	0.904 -- -- -- 0.798	0.232 -- -- -- 0.073	0.866 -- -- -- 0.739	0.8599 0.8567 0.8498 0.8287 0.7956	0.5251 0.5242 0.5008 0.3765 0.3886	-- -- -- -- --	-- -- -- -- --
Reuters-21578[32]	NB ROCCHO C4.5 K-NN SVM				0.720 0.799 0.794 0.823 0.864	-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
Reuters-21578[26]	BR CC ML-kNN ML-HARAM CNN-RNN	-- -- -- -- --	-- -- -- -- --	-- -- -- -- --	0.878 0.879 0.595 0.762 0.855	0.385 0.395 0.239 0.237 0.322	-- -- -- -- --	-- -- -- -- --
RCV1-v2[26]	BR CC ML-kNN ML-HARAM CNN-RNN	-- -- -- -- --	-- -- -- -- --	-- -- -- -- --	0.853 0.847 -- -- 0.849	0.687 0.693 -- -- 0.712	-- -- -- -- --	-- -- -- -- --
Arabic-HARD [46]	LGR Passive-Agg SVM Perceptron RF AdaBoost	-- -- 0.898 -- 0.839 --	-- -- 0.240 -- 0.132 --	-- -- 0.878 -- 0.787 --	-- -- -- -- -- --	-- -- -- -- -- --	-- -- -- -- -- --	0.94 0.92 0.94 0.92 0.91 0.87

Table 6.5 Continued

Arabic-ABSA [47]	Arabic-BERT-Linear-single Arabic-BERT-Linear-pair	0.910 --	0.910 --	0.910 --	-- --	-- --	-- --	85.93 89.51
Reuters-21578	BERT	0.96	0.88	0.84	--	--	--	--
RCV1-v2	BERT	0.86	0.57	0.616	--	--	--	--
Arabic-HARD	Arabic-BERT	0.949	0.95	0.949	--	--	--	--

Throughout our results in multi-label datasets, we will consider micro-f1 because, as we described in the previous section, an averaged micro is the most appropriate measure due to the high score, and Micro-F1 takes into consideration the number of categories that have a high frequency, making it suited for unbalanced data distribution. While Macro-F1 distributes equal weights to each label without taking frequency into account, that is the reason for the low score in the total accuracy score. Additionally, they have been used by different literature[31][32][26].

We have noticed from Table 6.2 that the subset of the top ten categories has achieved higher scores compared with the entire dataset. While the SVM has achieved the highest scores among all the models with all feature extraction techniques, the NB has gotten the lowest scores. Furthermore, within all the feature extraction techniques, we observe that TF-IDF and lemmatization obtained the highest accuracy compared with other techniques. PCA is a very important method in machine learning Whereas in text data, as we can see from the finding, it has achieved very low scores if we compare it with document frequency.

Additionally, as all the studies and literature showed, transformers outperformed all the previous techniques and traditional models in the text classification domain. As we can see from the finding above in Table 6.3, BERT outperformed all used algorithms in these experiments, in which we used two versions of BERT: the original version with news datasets and Arabic BERT that we used with an Arabic dataset. The comparison between original BERT and Arabic BERT was in favor of the Arabic version, which achieved 95% of the highest score among all the

results in the study. Lastly, while we built the Arabic model by adding a dense layer on top of the last layer of the Bert model and training and testing the model on the Arabic dataset, after obtaining a high percentage of accuracy, we saved the model and used it to predict another Arabic hotel review dataset and achieved a 91% score.

CHAPTER VII
CONCLUSIONS AND FUTURE WORK

In this study, we have explored the task of automated text classification, where document instances are classified into pre-determined classes. We address two different tasks: multi-label classification and single-label classification, by comparing the accuracy scores of standard machine learning models that have been applied with variations in feature extraction and dimensionality reduction techniques to the small multi-label Reuters21578 dataset. We also investigated the capabilities of contextually word embedding from the BERT model with input from the Arabic BERT task as well as RCV1-v2 news input from the original base BERT. Finally, we compared our findings to those in the literature. In the first approach, we built different models on a multi-label dataset, such as Naïve Base, Support Vector Machine, Random Forest and Decision Tree with different feature extraction techniques and two-dimensionality reduction methods. We conclude from these models that TF-IDF has outperformed all other techniques and that PCA does not affect textual data as much as document frequency does.

BERT has proven to be a useful tool in natural language processing. However, the majority of the study has focused on using BERT with the extensive English language corpus. In addition, an examination of the most recent classification of Arabic text using BERT was done. The objective was to identify the Arabic text classification Bert models, evaluate their performance, and assess their efficiency in comparison to the original English Bert models. The BERT model has achieved perfect results within small and huge datasets because it is bidirectional representation, which means BERT layers provide dynamic context-dependent word embeddings by taking the whole sequence as input and calculating the representation of every word by trying to extract information from the entire sequence. Furthermore, Arabic BERT has been archived 95%, which surpasses the original, but we can take into consideration that the original BERT applied to unbalanced and multi-label

Dataset, while Arabic BERT used to balanced and single-label dataset, which made it get higher results.

The study can be expanded by working on different versions of BERT, such as Arabic BERT-Medium, Arabic BERT-Mini, Arabic BERT-Large, Arabic BERT-Base, and ARABERT, that handle Arabic unbalanced datasets. It can also be used with BERT for feature extraction with advanced deep learning algorithms.

REFERENCES

[1] Minaee S, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu and J. Gao (2021), "Deep learning--based text classification: a comprehensive review", *ACM Comput Surveys*, vol. 54, no. 3, pp. 1–40, DOI: 10.1145/3439726.

[2] Sebastiani F. (2005), "Text categorization", *In, Encyclopedia of database technologies and applications*, Ed. Laura Rivero, Jorge Horacio Doorn, Viviana E. Ferraggine, IGI Global, pp. 683–687.

[3] LeCun Y, Y. Bengio and G. Hinton (2015), "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, DOI: 10.1038/nature14539.

[4] Jiang M, Y. Liang, X. Feng, X. Fan, Z. Pei, Y. Xue and R. Guan (2018), "Text classification based on deep belief network and softmax regression", *Neural Computing and Applications*, vol. 29, no. 1, pp. 61–70, DOI: 10.1007/s00521-016-2401-x.

[5] Pereira R. B., A. Plastino, B. Zadrozny and L. H. C. Merschmann (2018),"Categorizing feature selection methods for multi-label classification", *Artificial Intelligence Review*, vol. 49, no. 1, pp. 57–78, DOI: 10.1007/s10462-016-9516-4.

[6] Dhar A, H. Mukherjee, N. S. Dash and K. Roy (2021), "Text categorization: past and present", *Artificial Intelligence Review*, vol. 54, no. 4, pp. 3007–3054, DOI: 10.1007/s10462-020-09919-1.

[7] Yu Z., W. Wu, J. Xiao, J. Zhang, R.-Z. Huang, and O. Liu (2009), "Keyword combination extraction in text categorization based on ant colony optimization", *2009 International Conference of Soft Computing and Pattern Recognition,* , pp. 430–435, Malacca, Malaysia.

[8] Sebastiani F. (2002), "Machine learning in automated text categorization", *ACM Comput Surveys*, vol. 34, no. 1, pp. 1–47, DOI: 10.1145/505282.505283 .

[9] Shen D., J. T. Sun, Q. Yang and Z. Chen (2006), "A comparison of implicit and explicit links for web page classification", *Proceedings of the 15th international conference on World Wide Web*, pp. 643–650, Edinburgh, DOI: 10.1145/1135777.1135871.

[10] Yang Y., S. Slattery and R. Ghani (2002), "A study of approaches to hypertext categorization", *Journal of Intelligent Information Systems*, vol. 18, no. 2, pp. 219–241, DOI: 10.1023/A:1013685612819.

[11] Yu H., J. Han and K.-C. Chang (2004), "PEBL: Web page classification without negative examples", *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 1, pp. 70–81, DOI: 10.1109/TKDE.2004.1264823.

[12] Drucker H., D. Wu and V. N. Vapnik (2009), "Support vector machines for spam categorization", *IEEE Transactions on Knowledge and Data Engineering*, vol. 10, no. 5, pp. 1048–1054, DOI: 10.1109/72.788645 .

[13] Chuan Z., L. Xianliang, H. Mengshu, and Z. Xu (2005), "A LVQ-based neural network anti-spam email approach", *ACM SIGOPS Operating Systems Review*, vol. 39, no. 1, pp. 34–39, DOI: 10.1145/1044552.1044555.

[14] Gale W. A., K. W. Church and D. Yarowsky (1992), "A method for disambiguating word senses in a large corpus", *Computers and the Humanities*, vol. 26, no. 5, pp. 415–439, DOI: 10.1007/BF00136984.

[15] Escudero G., L. Marquez and G. Rigau (2000), "Boosting applied to word sense disambiguation", *European Conference on Machine Learning*, pp. 129–141, Berlin, DOI: 10.1007/3-540-45164-1_14.

[16] Bakshi R. K, N. Kaur, R. Kaur and G. Kaur (2016), "Opinion mining and sentiment analysis", *2016 3rd international conference on computing for sustainable global development (INDIACom)*, pp. 452–455, New Delhi.

[17] Pazzani M. J. and D. Billsus (2007), "Content-based recommendation systems", *In, The adaptive web*, Ed. Peter Brusilovsky, Alfred Kobsa, Wolfgang Nejdl, pp. 325–341, Springer, DOI: 10.1007/978-3-540-72079-9_10.

[18] Larkey L. S. (1999), "A patent search and classification system", *Proceedings of the fourth ACM conference on Digital libraries*, pp. 179–187, Amherst, Massachusetts.

[19] Belkin N. J. and W. B. Croft (1992), “Information filtering and information retrieval: Two sides of the same coin?”, *Communications of the ACM*, vol. 35, no. 12, pp. 29–38, DOI: 10.1145/138859.138861.

[20] Hayes P. J., P. M. Andersen, I. B. Nirenburg and L. M. Schmandt (1990), “Tcs: a shell for content-based text categorization”, *Sixth conference on artificial intelligence for applications*, pp. 320–326, USA, DOI: 10.1109/CAIA.1990.89206.

[21] Maybury M. (1999), *Advances in automatic text summarization*. MIT press, London.

[22] Cao Z., Li. W, Li. S. and W. Furu (2017), “Improving Multi-Document Summarization via Text Classification”, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, pp. 3053-3059, San Francisco, DOI: 10.1609/aaai.v31i1.10955.

[23] Aggarwal C. C. and C. Zhai (2012), “A survey of text classification algorithms”, *In, Mining text data*, Ed. Charu C. Aggarwal, ChengXiang Zhai, pp. 163–222, Springer, Boston, DOI: 10.1007/978-1-4614-3223-4_6.

[24] Patel A., S. Pathak and M. I. Khan (2021), “Automated text categorization”, *2021 3rd International Conference on Signal Processing and Communication (ICPSC)*, pp. 16–20, Coimbatore, India, DOI: 10.1109/ICSPC51351.2021.9451670.

[25] Tsoumakas G. and I. Katakis (2007), “Multi-label classification: An overview”, *International Journal of Data Warehousing and Mining (IJDWM)*, vol. 3, no. 3, pp. 1–13, DOI: 10.4018/jdwm.2007070101.

[26] Chen G., D. Ye, Z. Xing, J. Chen and E. Cambria (2017), “Ensemble application of convolutional and recurrent neural networks for multi-label text categorization”, *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2377–2383, USA, DOI: 10.1109/IJCNN.2017.7966144.

[27] Tao Li, M. Ogihara and Q. Li (2003), “A comparative study on content-based music genre classification”, *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pp. 282–289, USA, DOI: 10.1145/860435.860487.

[28] Boutell M. R., J. Luo, X. Shen and C. M. Brown (2004), "Learning multi-label scene classification", *Pattern Recognit.*, vol. 37, no. 9, pp. 1757–1771, DOI: 10.1016/j.patcog.2004.03.009.

[29] Carreras X. and L. Marquez (2001), "Boosting trees for anti-spam email filtering," *arXiv preprint cs/0109015*, pp. 58-64, DOI: 10.48550/arXiv.cs/0109015.

[30] Lewis D. D. and M. Ringuette (1994), "A comparison of two learning algorithms for text categorization", *Third annual symposium on document analysis and information retrieval*, pp. 81–93, Chicago.

[31] Yang Y. and X. Liu (1999), "A re-examination of text categorization methods," *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 42–49, Pittsburgh.

[32] Joachims T. (1998), "Text categorization with support vector machines: Learning with many relevant features, *European Conference on Machine Learning*, pp. 137–142, Berlin, DOI: 10.1007/BFb0026683.

[33] Liu J, W. C. Chang, Y. Wu and Y. Yang (2017), "Deep learning for extreme multi-label text classification", *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pp. 115–124, Tokyo, DOI: 10.1145/3077136.3080834.

[34] Lai S., L. Xu, K. Liu and J. Zhao (2015), "Recurrent convolutional neural networks for text classification", *Twenty-ninth AAAI conference on artificial intelligence*, pp. 2267–2273, China.

[35] Kowsari K, D. E. Brown, M. Heidarysafa, K. J. Meimandi, M. S. Gerber and L. E. Barnes (2017), "Hdltex: Hierarchical deep learning for text classification", *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pp. 364–371, Mexico, DOI: 10.1109/ICMLA.2017.0-134.

[36] Yao L., C. Mao and Y. Luo (2019), "Graph convolutional networks for text classification", *Proceedings of the AAAI conference on artificial intelligence*, pp. 7370–7377, California, DOI: 10.1609/aaai.v33i01.33017370.

[37] Howard J. and S. Ruder (2018), "Universal language model fine-tuning for text classification", *arXiv preprint arXiv1801.06146*, DoA. 5 .08 .2022, DOI: 10.48550/arXiv.1801.06146.

[38] Vaswani A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin (2017), "Attention is all you need", *Advances in neural information processing systems*, vol. 30, pp. 2–11 DOI: 10.48550/arXiv.1706.03762.

[39] Devlin .J, M.-W. Chang, K. Lee and K. Toutanova (2018), "Bert: Pre-training of deep bidirectional transformers for language understanding", *arXiv preprint arXiv1810.04805*, DoA. 5 .08 .2022, DOI: 10.48550/arXiv.1810.04805.

[40] Chang W.-C., H.-F. Yu, K. Zhong, Y. Yang and I. S. Dhillon (2020), "Taming pretrained transformers for extreme multi-label text classification", *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 3163–3171, USA, DOI: 10.1145/3394486.3403368.

[41] Chang W.-C., H.-F. Yu, K. Zhong, Y. Yang and I. Dhillon (2019), "X-bert: extreme multi-label text classification with using bidirectional encoder representations from transformers", *arXiv preprint arXiv1905.02331*, pp. 1-12.

[42] Erciyes N. E. and A. K. Görür (2021), "Deep learning methods with pre-trained word embeddings and pre-trained transformers for extreme multi-label text classification", *6th International Conference on Computer Science and Engineering (UBMK)*, pp. 50–55, Ankara, DOI: 10.1109/UBMK52708.2021.9558977.

[43] Qasim R, W. H. Bangyal, M. A. Alqarni and A. A. Almazroi (2022), "A fine-tuned BERT-based transfer learning approach for text classification", *Journal of Healthcare Engineering*, vol. 2022, pp. 2-17, DOI: 10.1155/2022/3498123.

[44] Lu Z, P. Du and J-Y. Nie (2020), "VGCN-BERT: augmenting BERT with graph embedding for text classification", *European Conference on Information Retrieval*, pp. 369–382, Cham, DOI: 10.1007/978-3-030-45439-5_25.

[45] Banerjee S, C. Akkaya, F. Perez-Sorrosal and K. Tsioutsiouliklis (2019), "Hierarchical transfer learning for multi-label text classification", *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6295–6300, Florence, Italy, DOI:10.18653/v1/P19-1633.

[46] Elnagar A., Y. S. Khalifa and A. Einea (2018), "Hotel Arabic-reviews dataset construction for sentiment analysis applications", *In, Intelligent natural language processing: Trends and applications*, pp. 35–52, Springer, Cham, DOI: 10.1007/978-3-319-67056-0_3.

[47] Abdelgwad M. M. (2021), "Arabic aspect based sentiment analysis using BERT", arXiv preprint *arXiv2107.13290,* pp. 2-19, DOI: 10.48550/arXiv.2107.13290.

[48] Abdelgwad M. M., T. H. A. Soliman, A. I. Taloba and M. F. Farghaly (2022), "Arabic aspect based sentiment analysis using bidirectional GRU based models", *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 9, pp. 6652–6662, DOI: 10.1016/j.jksuci.2021.08.030.

[49] Safaya A., M. Abdullatif and D. Yuret (2020), "Kuisail at semeval-2020 task 12: Bert-cnn for offensive speech identification in social media", *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pp. 2054–2059, Barcelona, DOI:10.18653/v1/2020.semeval-1.271.

[50] Antoun W, F. Baly and H. Hajj (2020), "Arabert: Transformer-based model for arabic language understanding", *arXiv preprint arXiv:2003.00104*, DoA. 10 .01 .2023, DOI: 10.48550/arXiv.2003.00104.

[51] Alsaleh D and S. Larabi-Marie-Sainte (2021), "Arabic text classification using convolutional neural network and genetic algorithms", *IEEE Access*, vol. 9, pp. 91670–91685, DOI: 10.1109/ACCESS.2021.3091376.

[52] Abdulghani F. and N. Abdullah (2022), "Hybrid deep learning model for Arabic text classification based on mutual information", *Journal of Information and Optimization Sciences*, vol. 43, no. 8, pp. 1901–1908, DOI: 10.1080/02522667.2022.2060910.

[53] Rocchio J. (1971), "Relevance feedback in information retrieval", *The Smart Retrieval System-Experiments in Automatic Document Processing*, pp. 313–323.

[54] Partalas I., A. Kosmopoulos, N. Baskiotis, T. Artieres, G. Paliouras, E. Gaussier, I. Androutsopoulos, M.-R. Amini and P. Galinari (2015), "Lshtc: A benchmark for large-scale text classification", *arXiv preprint arXiv1503.08581,* pp. 1-9, DOI: 10.48550/arXiv.1503.08581 .

[55] Schapire R. E. (1990), "The strength of weak learnability", *Machine Learning*, vol. 5, no. 2, pp. 197–227, DOI: 10.1007/BF00116037.

[56] Bloehdorn S. and A. Hotho (2004), "Boosting for text classification with semantic features", *International workshop on knowledge discovery on the web*, pp. 149–166, Springer, Berlin, DOI: 10.1007/11899402_10.

[57] Breiman L. (1996), "Bagging predictors", *Machine. Learning*, vol. 24, no. 2, pp. 123–140, DOI: 10.1007/BF00058655.

[58] Jiang S., G. Pang, M. Wu and L. Kuang (2012), "An improved K-nearest-neighbor algorithm for text categorization", *Expert Systems with Applications*, vol. 39, no. 1, pp. 1503–1509, DOI: 10.1016/j.eswa.2011.08.040.

[59] Tsoumakas G. and I. Vlahavas (2007), "Random k-labelsets: An ensemble method for multilabel classification", *European conference on machine learning*, pp. 406–417, Springer, Berlin, DOI: 10.1007/978-3-540-74958-5_38.

[60] Katakis I., G. Tsoumakas and I. Vlahavas (2008), "Multilabel text classification for automated tag suggestion", *Proceedings of the ECML/PKDD*, vol. 18, pp. 5, Thessaloniki.

[61] Alazaidah R. and F. K. Ahmad (2016), "Trending challenges in multi label classification", *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 10, pp. 127–131.

[62] Zhang M.-L. and K. Zhang (2010), "Multi-label learning by exploiting label dependency", *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 999–1008, *USA* DOI:10.1145/1835804.1835930.

[63] Tsoumakas G., I. Katakis and I. Vlahavas (2009), "Mining multi-label data", *In, Data Min. Knowl. Discov. Handb.*, Ed. Oded Maimon, Lior Rokach, pp. 667–685, Springer, Boston, DOI: 10.1007/978-0-387-09823-4_34.

[64] Clare A. and R. D. King(2001), "Knowledge discovery in multi-label phenotype data", *European conference on principles of data mining and knowledge discovery* , pp. 42–53, Berlin, DOI: 10.1007/3-540-44794 6_4.

[65] Thabtah F. A., P. Cowling and Y. Peng (2004), "MMAC: A new multi-class, multi-label associative classification approach", *Fourth IEEE International Conference on Data Mining (ICDM'04)*, pp. 217–224, Springer, Berlin, DOI: 10.1007/3-540-44794-6_4.

[66] Zhang M.-L. and Z.-H. Zhou (2005), “A k-nearest neighbor based algorithm for multi-label classification”, *2005 IEEE international conference on granular computing*, pp. 718–721, Beijing, DOI: 10.1109/GRC.2005.1547385.

[67] Elisseeff A. and J. Weston (2001), “A kernel method for multi-labelled classification”, *Advances in neural information processing systems*, vol. 14, PP. 681-687.

[68] Zhang M.-L. and Z.-H. Zhou (2006), “Multilabel neural networks with applications to functional genomics and text categorization”, *IEEE transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1338–1351, DOI: 10.1109/TKDE.2006.162.

[69] Schapire R. E. and Y. Singer (2000), “BoosTexter: A boosting-based system for text categorization”, *Machine Learning*, vol. 39, no. 2, pp. 135–168, DOI: 10.1023/A:1007649029923 .

[70] Zhang M.-L., J. M. Peña and V. Robles (2009), “Feature selection for multi-label naive Bayes classification”, *Information Sciences*, vol. 179, no. 19, pp. 3218–3229, DOI: 10.1016/j.ins.2009.06.010.

[71] Cheng W. and E. Hüllermeier (2009), “Combining instance-based learning and logistic regression for multilabel classification”, *Machine Learning*, vol. 76, no. 2, pp. 211–225, DOI: 10.1007/s10994-009-5127-5.

[72] Sutskever I., J. Martens and G. E. Hinton (2011), “Generating text with recurrent neural networks”, *ICML*, pp. 1017-1024, https://icml.cc/2011/papers/524_icmlpaper.pdf, DoA. 6 .01 .2022.

[73] Mandic D. and J. Chambers (2001), *Recurrent neural networks for prediction: learning algorithms, architectures and stability*, Wiley, Chichester, https://orca.cardiff.ac.uk/id/eprint/31639. DoA. 6 .01 .2022.

[74] Hochreiter S. and J. Schmidhuber (1997), “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, DOI: 10.1162/neco.1997.9.8.1735.

[75] Graves A. and J. Schmidhuber (2005), “Framewise phoneme classification with bidirectional LSTM and other neural network architectures,” *Neural networks*, vol. 18, no. 5–6, pp. 602–610, DOI: 10.1016/j.neunet.2005.06.042.

[76] Chung J., C. Gulcehre, K. Cho and Y. Bengio (2014), “Empirical evaluation of gated recurrent neural networks on sequence modeling”, arXiv preprint *arXiv1412.3555*, pp. 1-9, DoA. 6 .01 .2022, DOI: 10.48550/arXiv.1412.3555.

[77] Cho K. B., V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares and H. Schwenk, Y. Bengio (2014), “Learning phrase representations using RNN encoder-decoder for statistical machine translation”, *arXiv preprint arXiv1406.1078,* DOI: 10.48550/arXiv.1406.1078, DoA. 6 .01 .2022 .

[78] Jaderberg M., K. Simonyan, A. Vedaldi and A. Zisserman (2016), “Reading text in the wild with convolutional neural networks”, *International journal of computer vision*, vol. 116, no. 1, pp. 1–20, DOI: 10.1007/s11263-015-0823-z.

[79] LeCun Y., L. Bottou, Y. Bengio and P. Haffner (1998), “Gradient-based learning applied to document recognition”, *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, DOI: 10.1109/5.726791.

[80] Jones K. S. (1972), "A statistical interpretation of term specificity and its application in retrieval", *Journal of Documentation*, Vol. 28 No. 1, pp. 11-21. DOI: 10.1108/eb026526.

[81] Kowsari K., K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes and D. Brown (2019), “Text classification algorithms: A survey”, *Information*, vol. 10, no. 4, p. 150, DOI: 10.3390/info10040150.

[82] Guyon I. and A. Elisseeff (2003), “An introduction to variable and feature selection”, *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182 .

[83] Chandrashekar G. and F. Sahin (2014), “A survey on feature selection methods”, *Comput. Electr. Eng.*, vol. 40, no. 1, pp. 16–28, DOI: 10.1016/j.compeleceng.2013.11.024 .

[84] John G. H., R. Kohavi and K. Pfleger (1994), “Irrelevant features and the subset selection problem”, *Machine learning proceedings 1994*, Elsevier, New Brunswick, pp. 121–129, DOI: 10.1016/B978-1-55860-335-6.50023-4 .

[85] Yang Y. and J. O. Pedersen (1997), “A comparative study on feature selection in text categorization”, *Icml*, vol. 97, no. 412–420, p. 35.

[86] Abdi H. and L. J. Williams (2010), “Principal component analysis”, *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, DOI: 10.1002/wics.101 .

[87] Van L. Der Maaten, E. Postma and J. Van den Herik (2009), “Dimensionality reduction: a comparative”, *Journal of Machine Learning Research*, vol. 10, no. 66–71, p. 13.

[88] Jolliffe I. T. and J. Cadima (2016), “Principal component analysis: a review and recent developments”, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20150202, DOI: 10.1098/rsta.2015.0202.

[89] Ng A. (2015), “Principal components analysis”, *Generative Algorithms, Regularization and Model Selection CS*, vol. 229, p. 71.

[90] Cortes C. and V. Vapnik (1995), “Support-vector networks”, *Machine. Learning.*, vol. 20, no. 3, pp. 273–297, DOI: 10.1007/BF00994018.

[91] Noble W. S. (2006), “What is a support vector machine?”, *Nature biotechnology*, vol. 24, no. 12, pp. 1565–1567, DOI: 10.1038/nbt1206-1565.

[92] Mayoraz E. and E. Alpaydin (2006), “Support vector machines for multi-class classification”, *Lecture Notes in Computer Science*, vol. 1607, pp. 833–842 Springer, Berlin, DOI: 10.1007/BFb0100551.

[93] Pearson E. S. (1925), “Bayes’ theorem, examined in the light of experimental sampling”, *Biometrika*, pp. 388–442, DOI: 10.2307/2332088 .

[94] Hill B. M. (1968), “Posterior distribution of percentiles: Bayes’ theorem for sampling from a population”, *Journal of the American Statistical Association*, vol. 63, no. 322, pp. 677–691.

[95] McCallum A. and K. Nigam (1998), “A comparison of event models for naive bayes text classification”, *AAAI-98 workshop on learning for text categorization*, vol. 752, no. 1, pp. 41–48.

[96] Magerman D. M. (1995), “Statistical decision-tree models for parsing”, *arXiv preprint cmp-lg/9504030*,DoA. 18 .08 .2022, DOI: 10.48550/arXiv.cmp-lg/9504030.

[97] Quinlan J. R. (1986), “Induction of decision trees”, *Machine Learning*, vol. 1, no. 1, pp. 81–106, DOI: 10.1007/BF00116251.

[98] Ho T. K. (1995), “Random decision forests”, *Proceedings of 3rd international conference on document analysis and recognition*, vol. 1, pp. 278–282, Montreal, DOI: 10.1109/ICDAR.1995.598994.

[99] Lewis D. D., Y. Yang, T. Russell-Rose and F. Li (2004), “Rcv1: A new benchmark collection for text categorization research”, *Journal of machine learning research*, vol. 5, no. Apr, pp. 361–397.

[100] Pontiki M., D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. Al-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, O. d. Clercq, V. Hoste, M. Apidianaki, X. Tannier, N. Loukachevitch, E. Kotelnikov, N. Bel, S. M. Jiménez-Zafra and G. Eryiğit (2016), "Semeval-2016 task 5: Aspect based sentiment analysis", *International workshop on semantic evaluation*, pp. 19–30, San Diago.

[101] Qiu X., T. Sun, Y. Xu, Y. Shao, N. Dai and X. Huang (2020), "Pre-trained models for natural language processing: A survey", *Science China Technological Sciences*, vol. 63, no. 10, pp. 1872–1897, DOI: 10.1007/s11431-020-1647-3.

[102] Sutskever I., O. Vinyals and Q. V Le (2014), "Sequence to sequence learning with neural networks", *Advances in neural information processing systems*, vol. 27, pp. 1-9, DOI: 10.48550/arXiv.1912.06813.

[103] Kingma D. P. and J. Ba (2014), "Adam: A method for stochastic optimization", *arXiv preprint arXiv:1412.6980*,pp. 1-15, DOI: 10.48550/arXiv.1412.6980.

[104] Lever J., M. Krzywinski and N. Altman (2016), "Classification evaluation: it is important to understand both what a classification metric expresses and what it hides", *Nature Methods*, vol. 13, no. 8, pp. 603–604.

[105] Schütze H., C. D. Manning and P. Raghavan (2008), *Introduction to information retrieval*, Cambridge University Press Cambridge, New York.

APPENDICES

APPENDIX 1 DATASET SAMPLES

Table 1.1: Reuters-21578 Dataset Sample

ids	Categories	Text
test/14826	['trade']	ASIAN EXPORTERS FEAR DAMAGE FROM U.S.-JAPAN RIFT Mounting trade friction between the U.S……
test/14828	['grain']	CHINA DAILY SAYS VERMIN EAT 7-12 PCT GRAIN STOCKS A survey of 19 provinces…….
test/14829	['crude', 'nat-gas']	JAPAN TO REVISE LONG-TERM ENERGY DEMAND DOWNWARDS The Ministry of International Trade…….
test/14832	['corn', 'grain', 'rice', 'rubber', 'sugar', 'tin', 'trade']	THAI TRADE DEFICIT WIDENS IN FIRST QUARTER Thailand's trade deficit widened……
training/1	['cocoa']	BAHIA COCOA REVIEW Showers continued throughout the week in the Bahia cocoa zone, alleviating the drought since early……
training/10	['acq']	COMPUTER TERMINAL SYSTEMS <CPML> COMPLETES SALE Computer Terminal Systems ……
training/100	['money-supply']	N.Z. TRADING BANK DEPOSIT GROWTH RISES SLIGHTLY New Zealand's trading…….

Table 1.2: RCV-v2 Dataset Sample

Ids	Date	newsitem	C11	C12	C13	C14	C15
2286	########	Emerging evidence that Mexico's economy was back on the recovery track sent Mexican markets into a buzz of excitement Tuesday……	0	0	0	0	0
2287	########	Now being built in Argentina. Chrysler, which is cautiously trying to..	0	0	0	0	0
2288	########	CompuServe Corp. Tuesday reported a surprisingly large $29.6 million fiscal….	0	0	0	0	1
2289	########	CompuServe Corp. Tuesday reported a surprisingly large $29.6 million fiscal….	0	0	0	0	1
2286	########	If dining at Planet Hollywood made you feel like a movie star, now……	1	0	0	0	0

Table 1.3: HARD Dataset Sample

no	Hotelname	rating	user type	room type	nights	review
2	فندق 72	2	مسافرد منفرد	غرفة دوبلكس مزدوجه أو توأم	أقمت ليلة واحدة	ممتاز". النظافة " والطاقم متعاون
3	فندق 72	5	زوج	غرفة دوبلكس مزدوجه أو توأم	أقمت ليلة واحدة	استثنائي. سهولة إنهاء المعاملة في الاستقبال. لاشيئ
16	فندق 72	5	زوج	-	أقمت ليلتين	استثنائي. انصح بأختيار الاسويت و بالاخص غرفه رقم 801. نوعية الارضيه
20	فندق 72	1	زوج	غرفة قياسية مزدوجه	أقمت ليلة واحدة	استغرب تقييم " الفندق كخمس نجوم". لا شي. يستحق 2 نجمه
23	فندق 72	4	زوج	غرفة دوبلكس مزدوجه أو توأم	أقمت ليلتين	جيد. المكان جميل وهاديء. كل شي جيد ونظيف بس كان حوض السباحه لايعمل في هذي الفتره حسب كلامهم يقولوا فيه

Table 1.4: ABSA Dataset Sample

rid	polarity	category	sentence
287	negative	ROOMS_AMENITIES#GENERAL	['كانت رحلتي لهذا المرفق موفقه ، لكن كانت بعض ...
287	positive	FACILITIES#GENERAL	['كانت رحلتي لهذا المرفق موفقه ، لكن كانت بعض ...
1039	negative	ROOMS_AMENITIES#GENERAL	['رضى النزيل الاجهزة الكهربائية غير سليمة الشا...