REPUBLIC OF TÜRKİYE

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Information Technologies

FAKE NEWS DETECTION VIA AUTOMATED DEEP LEARNING

Yasir Abdulkareem Jamal JAMAL

Master's Thesis

Supervisor

Prof. Dr. Osman Nuri UÇAN

Istanbul, 2023

FAKE NEWS DETECTION VIA AUTOMATED DEEP LEARNING

Yasir Abdulkareem Jamal JAMAL

Information Technologies

Master’s Thesis

ALTINBAŞ UNIVERSITY

2023

The thesis/dissertation titled FAKE NEWS DETECTION VIA AUTOMATED DEEP LEARNING prepared by Yasir JAMAL and submitted on 03/04/2023 (DATE) has been **accepted unanimously/by majority of votes** for the degree of Master of Science in Information Technologies

Prof. Dr. Osman Nuri UÇAN
the Supervisor

Thesis Defense Committee Members:

Prof. Dr. OSMAN NURİ UÇAN	Electrical and Electronics Engineering, Altınbaş University	_____
Dr. Instructor Member ABDULLAHI ABDU IBRAHIM	Computer Engineering, Altınbaş University	_____
Dr. Instructor Member of SERDAR KARGIN	Biomedical Engineering, Arel Istanbul University	_____

I hereby declare that this thesis/dissertation meets all format and submission requirements of a …… Master's thesis.

Submission date of the thesis to the Graduate Education Institute:___/___/____

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Yasir Abdulkareem Jamal JAMAL

Signature

DEDICATION

First and foremost, I thank ALLAH Almighty for giving me the knowledge, ability, and opportunity to complete this research study and to persevere and complete it satisfactorily.

I would like to thank my supervisor Prof. Dr. Osman Nuri UÇAN for helping me in writing this dissertation, and I'm grateful to him for giving me the opportunity to work under his supervision.

My deep and sincere gratitude to my family for their continuous and unparalleled love, help and support. I am grateful to my mother, my father and my wife who look after me and pray for me day by day.

ABSTRACT

FAKE NEWS DETECTION VIA AUTOMATED DEEP LEARNING

JAMAL,Yasir Abdulkareem Jamal

M.Sc., Information Technologies, Altınbaş University,

Supervisor: Prof. Dr. Osman Nuri UÇAN

Date: April / 2023

Pages: 84

The phenomenon of fake news is one of the terrible and old problems that have appeared since ancient times, and which has increased recently with the widespread use of the Internet, especially news sites and social media. This problem has clear negative effects on communities and governments, which may lead to dire results. As a result, there are many research studies that discuss and try to reduce the seriousness of this problem by using machine learning and deep learning methods. This type of solution requires good experience, great effort, and a long-time during experimentation and repetition to obtain a good model and satisfactory results. Automated machine learning (AutoML) is one of the latest technologies in the field of machine learning which help solve problems related to the experience need, effort, and time, where this technology makes people with little experience in the field of machine learning to be able to work in this field. In addition, this technique reduces the experiences and repetition happening in the case of traditional machine learning, thus the time and effort will be reduced. In this work, we attempted to contribute to solving the fake news. For this purpose, we utilized the liar dataset, which has short news, for our experiments. We also utilized several AutoML libraries including automated deep learning (AutoDL) such as auto-sklearn, TPOT, Auto-Keras and AutoGluon to find the optimal models for detecting fake news. Additionally, we created traditional models using ML and

DL techniques where we employed TF-IDF, the feature extraction technique, with ML models, whereas word embedding techniques such as Word2Vec and GloVe were utilized along with DL models. We compared the results of the two approaches and concluded that the results were close to some extent, and this indicates clear progress in the field of AutoML.

Keywords: Fake News Detection, Automated Machine Learning, Automl, Automated Deep Learning, Autodl, Machine Learning, Deep Learning.

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

AutoML	:	Automated Machine Learning
AutoDL	:	Automated Deep Learning
ML	:	Machine Learning
DL	:	Deep Learning
LSTM	:	Long Short-Term Memory
BI-LSTM	:	Bidirectional Long Short-Term Memory
NB	:	Naïve Bayes
SVM	:	Support Vector Machine
CNN	:	Convolutional Neural Network
RNN	:	Recurrent Neural Network
NLP	:	Natural Language Processing
NLU	:	Natural Language Understanding
NLG	:	Natural Language Generation
TPOT	:	Tree-Based Pipeline Optimization Tool
TF-IDF	:	term frequency-inverse document frequency
GloVe	:	Global Vectors for Word Representation
NAS	:	Neural Architecture Search
HPO	:	Hyper-Parameter Optimization

1. INTRODUCTION

1.1 GENERAL

When someone needed news, they used to wait for the following day's newspaper. To be informed about a topic of interest, individuals now have a better and faster approach thanks to the expansion of online newspapers that update news practically quickly. Increasingly, people turned to online media like online websites and social networking platforms to obtain their news where important and breaking news is shared quickly. However, many news websites serve certain interests by disseminating inaccurate, only partially true, and occasionally fictitious news which may catch the attention of the intended audience. The spread of purposeful misinformation and other negative effects of fake news has made it a serious concern [1].

Unlike conventional mass media like newspapers, magazines, radio, and television, the use of social networks to disseminate news has as a side consequence of the pandemic's proliferation of fake news which shows itself to be a danger to logical truth, democracy, journalism, and the trustworthiness of governmental institutions due to human incapacity to discriminate between real and false information [2].

There are several websites where fake news stories are created. For instance, certain websites, like the aforementioned denverguardian.com, are created only to publish deceptively written and presented content. Frequently, these websites are given names that closely resemble those of trusted and well-known news organizations. Other satirical websites, like likewtoe5news.com, have stories that, when read out of context, may be mistaken for true information. However, some websites, like endingthefed.com, provide a mixture of real information—often with a partisan slant—and some fake objects. Fake news websites frequently disappear after a short time, and several that were significant in the lead-up to the 2016 American election are no longer online [3].

Consuming news via social media has pros and downsides. The positive side is that social media is utilized to find and consume news because of its cheap cost, the convenience of access, and rapid dissemination of information. the negative side is that social media encourages the mass circulation of "fake news," which refers to news of poor quality that

includes material that is intended to mislead. The widespread dissemination of false information has very detrimental effects on people and society. As a direct consequence of this, the academic research on identifying fake news on social media is gaining increasing attention [4].

The popularity of social media and the lack of a necessity for computer literacy create an environment for cybercrime by allowing for the dissemination of untrusted information online. News travels quickly and might be false or true. People lack the intelligence to recognize whether or not news is reliable [5]. The spread of false information over the internet has recently become a significant source of worry. It has a direct impact on individuals and has to be recognized [6].

Although fact-checking by humans may in fact assist readers in spotting fake news, they fall well short of the objective of early false news identification for the following reasons. First off, since it takes time, manual fact-checking often responds to false information after the fact. By the time manual fact-checking websites or technologies declare a news piece to be false, it has often already reached a large audience and harmed society; Secondly, fact-checking by humans is impractical to deal with the vast number of potentially false news pieces that are posted on the Internet every day. In light of this, automated detection methods are urgently required to enable real-time false news detection from the massive amount of news pieces generated each day [7].

In latest years, several machine learning (ML) and deep learning (DL) methods, which are based on data, were proposed to identify real and fake news, where many studies have been conducted, the primary emphasis of which has been the categorization and identification of fake news which are spread hugely on social networks [6] ,[8]. Machine learning (ML)-based automated detection methods have gained substantial interest from both the academic community and the business as a result of the recent fast growth of ML and DL approaches which have largely replaced human fact-checking [7]. There are several studies, which is concerning with detecting fake news, will be discussed in the next chapter. The objective of these studies is to find a solution to the issue of detecting fake news by using machine learning or deep learning approaches. However, a significant challenge for new users is created by the fact that Several machine learning algorithms' efficiencies is quite sensitive to a variety of designing choices. This is especially evident in the deep learning context,

where the ideal neural architectures, regularization strategies, training issues, and hyper-parameters of all these components have to be selected by the specialist to ensure that designed networks function as intended. For each application, this procedure must be repeated.

Even specialists often experience tiresome iterations of trial and error before settling on the suitable options which work best with a certain dataset. Making these kinds of choices in an automated, data-driven, and objective manner is considered as one of the prime aims of automated machine learning (AutoML). Providing the data is the only work required by the user; then the AutoML framework will choose the method that works perfectly for the given application.

As a result, AutoML makes cutting-edge ML techniques available to domain scientists who are interested in implementing machine learning but lack the resources to thoroughly understand the technology involved [9]. Our thesis's primary goal is to contribute to solving the fake news problem using the AutoML technique. For this purpose, we utilize four AutoML libraries to examine their performance. Two comparisons are presented in our work. The first one is between the AutoML tools, and the other one is between the AutoML approach and traditional machine learning approaches, both classic and deep.

1.2 TERMS RELATED TO FAKE NEWS

The expression "fake news" is around since a considerable amount of time, in addition to has a rich history that dates back centuries to the invention of the earliest writing systems. However, with the rise of social media during the last decade, there has been a shift in how news is disseminated that is quite different from how it was previously reported by traditional media. The various social media platforms have evolved into an ideal environment for trolling and for the dissemination of computational propaganda. There are a few other terms that are often used interchangeably with "fake news" [10]. Some of the most important terms will be reviewed within this section.

Rumors: a rumor is a piece of information that is spreading but whose truthfulness status has not yet been established at the date of publishing. As a result, rumor’s truth value is undetermined while it is spreading. When there is no proof to back a rumor or there is no formal

confirmation from reliable sources or sources that could have credibility in a certain context, it is said to be unverified [11].

Satire: Faked news that is intended to be amusing is called satire. It's possible that some individuals, out of ignorance, may take satire seriously [12]. So that viewers would understand the hilarious purpose, satire is meant to be noticeably distinct from true news. More often than not, hoaxes and satire create stories [13].

Hoax: it is a form of reporting that uses further sophisticated deceit strategies for deceiving its target audience where there are multiple sources that propagate fake news. It is probable that some individuals think the narrative to be real. As a result, the citizen is more inclined to trust it, As was the case during the US elections where false news flowed on social media regarding President Trump [12].

Fabrication: fabricated news is information that has been intentionally left out and generally comes from only a single source who is perhaps aware of the story's inaccuracy [12].

Propaganda: Readers are misled by propaganda to believe in a certain political or social goal. For the purpose of confusing readers, propaganda usually blends facts, ambiguities, and falsehoods [13].

Clickbait: It is common practice to utilize sensational headlines as clickbait to grab users' attention, lead them to click, and then send them to another website. Increasing the number of ad clicks will increase revenue [10].

1.3 FAKE NEWS IMPACT

The propagation of fake news has the potential to have a significant influence on society as well as the possibility to do a great deal of damage [7], [14]. One of the biggest challenges to business, democracy and journalism around the globe is fake news, which also has enormous collateral harm [15]. Fake news may be exploited for political or financial gain or personal benefit. in addition, it also may be exploited to damage a person's or organization’s reputation because its contents are willfully false. The timing and circumstances of the news's creation, the person who made the story and his or her social standing, as well as the online social network that was utilized, all have a significant role in how much harm false news does. Society might have negative impacts if fake news is not put an end to in its first stages [14]. In all categories of information, falsehood spread significantly faster, deeper, farther, and more widely than the truth, and false news about politics had more of an impact

than false news on financial information, science, terrorism, natural catastrophes, and urban legends [16].

There are several instances that demonstrate how false news affects political life and democracy. the most important examples in this regard is what happened in 2016 in the United States of America, specifically during the presidential elections, which saw Donald Trump elected as the country's leader, where there was a rise in fake news on social media [17]. Experts concurred that one of the numerous factors contributing to Hillary Clinton's poor performance versus Donald Trump in the US presidential elections was the fabrication of tales about her [18]. In 2014, the terrorist group ISIS started spreading its propaganda on every social media site you can think of. This is another example of how fake news can be used to change people's political views [17].

In the domains of trade and stock trading, the widespread dissemination of inaccurate information has increasingly detrimental implications [16]. for example, an erroneous news agency tweet that said President Obama had been hurt by explosives at the White House put markets on a roundtrip roller coaster. the Dow lost more than 140 points and Bond rates dropped. The Dow was trading with triple-digit gains and had regained its losses within six minutes. According to Reuters, the S&P 500's market capitalization lost $136.5 billion in value for a brief period. Dollar/yen also momentarily fell to a level around 98.60 in a deal that was closely related to it before rising to over 99 [19].

Fake news may contribute to the misallocation of resources during natural disasters and terrorist attacks, unfair elections, and the misalignment of business investments [16]. Figure 1.1 shows the fake tweet and the severe economic damage caused by this tweet.

Figure 1.1: The Fake Tweet And The Deterioration of Financial Markets [19].

Education also is greatly affected by fake news. Due to their growing use of social media, there is a high chance that adolescents and children encounter a massive wave of fake news, and they are also more prone to believe this news. This has a negative impact on youngsters since they may transmit false information to their younger siblings and classmates. Fake news that reaches children and adolescents may create several difficulties since it alters their thinking and affects their outlook on life as a result of receiving incorrect information. young adults and students have a higher likelihood to be exposed to viral postings and fake news because they lack the maturity to detect the fake news and are readily influenced by fake news sources, which impacts their learning culture. the Fake news may also influence behaviours. It encourages pupils to find justifications for rejecting and confusing the opinions of other students, exaggerating the facts, and spreading misinformation. This may cause anxiety in classrooms because pupils are cynical and uncertain of whom to trust. The Stanford research, published in November, was based on a survey in which 7,804 middle school to college-aged students were answered questions concerning online information. according to this research, young people's capacity to reason is "bleak" in light of the facts and contents they discovered on the web. Thus, this suggests that young people are incapable of recognizing the false information threat democracy itself [20].

Many governments throughout the globe have begun acting for struggling the fake news which is published widely on sites and social media. The United Kingdom investigated expansive regulatory proposals to fight the spread of harmful content, including misinformation, over the internet. The French law introduced expedited civil proceedings to fight the dissemination of misleading information before to referendums and major elections to combat foreign state-funded broadcast propaganda, as well as increasing the transparency of internet content distribution and funding. At the same time, this legislation sought to protect democratic legitimacy without compromising the basic liberties upon which it is founded [21].

In Ukraine, specifically in 2014, several information literacy initiatives that try to counteract fake news were utilized when Russia's invasion of Crimea was bolstered by a flow of rudimentary but influential disinformation. Many Ukrainians volunteered to obtain training in information literacy, and they banded together under the banner of defending their nation from the misinformation assault. By integrating information literacy education into civic life,

Ukraine has shown significant potential in limiting the harm caused by false information and other forms of disinformation. By thoughtfully adding media literacy training in prerequisite educational courses for different educational stages, it attained significant advancements towards updating the national educational system [22].

1.4 THESIS OUTLINE

Following is a summary of the structure of the present thesis:

The second chapter shows the related works achieved by the research community regarding detecting fake news via ML approaches, both traditional and deep. Additionally, studies and works regarding the auto-ml are mentioned in this chapter.

The third chapter mainly concentrates on the theoretical aspect, where it explains methods, materials, and technical words relevant to the present thesis.

The fourth chapter reports the experiments performed utilizing the different approaches and methodologies, as well as the results of these experiments. This chapter is concluded with a discussion of the results.

The fifth chapter provides a summary of this thesis as well as a discussion of the proposed future works.

2. LETIRETURE REVIEW

In this chapter we concentrate on two important aspects that have a strong relevance to our work. the first aspect shows related works that have been achieved in fake news detection scope using ML approaches, whether classical or DL models. We review several works that converted the Liar dataset from a classification with six labels to a binary classification, which is the same approach we followed in our work. in the second aspect, we shed light on research and works related to automated machine learning. In addition, part of these research is concerned with AutoML works towards natural language processing.

2.1 RELATED WORKS USING TRADITIOAL APPROACHES

The study [12] implemented different fake detection methods to recognize user behaviour by spreading fake news or rumours. On three corpus, liars and false news datasets, the comparison between several conventional machine learning approaches and deep learning techniques have been performed in which deep learning beat machine learning. Bi-LSTM achieved the highest rate in detecting fake news, 95% accuracy, and an F1 score in this comparison.

For determining fake news, the study [23] suggested the use of a multi-channel DL model called the Mc-DNN, which takes use of the headlines and contents of the articles on various channels. Many deep models developed by different academics and fact-checking websites are outperformed by the Mc-DNN model. Using the combination of CNN with GRU, CNN with RNN, CNN with Bi-GRU, CNN with LSTM, as well as CNN with BiLSTM, the performance of Mc-DNN is examined. According to the study, the CNN and BiLSTM Mc-DNN are the most accurate for detecting fake news, with accuracy rates of 94.68% and 99.23% on FND and ISOT datasets, respectively.

In three separate experiments, the research [24] applied ML classifiers, DL models, and transformers. In these research experiments, word embedding was used for the purpose of obtaining contextual features from the articles. The findings demonstrated that, in terms of accuracy, DL models beat ML classifiers and BERT transformer. The GRU and LSTM models were quite similar in accuracy, according to the findings. This research also

demonstrated that it is feasible identifying fake news with high accuracy via integrating an augmented linguistic feature set with ML or DL models.

To identify the competent fake news, this work [25] concentrated on several linguistic features. 26 important features were chosen, and different machine learning classifiers were applied for implementation. TF-IDF, hash-vectorizer and count-vectorizer, which are the techniques for feature extraction, were utilized. Then, this work evaluated those models using various sizes of training dataset to determine each model's accuracy before comparing them. For the experiment, four existing datasets were used. Using the Reuter dataset, an accuracy rate of 90.8% was obtained by the suggested framework. In addition, The Buzzfeed dataset had the greatest accuracy, at 90%. Accuracy rates for the Mc_Intire and Random Political datasets were 86.9% and 93.8, respectively.

Based on the DL approach, the research [26] suggested a method for revealing fake news . Three stages were carried out in this proposed system: text encoding, feature extraction, and classification. By employing GLOVE for word representation, the text encoding procedure was performed on the input news words. To be enrolled in the suggested deep learning models, the encoded words must subsequently be embedded into a certain word length. The suggested deep learning models include automatic tasks for feature extraction and classification. The research proposed four diverse deep learning models, involving GRU, LSTM, Convolutional Neural Networks (CNNs) and Concatenated CNNs to find an optimal model which outperforms earlier works. The Concatenated CNNs accomplished an accuracy of 99.6% and trained more quickly than others when the proposed deep learning models were tested on the FNC and FNs datasets. Precision, recall, F1, and accuracy are only a few of the assessment criteria that were applied to assess the suggested models' performance.

The study [27] evaluated the efficiency of three classifiers on two feature sets in identifying false news. For the evaluation, the liar dataset was employed as a benchmark dataset. The dataset has converted from classification with six classes to a binary classification, where the true label represented mostly-true and true, whereas the fake label represented half-true, barely-true, false, and pants fire. Two experiments have been performed in this work, in which the speakers' statement feature set was used in the first experiment, whereas in the other one, counts of the speaker's statement history and statement features were employed. The results of J48 and BayesNet F-score were enhanced by 9.7% and 10.6%, respectively,

with the inclusion of the history feature set. The better possible scores of 69.7% F1 Score were obtained by utilizing the J48 classifier on the count of the speaker's statement history and the politician's brief statement.

The study [28] proposed an ensemble—based deep learning model to classify news as fake or real using LIAR dataset. This study only utilized the records which their class are false or true. the total number of records which used were 4557, where 2053 records as true label, and 2504 records as false label, respectively. Due to the nature of the dataset attributes, two deep learning models were used. For the textual attribute "statement," Bi-LSTM-GRU-dense deep learning model was used, while for the remaining attributes, dense deep learning model was used. Experimental results showed that the proposed study achieved an accuracy of 0.898, recall of 0.916, precision of 0.913, and F-score of 0.914, respectively, using only statement attribute.

The research [29] looked at how well five machine learning models performed to differentiate between real and fake news using the Liar dataset. As part of this work, the labels marked as barely- true, false, and pants-fire were changed to fake labels, while the ones marked as true, mainly-true, and half-true were modified as real labels. The converted dataset contained 44% as fake statements and 56% as real statements. two feature extraction methods, TF-IDF and Count vectorizer, were applied, and the statement feature was used in this work. When utilizing CV, Multinomial Naïve Bayes produced the best performance, with an accuracy of 60%. With an average accuracy of 61%, the logistic regression classifier demonstrated the highest level of performance when the TF-IDF method was employed.

Regarding the research [30], LIAR dataset was used for experimental analysis for spotting fake news. The dataset was converted from multi class to a binary class. Statement and label fields were used. Features of news text were used, and Three types of linguistic features including psycho-linguistic, writing pattern, and quantity features were considered. for final machine learning classification, these features were combined with news text. count-vectorizer, TF-IDF and hash-vectorizer techniques were employed for feature extraction. By employing the random forest algorithm as well as TF-IDF technique, the suggested method had the highest of 72.8% accuracy.

2.2 AUTOMATED MACHINE LERNING WORKS

In the domain of ML, the automated machine learning (AutoML) is one of the most concentrated areas of research. As a direct result of AutoML, both the number of users and interest in ML applications have significantly increased. However, given that AutoML is a relatively new technology that is still in the process of development and that a considerable amount of research is being conducted in this specific field, there may be a variety of benefits and drawbacks that may be discovered in any AutoML library. In addition, the performance of the AutoML tools that are used for various fields might vary. As a result, several different studies were conducted to assess the usefulness of various AutoML libraries [31].

An introduction to automated machine learning for supervised learning was presented by the research [32] in addition to giving historical overview of the advancement that has been made in this subject. It summarized the major findings in the early years of AutoML, additionally offering an overview of the significant accomplishments made throughout this first decade. This study also described the main paradigms of AutoML, provided the fundamentals of the AutoML task and presented the most representative methodologies.

The study [33] provided a thorough and current analysis of the state-of-the-art in AutoML. Also, it described AutoML approaches in accordance with the DL pipeline, encompassing data preparation, feature engineering, hyperparameter optimization, and neural architecture search (NAS), with an emphasis on NAS since it is a popular sub-topic of AutoML at the moment. On the ImageNet and CIFAR-10 and datasets, it reviewed the performance of the typical NAS algorithms and further covered a number of NAS-related topics. For future study, it outlined some unresolved issues related to AutoML techniques now in use.

The study [34] included an empirical and theoretical overview of AutoML's present condition. It offered the first empirical assessment of Combined Algorithms Selection and Hyperparameter Optimization (CASH) on 114 publicly available datasets. In terms of examined AutoML frameworks and the number of datasets, these frameworks were evaluated. Theoretically, essential techniques employed by such frameworks were presented and summarized.

According to the study [35], provided a thorough analysis of the most recent initiatives to address the CASH issue. this study also drew attention to the ongoing research on automating

the remaining phases of the whole complicated machine learning pipeline (AutoML), from data understanding to model development. it also provided thorough coverage of the many frameworks and technologies launched in this field. For realizing the vision and objectives of the AutoML process, the study talked about some of the research paths and unresolved issues that need to be handled.

The research [36] provided a current survey on automated machine learning. it also suggested a broad AutoML framework that can both direct the design of new techniques and covers the majority of current approaches. The research classified and reviewed the prior studies in terms of the problem formulation as well as the techniques utilized.

Based on automated machine learning and classical approaches (traditional techniques for ML/DL), the research [31] evaluated the implementation of a classification model for sentiment analysis. As AutoML frameworks, TPOT and HyperOpt SkLearn were used for building the sentiment analysis models. Scikit learn libraries were used as a traditional method. In addition, Keras and Auto-Keras libraries were employed to implement the deep learning models. these libraries were applied for building two binary classification and two multiclass classification models. then, the results were evaluated by each AutoML and Traditional methods. as a result, the researchers identified that using machine learning or deep learning approaches for building models manually is better than using automated machine learning approaches. According to the findings, it was evident that AutoML frameworks are not appropriate for every situation and that they should only be used to determine the optimal algorithmic method.

Through the examination of several of the user's tweets, the purpose of the research [37], which finished in fourth place overall in the PAN competition, is to determine whether or not the user is engaging in the promotion of hatred on Twitter. a number of transformer methods were employed for extracting feature embeddings from these tweets. For classifying these embedding features, an AutoML classifier was employed. an Accuracy of 82% and 72% was attained for gold standard test data, and when employing 5-fold cross-validation, an accuracy of 85% and 75% was attained for Spanish and English, respectively.

In the study [38], Researchers examined a number of AutoML tools with regard to their capabilities in the standard machine learning pipeline. In addition to this, they tested tools

across a wide variety of datasets on several data segments. they found that the majority of AutoML tools provide results that are acceptable in terms of the performance they deliver across a variety of datasets, but at this point in time, there is no such thing as a flawless tool; none of the available options has managed to surpass the competition across the board.

The research [39] indicated that AutoML techniques have shown growing success with tabular data, and it also mentioned whether the AutoML approach can be used in text classification issues effectively. for this reason, four AutoML libraries were applied on a number of public datasets and two comparisons were made; a comparison between AutoML libraries' performance to find how these libraries perform. the second comparison was between the performance of humans and AutoML tools performance. The findings found that AutoGluon tool outperformed the rest of the AutoML tools whereas autosklearn ranked second. additionally, these findings indicated that AutoML cannot typically beat humans in text classification problems, noting that some of these problems can be solved equally or better by AutoML approaches. the researchers saw this difference in performance will be reduced throughout the following years with the development of automated approaches increasingly.

3. METHODLOGY

Within the scope of this chapter, we will explain the methodology which is employed for:

a. Creating the models using classic ML and DL techniques.

b. Finding optimal models using AutoML frameworks.

As we showed in chapter one, our contribution concentrates on building traditional models, which require human efforts, and testing AutoML frameworks to obtain the optimal models. These two approaches worked in conjunction with internal embedding, NLP features and word embeddings for capturing the linguistic features of news for evaluating the performance of traditional and AutoML approaches.

Also, as shown in chapter two, many researchers proposed traditional methods to solve fake news problem but did not pay attention to use AutoML tools for solving this phenomenon. For this purpose, we will utilize the liar Dataset which is verified by Politifact.com and we will discuss the models and frameworks that were developed and some related theories.

Figure 3.1 depicts the thesis's overall structure. We begin by introducing our Dataset and detailing how it is collected; next, we discuss how we pre-processed the data making use of natural language processing, word embeddings, and internal embeddings; finally, we outline the techniques, frameworks, and approaches that we will employ. The theoretical explanations for the methodology's components will be discussed in the following sections.

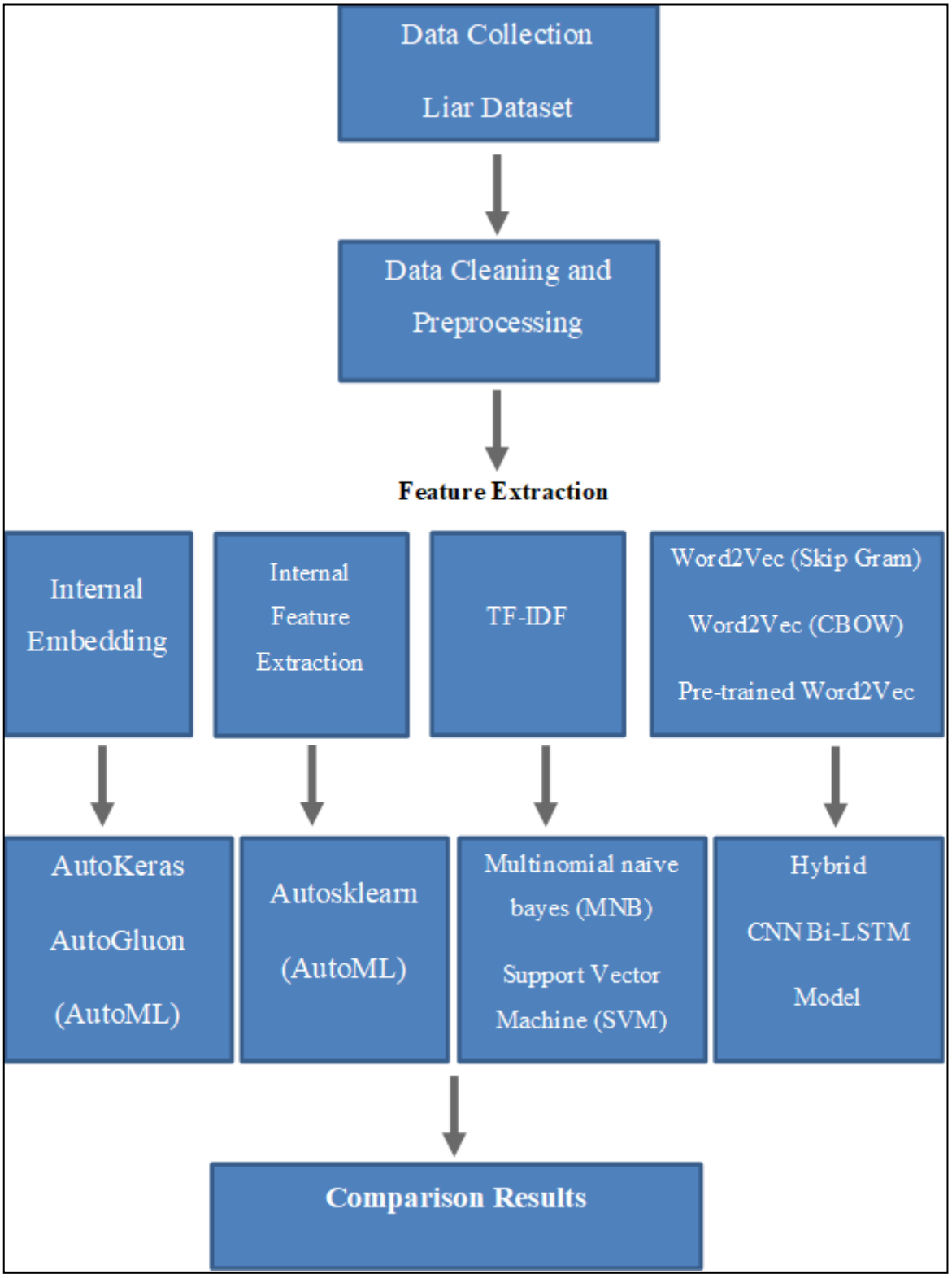


Figure 3.1: Overview of The Methodology.

3.1 DATA COLLECTION

The first stage in this work is the collection of data, which is covered in this section. The datasets we used is Liar dataset [40]. in general, datasets are collected using programming techniques by either scraping news websites or by sending request to an API [41]. We did not use these techniques during our work, instead we just used the datasets that were uploaded to (www.cs.ucsb.edu) website.

3.1.1 Liar Dataset

The LIAR Dataset is a free, accessible resource that has been utilized in studies on the identification of act-checking and fake news. It was made in 2017 and includes 12.8K short statements from POLITIFACT.COM in different contexts. for each statement, POLITIFACT.COM also provided thorough reports and links to the source documents. Every statement is verified for truthfulness by a POLITIFACT.COM editor. The statement dates are primarily from 2007-2016 and labelled into six classes [40]. Truth-O-Meter is used to determine how true each statement is. There are six possible ratings on the truthfulness scale, decreasing in intensity [42] . Table 3.2 describe the type of labels [43]. The person making the statement has to prove it, and the rating of the statement is based on what is known at the time it is made. When submitting the report to an assigning editor, a rating will be offered by the reporter who conducted the research and wrote the fact-check. Both the reporter and the editor go through the report and usually make changes and add more information. They agree on how it should be rated. After a rating has been given, the fact check is shared with another two editors by the assigning editor. The fact-check is reviewed by the three editors and the reporter, who talk about it and answer the questions below [43].

a. Is the statement true literally

b. Is there a different way to read the statement? Is the statement susceptible to interpretation

c. Was there any proof offered by the speaker? Did the speaker demonstrate that the statement was true

d. How have we responded to other statements before? What is the jurisprudence of PolitiFact

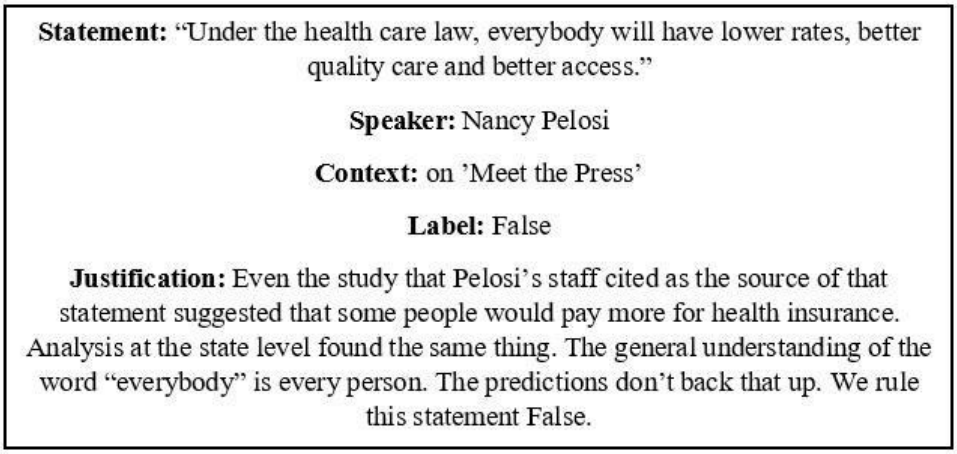
After discussing the rating, the three editors will vote on it (with the majority rule being two votes) where they change it to a different rating, and sometimes they stick with the reporter's suggestion. After a few more rounds of revisions, the report is released [43]. With the exception of 1,050 occurrences of "pants-fire," The LIAR dataset's labelling distribution is quite balanced. The number of instances for the remaining labels varies between 2,063 and 2,638 [40]. Figure 3.2 show an example of one of LIAR dataset records. Table 3.1 explains liar dataset columns.

Table 3.1: The Liar Dataset Columns [42].

The feature	The description
Id	The statement identifier
Statement	It contains short statements in English language
Label	the label of statement
Subject	The statement subject
Speaker	The statement source
Speaker job title	The job title of the source
Party affiliation	The Information related to party affiliation
State info	State of source
Barely true counts	how many barely true statements that source has made
False counts	how many false statements that source has made
Half true counts	how many half-true statements that source has made
Mostly true counts	how many mostly-true statements that source has made
Pants on fire counts	how many pants-fire statements that source has made
Context	The place of the statement's making

Table 3.2: The Label Types [43].

Label	description
TRUE	The statement is accurate and there's nothing significant missing.
MOSTLY TRUE	The statement is accurate but needs clarification or additional information.
HALF TRUE	The statement is partially accurate but leaves out important details or takes things out of context.
MOSTLY FALSE	The statement contains an element of truth but ignores critical facts that would give a different impression.
FALSE	The statement is not accurate.
PANTS ON FIRE	The statement is not accurate and makes a ridiculous claim.


Statement: "Under the health care law, everybody will have lower rates, better quality care and better access."

Speaker: Nancy Pelosi

Context: on 'Meet the Press'

Label: False

Justification: Even the study that Pelosi's staff cited as the source of that statement suggested that some people would pay more for health insurance. Analysis at the state level found the same thing. The general understanding of the word "everybody" is every person. The predictions don't back that up. We rule this statement False.

Figure 3.2: An Instance of One of Liar Dataset Records [40].

3.2 NATURAL LANGUAGE PROCESSING (NLP)

The term "natural language" describes how people communicate with each other by speech or text. Communication, which is speech or text, includes massive amounts of information. The way someone talks, the words they use, etc., all convey some kind of information. Even though text contains a wealth of information, machines cannot interpret it, thus we must transform it into a language they can comprehend. To do this, we employ natural language

processing (NLP) [44]. In fact, NLP is a tract of Artificial Intelligence and Linguistics, devoted to make computers understand the statements or words written in human languages [45]. NLP is a clever and practical method that allows computers to study human language and derive meaning from it [46].

The growth of computer science in the 1940s may have been the start of NLP, and improvements in artificial intelligence have also contributed to our growing knowledge of NLP. Natural language understanding (NLU) was added to the challenge area as the 20th century went on, expanding the area of NLP and enabling computers to respond to commands properly. For instance, a chatbot that utilizes natural language understanding (NLU) would be able to take a request like "find food near me" and turn it into a series of steps that would provide the desired outcome for the user. As we go closer to the present, we see that over the past 20 years, NLP has seen an increase in attention with the explosive growth of machine learning. Part of this is due to the broader availability of the huge repositories of labelled data sets, and computational power has increased, contributing to this in part [47].

There are two main branches of NLP: Natural Language Generation (NLG) and Natural Language Understanding (NLU). These categories develop the work of understanding and producing text. The general categorization of NLP is shown in Figure 3.3. With the use of natural language understanding, computers can understand and analyze human speech by extracting concepts, emotions, entities, keywords etc. Using an internal representation, NLG creates meaningful phrases, sentences, and paragraphs [45]. NLP is being utilized to assist us in our day-to-day lives through many applications such as [44]:

a. Translation: Google Translate

b. Spell checking: Grammarly, Microsoft Office.

c. Personal assistants like: Google Assistant, Cortana, Siri.

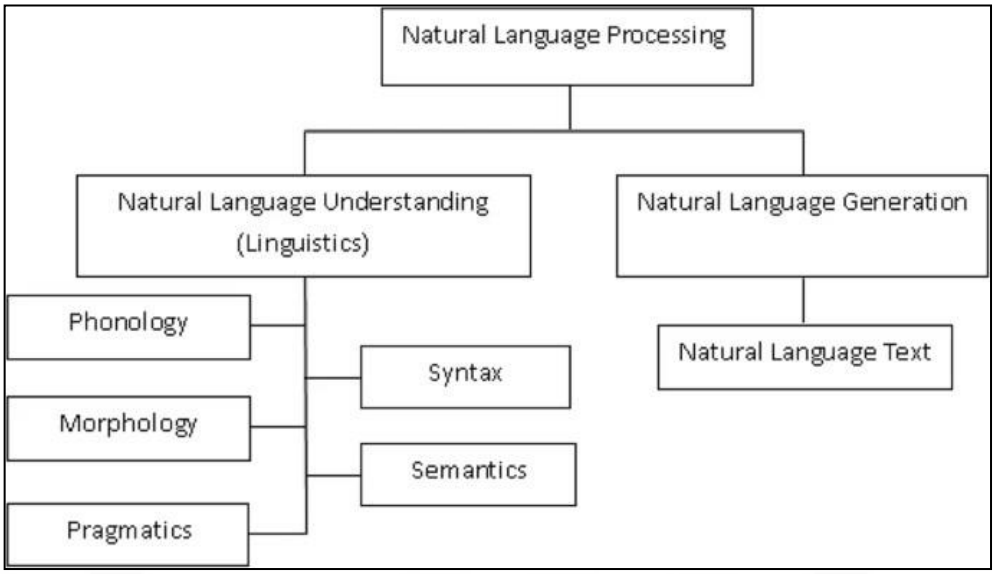


Figure 3.3: The General Categorization of NLP [45].

In NLP, stemming and lemmatization are used to reduce words to their root form., so that there is less variation between similar data. And then converted into numerical format using encoders like One-hot encoding, Count Vectorizer and TF-IDF Vectorizer [47]. these techniques will be explained in the next sections in this chapter.

3.2.1 Data Preprocessing

The fundamental step in this process is how the computer reads the text, or how a word is changed into a numerical representation. various pre-processing steps usually are necessary to have a suitable feature vector representation [42]. The purpose of preprocessing is to break down the text into individual words and represent each document as a feature vector. in the proposed classifiers, the text documents are represented as transactions. The most crucial part of the preprocessing phase for indexing texts is selecting the keyword, which is the process of choosing features. The success of this stage will determine the effectiveness of the categorization stage, which comes next. Important keywords that convey meaning should be chosen, and words that do not help differentiate across texts should be discarded [48]. In addition to being a crucial stage in getting the corpus ready for modelling, text preprocessing is also a crucial area that directly influences the outcomes of NLP applications [49].

We used several text preprocessing steps in our work which included spell correction, contractions adding, stop-word removal, lowercase conversion, stemming and tokenization.

The first step in the text preprocessing pipeline is the spell correction for words in the corpus. The review data should be checked for misspelt terms that need to be fixed since they may radically alter the meaning of a phrase or review [50].

Lowercase conversion is another often used preprocessing procedure for text categorization. All uppercase letters are often changed to their lowercase equivalents prior to the categorization since it is thought that there is no difference between words written in uppercase or lowercase [51].

Another step is the processing of contraction. contractions are the concise form of characters or words. In English, the location of missing letters is indicated by using an apostrophe. In written English, we often shorten phrases or eliminate whole words, creating contractions such as doesn't for does not [51].

The next step is stemming process in which Stemming methods are used to determine a word's root or stem. Stemming converts words to their stems, a process that requires a substantial amount of language-specific linguistic expertise. Words that share the same stem or word root are commonly employed to express the same or similar ideas in literature, which is why stems may be used to conflate words. For instance, the stem of the words: user, users, used, and utilizing is 'USE'. The Porter Stemmer method [52], the most generally used algorithm in English, is used in this study [48].

Stop words are words that occur repeatedly in texts but have no meaningful connection to the subject matter (e.g., prepositions, conjunctions, etc.). Since stop words are often considered to be irrelevant, they are typically removed prior to categorization in text classification studies. In the same way as stemming is unique to the language being studied, stop words are also unique [51].

The last step in our text preprocessing is the tokenization. In text processing, Tokenization is the process of breaking down a text into tokens, which may be words, sentences, or other meaningful units. thus, tokenization is a segmentation technique. Alphanumeric characters and alphabetic characters bounded by non-alphanumeric characters are considered in the segmentation. (e.g., whitespace, punctuations) [51].

3.2.2 Features

Formal feature extraction techniques can be employed to the data once it has been processed and tokenized. Texts and documents are examples of what are called "unstructured data sets" which must be converted into a structured feature space by applying mathematical modelling [53]. characteristics are extracted from the text in an iterative procedure that seeks to choose and combine multiple pieces of information to provide the most accurate description of the source text [42]. The common techniques of feature extractions are Term Frequency-Inverse Document Frequency (TF-IDF), Word2Vec, and Global Vectors for Word Representation (GloVe) which are either weighted word or word embedding techniques [53]. In the next sections, we will review the methods we used for extracting text features.

3.2.2.1 Tf-idf technique

Features for the extraction of keywords and keyphrases may be divided intuitively into two major groups: Features based on phraseness, and Features based on informativeness. For explanation, A keyphrase connotes a multi-word lexeme (e.g., hard disk, computer science), whereas a keyword is a single-word term (e.g., disk, computer). Features based on phraseness provide insight into the degree to which various components of a phrase are bonded together or the degree to which the generated phrase qualifies as a phrase. We may determine the present keyword or keyphrase's importance using features based on informativeness.

It is possible to further categorize features based on informativeness into three groups: based on placement in the text, based on term weight, and Miscellaneous. What matters to us, in this case, is type, which establishes the significance of a word in a text as well as in a corpus (based on term weight). In a text corpus, term weighting techniques can help detect stopwords and keywords [54]. A numerical statistic called Term Frequency-Inverse Document Frequency aims to show how significant a word is to a document inside a corpus or group of documents. In text mining and information retrieval, it is frequently employed as a weighting factor [55]. To put it simply, the TF-IDF technique compares the relative frequency of words in a given text to the inverse proportion of that word over the whole corpus of documents. This computation, intuitively, calculates the degree of relevance of a word in a certain document. When compared to common words like articles and

prepositions, words that are often used in a single or small set of texts typically have higher TF-IDF scores [56].

Equation 1 shows how to calculate TF-IDF [57]:

TF-IDF(t,d,D) = tf(t,d) X idf (t,D) (1)

Where:

a. TF-IDF(t,d,D) = TF-IDF weight of a term calculation result

b. tf(t,d) (term frequency) = how often a term appears in a text or category.

c. idf(t,D) (inverse document frequency) = calculation of log multiplied by inverse probability of a term being found in any essay.

The TF-IDF training set example is shown in Table 3.3. Suppose we need to compute the weight of term “learning” in the first and third document. the word “learning” occurs twice in the first document and first in the third document, and that term were written in two documents. Hence, the computation is as shown in:

a. TF-IDF(learning,1) = 2 * log (3/2) = 0.46

b. TF-IDF(learning,3) = 1 * log (3/2) = 0.23

In table 3.3, the “learning” word appeared with two various weights in the document 1 and document 3. This occurs since the word “learning” has more affection in the first document.

Table 3.3: TF-IDF Technique Example.

Seq.	Doc.
1	Keep learning and learning
2	The life is beautiful
3	The learning is very important for us

3.2.2.2 Word2vec technique

Computers are required to process a significant quantity of data relating to natural language to perform well in the field of NLP. Natural language must be converted into a digital language for computers to understand it since computers cannot read like humans. it is the quantifying words process. However, even though the computer "understands" real language, it is still unable to comprehend the semantic relationships between words. The fact that Google resolved to address this issue by announcing Word2vec at the tail end of 2013 is an effective solution [58].

The word2vec approach is an innovative model architecture for obtaining continuous vector representations of words from massive datasets. Similarity between words is used to evaluate the accuracy of these representations. word2vec focused on distributed representations of words learned by neural networks. word2vec model has two architectural variants: continuous-bag-of-words (CBOW) which try to predict the current word based on the context, and skip-gram which use the current word to predict surrounding words. CBOW architecture consists of input, projection, and output layers. the projection layer is shared for all words. thus, all words get projected into the same position (their vectors are averaged). in this architecture, the order of words in the history does not influence the projection. The Continuous Skip-gram Model is the second architecture. It's similar to CBOW, except rather than trying to predict the current word based on the context, it seeks to maximize the categorization of a word based on another word in the same phrase [59].The figure 3.4 explains the architecture of the two types of word2vec technique.

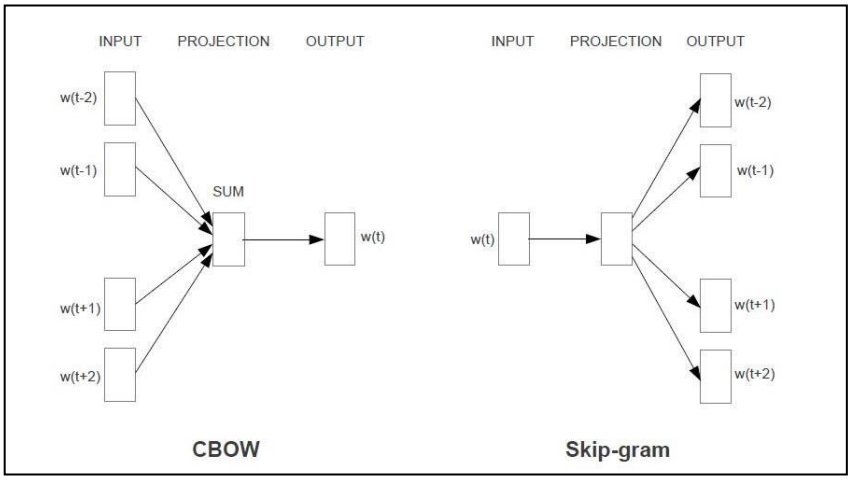


Figure 3.4: Word2Vec Technique [59].

3.2.2.3 Glove technique

The GloVe is a worldwide log bilinear regression method that combines the advantages of global matrix factorization and local context window techniques. By limiting training to the nonzero elements of a word-word cooccurrence matrix rather than the full sparse matrix or specific context windows in a huge corpus, this technique effectively makes use of statistical data. The model generates a vector space that has a significant substructure. On named entity recognition and similarity tasks, it performs better than comparable models.

Despite new techniques for learning vector space representations of words have effectively captured fine-grained semantic and syntactic regularities using vector arithmetic, the source of these regularities has remained a mystery. The GloVe method examined and identified the model characteristics required for such regularities to appear in word vectors [60].

3.3 MODELS

In this section, an overview of ML, DL, and AutoML will be given, and the models of ML and DL as well as AutoML libraries, which will be utilized for the experiments, will be explained to give insight concerning these algorithmic methods and technologies. Also, some pseudocode and/or mathematics that show the difficulty of ML, DL, and AutoML approaches will be included in this section. Because the solution to our problem is concerned with classifying the text, we will discuss the models that are relevant to this kind of problem.

3.3.1 Machine Learning

The process of acquiring new or changing existing behaviours, attitudes, knowledge, talents, or preferences is the general definition of learning. A subtype of artificial intelligence called machine learning (ML) enables computers to reason and learn on their own. It all boils down to changing the actions that computers do to improve their accuracy, which is based on how often the chosen actions result in the correct ones. The broad field of machine learning is supported by a number of research areas [61] .

Huge training data sets, effective model training techniques, and accurate inference make up a productive machine learning system. Several variables, including the diversity of the machine learning process, may affect how effectively the process works. Each process may benefit from the variation in ensuring flawless machine learning. The training set's variety guarantees it could supply the model with more discriminative data. Due to the diversity of the learned model, each parameter or model may gather unique or complementary information. There may be a wide range of possibilities available thanks to the diversity of inferences, each of which relates to a certain likely local optimal result [62].

Machine learning is a fast-growing subject of computer algorithms intended to emulate human intelligence by learning from their surroundings. The workhorses of the vast new data era, they are so called. A wide range of industries, including pattern recognition, computer vision, finance, entertainment, biological and medical applications, spacecraft engineering, and computational biology, have successfully employed machine learning techniques [63].

3.3.1.1 Machine learning algorithms

In our work, we essentially concentrate on the Supervised Learning algorithms with classification type as the training data's labels are previously known, and we have a classification issue to solve. For dealing with binary or multi-class text classification challenges, Support Vector Machines and Naive Bayes are among the most popular and successful approaches. We will examine the theoretical aspect of these algorithms in the sections that follow.

Naïve bayes algorithm:

One of the ML algorithms is the naïve bayes (NB). it is referred to described as "naive" because it implies that the appearance of one feature has no bearing on the appearance of other features. Using the NB method, it is possible to rapidly generate and forecast models. naive bayes classifier (NBC) has good outcomes for textual data analysis. Due to its efficient grating assumptions and rapid and simple implementation, NB is a particularly popular approach to categorize text. with naive bayes technique, A limited amount of training data is needed to estimate the necessary parameters. A form of NB classifier that is frequently used as a baseline to classify text is a multinomial nave bayes classifier (MNB) [64]. When compared to other naive bayes classifiers, such as Bernoulli nave bayes, research shown that multinomial naive bayes gained a marginal advantage in the area of text categorization [65],[66]. NB is a straightforward probabilistic classifier that determines a set of probabilities by counting the frequency and combinations of values for each dataset. The training dataset is utilized for training a classifier technique by using known values for forecasting future, unknown values [67]. NB classifiers are a collection of classifiers built on the well-known probability theorem of Bayes [64]. The conditional probability is calculated using Bayes' theorem, a mathematical formula that bears the name of the British mathematician Thomas Bayes from the 18th century [68].

$$P(A|B) = \frac{P(A)P(B|A)}{P(B)} \tag{3.1}$$

Where:

The likelihood of the incidence of event A when event B happens is represented as P(A|B),

The likelihood that event A will take place is denoted by the probability function P(A),

The likelihood of the incidence of event B when event A happens is represented as P(B|A),

The likelihood that event B will take place is denoted by the probability function P(B) [68].

Support Vector Machine Algorithm*:*

The Support Vector Machine, or SVM for short, is a popular method of supervised machine learning that may be used to a wide variety of classification issues. the SVM seeks to identify the best line in 2D or the most suitable hyperplane in an N-dimensional space for

categorizing data points. Such a hyperplane is selected to separate the two groups of points by the largest possible margin. On the basis of their placement on either side of the plane, data points may be divided into several groups. Maximizing the distance between the hyperplane and data points is the primary goal of the SVM approach. the hinge loss function is the function that is responsible for maximizing the margin [69].

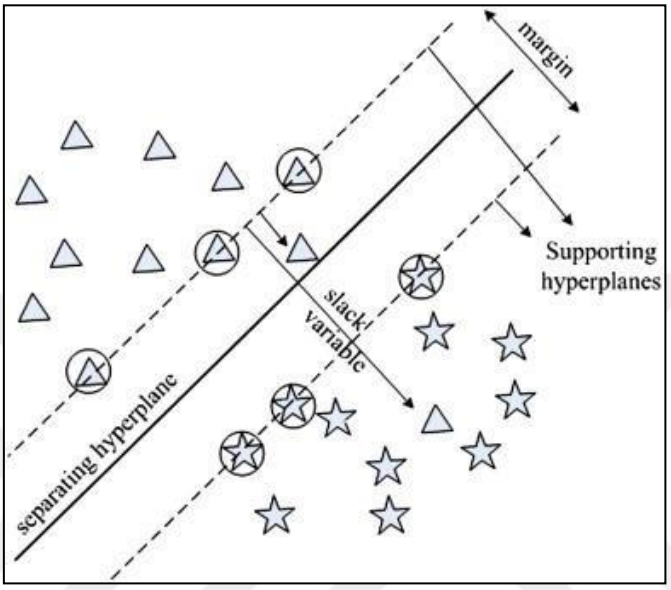


Figure 3.5: The SVM Algorithm Concept [70].

Figure 3.5 provides an illustration of the SVM method in practice, showing how two classes (stars and triangles) must be divided using a linear hyperplane. The support vectors are on two class boundary hyperplanes (dashed lines) and have a circle as their border. Between and parallel to the class boundary hyperplanes is the ideal separating hyperplane, which is a solid line. The "margin" is the space between the class border hyperplanes, while "slack variables" are those that exist between a class boundary and its outliers [70].

3.3.2 Deep Learning Approach

Neurons are cells found in the nervous systems of living things that are responsible for processing and transmitting information within the body. Every neuron has a simple function, but when they all work together, the results are complex [71]. About 100 billion neurons make up an individual's brain, which is very high for a living organism [72].

In a neural network, a neuron serves as the basic data processor. It takes in information from a number of different sources, combines it in some way, applies some operation—typically one that is not linear—to the resulting data, and then sends that information out as output. There are four basic main components of all natural neurons.

Dendrites, soma, axon, and synapses are these parts. The input channels are the dendrites. Synapses between different neurons provide the information for these input channels. After receiving these signals, the soma gradually processes them. The soma converts the result of this processing into an output, which is subsequently sent to neighbouring neurons through axon and synapses. The relationship between these four parts is seen in Figure 3.6. Now, neural networks are the straightforward grouping of very basic artificial neurons. This grouping is accomplished by constructing connected layers [73].

The structure and function of the human brain's neuronal network serves as a model for the design of Artificial Neural Networks. Via neuro synaptic connections, information is processed and sent from one neuron to the next. ANN are structured similarly, with several layers of cells interconnected with one another. In a neural network, the output or information from one layer is used as input by the next layer, and so on, until it reaches the final layer, which produces the output. As a means of facilitating additional processing, the output of the inner layers is given non-linearity as the input to the outer layer [74].

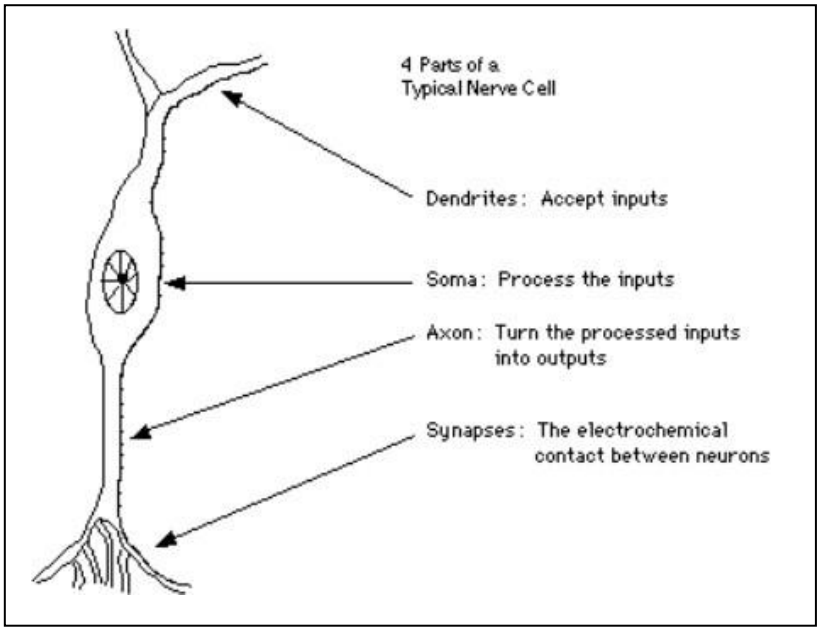


Figure 3.6: The Neuron Components [73].

Neurons are the processing units that make up an ANN. An artificial neuron attempts to imitate the way a natural neuron is built and how it works. A neuron has inputs, called dendrites, and one output, called an axon (synapse via axon). The activation of the neuron is determined by the neuron's function [75]. The building blocks of an ANN are as follows [76]:

a. An input layer: it takes in the data.

b. A hidden layer(s): between the input and output layers, there may be one or more hidden layers of neurons.

c. An output layer: in most cases, it consists of a single neuron whose output is between zero and one. But it's possible for there to be more than one output.

A deep neural network (DNN) consists of a number of hidden layers, which means that more neurons are used in the model's building. Thus, the DNN is an ANN with several hidden layers, and DL is the process of training DNNs. In DL, a deep topology is employed for mapping associations between input variables and the result [77]. Figure 3.7 show the structure of both neural network and Deep learning (deep neural network).

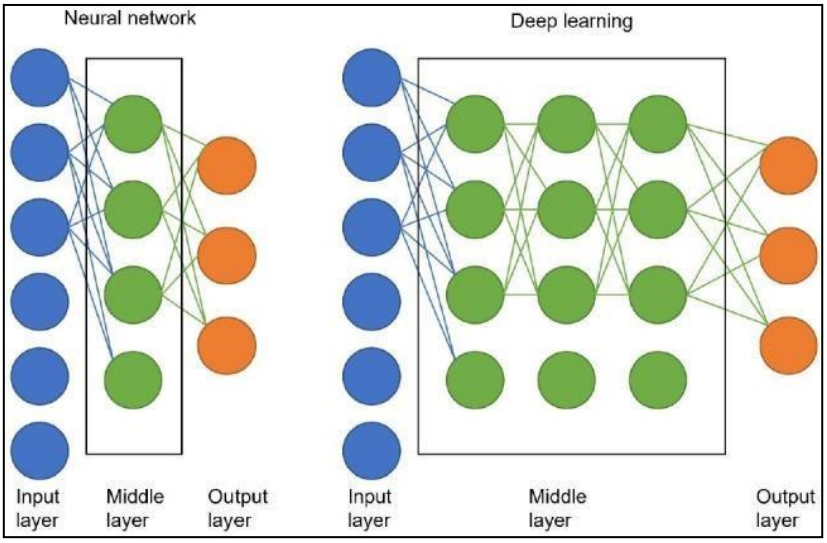


Figure 3.7: Aneural Network and Deep Learning (Deep Neural Network) [78].

3.3.2.1 Bidirectional-lstm model

When it comes to predicting the next word in a phrase based on the words that have previously been provided, Among the most popular DL models is the recurrent neural network (RNN)in this regard. This network is mainly employed for NLP-related issues. RNN uses back-propagation in the same way as conventional neural networks do. RNN, on the other hand, has issues with vanishing problems and gradient exploding. Because of these two obstacles, RNN models may be difficult to train and fine-tune their parameters. Back-propagation is where these sorts of problems usually manifest [79].

An RNN model called Long Short-Term Memory (LSTM) [80] was designed to address the aforementioned issues. RNN's structure is modified by LSTM. It changes the RNN layer into a structure with a gate and a memory cell. LSTM is designed to permanently store information in the memory cell for later usage and modification. By using this novel structure, LSTM is able to address the gradient exploding and vanishing issues that plague RNNs [79]. The LSTM architecture is seen in Figure 3.8 [81].

One RNN method, Bi-LSTM, is designed to fix LSTM's problems with text sequence characteristics. Unlike traditional LSTM, which only allows for one direction of information flow, Bi-LSTM allows for two-way information flow (from the back to the front) by employing two hidden states and allowing data to flow in both ways [79]. The Bi-LSTM design is seen in Figure 3.9 [81]. Bi-LSTM is superior to the conventional RNN model, which requires decay to include future data since it retains input data of both the previous and subsequent sequences [79].

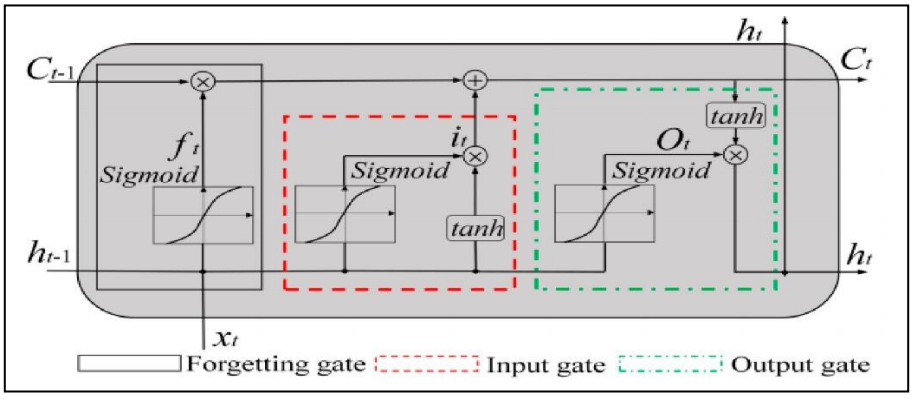


Figure 3.8: The LSTM Architecture [81].

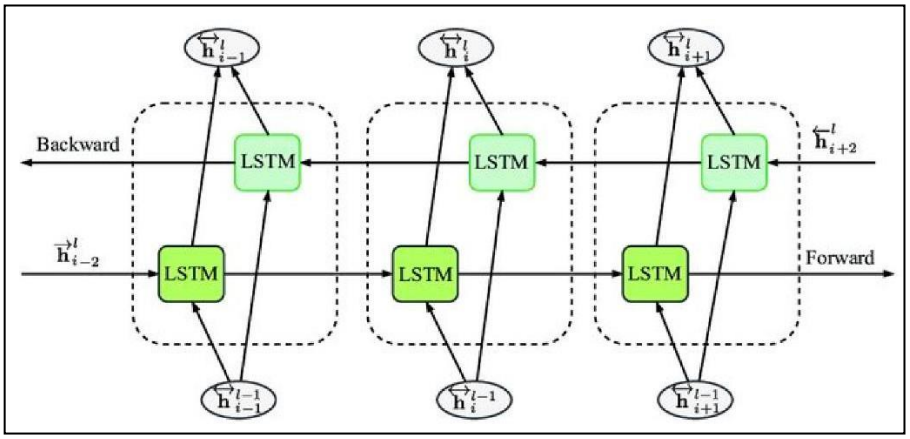


Figure 3.9: The Bi-LSTM Architecture [81].

3.3.2.2 Convolutional neural network

For computer vision and image processing, CNNs, which are themselves a type of neural network, have proven to be highly effective and innovative. With the help of Figure 3.10, we can see the basic structure of a CNN. The graphic explains the structure of a CNN, which includes the input layer, the convolutional layer, the pooling layer, and the fully connected layer. The input layer's job is to receive the image's pixel value as an input. The next layer, called the convolution layer (CONV), is responsible for generating output based on the values of its kernel or filter. Convolution's output is sent into a Pooling Layer (POOL), which reduces the dimensionality of the representation and speeds up the calculation. Max pooling, which takes the maximum value from each window, is the most common form of pooling. in the fully connected layer, every neuron in this layer is linked to all the activations of the preceding layer, as is typical of neural networks [82].

CNNs were first created for the purpose of processing images. Then, several research used CNN for NLP tasks. The typical CNN used for NLP applications differs in that it employs 1D convolution layers and 1D pooling layers instead of 2D ones [83].

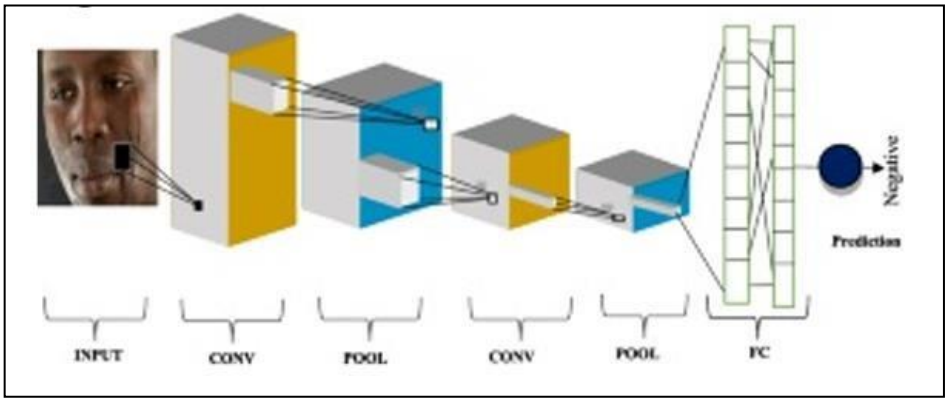


Figure 3.10: The Cnn Architecture [82].

Let's say we have a sequence of words $W_{i:n} = W_1, W_2, W_3...W_n$, and there is a d-dimensional embedding vector is for Each word W_i. A 1d convolution of width k moves along the words in the sentence like a sliding window of size k i.e. each window contains k words. Each window of k words. The convolution filter or kernel is applied to each window. For example, consider a i_{th} window of words w_i , $w_i+1...w_{i+k}$ This i_{th} window is represented by xi = [w_i , w_{i+1}, ...w_{i+k}] ∈ R^{d*k}.

The scalar feature c_i is generated when applying a convolutional filter v of size R d∗k to i^th window.

$$ci = g(v * x_i) \in R \tag{3.2}$$

in this case, g is a nonlinear function that is reLu function in general. Applying this filter to each conceivable window of words yields a resulting sequence of features c.

$$c = [c_1, c_2...c_{n-k+1}] \in R^{n-k+1} \tag{3.3}$$

Practically, as illustrated in Figure 3.11, many filters v1, v2,...vl are used for producing output $r_i \in R^{l*(n-k+1)}$. This represents the result of CNN's i_{th} layer. next, a pooling procedure, often max-polling, is performed. The max-pooling technique extracts maximum values from a sliding window of a defined size that comprises a succession of features c. The objective is to extract the most significant features from a series of features [84].

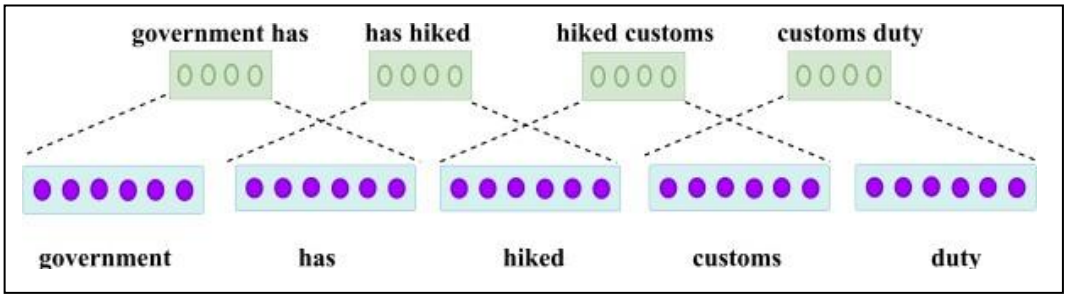


Figure 3.11: Asentence Convolution Example [84].

3.4 AUTOMATED MACHINE LEARNING

The model employed for the prediction objective is the heart of any machine learning pipeline. But for the same problem, there are different model configurations that may provide different degrees of accuracy. Consequently, it is essential to establish the optimal model considering the accuracy-execution time tradeoff. When a model is complete, the last step is a manual optimization of its hyperparameters for producing the final model. These previously mentioned stages can be automated using automated machine learning (AutoML), which helps reduce reliance on humans. These previously mentioned stages can be automated using AutoML, which helps reduce reliance on humans [85]. Figure 3.12 shows how the task performed manually.

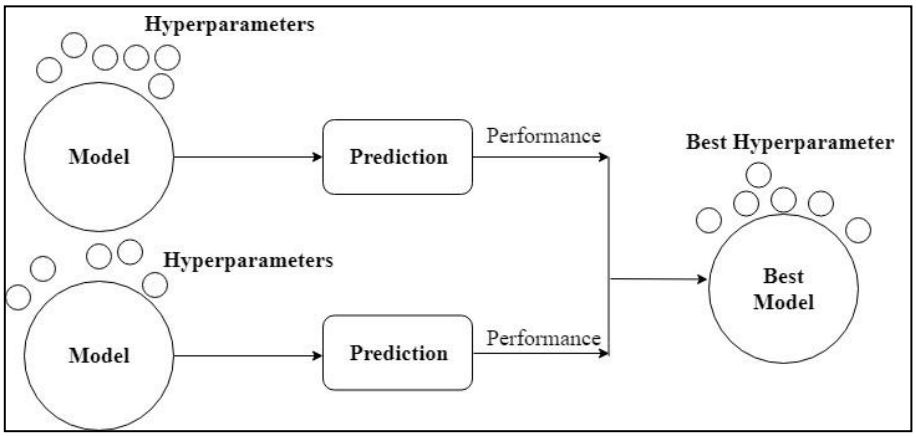


Figure 3.12: Manual Task By Humans [85].

AutoML intends to automate the whole machine learning process. Figure 3.13 [33] illustrates the several stages included in the AutoML pipeline. These stages include data preparation, feature engineering, model generation, and model evaluation.

Data preparation is the first stage in the ML pipeline. The process of preparing the data may be broken down into three distinct steps: collecting the data, cleaning the data, and augmenting the data. The gathering of data is an essential phase in the process of building a new dataset or expanding an existing dataset. The procedure of data cleaning is employed to filter noisy data in order to ensure that the training of models that come after it is not affected. When it comes to boosting model robustness and performance, data augmentation is crucial. Right now, no AutoML tool is particularly good at handling the data preprocessing duty, hence a lot of work in this area is being done manually [33].

Data preparation for machine learning relies heavily on a process known as feature engineering. Better predictive performance may be achieved by the process of creating appropriate features from existing features [86].

Feature engineering is crucial yet labor-intensive. choosing the right features is crucial to the success of ML methods. Accordingly, the majority of the work involved in deploying ML algorithms is thus spent on designing preprocessing pipelines and data transformations which provide a representation of the data that can allow efficient machine learning [87].

Feature engineering divides into these three subfields: feature selection, feature extraction, and feature construction. Extraction, selection, and construction of features aim to achieve three goals: data reduction, emphasis on important data, and enhancing data quality and hence the efficacy of data mining algorithms. Choosing a subset of the original features to minimize the feature space according to some criteria is what we mean when we speak about feature selection. In order to build a collection of new features, a procedure called feature extraction/construction must be carried out. They may function alone or in tandem with one another. All try to enhance performance, including estimated accuracy, ease of understanding of acquired knowledge, and visualization.

It is possible to use feature selection, extraction, and construction together. many times, the number of features may grow during feature construction, and although the newly generated features are more expressive, they can also contain unnecessary features. With the use of feature selection, it is possible to automatically eliminate unnecessary ones [88].

As can be seen in Figure 3.12, model generation consists of two parts: the search space and optimization techniques. Model structures that can, in theory, be designed and optimized are

set by the search space. Models may be broken down into two groups: classical ML models like naive bayes, and deep neural networks. According to parameters, the optimization algorithms use two sorts of parameters: hyperparameters which are employed for training process like the learning rate, and parameters specified for the design of the model like filter size for DNN [33].

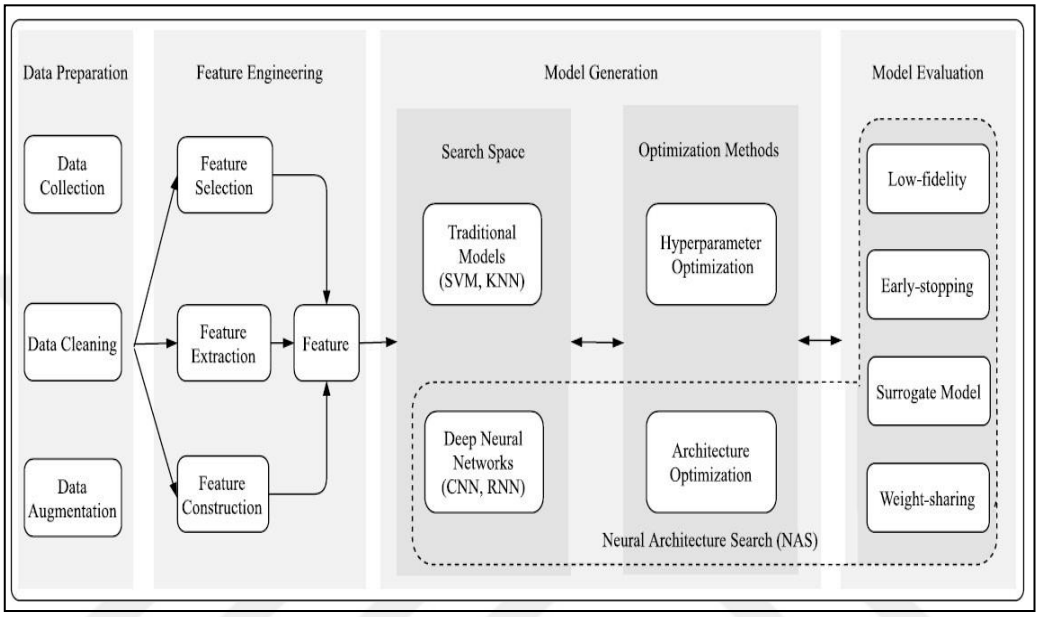


Figure 3.13: Model Generation Parts [33].

3.4.1 Hyper-Parameter Optimization

Hyper-parameters settings used by most ML algorithms have a significant impact on how well they function once training has started. These values might be continuous like the error rate of a neural network, discrete like number of neighbours in nearest neighbours, or categorical like the kernel function of a support vector machine. Hyperparameters affect a model's performance as well as how much time and memory it needs. On a validation subset of the data, hyper-parameters are often tweaked by a human expert who blends expertise with experimentation [89].

Hyper-parameter optimization (HPO) refers to the challenge of finding an appropriate value for a hyper-parameter [90]. There are several significant use cases for automated hyperparameter optimization., where it can lower the amount of labor-intensive manual work involved in implementing machine learning. In relation to AutoML, this is very significant.

Additionally, adapting ML algorithms to the work at hand may enhance their effectiveness, which has resulted in the latest outcomes for major ML benchmarks in a number of recent research [91].

3.4.2 Neural Architecture Search

Deep Neural Networks, often known as DNNs, have been shown to be very successful in a variety of applications that deal with the actual world including NLP, image categorization, and speech recognition, where they have been highly effective in a variety of fields. DNNs' architectures, which are often manually constructed with extensive skill, have a significant impact on their performance. But because it involves a lot of trial-and-error, it requires a considerable amount of effort. Additionally, it is difficult to implement because it requires a specialized level of knowledge.

Neural architecture search (NAS) is a form of technology which can automatically design the architectures of DNNs, where its goal is automating the architecture designs of DNNs [92].[93] divided the neural architecture search methods in accordance with the following dimensions: the space of search, the strategy of search and the strategy of performance estimation.

The neural architecture search space, which specifies the collection of all architectures that may be discovered by the search process, is regarded as the core component of any NAS method. A challenging trade-off occurs when creating a robust architectural search space: as the quantity of distinct architectures, which they are possible to represent, rises with more degrees of freedom, the difficulty of the optimization issue also rises. Inside the search space, High-performance architectures cannot be available If the architecture search space is excessively limited [94].

The search strategy, once the search space has been established, outlines how to guide the search to quickly find the ideal model architecture for optimum performance [33]. Additionally, exploring the search space is explained by the search strategy. The space of neural architectures can be explored using a variety of different search techniques such as gradient-based methods, reinforcement learning, evolutionary methods, Bayesian optimization, and random search [93]. Additionally, the strategy of Search is known as the Architecture Optimization Method [33].

The strategy of performance estimation: Finding systems with high prediction performance on data that has not been seen is a common use of NAS. The process of evaluating this performance is known as performance estimation. The simplest approach is to train and validate the model on data, but this approach has some disadvantages, where it is very computationally costly as well as it leads to a reduction in the number of architectures which can be explored. As a result, a lot of current research concentrates on creating methods which lower the expense of such performance assessments.

Figure 3.14 explains the NAS methods, where architecture A is chosen by the search strategy A from the predetermined search space A, and then the strategy of performance estimation receives the architecture and gives the search strategy an estimate of how well A will work [93].

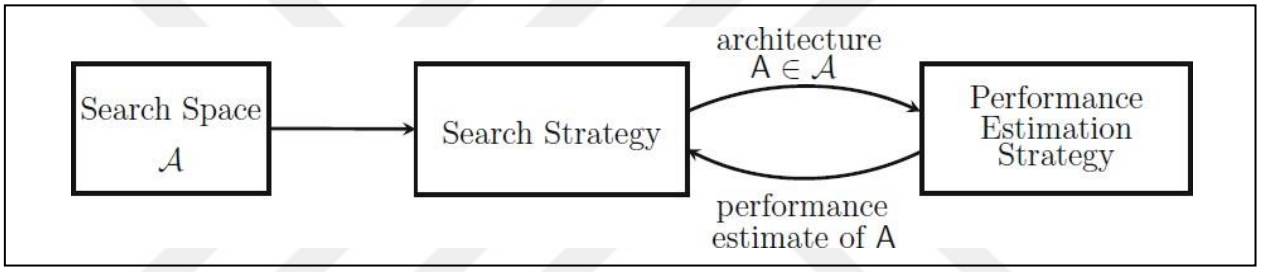


Figure 3.14: Neural Architecture Search methods [93].

3.4.3 Automated Machine Learning Frameworks

3.4.3.1 Auto-keras framework

Autokeras is a free and open-source AutoML framework which is possible to download and install locally. It has a very simple user interface. Auto-Keras concentrates on deep learning tasks and attempts to make machine learning methods accessible to users with little or no prior knowledge with the technology.

The Auto-Keras system's architecture is shown in the figure 3.15. It is made to fully use the computing power of both the GPU and the CPU while also making optimal use of the memory by just keeping what is presently needed in the RAM and saving the remaining on storage media like hard drives. The API is located at the top and is immediately contacted by users. To fulfil certain functions, it is in charge of contacting the appropriate middle-level modules.

The module of NAS algorithm known as The Searcher contains the Bayesian Optimizer and the Gaussian Process. The CPU is used to conduct these search algorithms. The GPU computation is handled by a module called the Model Trainer. For parallelism, it uses the training data to train specific neural networks in a separate process. The Searcher controls the Graph, a module that processes the computational graphs of neural networks, in order to perform network morphism operations. For quicker access, the current neural architecture in the graph is stored on RAM. Due to the large size of neural networks, it is impractical to keep all of the trained models in memory. Therefore, the model storage saves these trained models on storage devices [95].

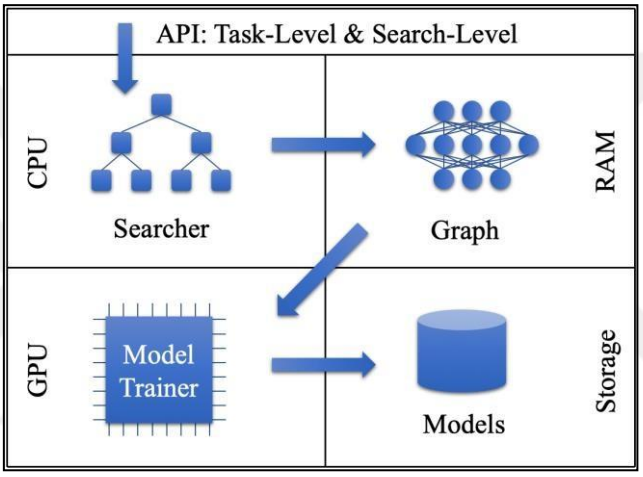


Figure 3.15: Auto-Keras Architecture [95].

3.4.3.2 Autogluon framework

In January 2020, Amazon created the Auto Gluon framework. It is very easy to use and completely free. With its assistance, models are built for text classification, image classification, object detection, and supervised learning. AutoGluon helps save time by automating time-consuming ML tasks. These tasks include dealing with missing or incorrect data, hyperparameter tuning, and finishing up an algorithm [96]. Users can train and use highly accurate ML and DL models on tabular data, text, image, and time series with only a few lines of code [97]. Tabular, textual, and image data are each handled by their own specialized sub-modules inside AutoGluon. What interests us in our work is the (autogluon.text) sub-module which is concerned with solving the NLP issues using (TextPredictor) class [98].

3.4.3.3 Auto-sklearn framework

Matthias Feurer, et al. created the open-source Auto-Sklearn, which was presented in the study "Efficient and Robust Automated Machine Learning" in 2015. It enables Python users to execute AutoML. for data transformations and machine learning techniques, it employs Scikit-Learn library. It also employs a Bayesian search technique for effectively finding the best model pipeline for a given dataset [99].

To be effective in practice, AutoML methods should automatically choose the best algorithm, feature preprocessing steps, and configure the appropriate hyperparameters for each dataset. With the use of effective Bayesian optimization techniques, Auto-sklearn has begun to tackle this AutoML challenge. This led to the introduction of this AutoML system based on Scikit-Learn (utilizing 4 data preprocessing techniques, 14 feature preprocessing techniques, and 15 classifiers, resulting in a structured hypothesis space with 110 hyperparameters).

For the AutoML technique, autosklearn makes two enhancements. First, it starts out to warm up the Bayesian optimization process by including a meta-learning phase, thus the efficiency will be increased significantly. Second, it has a phase for automatically creating ensembles that enables us to apply every classifier that was discovered through Bayesian optimization.

To be more specific, the meta-learning phase is used to choose instances of a particular machine learning framework that will have the best chance of succeeding with a fresh dataset. To be more precise, both a collection of meta-features and performance data and are gathered for a number of datasets.

These features are characteristics of the dataset which can be effectively calculated and that assist in choosing which method to apply to a new dataset. For improving an ML framework, this meta-learning strategy works in conjunction with Bayesian optimization. While meta-learning may immediately recommend certain instantiations of the machine learning framework that are expected to perform well, it is unable to provide fine-grained performance data.

However, Bayesian optimization starts slowly for hyperparameter fields as big as those of complete ML frameworks but may improve performance with time. Concerning the second improvement, we note that while Bayesian hyperparameter optimization is data-efficient in

finding the best-performing hyperparameter setting, it is a very wasteful procedure when the goal is to simply make good predictions. This is because all the models it trains during the search are lost, typically even some that perform nearly as well as the best.

Rather than discarding these models, it is proposed to store them and to use an efficient post-processing method (which can be run in a second process on-the-fly) to construct an ensemble out of them. This automatic ensemble construction avoids to commit itself to a single hyperparameter setting and is thus more robust (and less prone to overfitting) than using the point estimate that standard hyperparameter optimization yields. [100]. The whole procedure for auto-sklearn, including both improvements, is shown in figure 3.16.

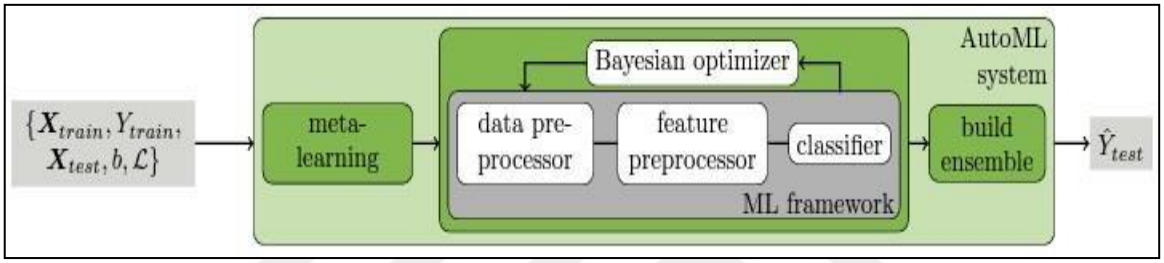


Figure 3.16: Auto-Sklearn Approach [100].

3.4.3.4 Tpot framework

The Tree-Based Pipeline Optimization Tool (TPOT) [101] is an open-source, Python-based AutoML tool that improves ML pipelines via the use of genetic programming in order to achieve the highest possible accuracy in a supervised classification problem. it automates the most time-consuming aspect of machine learning by automatically examining millions of different pipelines to discover the optimal pipeline for our data. After TPOT tool completes its search, it presents us with the Python code for the best pipeline it discovered, allowing us to modify the pipeline from there. TPOT is developed on top of scikit-learn library, thus the generated code should be familiar [101], [102].

Using a type of genetic programming (GP), popular evolutionary computation algorithm which construct computer programs automatically, TPOT leverage the ML pipelines. To combine the operators (Supervised Classification Operators, Feature Preprocessing Operators, and Feature Selection Operators) into the ML pipeline, TPOT treats them as GP primitives and construct GP trees from them. A tree-based pipeline, such as the one seen in

figure 3.17, receives two copies of the dataset, which are then updated sequentially by each operator, merged into a single data set, and then utilized for making classifications [101].

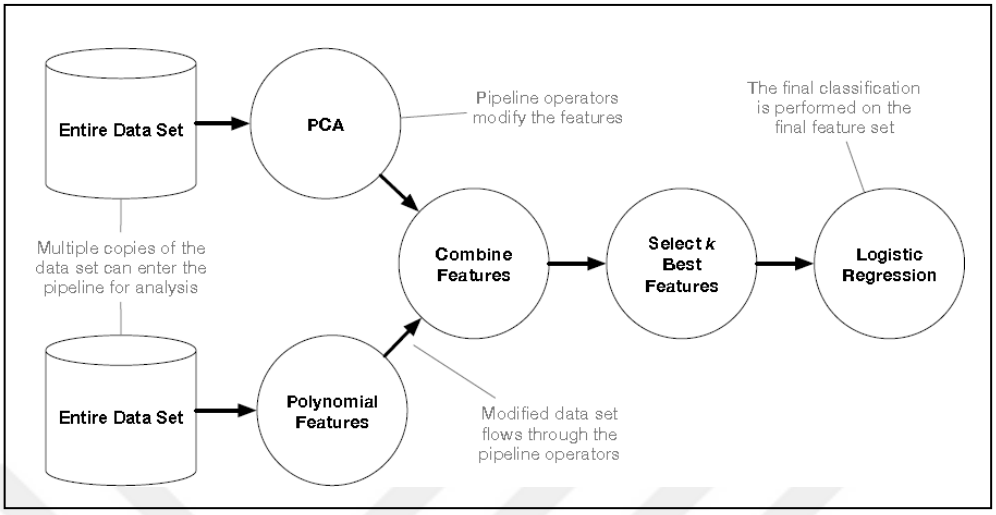


Figure 3.17: An Example Tree-Based Pipeline From TPOT [101].

TPOT tool utilizes genetic programming (GP) algorithm to generate and optimize tree-based pipelines automatically. A typical genetic programming procedure is followed by the TPOT GP algorithm. The genetic programming technique first creates 100 random tree-based pipelines and assesses their balanced cross-validation accuracy on the data set. The top 20 pipelines in the population are chosen for each generation of the GP algorithm using the NSGA-II selection scheme, where pipelines are chosen for maximizing the accuracy of the classification on the data set while reducing the total number of operators in the pipeline simultaneously. Each of the top 20 chosen pipelines produces 5 copies (i.e., offspring) into the population of the next generation, five percent of those offspring cross over with another offspring using one-point crossover, then ninety percent of the remaining unaffected offspring are altered randomly by a point, insert, or shrink mutation (1/3 probability of each). A Pareto front found at any time throughout the GP run is updated by the algorithm every generation. After 100 generations of running through the evaluate-select-crossover-mutate procedure, the algorithm settles on the highest-accuracy pipeline from the Pareto front as the representative "best" pipeline from the run. This procedure is repeated in order to add and tune pipeline operators which enhance the accuracy of the classification while removing those that degrade it [101].

3.5 EVALUATION METRICS

One of the most important steps in any predictive modelling pipeline is evaluating the results achieved by the ML model. This conclusion typically cannot be drawn depending on classification accuracy alone. For measuring the model's efficacy, several evaluation measures should be employed. One of the most useful tools for organizing and arranging the evaluation is the confusion matrix. Based on the true values, this matrix presents a review of how well the model performed when applied to the test data. Additionally, It reviews the model's success and provides helpful results of false negative(FN), true positive(TP), false positive(FP), and true negative (TN) [103]. These four terms will be explained in the next paragraph. The confusion matrix is represented by a table which can be created for a classifier for describing its performance [104]. The figure 3.18 displays the confusion matrix components [105].

		predicted	
		Negative	Positive
actual	Negative	TN	FP
	Positive	FN	TP

Figure 3.18: Confusion Matrix Components [105].

The matrix consists of the following terms: [104]

True Positives (TP) where both actual and prediction are yes.

False Negatives (FN) where both actual and prediction are no.

False Positives (FP) where actual is no, and prediction is yes.

True Negatives (TN) where actual is yes, and prediction is no.

Several measures are employed to determine whether a model works efficiently. These measures are called metrics. the metrics can be derived from a confusion matrix, and they provide us with the necessary approximation. Four well-known metrics were used in this thesis, which are as follows:

a. Accuracy

This measure is used in machine learning for evaluating the performance of classification models [106]. It is expressed as the percentage of correct predictions in proportion to all of the predictions produced by the model. Equation below is used for calculating the accuracy [103].

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

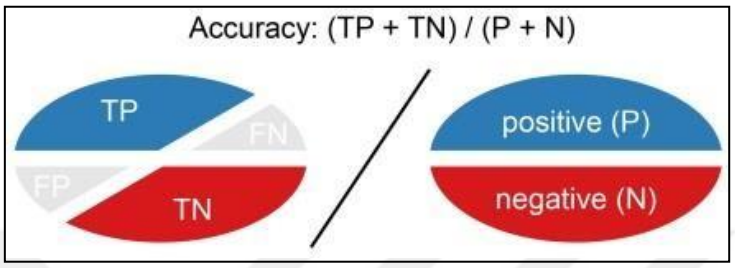


Figure 3.19: The Accuracy Metrics [105].

b. Precision

For machine learning classification models, precision is a statistical metric that quantifies correct positive predictions. Its definition is that it is the proportion of successfully predicted positive samples to all correct positive samples. It reveals what proportion of samples predicted as a certain class label has been samples of that class label. Low false-positive rates are obtained when the precision value is high. This statistic is crucial for figuring out when False Positives result in excessive costs [107]. Equation below is used for calculating the precision:

$$\text{Precision} = \frac{TP}{TP + FP}$$

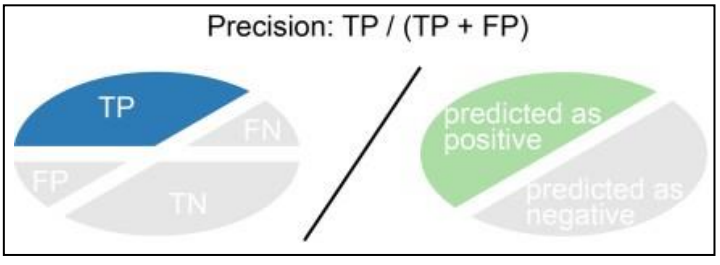


Figure 3.20: The Precision Metrics [105].

c. Recall

It is calculated as the total number of properly categorized positive samples divided by all the positive samples found in the dataset. Sensitivity is another name for the recall measure [106]. Equation below may be used to calculate the recall:

$$\text{Recall} = \frac{TP}{FN + TP}$$

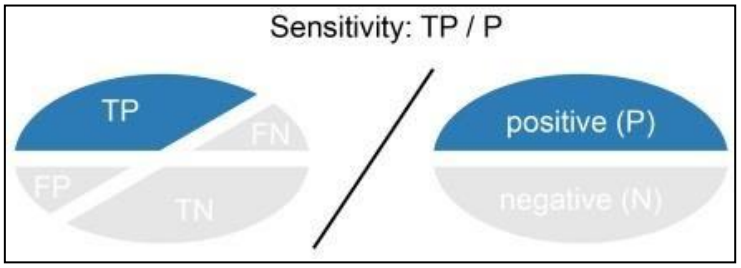


Figure 3.21: The Recall Metrics [105].

d. F1 Score

In text classification issues, F1 Score is one of the most popular metrics of evaluation. The F1 score shows how accurate the model is in each class. The F1 score is frequently employed when the dataset is unbalanced. It represents the harmonic mean between recall and precision. Thus, it balances recall and precision. Equation below is used for calculating the F1 score [103],[106]:

$$\text{F1} = 2 * \frac{precision * recall}{precision + recall} \tag{3.4}$$

4. EXPERIMENTS AND RESULTS

In this chapter, we present the results that obtained from our experiments. The experiments are categorized into three parts according to the approach we apply (ML, DL, and AutoML approaches). For each part, we applied our experiments on the liar dataset. we also used a number of text feature extractions techniques when implementing these experiments. For each experiment we conducted, we provided the accuracy, recall, precision, and F1-score. We execute our codes by using python language with version 3.8.10 for two reasons. First, Python is programming language which is widely used for analyzing the data[108] .Second, python language supports various AutoML libraries [31]. The programming environment we use is Google Colab. which is a product from google research.

4.1 LIAR DATASET

The Liar dataset consists of three files: train, valid and test. We collected these files into one file with 12836 rows Lair dataset consists of fourteen columns. We dropped unneeded columns because We only care about two columns in our work which are statement column which include the news, and label column which describes news truthiness. We removed duplicated rows which include same statement; therefore, the number of rows became 12786. the table 4.1 explain the data distribution through which we can notice that the data distribution is approximately balanced between classes except pats-fire class which has lower examples. to begin with, for building a binary-label classifiers, we followed the same pattern in the previous papers and thesis, and we changed the classes grouping for Liar dataset, where we grouped the classes “pants-fire”, “false” and “barely-true” as “Fake”, and we grouped “half-true”, “mostly-true”, “true” as “Real”. figure 4.1 shows the new data distribution for Liar dataset. To build our classification models, we divided our data as following: 90% for the training and 10% for the testing.

Table 4.1: Data Distribution for Liar Dataset.

Label	Count
True	2055
Mostly-True	2463
Half-True	2627
Barely-True	2107
False	2484
Pants-Fire	1050

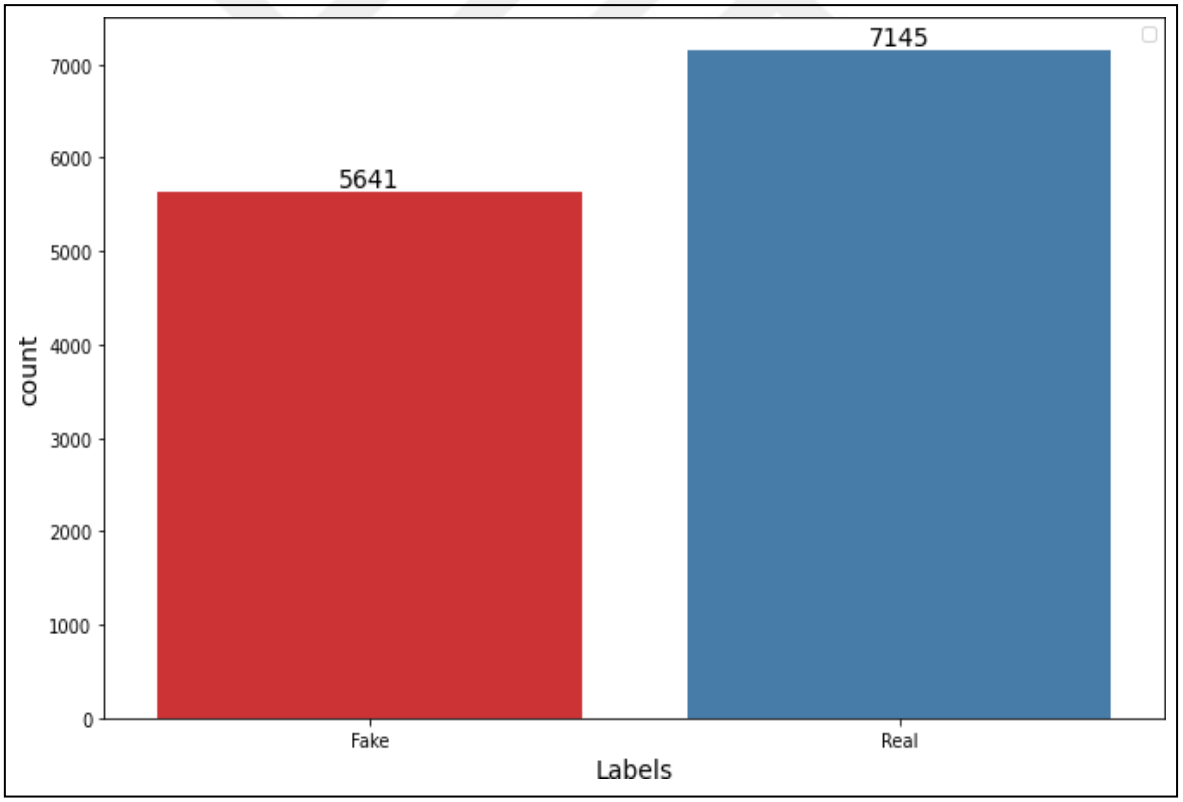


Figure 4.1: Data Distribution for Liar Dataset With Tow Labels.

4.2 DATA PREPROCESSING

The essential step in most machine learning processes is the preparation of data. While many AutoML frameworks execute this preprocessing, at the time, this job is not performed particularly effectively by the various AutoML technologies and still needs some human interaction [38]. Therefore, we performed preprocessing on the dataset to make it ready for using with both traditional machine learning and AutoML approaches. the Liar dataset was read using Pandas library data frames from the CSV file for cleaning and preprocessing. We use pandas as it is very efficient, and it is used with many studies. First, we removed all columns except for only two: statement and label columns because out work concentrated on the piece of the news only. then We lowercased the text in the data frame to reduce the size of the vocabulary of our text data. Duplicate rows were dropped from the statement column of the data frame. Next, we removed punctuation so that we don't have different forms of the same word like '!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'. If we don't remove the punctuation, then been. been, been! will be treated separately.

We also removed numbers by using regular expressions. We also removed stopwords from our text. The NLTK library has a set of stopwords, and we can use these to perform this step. we removed all the white spaces in the rows by using the join and split functions. Stemming is the process of getting the root form of a word. Stem or root is the part to which inflectional affixes (-ed, -ize, -de, -s, etc.) are added. The next step is removing the prefix or suffix of a word to create the stem of a word. For this, the Porter Stemmer technique is applied. The following example explains the preprocessing step applied on one sentence from the liar dataset:

Before text preprocessing steps: The economic turnaround started at the end of my term.

After text preprocessing steps: econom turnaround start end term.

Using wordclouds, the figure 4.2 showed the higher 100 words which used with the fake news, while the figure 4.3 explained the higher 100 words used with the real news. we noticed that there are several words used frequently in both fake and real news such as say, state. we performed removing (say) word and we noticed improvement in the accuracy.

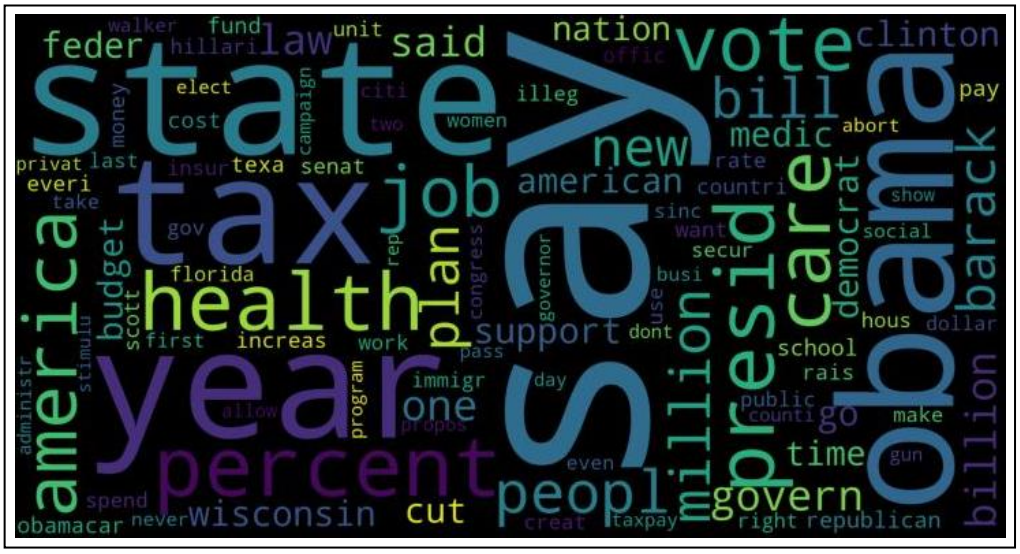


Figure 4.2: Fake News Word Cloud.

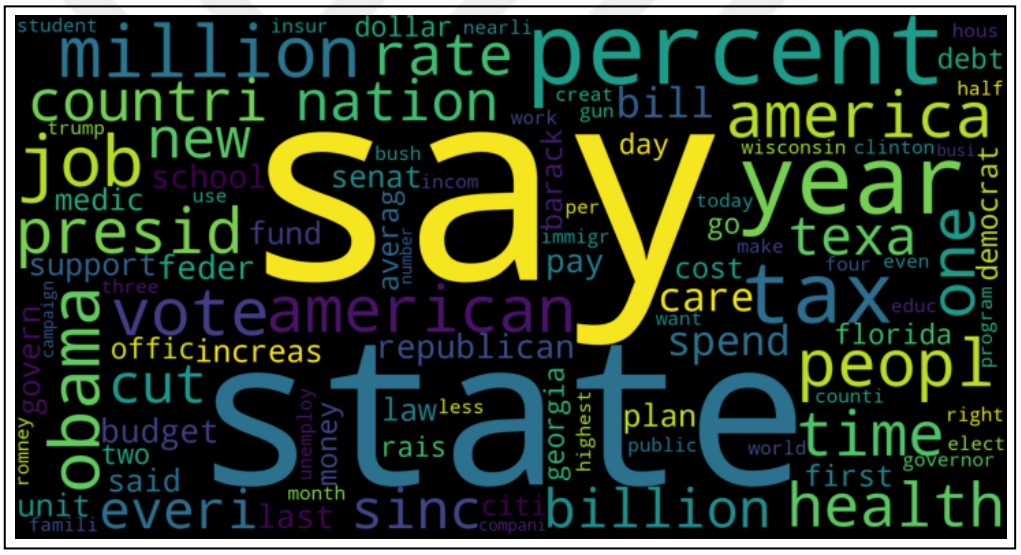


Figure 4.3: Real News Word Cloud.

4.3 FEATURE EXTRACTION

Once text preprocessing steps are completed, the text is ready to extract features form it. Different features can be extracted from it. Several techniques were tested for this purpose. We applied TF-IDF techniques for machine learning models. For TF-IDF method, we set parameters as following:

max feature = 3000,

ngram range = (1,1).

In the case of deep learning models, we utilized Word2Vec (skip-gram and CBOW), pre-trained Word2Vec and pre-trained GloVe word embedding techniques.

In case of AutoML, the process of extracting the features is done internally with auto-sklearn, AutoGluon, and Auto-Keras, where they have the possibility to process raw text automatically including feature extraction operation without human interaction. TPOT framework does not have this possibility (extracting the features internally), therefore we use TF-IDF with it.

For Word2Vec technique, we experimented with both skip-gram and CBOW cases. The parameters we used for the Word2Vec model are:

Minimum count = 1 which means we kept all words from the corpus,
windows = 5 which means the maximum distance between the current and predicted word within a sentence,
Vector size = 300, which indicates to word vectors length,
sg = 0 or 1, this value specifies what type of Word2Vec we execute. 0 is for CBOW and 1 is for skip-gram.
Final text embeddings were also limited to a dimension of 300, while null items were used to pad shorter statements.

4.4 THE EXPERIMENTS

This section will include all the experiments we perform in our work. The section will be divided into three parts. The first part will contain ML experiments, and the second part will include DL experiments. AutoML experiments will be presented in the third part.

4.4.1 Machine Learning Algorithms and Tf-idf Technique

4.4.1.1 Support vector machine algorithm

Support Vector Classifier (SVC) is used in this part of our experiments. Finding the best parameters for SVC involved several experiments. we employ the grid search technique for obtaining the best values for model parameters (using the grid search was only for finding the best values for parameters, not applying a cross-validation technique). after that, we train the SVC on the training data part using the following parameters:

kernel: sigmoid,
C = 30 (Regularization parameter),
gamma: 0.01 (Kernel coefficient for sigmoid).

We use TF-IDF technique with support vector classifier (SVC) algorithm experiments. The results are listed in table 4.2 down below.

Table 4.2: SVM With TF-IDF Report.

	Precision (%)	**Recall (%)**	**F1-score (%)**	**support**
Fake	64 %	38 %	48 %	565
Real	63 %	83 %	71 %	714
accuracy			63 %	1279
macro avg	63 %	61 %	60 %	1279
weighted avg	63 %	63 %	61 %	1279

4.4.1.2 Naïve bayes algorithm

We use Multinomial Naïve Bayes (MNB) in this part of the experiments since this classifier is suitable for classification with discrete features (e.g., word counts for text classification). Finding the best parameters for MNB involved several experiments. As in the case of SVC experiment above, we employ the grid search technique for obtaining the best values for model parameters (using the grid search was only for finding the best values for parameters, not applying a cross-validation technique). We then trained the MNB classifier on the training data part using the following parameters:

Alpha = 5 (smoothing parameter),
Fit prior: False (Whether to learn class prior probabilities or not).

TF-IDF technique are used with Multinomial Naïve Bayes (MNB) algorithm experiments. The results are detailed in Table 4.3 down below.

Table 4.3: MNB With TF-IDF Report.

	Precision (%)	**Recall (%)**	**F1-score (%)**	**support**
Fake	58 %	50 %	54 %	565
Real	64 %	72 %	68 %	714
accuracy			62 %	1279
macro avg	61 %	61 %	61 %	1279
weighted avg	62 %	62 %	62 %	1279

We show below how a heatmap representation of SVM AND MNB algorithms looks.

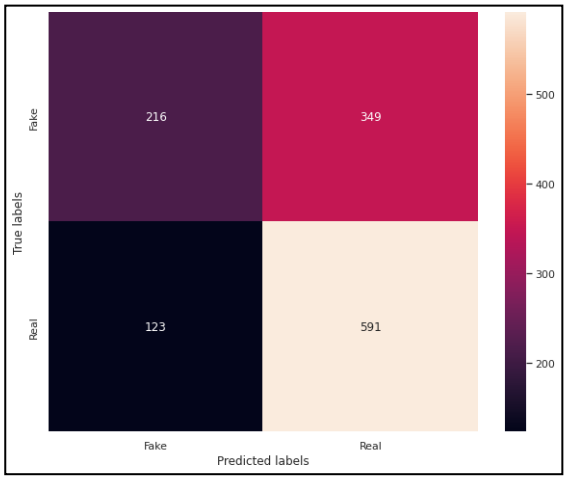


Figure 4.4: Heatmap of SVM And TF-IDF.

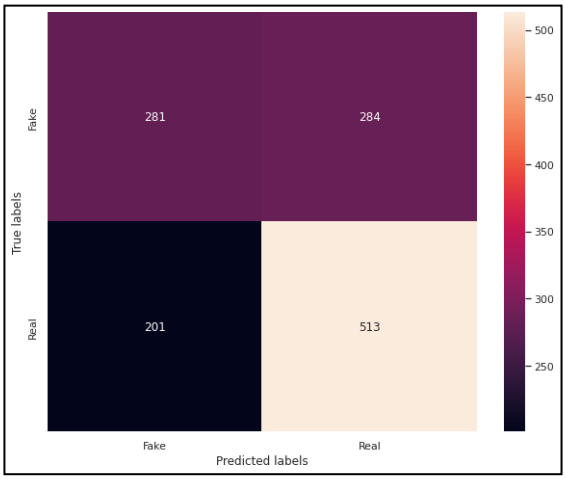


Figure 4.5: Heatmap of MNB and TF-IDF.

4.4.2 Deep Learning with Word Embeddings

In this section of our experiments, we concentrate on Deep Learning as well as an additional method of feature extraction that is known as Word Embeddings. In chapter 3, we explained Word Embeddings method and we also concentrate on Word2Vec technique which has been used with our deep learning model. we used trained and pre-trained word2vec techniques with 300-dimension word vectors. Following the production of our word vectors, we turned our attention to the design of our neural networks with the Keras library. we built our model using Sequential model. What follows is a description of the design of the hybrid CNN and BiLSTM model that was employed. We built our model using Sequential model. The model architecture consists of the following:

a. The nontrainable embedding layer is the first layer in the structure, which is fed with the word embeddings (trained such as Word2Vec-SG and Word2Vec-CBOW, and pre-trained such as pre-trained word2vec and pre-trained GloVe).

b. a CONV1D layer with 1024 neurons, kernel size = 4. The activation function applied is ReLu.

c. a Bi-LSTM layer of size 512.

d. a fully connected output layer with one neuron. The activation function applied is sigmoid.

The figure summarizes the architecture of the model. the optimization algorithm we used is Adam optimizer with a learning rate equals to 0.00001. We train the model for 200 epochs, and we stop the training using the early stopping method when the model's performance stops increasing on the holdout validation set. With a value of 3, the early-stopping patience parameter is set. Since our problem is binary classification, Binary Crossentropy Loss Function is used. The results are detailed in table 4.4, table 4.5, table 4.6 and table 4.7 respectively down below.

Table 4.4: Hybrid CNN-Bi-LSTM with Word2Vec (Skip-Gram) Report.

	Precision (%)	**Recall (%)**	**F1-score (%)**	**support**
Fake	55 %	53 %	54 %	565
Real	64 %	66 %	65 %	714
accuracy			60 %	1279
macro avg	60 %	60 %	60 %	1279
weighted avg	60 %	60 %	60 %	1279

Table 4.5: Hybrid CNN-Bi-LSTM with Word2Vec (CBOW) Report.

	Precision (%)	**Recall (%)**	**F1-score (%)**	**support**
Fake	56 %	20 %	30 %	565
Real	58 %	88 %	70 %	714
accuracy			58 %	1279
macro avg	57 %	54 %	50 %	1279
weighted avg	57 %	58 %	52 %	1279

Table 4.6: Hybrid CNN-Bi-LSTM With Pre-Trained Word2Vec Report.

	Precision (%)	**Recall (%)**	**F1-score (%)**	**support**
Fake	59 %	34 %	43 %	565
Real	61 %	81 %	70 %	714
accuracy			60 %	1279
macro avg	60 %	58 %	56 %	1279
weighted avg	60 %	60 %	58 %	1279

Table 4.7: Hybrid CNN-Bi-LSTM With Pre-Trained Glove Report.

	Precision (%)	**Recall (%)**	**F1-score (%)**	**support**
Fake	57 %	28 %	37 %	565
Real	59 %	83 %	69 %	714
accuracy			59 %	1279
macro avg	58 %	56 %	53 %	1279
weighted avg	58 %	59 %	55 %	1279

We show below how a heatmap representation of hybrid CNN-Bi-LSTM model with different feature extraction techniques looks.

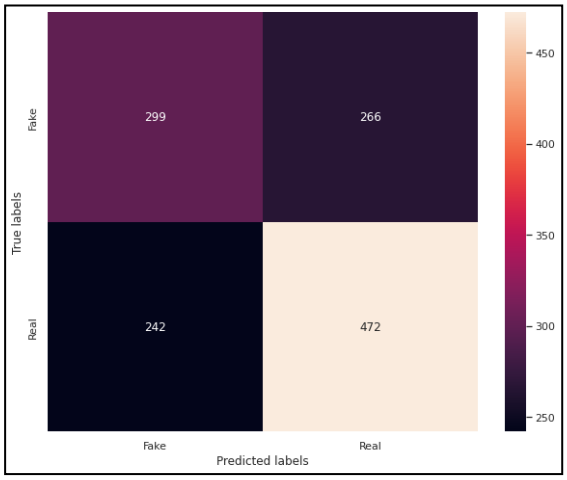


Figure 4.6: Heatmap of Hybrid CNN-Bi-LSTM Model And Word2Vec (Skip Gram).

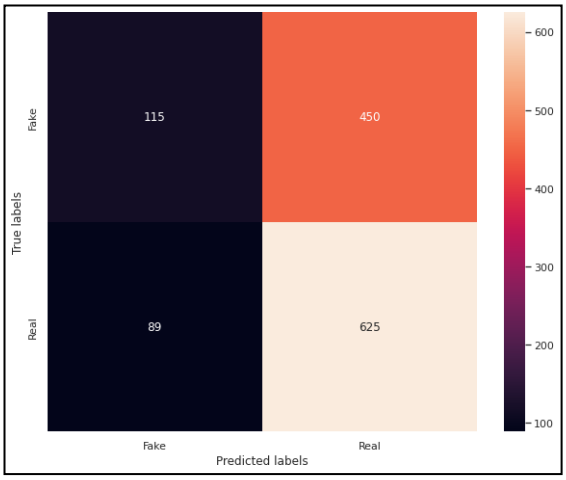


Figure 4.7: Heatmap of Hybrid CNN-Bi-LSTM And Word2Vec (CBOW).

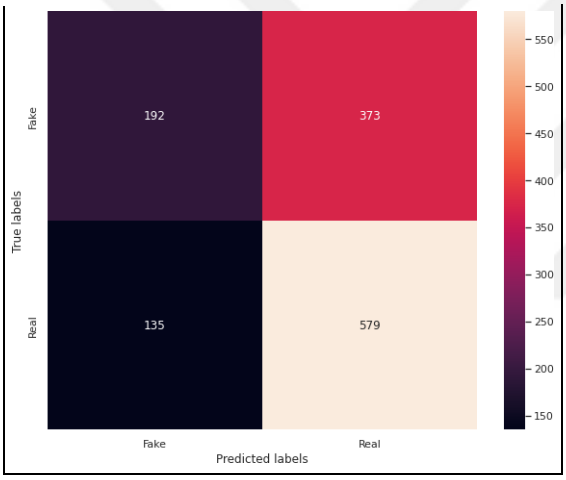


Figure 4.8: Heatmap of Hybrid CNN-Bi-LSTM and Pre-Trained Word2Vec.

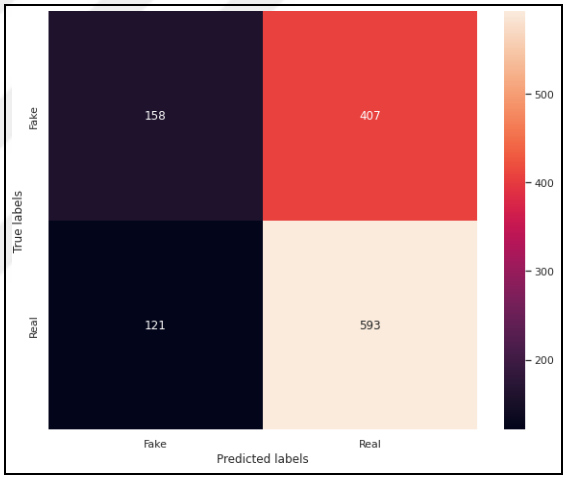


Figure 4.9: Heatmap of Hybrid Cnn-Bi-Lstm and Pre-Trained Glove.

4.4.3 Automated Machine Learning Experiments

4.4.3.1 Auto-sklearn experiments

Installing and importing the Auto-Sklearn library is the first step in this prat of our experiments. the auto-sklearn framework version number we installed is 0.15.0. Depending on our problem, we employ the AutoSklearnClassifier class to create an object, then we configure and fit it on the dataset to let Auto-Sklearn discover top-performing models.

As inputs to the AutoSklearnClassifier, there are several parameters are offered. through time_left_for_this_task parameter, the auto-sklearn library allows setting a time limit for the task. this period of time is for searching for suitable models. A time of 60 minutes was used in our experiments as it is in the default configuration, where this time is appropriate for our dataset size. we set memory_limit parameter to 100MB to avoid stopping in the training because the fitting process of the machine learning algorithm will be halted by autosklearn if this algorithm attempts to allocate more memory than the specified by memory_limit parameter. we also set per_run_time_limit parameter to 30 second.

This parameter restricts the amount of time for a single call to the ML model. If the ML algorithm exceeds the time restriction, the model fitting process will be halted. The ensemble_size argument was set to 20. an ensemble of best performing models, which is found throughout the search, will be created by the search automatically. Table 4.8 display the results.

Table 4.8: Auto-Sklearn Report.

	precision	**recall**	**F1-score**	**support**
Fake	64 %	35 %	45 %	565
Real	62 %	85 %	72 %	714
accuracy			63 %	1279
macro avg	63 %	60 %	58 %	1279
weighted avg	63 %	63 %	60 %	1279

4.4.3.2 Tpot experiments

With this part, we use the TPOT tool along with the TF-IDF technique. We used the following parameters for TPOT tool:

population size = 20,
generations = 20,
config_dict = 'TPOT sparse',
cv = 5.

The results are listed in table 4.9.

Table 4.9: TPOT Report.

	precision	**recall**	**F1-score**	**support**
Fake	59 %	48 %	53 %	565
Real	64 %	73 %	68 %	714
accuracy			62 %	1279
macro avg	61 %	61 %	60 %	1279
weighted avg	62 %	62 %	61 %	1279

4.4.3.3 Auto-keras experiments

In this section, the AutoKeras framework mentioned in the chapter 3 was tested in this part of our experiments. The maximum number of different Keras models for trying (Max_Trial parameter) was set to 200 trials. The search may finish before reaching the maximum number of trails. The epochs parameter was set to 200. The epochs parameter is the number of epochs to train each model during the search.

The Text Classifier, which is AutoKeras text classification class, was used in AutoKeras experiments. we applied the same approach with the splitting of data which used by the Liar dataset creator. the splitting of data was applied as the following: training (80%), validation (10%) and testing (10%). in case of max trail and number of epochs, we tried to give chance to make auto keras choose the best model. using auto keras framework is because it is based on keras library which we used in deep learning traditional models.

We used the early stopping approach with patience equals to 3 to stop the training when there is no improvement in the result of every trial. The results are listed in table 4.10.

Table 4.10: Auto-Keras Report.

	precision	**recall**	**F1-score**	**support**
Fake	55 %	33 %	42 %	565
Real	60 %	79 %	68 %	714
accuracy			59 %	1279
macro avg	58 %	56 %	55 %	1279
weighted avg	58 %	59 %	56 %	1279

4.4.3.4 Autogluon experiments

The AutoGluon framework mentioned in the chapter 3 was tested in this part of our experiments. we used TextPredictor class to perform AutoGluon experiments. the function of TextPredictor class is only for natural language processing problems. we set the time limit parameter to 5 hours. this time means how long Approximately the distillation process should run for.

We applied the same approach with the splitting of data which used by the Liar dataset creator. the splitting of data was applied as the following: training (80%), validation (10%) and testing (10%). The results are listed in table 4.11.

Table 4.11: Autogluon Report.

	precision	**recall**	**F1-score**	**support**
Fake	60 %	48 %	54 %	565
Real	65 %	74 %	69 %	714
accuracy			63 %	1279
macro avg	62 %	61 %	61 %	1279
weighted avg	63 %	63 %	62 %	1279

We show below how a heatmap representation of optimal models obtained by AutoML libraries looks.

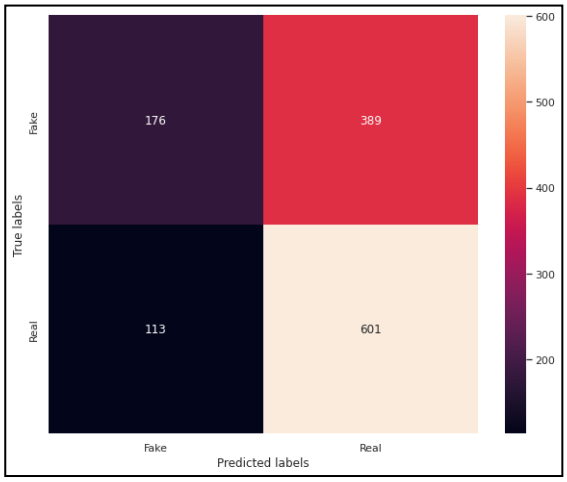


Figure 4.10: Heatmap Related to The Best Model Which is Found by Auto-Sklearn.

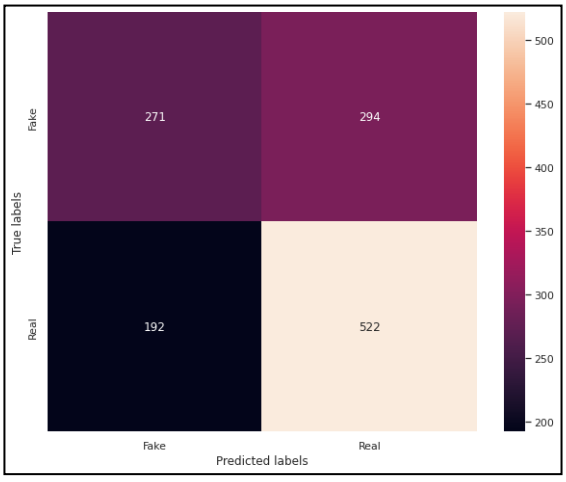


Figure 4.11: Heatmap Related to The Best Model Which is Found by TPOT.

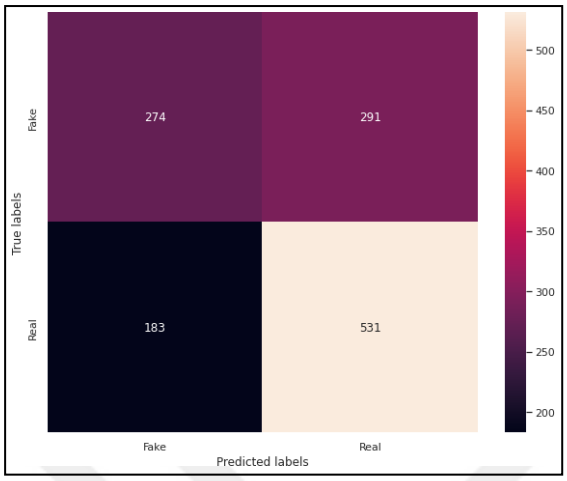


Figure 4.12: Heatmap Related to The Best Model Which is Found by Autogluon.

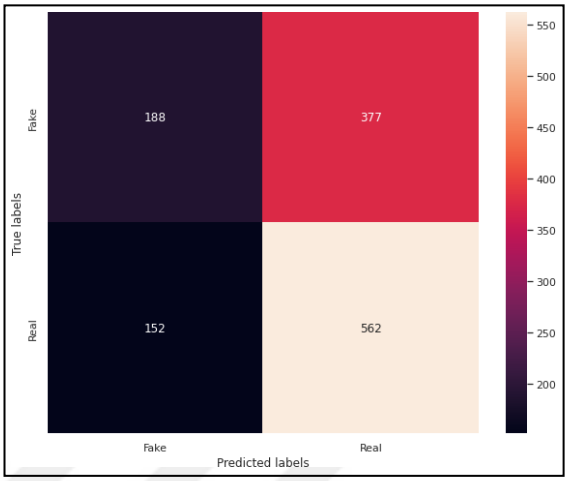


Figure 4.13: Heatmap Related to The Best Model Which is Found by Auto-Keras.

An overview of the results from the experiments conducted can be shown by table 4.12.

Table 4.12: Summary of The Experiments Results.

Model/Framework	Test accuracy	Precision	Recall	F1-score	Validation accuracy
SVM	63.10	62.87	82.77	71.46	69.30
MNB	62.08	64.37	71.85	67.90	69.10
Hybrid CNN-Bi-LSTM and SG	60.28	63.96	66.11	65.01	65.75
Hybrid CNN-Bi-LSTM and CBOW	57.86	58.14	87.54	69.87	66.80
Hybrid CNN-Bi-LSTM and pre-trained Word2Vec	60.28	60.82	81.09	69.51	70.25
Hybrid CNN-Bi-LSTM and pre-trained GloVe	58.72	59.30	83.05	69.19	69.96

Table 4.13: Summary of The Experiments Results (Table continued)

TPOT	62.00	63.97	73.11	68.24	68.01
auto-sklearn	60.75	60.71	84.17	70.54	87.03
Auto-Keras	58.64	59.85	78.71	68.00	60.83
AutoGluon	62.94	64.60	74.37	69.14	62.31

There are some important points which we have to mention regarding with the results above:

a. Through the above results, we note that the results are not good enough. From our point of view two main reasons led to not achieving the desired results, the first reason is the limited number of records that have been employed in the training. The second reason is that we only used the statement feature, which is characterized by its short and contains a limited number of words, and this in turn will affect the process of extracting features from the text.

b. When comparing the results collected from the two approaches (traditional and AutoML), we note that the results are somewhat close, which is why we are optimistic about the automated machine learning future.

c. Compared to the performance of all experiments, SVM algorithm outperformed the others with accuracy equal to (63.10), and AutoGluon comes in the second place with a very slight difference from the SVM algorithm, as it achieved an accuracy equal to (62.94). But in terms of overfitting, the SVM has suffered something of this problem, unlike the AutoGluon which did not suffer from the overfitting.

d. When we come to compare the performance of AutoML libraries, it is clear that AutoGluon has surpassed the rest in terms of performance and not suffering from the overfitting. What is remarkable is that the auto-sklearn has suffered greatly from the overfitting, where the test accuracy was (60.75), while the train accuracy was (87.03).

5. CONCLUSION AND FUTURE WORK

This chapter wraps up the thesis by presenting a conclusion to the work that we performed as well as a suggestion of some potential future work. The purpose of this thesis was to detect fake news on online sources such as news websites and social media using traditional machine learning algorithms and automated machine learning frameworks including automated deep learning frameworks, but our focus mainly has been on solving the fake news problem with many automated machine learning tools. To our knowledge this is the first work to address the problem of fake news with these tools. In the first chapter, we shed light about the phenomenon of fake news that invaded the internet and showed its great negative impact on society in diverse fields of life. In the second chapter, we reviewed several works that focused on detecting fake news, whether these works depended on machine learning or deep learning approaches. We also highlighted on the automated machine learning works, especially these are related to natural language processing.

In this work, we used liar dataset as a data source to apply our experiments. This dataset was created in 2017 form PolitiFact organization and included a short text news from variety media sources like news websites and social media platforms. We followed the same approach used by many studies, by changing the dataset from the six-way classification to the binary classification (as real and fake classes). Unlike many models that were produced using several features, we used only the statement attribute (the news text) for differentiating between fake and real news.

For building the traditional models, we employed NB and SVM algorithms to build classical machine learning models. In case of deep learning approach, we created hybrid CCN-BI-LSTM model which consisted of several layers such as embedding, 1D convolutional, Bi-LSTM, and dense layers. For finding the optimal models by automated machine learning, four frameworks were used.

For the natural language processing task, there are various approaches we used for extracting the features from the text (text representation). for classic ML methods, NB and SVM methods, we used TF-IDF technique. As for the deep learning model, we used two of word embedding techniques: Word2Vec (skip-gram and CBOW) and GloVe. we applied them with trained and pre-trained approaches. On the other side, which is related to automated

machine learning tools, there are different approaches for text representation. according to Autokeras and AutoGluon, they can generate text representations automatically (internal embeddings) without any human effort. Auto-sklearn now supports text representation which means it can extract features automatically, unlike TPOT, which does not have the possibility to perform the feature extraction operation, therefore we employed the TF-IDF technique as is the case with the SVM and NB for feature extraction operation.

Although the accuracy is not great in all the experiments we conducted, it is close to the accuracy obtained by the experiments reviewed in previous work in the second chapter. There is no doubt that automatic machine learning tools are very useful in terms of shortening work time and reducing the effort spent on building machine learning systems, and this appears clearly in the case of building deep learning models which need many trials for obtaining the best model. During the experiments, we noticed that AutoGluon beat the other AutoML libraires, the DL model and NB model. In the case of SVM algorithm, SVM outperformed AutoGluon with a very simple superiority, but the SVM suffered from overfitting unlike AutoGluon tool, which did not suffer from any overfitting.

It would be very useful to explore more AutoML frameworks, test how effective they are, and then compare the outcomes of these tests. No doubt conducting experiments on databases with different sizes and classes will show the strengths and weaknesses of the AutoML libraries. Certainly, using more than one of the dataset's features and not being limited to just one will lead to different results.

REFERENCES

[1] J. Y. Khan, M. Khondaker, T. Islam, A. Iqbal, and S. Afroz, "A benchmark study on machine learning methods for fake news detection," *arXiv Prepr. arXiv1905.04749*, vol. 2, 2019.

[2] N. R. de Oliveira, P. S. Pisa, M. A. Lopez, D. S. V de Medeiros, and D. M. F. Mattos, "Identifying fake news on social networks based on natural language processing: trends and challenges," *Information*, vol. 12, no. 1, p. 38, 2021.

[3] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *J. Econ. Perspect.*, vol. 31, no. 2, pp. 211–236, 2017.

[4] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD Explor. Newsl.*, vol. 19, no. 1, pp. 22–36, 2017.

[5] M. Rani and C. Virmani, "Detection of Fake News on Social Media: A Review," *Available SSRN 4143832*, 2022.

[6] M. Choudhary, S. S. Chouhan, E. S. Pilli, and S. K. Vipparthi, "BerConvoNet: A deep learning framework for fake news classification," *Appl. Soft Comput.*, vol. 110, p. 107614, 2021.

[7] Y. Liu, "Early detection of fake news on social media." New Jersey Institute of Technology, 2019.

[8] N. Kanagavalli, S. B. Priya, and D. Jeyakumar, "Design of Hyperparameter Tuned Deep Learning based Automated Fake News Detection in Social Networking Data," in *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*, 2022, pp. 958–963.

[9] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019.

[10] S. Hangloo and B. Arora, "Fake News Detection Tools and Methods--A Review," *arXiv Prepr. arXiv2112.11185*, 2021.

[11] A. Zubiaga, A. Aker, K. Bontcheva, M. Liakata, and R. Procter, “Detection and resolution of rumours in social media: A survey,” *ACM Comput. Surv.*, vol. 51, no. 2, pp. 1–36, 2018.

[12] S. Mishra, P. Shukla, and R. Agarwal, “Analyzing machine learning enabled fake news detection techniques for diversified datasets,” *Wirel. Commun. Mob. Comput.*, vol. 2022, pp. 1–18, 2022.

[13] H. Rashkin, E. Choi, J. Y. Jang, S. Volkova, and Y. Choi, “Truth of varying shades: Analyzing language in fake news and political fact-checking,” in *Proceedings of the 2017 conference on empirical methods in natural language processing*, 2017, pp. 2931–2937.

[14] C. M. M. Kotteti, “Fake News Detection in Social Media Using Machine Learning and Deep Learning,” 2020.

[15] J. A. Nasir, O. S. Khan, and I. Varlamis, “Fake news detection: A hybrid CNN-RNN based deep learning approach,” *Int. J. Inf. Manag. Data Insights*, vol. 1, no. 1, p. 100007, 2021.

[16] S. Vosoughi, D. Roy, and S. Aral, “The spread of true and false news online,” *Science (80-.).*, vol. 359, no. 6380, pp. 1146–1151, 2018.

[17] H. Mjaaland, “Detecting Fake News and Rumors in Twitter Using Deep Neural Networks.” University of Stavanger, Norway, 2020.

[18] P. Chennam Lakhsmikumar, “Fake news detection a deep neural network.” University of Stavanger, Norway, 2019.

[19] “False Rumor of Explosion at White House Causes Stocks to Briefly Plunge; AP Confirms Its Twitter Feed Was Hacked,” 2013. https://www.cnbc.com/id/100646197

[20] F. News, “Effect of Fake News on Education and Teenager Students,” p. 4, 2020, [Online]. Available: https://writingbros.com/essay-examples/effect-of-fake-news-on-education-and-teenager-students/

[21] R. Craufurd Smith, “Fake news, French Law and democratic legitimacy: lessons for

the United Kingdom?," *J. Media Law*, vol. 11, no. 1, pp. 52–81, Jan. 2019, doi: 10.1080/17577632.2019.1679424.

[22] M. Haigh, T. Haigh, M. Dorosh, and T. Matychak, "Beyond fake news: Learning from information literacy programs in ukraine," *Adv. Librariansh.*, vol. 50, pp. 163–182, 2021, doi: 10.1108/S0065-283020210000050007/FULL/HTML.

[23] J. V. Tembhurne, M. M. Almin, and T. Diwan, "Mc-DNN: Fake news detection using multi-channel deep neural networks," *Int. J. Semant. Web Inf. Syst.*, vol. 18, no. 1, pp. 1–20, 2022.

[24] E. Amer, K.-S. Kwak, and S. El-Sappagh, "Context-based fake news detection model relying on deep learning models," *Electronics*, vol. 11, no. 8, p. 1255, 2022.

[25] S. Garg and D. K. Sharma, "Linguistic features based framework for automatic fake news detection," *Comput. Ind. Eng.*, vol. 172, p. 108432, 2022.

[26] A. Sedik, A. A. Abohany, K. M. Sallam, K. Munasinghe, and T. Medhat, "Deep fake news detection system based on concatenated and recurrent modalities," *Expert Syst. Appl.*, vol. 208, p. 117953, 2022.

[27] H. Alquran and S. Banitaan, "Fake News Detection in Social Networks Using Data Mining Techniques," in *2022 IEEE World AI IoT Congress (AIIoT)*, 2022, pp. 155–160.

[28] N. Aslam, I. Ullah Khan, F. S. Alotaibi, L. A. Aldaej, and A. K. Aldubaikil, "Fake detect: A deep learning ensemble model for fake news detection," *Complexity*, vol. 2021, pp. 1–8, 2021.

[29] P. Agarwal, S. Reddivari, and K. Reddivari, "Fake News Detection: An Investigation based on Machine Learning," in *2022 IEEE 23rd International Conference on Information Reuse and Integration for Data Science (IRI)*, 2022, pp. 61–62.

[30] D. K. Sharma, P. Shrivastava, and S. Garg, "Utilizing Word Embedding and Linguistic Features for Fake News Detection," in *2022 9th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2022, pp. 844–848.

[31] K. T. Y. Mahima, T. Ginige, and K. De Zoysa, “Evaluation of sentiment analysis based on AutoML and traditional approaches,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 2, 2021.

[32] H. J. Escalante, “Automated Machine Learning--a brief review at the end of the early years,” *arXiv Prepr. arXiv2008.08516*, 2020.

[33] X. He, K. Zhao, and X. Chu, “AutoML: A survey of the state-of-the-art,” *Knowledge-Based Syst.*, vol. 212, p. 106622, 2021.

[34] M.-A. Zöller and M. F. Huber, “Benchmark and survey of automated machine learning frameworks,” *J. Artif. Intell. Res.*, vol. 70, pp. 409–472, 2021.

[35] R. Elshawi, M. Maher, and S. Sakr, “Automated machine learning: State-of-the-art and open challenges,” *arXiv Prepr. arXiv1906.02287*, 2019.

[36] Q. Yao *et al.*, “Taking human out of learning applications: A survey on automated machine learning,” *arXiv Prepr. arXiv1810.13306*, 2018.

[37] T. Anwar, “Identify Hate Speech Spreaders on Twitter using Transformer Embeddings Features and AutoML Classifiers.,” in *CLEF (Working Notes)*, 2021, pp. 1808–1812.

[38] A. Truong, A. Walters, J. Goodsitt, K. Hines, C. B. Bruss, and R. Farivar, “Towards automated machine learning: Evaluation and comparison of AutoML approaches and tools,” in *2019 IEEE 31st international conference on tools with artificial intelligence (ICTAI)*, 2019, pp. 1471–1479.

[39] M. Blohm, M. Hanussek, and M. Kintz, “Leveraging automated machine learning for text classification: Evaluation of automl tools and comparison with human performance,” *arXiv Prepr. arXiv2012.03575*, 2020.

[40] W. Y. Wang, “‘ liar, liar pants on fire’: A new benchmark dataset for fake news detection,” *arXiv Prepr. arXiv1705.00648*, 2017.

[41] Μ. Π. Ζαγκότσης, “Fake News Detection using Deep Learning and Machine Learning Methods-A comparative study on short and long texts.” Αριστοτέλειο Πανεπιστήμιο

Θεσσαλονίκης, 2019.

[42] E.-R. Stefan, “Fake news detection and analysis.” Universitat Politècnica de Catalunya, 2022.

[43] “PolitiFact | The Principles of the Truth-O-Meter: PolitiFact’s methodology for independent fact-checking.” https://www.politifact.com/article/2018/feb/12/principles-truth-o-meter-politifacts-methodology-i/ (accessed Mar. 03, 2023).

[44] M. A. Shariff, “Channel Islands.” CALIFORNIA STATE UNIVERSITY, 2020.

[45] D. Khurana, A. Koli, K. Khatter, and S. Singh, “Natural language processing: State of the art, current trends and challenges,” *Multimed. Tools Appl.*, pp. 1–32, 2022.

[46] M. M. Lopez and J. Kalita, “Deep Learning applied to NLP,” *arXiv Prepr. arXiv1703.03091*, 2017.

[47] T. Beysolow, “Applied natural language processing with python,” 2018.

[48] V. Srividhya and R. Anitha, “Evaluating preprocessing techniques in text categorization,” *Int. J. Comput. Sci. Appl.*, vol. 47, no. 11, pp. 49–51, 2010.

[49] C. P. Chai, “Comparison of text preprocessing methods,” *Nat. Lang. Eng.*, pp. 1–45, 2022.

[50] S. Gharatkar, A. Ingle, T. Naik, and A. Save, “Review preprocessing using data cleaning and stemming technique,” in *2017 international conference on innovations in information, embedded and communication systems (iciiecs)*, 2017, pp. 1–4.

[51] A. K. Uysal and S. Gunal, “The impact of preprocessing on text classification,” *Inf. Process. Manag.*, vol. 50, no. 1, pp. 104–112, 2014.

[52] M. F. Porter, “An algorithm for suffix stripping,” *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[53] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, “Text classification algorithms: A survey,” *Information*, vol. 10, no. 4, p. 150,

2019.

[54] S. Siddiqi and A. Sharan, “Keyword and keyphrase extraction techniques: a literature review,” *Int. J. Comput. Appl.*, vol. 109, no. 2, 2015.

[55] A. Mishra and S. Vishwakarma, “Analysis of tf-idf model and its variant for document retrieval,” in *2015 international conference on computational intelligence and communication networks (cicn)*, 2015, pp. 772–776.

[56] J. Ramos, “Using tf-idf to determine word relevance in document queries,” in *Proceedings of the first instructional conference on machine learning*, 2003, vol. 242, no. 1, pp. 29–48.

[57] A. A. Hakim, A. Erwin, K. I. Eng, M. Galinium, and W. Muliady, “Automated document classification for news article in Bahasa Indonesia based on term frequency inverse document frequency (TF-IDF) approach,” in *2014 6th international conference on information technology and electrical engineering (ICITEE)*, 2014, pp. 1–4.

[58] H. Tian and L. Wu, “Microblog emotional analysis based on TF-IWF weighted Word2vec model,” in *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*, 2018, pp. 893–896.

[59] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv Prepr. arXiv1301.3781*, 2013.

[60] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[61] J. Alzubi, A. Nayyar, and A. Kumar, “Machine learning from theory to algorithms: an overview,” in *Journal of physics: conference series*, 2018, vol. 1142, p. 12012.

[62] Z. Gong, P. Zhong, and W. Hu, “Diversity in machine learning,” *IEEE Access*, vol. 7, pp. 64323–64350, 2019.

[63] I. El Naqa and M. J. Murphy, *What is machine learning?* Springer, 2015.

[64] M. Abbas, K. A. Memon, A. A. Jamali, S. Memon, and A. Ahmed, "Multinomial Naive Bayes classification model for sentiment analysis," *IJCSNS Int. J. Comput. Sci. Netw. Secur*, vol. 19, no. 3, p. 62, 2019.

[65] G. Singh, B. Kumar, L. Gaur, and A. Tyagi, "Comparison between multinomial and Bernoulli naïve Bayes for text classification," in *2019 International Conference on Automation, Computational and Technology Management (ICACTM)*, 2019, pp. 593–596.

[66] A. Rahman and M. S. Hossen, "Sentiment analysis on movie review data using machine learning approach," in *2019 International Conference on Bangla Speech and Language Processing (ICBSLP)*, 2019, pp. 1–4.

[67] G. Dimitoglou, J. A. Adams, and C. M. Jim, "Comparison of the C4. 5 and a Naïve Bayes classifier for the prediction of lung cancer survivability," *arXiv Prepr. arXiv1206.1121*, 2012.

[68] M. M. Saritas and A. Yasar, "Performance analysis of ANN and Naive Bayes classification algorithm for data classification," *Int. J. Intell. Syst. Appl. Eng.*, vol. 7, no. 2, pp. 88–91, 2019.

[69] A. Agarwal and A. Dixit, "Fake news detection: an ensemble learning approach," in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2020, pp. 1178–1183.

[70] L. Yu, A. Porwal, E.-J. Holden, and M. C. Dentith, "Towards automatic lithological classification from remote sensing data using support vector machines," *Comput. Geosci.*, vol. 45, pp. 229–239, 2012.

[71] M. Ben-Ari and F. Mondada, *Elements of robotics*. Springer Nature, 2017.

[72] S. Herculano-Houzel, "The human brain in numbers: a linearly scaled-up primate brain," *Front. Hum. Neurosci.*, p. 31, 2009.

[73] S. Kohli, S. Miglani, and R. Rapariya, "Basics of artificial neural network," *Int. J. Comput. Sci. Mob. Comput.*, vol. 3, no. 9, pp. 745–751, 2014.

[74] S. Sharma, S. Sharma, and A. Athaiya, “Activation functions in neural networks,” *Towar. Data Sci*, vol. 6, no. 12, pp. 310–316, 2017.

[75] H. Kukreja, N. Bharath, C. S. Siddesh, and S. Kuldeep, “An introduction to artificial neural network,” *Int J Adv Res Innov Ideas Educ*, vol. 1, pp. 27–30, 2016.

[76] M. Shukla and M. Abdelrahman, “Artificial neural networks based steady state security analysis of power systems,” in *Thirty-Sixth Southeastern Symposium on System Theory, 2004. Proceedings of the*, 2004, pp. 266–269.

[77] O. A. Montesinos López, A. Montesinos López, and J. Crossa, “Fundamentals of Artificial Neural Networks and Deep Learning,” in *Multivariate Statistical Machine Learning Methods for Genomic Prediction*, Springer, 2022, pp. 379–425.

[78] J. Y. Ryu, H. Y. Chung, and K. Y. Choi, “Potential role of artificial intelligence in craniofacial surgery,” *Arch. Craniofacial Surg.*, vol. 22, no. 5, p. 223, 2021.

[79] S. Tam, R. Ben Said, and Ö. Ö. Tanriöver, “A convbilstm deep learning model-based approach for twitter sentiment classification,” *IEEE Access*, vol. 9, pp. 41283–41293, 2021.

[80] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[81] T. H. H. Aldhyani and H. Alkahtani, “A bidirectional long short-term memory model algorithm for predicting COVID-19 in gulf countries,” *Life*, vol. 11, no. 11, p. 1118, 2021.

[82] A. Yadav and D. K. Vishwakarma, “Sentiment analysis using deep learning architectures: a review,” *Artif. Intell. Rev.*, vol. 53, no. 6, pp. 4335–4385, 2020.

[83] P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao, and B. Xu, “Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling,” *arXiv Prepr. arXiv1611.06639*, 2016.

[84] M. K. Balwant, “Bidirectional LSTM based on POS tags and CNN architecture for fake news detection,” in *2019 10th International conference on computing,*

communication and networking technologies (ICCCNT), 2019, pp. 1–6.

[85] K. Chauhan *et al.*, "Automated machine learning: The new wave of machine learning," in *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, 2020, pp. 205–212.

[86] F. Nargesian, H. Samulowitz, U. Khurana, E. B. Khalil, and D. S. Turaga, "Learning Feature Engineering for Classification.," in *Ijcai*, 2017, vol. 17, pp. 2529–2535.

[87] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.

[88] H. Motoda and H. Liu, "Feature selection, extraction and construction," *Commun. IICM (Institute Inf. Comput. Mach. Taiwan)*, vol. 5, no. 67–72, p. 2, 2002.

[89] J. Madrid, "Autotext: AutoML for text classification." 2019.

[90] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization.," *J. Mach. Learn. Res.*, vol. 13, no. 2, 2012.

[91] M. Feurer and F. Hutter, "Hyperparameter optimization," *Autom. Mach. Learn. Methods, Syst. challenges*, pp. 3–33, 2019.

[92] Y. Liu, Y. Sun, B. Xue, M. Zhang, G. G. Yen, and K. C. Tan, "A survey on evolutionary neural architecture search," *IEEE Trans. neural networks Learn. Syst.*, 2021.

[93] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20, no. 1, pp. 1997–2017, 2019.

[94] G. J. van Wyk and A. S. Bosman, "Evolutionary neural architecture search for image restoration," in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019, pp. 1–8.

[95] H. Jin, Q. Song, and X. Hu, "Auto-keras: An efficient neural architecture search system," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 1946–1956.

[96] A. Doke and M. Gaikwad, “Survey on automated machine learning (AutoML) and meta learning,” in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 2021, pp. 1–5.

[97] “GitHub - autogluon/autogluon: AutoGluon: AutoML for Image, Text, Time Series, and Tabular Data.” https://github.com/autogluon/autogluon (accessed Apr. 09, 2023).

[98] “AutoGluon: AutoML for Text, Image, Time Series, and Tabular Data — AutoGluon Documentation 0.7.0 documentation.” https://auto.gluon.ai/stable/index.html (accessed Mar. 01, 2023).

[99] L. Ferreira, A. Pilastri, C. M. Martins, P. M. Pires, and P. Cortez, “A comparison of AutoML tools for machine learning, deep learning and XGBoost,” in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–8.

[100] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, “Efficient and robust automated machine learning,” *Adv. Neural Inf. Process. Syst.*, vol. 28, 2015.

[101] R. S. Olson and J. H. Moore, “TPOT: A tree-based pipeline optimization tool for automating machine learning,” in *Workshop on automatic machine learning*, 2016, pp. 66–74.

[102] “TPOT.” http://epistasislab.github.io/tpot/ (accessed Mar. 02, 2023).

[103] M. F. Mridha, A. J. Keya, M. A. Hamid, M. M. Monowar, and M. S. Rahman, “A comprehensive review on fake news detection with deep learning,” *IEEE Access*, vol. 9, pp. 156151–156170, 2021.

[104] J. R. Maria Navin and R. Pankaja, “Performance analysis of text classification algorithms using confusion matrix,” *Int. J. Eng. Tech. Res. IJETR*, vol. 6, pp. 75–78, 2016.

[105] T. Saito and M. Rehmsmeier, “Basic evaluation measures from the confusion matrix,” *Access link https//classeval. wordpress. com/introduction/basic-evaluation-measures*, 2017.

[106] S. A. Alameri and M. Mohd, “Comparison of fake news detection using machine learning and deep learning techniques,” in *2021 3rd International Cyber Resilience Conference (CRC)*, 2021, pp. 1–6.

[107] A. Mohapatra, N. Thota, and P. Prakasam, “Fake news detection and classification using hybrid BiLSTM and self-attention model,” *Multimed. Tools Appl.*, vol. 81, no. 13, pp. 18503–18519, 2022.

[108] A. E. Lillie and E. R. Middelboe, “Fake news detection using stance classification: A survey,” *arXiv Prepr. arXiv1907.00181*, 2019.