KARADENİZ TECHNICAL UNIVERSITY
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING

STANCE CLASSIFICATION FOR FAKE NEWS DETECTION IN SOCIAL MEDIA

MASTER THESIS

Maysaa ALSAFADI

NOVEMBER 2023
TRABZON

KARADENİZ TECHNICAL UNIVERSITY

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING

STANCE CLASSIFICATION FOR FAKE NEWS DETECTION IN SOCIAL MEDIA

Maysaa ALSAFADI

ORCID : - - -

This thesis is accepted to give the degree of

"MASTER OF SCIENCE"

By

The Graduate School of Natural and Applied Sciences at
Karadeniz Technical University

The Date of Submission : **28 / 09 /2023**

The Date of Examination : **07 / 11 /2023**

Supervisor : **Asst. Prof. Dr. Murat AYKUT**
ORCID : - - -

Trabzon 2023

ACKNOWLEDGMENTS

I want to convey my profound appreciation to the numerous people who have assisted me in various ways throughout my journey towards finishing this thesis.

First and foremost, I want to express my sincere gratitude to my thesis supervisor, Asst. Prof. Dr. Murat AYKUT, for their invaluable guidance and unwavering support throughout my master's program. Their assistance has been instrumental in helping me overcome the challenges I encountered along the way. I am truly appreciative of their dedication and commitment to my academic growth.

I am at a loss for words when it comes to acknowledging the sacrifices, love, and affection of my wonderful family. Despite the borders that prevented us from meeting in person, they have always been by my side, providing unwavering love and support throughout this journey. Their presence has given me the strength to persevere even in the most challenging times.

Maysaa ALSAFADI
Trabzon, 2023

THESIS STATEMENT

This study, titled 'Stance Classification for Fake News Detection in Social Media,' which I presented as a master's thesis, received full support from my supervisor, Asst. Prof. Dr. Murat AYKUT. I completed it under his guidance, conducting experiments in the relevant laboratories. I have meticulously referenced all information obtained from other sources within the text and the bibliography. Throughout the research process, I adhered to scientific and ethical principles, and I made all necessary decisions when challenges arose. I hereby acknowledge and accept any legal consequences. 07/11/2023.

Maysaa ALSAFADI
Trabzon, 2023

TABLE OF CONTENTS

Yüksek Lisans Tezi

ÖZET

SOSYAL MEDYADA SAHTE HABER TESPİTİ İÇİN DURUM SINIFLANDIRMASI

Maysaa ALSAFADI

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı
Danışman: Dr. Öğr. Üyesi Murat AYKUT
2023, 98 Sayfa

Bugünün dijital ortamında, çeşitli kaynakların ve uygulamaların sunulması, haberlerin neden olmaktadır hızla yayılmasını hızlandırdı. Bu, bireylerin sosyal medya aracılığıyla görüşlerini paylaşmalarına olanak tanımış, bazen otomatik botlar veya sahte kullanıcılar kullanılarak gerçekleştirilmiştir. Sahte Haber Algılama, haberin güvenilir olup olmadığını belirlemeyi amaçlayan bir sınıflandırma görevidir. Son yıllarda, Arap dillerinde FND'ye artan bir ilgi olmuş ve birçok algılama yaklaşımı çeşitli veri setlerinde sahte haberleri tanıma yeteneğini göstermiştir.

Araştırmamızda, kelime gömme tekniklerini ve hibrit derin öğrenme modelini, optimize edilmiş ön işleme teknikleri ile birleştirdik. Ayrıca haber makalelerinden doğru veri elde etmek için özellik çıkarma ve seçme yöntemlerini kullandık. Kamuya açık AFND veri seti üzerinde gerçekleştirilen deneyler, uyguladığımız yöntemlerle etkileyici performans sergilemiştir. Bulgularımız, geleneksel makine öğrenimi yaklaşımları olan Lojistik Regresyon, Doğrusal SVM, Naif Bayes ve XGBoost'a karşı LSTM, BiLSTM ve CNN-BiLSTM gibi derin öğrenme modellerinin performansını göstermektedir. Özellikle, CNN-BiLSTM modeli en yüksek doğruluk elde ederek %95'e ulaşmıştır.

Anahtar Kelimeler: Sahte haber tespiti (FND), Derin Öğrenme (DL), İki Yönlü Uzun KısaVadeli Bellek (BiLSTM), Evrişimli Sinir Ağı (CNN), Arapça Dil.

Master Thesis

SUMMARY

STANCE CLASSIFICATION FOR FAKE NEWS DETECTION IN SOCIAL MEDIA

Maysaa ALSAFADI

Karadeniz Technical University
The Graduate School of Natural and Applied Sciences
Computer Engineering Graduate Programs
Supervisor: Asst. Prof. Dr. Murat AYKUT
2023, 98 Pages

In today's digital landscape, the availability of a diverse range of resources has accelerated the growth of news dissemination. This has enabled individuals to share their opinions through social media, sometimes using automated bots or fake user. Fake News Detection is a classification task which aims to determine whether news is credible or not.

In recent years, there has been a growing interest in FND in Arabic languages, and many detection approaches have demonstrated the ability to identify fake news across various datasets. In our research, we combined word embedding techniques and a hybrid deep learning model, with optimizing preprocessing techniques. We also utilized feature extraction and selection to obtain accurate data from news articles. Experiments conducted on the publicly available AFND dataset have revealed impressive performance using the methods we applied. Our findings showcase the performance of deep learning models, including LSTM, BiLSTM, and CNN-BiLSTM, in comparison to traditional machine learning approaches like Logistic Regression, linear SVM, Naïve Bayes, and XGBoost. Notably, the CNN-BiLSTM model achieved the highest accuracy result, reaching 95%.

Keywords: Fake News Detection (FND), Deep Learning (DL), Bidirectional Long Short-Term Memory (BiLSTM), Convolutional Neural Network (CNN), Arabic Language.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

FND	: Fake News Detection
AFND	: Arabic Fake News Detection
LR	: Logistic Regression
ML	: Machine Learning
DL	: Deep Learning
NLP	: Natural Language Processing
NV	: Naïve Baye
LSVM	: Linear Support Vector Machine
XGBoost	: Extreme Gradient Boost
LSTM	: Long Short-Term Memory
BiLSTM	: Bidirectional Long Short-Term Memory
CNN	: Convolutional Neural Networks
RNN	: Recurrent Neural Network
GRU	: Gated Recurrent Units
LIAR	: Liar, Liar Pants on Fire
NLTK	: Natural Language Toolkit
FNI	: Fake News Identification
BERT	: Bidirectional Encoder Representations from Transformers
TF-IDF	: Term Frequency–Inverse Document Frequency
FNC	: Fake News Corpus
ASTD	: Arabic Sentiment Tweets Dataset
AJGT	: Arabic Jordanian General Tweets
ANS	: Arabic News Stance
ISOT	: The Information Security And Object Technology
FNC-1	: Fake News Challenge
SST	: Stanford Sentiment Treebank
RF	: Random Forest
DT	: Decision Tree
K-NN	: K Nearest Neighbor

PML	: Padding Max Length
IR	: Arabic Information Retrieval
SGD	: Stochastic Gradient Descent

1. GENERAL INFORMATION

1.1. Introduction

In recent years, the proliferation of social media platforms has significantly transformed the way information is disseminated and consumed. While social media has undoubtedly empowered individuals to express their opinions and engage in discussions, it has also given rise to a concerning phenomenon: the spread of fake news. Fake news refers to fabricated or misleading information that is presented as factual, often with the intent to deceive or manipulate readers. The widespread dissemination of fake news on social media platforms has raised serious concerns about its potential impact on public opinion, political discourse, and societal well-being.

The challenges of fake news research are rooted in the difficulty of providing a universal definition for this pervasive phenomenon. Scholars have approached fake news from different perspectives, resulting in various interpretations. Some researchers view fake news as "a news article that is intentionally and verifiably false," as mentioned by Su et al. (2020), categorizing it as deceptive news. Others, such as Paka et al. (2021), define it as "a news article or message published and propagated through media, carrying false information regardless of the means and motives behind it." This definition overlaps with terms like false news, disinformation, misinformation, satire news, or even stories that individuals simply dislike, considering them improper. Furthermore, the evolving nature of news itself poses challenges in defining it, as it can encompass anything from a factual account of recent, significant events to a sensationalized narrative of something novel or deviant. In particular, the digitization of news has further complicated traditional definitions of news, blurring the lines between legitimate reporting and falsehoods, making the detection and understanding of fake news an intricate and evolving task (Zhou et al., 2020).

To combat the menace of fake news, researchers and practitioners have been turned to various techniques, one of which is stance classification. Stance classification involves determining the position or perspective expressed in a piece of content, such as an article, tweet, or post, with respect to a particular claim or topic. By accurately identifying the stance of a piece of content, it becomes possible to analyze and assess its credibility, authenticity, and potential for spreading misinformation.

The application of stance classification specifically for fake news detection in social media has gained significant attention due to the massive amount of user-generated content and the speed at which information spreads across platforms. Stance classification models can be trained to classify content into different stance categories, such as supporting, opposing, or neutral, with respect to specific claims or topics. These models utilize natural language processing (NLP) techniques, machine learning algorithms, and large annotated datasets to learn patterns and features that differentiate between genuine and deceptive content.

The classification of stances in social media for fake news detection is a challenging task due to several reasons. First, social media content is often characterized by brevity, colloquial language, and unconventional grammar, making it different from traditional news articles or formal texts. Second, fake news perpetrators often employ sophisticated techniques, such as using partial truths or exploiting emotional triggers, to make their content more persuasive and believable. This necessitates the development of robust and adaptive stance classification models that can effectively handle these complexities.

The implications of accurate stance classification for fake news detection are far-reaching. By identifying the stance of a piece of content, social media platforms can employ targeted interventions, such as warning labels, fact-checking notifications, or reducing the visibility of potentially deceptive content. Furthermore, researchers can leverage stance classification data to gain insights into the prevalence and spread of fake news, the tactics employed by misinformation propagators, and the underlying motivations behind the creation and dissemination of false information. In conclusion, the rapid spread of fake news on social media platforms poses a significant challenge to the credibility and trustworthiness of online information. Stance classification offers a promising approach to tackle this issue by enabling the automated identification and analysis of the positions expressed in social media content. By developing robust stance classification models, we can empower social media platforms, researchers, and individuals to make informed decisions about the veracity of information, combat the spread of fake news, and uphold the integrity of public discourse.

Stance Classification has been defined in various ways in previous studies. Gayathri et al. (2013) described it as the process of automatically assigning electronic documents to predefined categories or classes based on their content. It involves labeling a text based on

its position regarding a specific target, which can be in favor, against, or neutral (Du et al., 2017). Stance detection in natural language texts entails analyzing the viewpoint or attitude of the text's author toward a specific subject or group of subjects (Küçük et al., 2020). Common stances include expressions of support, denial, commentary, or questioning of an existing claim or fact. Identifying the stances authors adopt, particularly in response to claims such as those found in online commentary, provides valuable insights, as it can unveil rumors and false assertions when closely observed (Aker et al., 2017).

Stance detection, also known as stance classification, is a computational technique used to identify the stance or perspective expressed in a piece of text. It involves determining whether the text supports, opposes, or remains neutral toward a particular claim, topic, or target. Stance detection plays a crucial role in various domains, including fake news detection, sentiment analysis, and opinion mining. Accurately categorizing the stance of a piece of text enables researchers and practitioners to gain valuable insights into public opinion, ideological differences, and the spread of misinformation.

Stance detection models typically leverage natural language processing (NLP) techniques, machine learning algorithms, and annotated datasets to learn patterns and features that differentiate between different stances. This computational approach allows for the automated analysis of large volumes of text data, enabling faster and more efficient understanding of people's attitudes and beliefs (Kuyumcu et al., 2019).

Text classification is a fundamental task in NLP that involves assigning predefined categories or labels to a given piece of text. It is a vital technique for organizing and making sense of unstructured textual data. Text classification has a wide range of applications, such as sentiment analysis, topic categorization, spam detection, and document classification. By leveraging machine learning algorithms and NLP techniques, text classification models learn patterns and features from labeled training data to make predictions on new, unseen text instances. These models extract relevant information, such as keywords, linguistic structures, and contextual cues, to determine the most appropriate category or label for a given text. Text classification enables automated processing and analysis of large volumes of textual data, providing valuable insights and facilitating decision-making in various domains, including customer feedback analysis, content moderation, and information retrieval systems.

Fake News Detection (FND) refers to the process of identifying and classifying misleading or fabricated information presented as factual news. With the increasing of

social media and the ease of content sharing, the spread of fake news has become a significant concern in today's digital landscape. Fake news can have detrimental effects on public opinion, political discourse, and societal trust. FND is a critical task in NLP that aims to identify and verify the accuracy of major statements in a news article to determine the news's truthfulness. FND serves a role in preventing potential harm to specific societal segments by addressing the misrepresentation of facts, which can lead to social, political, and national consequences (Singh, P., et al., 2022). To combat this problem, researchers and practitioners employ various techniques and technologies to detect and mitigate the impact of fake news. These techniques often involve the application of NLP, machine learning, and data mining algorithms to analyze textual content, identify patterns, and assess the credibility and authenticity of the information. Fake news detection models leverage features such as linguistic cues, source credibility,
fact-checking, and social network analysis to distinguish between genuine and deceptive news. By detecting fake news, we can empower individuals, media organizations, and social media platforms to make informed decisions, promote media literacy, and maintain the integrity of public information.

In this research, we applied various machine learning and deep learning algorithms to obtain a best solution to the FND task using a large Arabic dataset, AFND (Khalil, A., et al., 2022). We considered both the Naïve Bayes model and the Linear Support Vector Classifier (SVM) model. The Naïve Bayes algorithm is a probabilistic method widely used for text classification tasks. It leverages the TF-IDF vectorization technique to convert text data into numerical features. On the other hand, the Linear SVM model utilizes a hyperplane to separate data into different classes and is known for its effectiveness in linearly separable datasets. We built pipelines for each model, combining the TF-IDF vectorization step with the respective classifier. These pipelines allowed for seamless integration of data transformation and model training. Our aim was to identify the most suitable approach for our fake news detection task by evaluating the performance of these models.

In addition to the Naïve Bayes and Linear SVM models, we also explored the effectiveness of more advanced neural network architectures for text classification. We investigated the LSTM model, which is a type of RNN capable of capturing sequential dependencies in text data. The LSTM model has shown promising results in various natural language processing tasks, including text classification. Furthermore, we

experimented with a Bidirectional RNN, which enhances the LSTM model by processing the input data in both forward and backward directions, allowing the model to capture contextual information from both ends of the sequence. Additionally, we applied the CNN-BiLSTM architecture, which combines convolutional neural networks (CNNs) with Bidirectional LSTMs, aiming to capture both local and global dependencies in the text.

Our experiments illustrated the superior performance of deep learning models in comparison to traditional machine learning approaches. During the model training phase, we observed notable instances of underfitting and overfitting, suggesting the presence of noise and complexities within the dataset. Finally, we compared the results of this study with some previous studies that worked on the Arabic Fake News Dataset (AFND) and English language dataset.

1.2. Research Motivation

The primary motivation for focusing on the topic of fake news detection, particularly for Arabic resources and information, arises from the critical need to preserve the integrity of information in the digital age. With millions of Arabic-speaking internet users and a diverse range of dialects, the Arabic-speaking world has become a hotbed for the spread of false information, misinformation, and disinformation. The consequences of this phenomenon are far-reaching, impacting public opinion, social cohesion, political stability, and even public health. Detecting fake news in Arabic is not only a technological challenge but also a societal imperative. Researchers aim to develop robust and culturally sensitive algorithms capable of effectively discerning deceptive content from credible information. Ultimately, the motivation behind Arabic fake news detection research is to empower the Arabic-speaking population and contribute to the global fight against the harmful effects of misinformation.

1.3. Literature Survey and Related Work

Recently, the auto-classification and detection of fake news have become serious concerns due to the variety and rapid emergence of fake resources, as well as their potential to cause significant public and national losses with negative effects. This issue

gained prominence, particularly in the wake of numerous concerning studies released since the 2016 US election.

To distinguish fake news from real news on social media and other online platforms, researchers employ various machine learning and deep learning techniques to develop new models. It is imperative to enhance systems that can automatically detect fake data when it surfaces online.

Several researchers have focused on detecting fake news on online platforms, utilizing datasets in multiple languages. However, certain limitations exist in the proposed modules, particularly in terms of accuracy.

(Zhou, X., et al., 2020) surveyed several studies that targeted the detection of fake and non-fake news. They explained fundamental theories, detection methods, and opportunities across various disciplines. Their comprehensive literature survey identified well-known theories that can potentially be used to study fake news-related theories and user-related theories. They detailed the detection of fake news from four perspectives:

a. Knowledge-based methods: Detect fake news by verifying if the knowledge within the news content is consistent with facts.
b. Style-based methods: Focus on how fake news is written.
c. Propagation-based methods: Detect fake news based on how it spreads online.
d. Source-based methods: Detect fake news by investigating the credibility of news sources at various stages.

On the other hand, the survey prepared by (Patil, R., et al., 2023) has focused on studying how the NLP field has evolved from rule-based and statistical approaches to more context-sensitive learned representations. This survey covers the history of text representations from the 1970s onwards, starting from regular expressions and extending to the latest vector representations used to encode raw text data, as illustrated in Figure 1. They focused on the initial stages of this evolution and showed how, due to changes in text representation, the NLP field progressed from merely comprehending fragments to comprehending all aspects of text. The neural model automatically learns the features and categorizes them into context-sensitive, context-insensitive, and pre-trained embeddings.

Moreover, it has been shown that employing various word embeddings, including domain-specific and knowledge-enriched embeddings, is valuable for capturing semantic

and syntactic connections among words within extensive collections of text, leveraging both local contextual cues and external knowledge references.

In comparison to other machine learning methods, Deep Learning (DL) techniques are known to be more effective in detecting fake news. In their recent study, (Mridha, M. F., et al., 2021) examined various advanced and state-of-the-art mechanisms for detecting fake news. The study begins by highlighting the severe consequences of fake news, and then delves into the dataset and NLP techniques used in previous research. An extensive overview of deep learning-based strategies is provided to categorize typical methodologies, along with a discussion on the prominent evaluation metrics in fake news detection. Therefore, the study analyzes fake news detection models based on NLP and advanced DL techniques, presenting a taxonomy of fake news detection approaches.

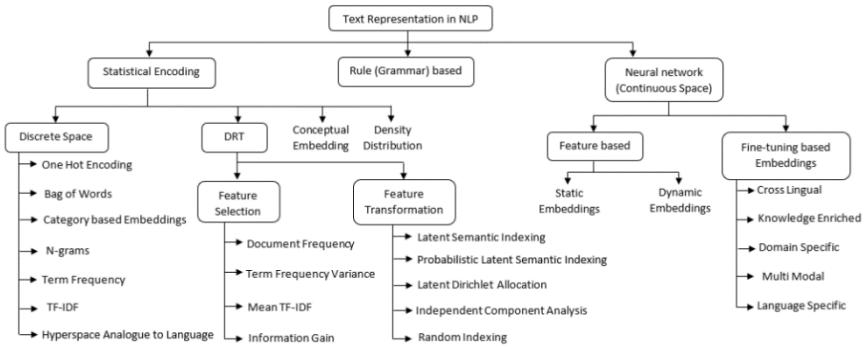


Figure 1. Embedding Taxonomy (Patil, R., et al., 2023).

Regarding the published studies and models, most of these studies have focused on determining the credibility of news articles in the English language, as demonstrated in the study by Wang (2017).

The lack of labeled benchmark datasets has limited progress in combating fake news using statistical approaches. To address this limitation, the authors collected a large dataset of 12.7K manually labeled short statements from POLITIFACT.COM, a reputable fact-checking website. The LIAR dataset not only serves as a benchmark for fake news detection but also facilitates fact-checking research. It is significantly larger than previously available public datasets of similar types. The dataset consists of authentic, real-world statements from various contexts and diverse speakers, enabling research on developing broad-coverage fake news detectors.

1.3.1. Arabic Fake News Datasets

As we have noticed, research related to the Arabic language is limited due to the absence of a labeled Arabic fake news dataset. This absence remains a significant obstacle to the advancement of computational solutions aimed at mitigating the spread of false information in Arabic content. Such datasets play a pivotal role in enhancing our understanding of Arabic fake news while fostering efforts to ensure the dissemination of reliable and trustworthy information in the Arabic language. However, further efforts are needed to expand the size, diversity, and availability of Arabic fake news datasets to better address the unique complexities and challenges specific to the Arab world.

Below, Table 1 provides an overview of the existing datasets available for the Arabic fake news detection task. Addressing this gap, Khalil et al. (2022) introduced an Arabic corpus designed specifically for training deep learning models. This dataset, known as the Arabic Fake News Dataset (AFND), stands out as the most comprehensive single-labeled and diverse collection for Arabic fake news. It comprises a staggering 606,912 articles sourced from 134 online Arabic news outlets spanning 19 different Arab countries. These countries encompass the Levant (Jordan, Palestine, Lebanon, and Syria), Arab Gulf states (United Arab Emirates, Saudi Arabia, Kuwait, Qatar, Bahrain, and Oman), Arab Maghreb nations (Algeria, Morocco, Tunisia, Libya, and Mauritania), as well as Yemen, Iraq, Egypt, and Sudan. The dataset was meticulously gathered from diverse origins, including news websites, social media platforms, and online forums, covering a wide array of subjects such as politics, health, sports, and entertainment.

Nabil, M., et al. (2015) introduced ASTD, a dataset for Arabic social sentiment analysis collected from Twitter. The dataset consists of approximately 10,000 tweets classified into four categories: objective, subjective positive, subjective negative, and subjective mixed. This study included two sets of benchmark experiments: (1) a four-way sentiment classification task and (2) a two-stage classification task. Additionally, the authors constructed a seed sentiment lexicon based on the dataset.

Salem, F. et al. (2019) presented the creation of a fake news dataset specifically focused on the Syrian war. The authors highlighted the limitations of existing fake news datasets, which primarily revolve around US politics, entertainment news, or satire. They emphasized the need for a dataset that covers the specific nature of news reporting on wars

and the scarcity of available sources for manual labeling. To address these challenges, the authors have introduced the FA-KES dataset, comprising news articles from various media outlets representing mobilization press, loyalist press, and diverse print media. Instead of relying on manual labeling, they employ a semi-supervised fact-checking approach. Crowd workers extract specific information from the articles, aiding in matching them to "ground truth" information obtained from the Syrian Violations Documentation Center. Unsupervised machine learning is utilized to cluster the articles into separate sets, resulting in a carefully annotated dataset of 804 articles labeled as true or fake.

Similarly, Ali, Z. S., et al. (2021) have proposed a new dataset called "AraFacts," which is the first extensive collection of naturally occurring claims in the Arabic language. These claims were gathered from five different Arabic fact-checking websites, including sources like "Fa-tabyyano" and "Misbar," and span claims made since 2016. The dataset consists of 6,222 claims, each paired with factual labels and additional metadata, such as the content of fact-checking articles, the category of the claim, and links to posts or web pages spreading the claims. Given that the data originates from multiple fact-checking websites, it provides valuable diversity.

Additionally, Alkhair, M. et al. (2019) have introduced a new Arabic Fake News Corpus designed for research purposes. This corpus focuses on the issue of fake news in the Arabic world, particularly on YouTube. The paper tests the feasibility of distinguishing between fake news and non-fake news comments using three machine learning classifiers: Support Vector Machine (SVM), Decision Tree (DT), and Multinomial Naïve Bayes (MNB). They observe that the performance of these classifiers varies depending on the topic of the rumor.

Alhindi, T. et al. (2021) created AraStance, a comprehensive and diverse Arabic Stance Detection dataset aimed at addressing the growing issue of misinformation and disinformation on the internet. AraStance consists of 4,063 claim-article pairs gathered from various reputable sources, including fact-checking websites and news sites. It covers both false and true claims across multiple domains and Arab countries while maintaining a well-balanced distribution between related and unrelated documents with respect to the claims. In their paper, the authors benchmarked AraStance alongside two other stance

detection datasets using BERT-based models, achieving an accuracy rate of 85% and a macro F1 score of 78%.

Similarly, Saadany, H. et al. (2020) introduced a new dataset comprising 3,185 articles scraped from two Arabic satirical news websites, "Al-Hudood" and "Al-Ahram Al-Mexici," along with a real news dataset containing 3,710 articles from official news sites such as "BBC-Arabic," "CNN-Arabic," and "Al-Jazeera News." Both datasets focus on political issues related to the Middle East.

Table 1. Arabic Datasets for Fake News Detection Task

Dataset Name/Authors	Number of articles/sentences	Class distributions	Input Data
Arabic-FakeNews (Alnabrisi, I. et al, 2023)	8k news	6k Real 2k Fake	Text
AFND (Khalil, A., et al., 2021)	606912 articles	207310 Credible 167233 Not Credible 232369 Undecided	Text
Social Media Rumors in Arabic Dataset (SMARD) (Fatima H. et al., 2021)	138 verified claims and identified 9.4K relevant tweets to those claims.	1,831 True Tweets 1,753 False Tweets 5,830 Other Tweets	Text
Dataset for Arabic Fake News (Rasha A. et al., 2021)	323 articles	100 reliable news 223 unreliable news	Text
AraFacts (Ali, Z. S., et al., 2021)	6,222 claims	0 class have 2081 2 class have 1701 1 class have 2440	Text, Image, Video
CLEF-2020 CheckThat! Lab Arabic Subtask (Hasanain, M. et al., 2020)	7.5K tweets and 14,742 Web pages	2062 Positive 5438 Negative	Text
(Hadeel S. et al., 2020)	6895 articles	3185 news 3710 real news	Text
ArSentD-LEV (Baly, R. et al., 2019)	4,000 tweets	16.3% Very Negative 30.8% Negative 22.13% Natural 20.1% Postitive 10.7% Very Postive	Text

Continued Table 1.

AraCOVID19-MFH (Ameur, M. S. H., 2021)	10,828 Arabic tweets	11463 Yes	Text
		54172 No	
		16164 Can't Decide	
		3308 News	
		7520 Opinion	
		5132 MSA	
		1346 North Africa	
		4104 Middle East	
		3933 Maybe	
FA-KES (Fatima A., et al., 2019)	804 articles	426 True articles	Text
		378 Fake articles	
(Alkhair, M., et al., 2019)	4079 YouTube comments	2363 An Egyptian dancer	Text
		1216 Algerian president	
		500 An Egyptian comedian	
Automating Credibility Assessment of Arabic News (Hammad, M., et al., 2013)	9358 articles	2606 credible	Text
		6752 not credible	

1.3.2. Fake News Detection Systems

The growing need to assess the credibility of news articles, particularly in the context of Arabic language users and the rise of e-publishing, has become increasingly evident. The absence of laws and regulations governing the credibility of e-published content, combined with political conflicts arising after the Arab Spring, has heightened the demand for automated credibility assessment.

In their work, Hammad, M., et al. (2013), the researchers, have introduced a system for automating the credibility assessment of news articles based on two essential criteria: (i) whether the article indicates the source of information and (ii) whether it indicates the time of occurrence of the reported event. For each criterion, the authors developed a classification model to classify news articles as either violating or adhering to the criteria. These models were built and evaluated using news articles previously assessed by MCE Watch, a manual service for news credibility assessment. Experimental evaluations

demonstrated that the proposed models achieved an accuracy exceeding 82% for both criteria.

Meanwhile, in parallel efforts, other researchers have been addressing the prediction of fake news by developing systems designed to assess the truthfulness of news articles shortly after their dissemination on social media. In their model called DSS, Davoudi, M., et al. (2022), have incorporated three key components: Dynamic analysis, Static analysis, and Structural analysis. Additionally, they introduced an innovative approach to gauge the sentiments expressed in responses related to news articles by constructing a stance network and extracting various graph-based features. To evaluate their proposed model, they employed the FakeNewsNet repository, which includes two prominent datasets in this field, namely PolitiFact and GossipCop. The results of their evaluation have shown promising performance, surpassing state-of-the-art methods by 8.2% on PolitiFact and 3% on the GossipCop datasets.

Several research works have introduced innovations in terms of comparing machine learning and deep learning models, evaluating datasets of various sizes, employing statistical tests for performance comparison, and proposing stacking models to enhance detection accuracy. Khalil, A., et al. (2021), utilized the AFND dataset to employ a variety of machine learning (ML) and deep learning (DL) algorithms for different purposes:

a. Capsule networks, a type of CNN model, which was improved using a static routing variant for tasks involving both image and text classification.
b. Deep double Q-learning (DDQN), a reinforcement learning (RL) algorithm that makes use of CNNs.
c. Deep pyramid CNN (DPCNN), a CNN model designed for word-level training.
d. Information Distilled LSTM (ID-LSTM), an RL model that incorporates the LSTM approach.

Apart from utilizing these deep learning models, they also incorporated other classification models into this experiment. These models encompassed support vector machines (SVC), linear SVC, K-Means, and variational Bayesian estimation of a Gaussian mixture (Bayesian Gaussian mixture). The findings revealed that the capsule network model outperformed the others, achieving the highest accuracy rates for both binary and multi-classification tasks, with accuracies of 79% and 70.9%, respectively.

Jiang, T. A. O., et al. (2021), evaluated the performance of five machine learning models and three deep learning models on two different datasets, utilizing various text representation techniques. They used evaluation metrics such as accuracy, precision, recall, and F1-score to assess the models' performance. Furthermore, they applied a corrected version of McNemar's test to determine significant differences. Based on their experiments, the authors proposed a stacking model that achieved high testing accuracy of 99.94% and 96.05% on the ISOT and KDnugget datasets, respectively, outperforming baseline methods.

- Machine Learning-Based Fake News Detection

While machine learning models for detection the fake news have been developed in various languages, Arabic has received limited attention in this regard. To address this gap, Himdi, H. et al. (2022) constructed a supervised machine learning model that focuses on evaluating the credibility of Arabic news articles through textual analysis. The authors introduced an Arabic dataset with fake news articles created through crowdsourcing and developed a unique approach for extracting textual features. And thier model achieved an impressive accuracy rate of over 75%, surpassing human performance in the same task. The study's findings underscore the significance of linguistic features in identifying deceptive text in fake news, representing a substantial advancement in Arabic fake news detection.

Additionally, Sharifani, K., et al. (2022), delved into the application of machine learning techniques in the field of NLP to enhance the accuracy and reliability of fabricated news detection models. They employed three classifiers: Passive Aggressive, Support Vector Machine, and Naïve Bayes on two publicly available datasets, experimenting with various features to extract optimal text characteristics for improved classification. Their study aimed to explore the interplay between NLP and machine learning, offering insights into the essential attributes of both. As a result of their developed system, the best-performing classifier was Passive Aggressive (PA), achieving an accuracy rate of up to 93%.

Twitter serves as a social networking and microblogging platform that enables individuals to express and disseminate their thoughts through tweets. Given its brevity, Twitter provides a valuable platform for collecting and analyzing data, especially for

research purposes. Consequently, numerous studies have accessed Twitter's data repository to analyze misleading or deceptive text.

In a research effort by Jardaneh, G. et al. (2019), the authors employed machine learning to distinguish Arabic fake news tweets using a supervised classification model. They employed user- and content-related features and incorporated sentiment analysis to generate new features. Their work involved four algorithms: AdaBoost, Random Forest, Logistic Regression, and Decision Tree. The evaluation results demonstrated that their model could filter out fake news with an accuracy of 76%.

In research shared by Samih, Y., et al. (2021), tweets have been filtered based on user-stated locations. They focused on the 5,000 most active users with at least 10 tweets. The experiment was conducted on two training sets, which were separated based on whether they could crawl users' timelines or not. Four different classification setups were employed: FastText, SVM with retweeted accounts as features (SVMRT), SVM with all words as features (SVMT EXT), and fine-tuned BERT embeddings with a dense neural layer and SoftMax output (BERT). Results showed that BERT yielded the best performance in terms of accuracy, macro precision, recall, and F1-score for most topics.

The researchers (Singh, P., et al., 2022) have concentrated on fake news detection using a novel multimodal stacked ensemble-based approach named SEMI-FND. They introduced a framework that amalgamates textual and image analysis, leveraging BERT and ELECTRA models for text analysis and NasNet Mobile for image analysis. The approach attains high accuracies of 85.80% and 86.83% on the Twitter MediaEval and Weibo datasets, respectively, surpassing recent works with similar objectives. Notably, the proposed framework diminishes the number of parameters employed in training by at least 20%, with a substantial 60% reduction in text-related parameters. The authors conclude that the stacked ensemble significantly enhances fake news detection while improving speed and reducing computational requirements.

In a separate study, Ahuja, N., et al. (2020) introduced S-HAN, a model specifically designed for fake news detection on social media platforms. The proposed model, S-HAN, represents an enhanced version of Hierarchical Attention Networks (HAN), incorporating stacked bidirectional Gated Recurrent Units (GRU). This modification empowers the model to pinpoint important words and sentences within news articles, thereby facilitating

the detection of misinformation. Experimental results underscore that S-HAN outperforms standard baseline deep learning models, achieving an accuracy of 93.63% on a real-world dataset.

The studies by Rahab, H., et al. (2021) and Hawashin, B., et al. (2023) have addressed the pressing issue of FND in the context of Arabic text on social media. Rahab, H., et al. (2021) provided an overview of the methods employed in recent years to combat Arabic spam content and fake news. Twitter emerged as the most extensively investigated platform in this context. Their survey explored the works of Hawashin, B., et al. (2023), who introduced an optimized model for Arabic fake news classification task. Through experiments, they demonstrated that optimizing feature selection significantly improves detection accuracy. Importantly, this performance enhancement brings the results closer to those achieved by deep learning methods, all while simplifying model complexity and reducing training time.

A pioneering solution has been introduced in the paper by Abd Elaziz, M. et al. (2023) in the form of a Hybrid Multitask Learning Framework, synergizing with the innovative Fire Hawk Optimizer. Through the integration of multi-task learning and a pre-trained transformer-based model, the framework effectively extracts contextual features from Arabic social media posts, paving the way for a comprehensive understanding of the disinformation landscape. To enhance the feature selection process, a modified version of the Fire Hawk Optimizer is employed, resulting in an elevated disinformation detection rate. Rigorous evaluations conducted on diverse Arabic social media datasets demonstrate the framework's prowess, achieving an impressive accuracy of 59% and excelling in precision, recall, and F-measure metrics.

Many studies are also focused on enhancing transformer architecture models. In the study by Raza, S., et al. (2022), they introduced FND-NS (Fake News Detection through News Content and Social Context) model, that adapts bidirectional and auto-regressive Transformers (BART) for the task of FND. This model leverages data from both social contexts and news articles to identify fake news. Comprehensive experiments were conducted using real-world datasets: NELA-GT-2019 and Fakeddit. The proposed model's performance results demonstrated an accuracy of 74.89%, precision of 72.40%, a recall value of 77.68%, AUC of 70.4%, and an F1-score of 74.95%.

In another study, Nassif, A. B., et al. (2022) developed and evaluated transformer-based classifiers to detect fake news. They utilized eight state-of-the-art Arabic contextualized embedding models, including XLM-Roberta and GigaBERTv4, which are multilingual models. Additionally, other models such as Arabert, Arabic-Bert, ArBert, MARBert, Araelectra, and QaribBert base, trained on Arabic text using the Bert structure, were employed.

Antoun, W., et al. (2020), in their research, presented a methodology for pre-training BERT specifically for the Arabic language, aiming to achieve results similar to what BERT accomplished for the English language. This study introduces a representation model for the Arabic language aimed at advancing the state-of-the-art in diffirence Arabic Natural Language Understanding (NLU) tasks. The ARABERT model serves as a foundational element for achieving state-of-the-art results across multiple NLP tasks in various languages. The proposed model underwent evaluation across three Arabic NLU downstream tasks: Sentiment Analysis (SA), Question Answering (QA), and Named Entity Recognition (NER). Datasets encompassing both Dialectal Arabic (DA) and Modern Standard Arabic (MSA) were utilized. The results of the AraBERT model demonstrate state-of-the-art performance in most of the evaluated Arabic NLP tasks.

In a related study by Al-Yahya, M. et al. (2021), a comparative investigation was conducted, exploring transformer-based and neural networks approaches using four datasets: ArCOV19-rumors, AraNews, Arabic news stance (ANS), and Covid-19-fakes evaluation. The research aimed to determine the effectiveness of the latest deep learning models and large-scale language models for AFND. Multiple text preprocessing pipelines were employed, including the utilization of fastai as a lemmatizer for Arabic, considering the language's intricate morphology. Lemmatization is a crucial step in text mining applications. However, it's worth noting that this study faced certain limitations, including a small dataset size, issues related to tweet repetition and unavailable tweets, and the presence of noise in the tweet data. The results of this study demonstrated that transformer-based models outperformed neural network-based models, leading to a significant increase in the F1 score, from 0.83 (with GRU as the best neural network-based model) to 0.95 (with QARiB as the best transformer-based model), resulting in a 16% improvement in accuracy. Additionally, the study found that AraBERT v02 exhibited the best

generalization compared to all other models. Furthermore, the research revealed that linear models were the most effective for handling small-sized datasets without overfitting.

In contrast, Safaya, A., et al. (2020) described an approach that combines pre-trained BERT models with Convolutional Neural Networks for identifying offensive speech in social media. They focused on sub-task A of the Multilingual Offensive Language Identification shared task (OffensEval 2020) within SemEval 2020. The authors demonstrated that the combination of CNN with BERT yielded superior results compared to using BERT alone. Their system achieved the 4th rank with a macro-averaged F1-Score of 0.897 for Arabic, the 4th rank with a score of 0.843 for Greek, and the 3rd rank with a score of 0.814 for Turkish. Additionally, the authors introduced ArabicBERT, a collection of pre-trained transformer language models for Arabic, which they have made available to the community.

- Deep LearningBased Fake News Detection

(Seyam, A., et al., 2021) conducted a survey of numerous studies pertaining to the most effective Deep Learning (DL) models employed in the detection of false news and information. Their primary focus centered on DL models and associated techniques. Their research revealed that the accuracy of an individual module may fluctuate based on the dataset used for its evaluation. Consequently, there exists no universally applicable assessment dataset suitable for testing multiple modules and appraising the accuracy of various methodologies. This underscores the necessity of not exclusively favoring one module over another based solely on accuracy, unless they undergo comprehensive testing and evaluation using consistent features and datasets from all perspectives.

Another study emphasizes the significance of advanced DL techniques in decreasing the spread of Arabic fake news and explores their potential applications across various domains. (Najadat, H. et al., 2022) focused on employing deep learning models, including LSTM and CNN, to ascertain the relevance of news headlines to their corresponding articles. The dataset used in this study centers around the war in Syria and Middle East political issues, comprising 422 claims and 3,042 articles. The results obtained from these models are promising. The research process involves data cleaning, feature extraction, and the utilization of word embeddings. The dataset is divided into training, validation, and testing sets, allocated at 60%, 20%, and 20%, respectively. The findings indicate that the

combination of CNN and LSTM (AFND-CNN-LSTM) outperforms the use of LSTM alone (AFND-LSTM), achieving an accuracy of 70% compared to 68.2%.

In addition, (Huang, Y. F., et al., 2020) have introduced a novel system for detecting fake news, centered around an ensemble learning model that incorporates deep learning techniques. This ensemble learning model combines four distinct components: embedding LSTM, depth LSTM, LIWC CNN, and N-gram CNN. To enhance the accuracy of FND, the authors fine-tuned the weights of the ensemble learning model through the application of the Self-Adaptive Harmony Search (SAHS) algorithm. The experimental results achieving an impressive accuracy rate of 99.4%. Moreover, they delved into the challenge of cross-domain difficulty and achieved a remarkable accuracy of 72.3% in that context. The authors concluded that the quality of the training data and the choice of extracted features are pivotal factors in elevating the performance of deep learning models. They observed that LIWC vectors and n-grams provide more discriminative power than embedding words and sentence depths within datasets specific to a single domain. Furthermore, the ensemble learning model consistently outperforms individual models, emphasizing the importance of optimizing weights using the SAHS algorithm to enhance accuracy

In an alternative deep learning methodology, (Harrag, F. et al., 2022) have introduced a fact-checking technique tailored for assessing the veracity of Arabic news articles or claims. Their approach relies on a Convolutional Neural Network and draws upon a comprehensive Arabic dataset that covers various facets, such as stance detection, rationale identification, retrieval of pertinent documents, and the actual fact-checking process. Through rigorous training on specific features, their model attained an impressive accuracy rate of 91%, surpassing the performance of state-of-the-art methods on the same Arabic dataset.

CNN-BiLSTM stands out as one of the popular and effective DL models, and numerous researches have employed, introducing various architectural modifications. In their work, (Ouassil, M. A., et al., 2022) have presented a FND model that amalgamates word embedding techniques with a hybrid DL model. The have underscored the pervasiveness of fake news in online sources and the critical need for its detection and prevention. Their DL model have combined a hybrid CNN and BiLSTM model with word embedding techniques. Their findings revealed that the most favorable outcomes were achieved by blending a pre-trained Word2Vec CBOW model and a Word2Vec Skip-Word

model with CNN on BiLSTM layers. The combined approach yielded an impressive accuracy rate of up to 97%.

Another deep learning approach, (Farha, I. A., et al., 2020) developed a model capable of detecting hate speech and offensive language while also predicting sentiment. They explored diverse methods for hate speech and offensive language detection, including deep learning (specifically BiLSTM and CNN-BiLSTM), transfer learning, and multitask learning. Their proposed multitask learning architecture proved to be the most effective for both hate speech and offensive language detection. Furthermore, they examined the impact of incorporating sentimental information, which demonstrated its utility.

The studies conducted by (Baniata, L., et al., 2016) and (Alharbi, O., 2021) have introduced a DL model for Arabic sentiment analysis. In both studies, the proposed approche combines CNN and BiLSTM. In the study by (Baniata, L., et al., 2016), a comparison between two architectures, CNN-BiLSTM and BiLSTM-CNN, revealed that CNN-BiLSTM achieved a test accuracy of 86.43%.

In contrast, (Alharbi, O., 2021) introduced a modification to the traditional deep learning architecture by substituting the fully connected layer with an SVM classifier. This classifier makes use of embedded vectors derived from CNN and BiLSTM for the purpose of polarity classification in Arabic reviews.

(Kaliyar, R. K., et al., 2021) have introduced a hybrid model designed for the effective detection of fake news, with a particular focus on COVID-19-related news articles. This innovative model utilizes multiple branches of a CNN with LSTM layers, each employing various kernel sizes and filters to enhance its depth and feature extraction capabilities. Three dense layers are also incorporated into the model.

The authors have meticulously curated a novel dataset named FN-COV, which comprises 69,976 fake and real news articles related to the COVID-19 pandemic. These articles are tagged with keywords such as social distancing, COVID-19, and quarantine. To evaluate the model's performance, they utilized another real-time fake news dataset known as PHEME. The combined kernels and layers in the C-LSTM network have exhibited promising results for both datasets. With this proposed model, the authors achieved an accuracy rate of 91.88% on the PHEME dataset, surpassing existing models, and an accuracy of 98.62% on the FN-COV dataset. This underscores the effectiveness of the hybrid model in detecting fake news across both general and COVID-19-specific contexts.

In conclusion, the authors demonstrate the effectiveness of their C-LSTM model in capturing temporal semantics and phrase-level representations, yielding optimized accuracies with minimal loss. Additionally, they contribute a valuable novel dataset of fake news articles during the COVID-19 pandemic.

The experimental results validate the efficacy of the proposed model for detecting fake news using both the PHEME and FN-COV datasets. (Truică, C. O., et al., 2023) have introduced an end-to-end solution encompassing two novel deep learning architectures for fake news detection. Additionally, they have presented a real-time algorithm named MCWDST (Minimum-Cost Weighted Directed Spanning Tree) and a ranking function designed to identify and immunize harmful nodes within the network. The deep learning architectures employ convolutional and BiLSTM layers to capture semantic, syntactic, and contextual information, achieving state-of-the-art performance on five real-world datasets. The MCWDST algorithm constructs a minimum cost weighted directed spanning tree for a detected node, while the ranking function takes network information and harmful content into account, effectively mitigating the spread of fake news. This comprehensive solution demonstrates effectiveness in real-time fake news detection and combating.

In a separate study, (Shang, L. et al., 2020) have introduced an optimized sentiment analysis model, CNN-BiLSTM-Attention, tailored for accurately assessing the emotional tendencies of film reviews. This model amalgamates the strengths of CNN and LSTM models. CNN excels in capturing local correlations, while LSTM can process sequences of varying lengths and capture long-range dependencies. By incorporating an attention mechanism, the proposed model achieves improved accuracy in analyzing sentiment characteristics of texts compared to CNN and LSTM models alone. Experimental results clearly demonstrate the effectiveness of the proposed model, albeit highlighting increased complexity and longer training times. The study concludes that the optimized model holds practical significance for sentiment analysis of film reviews and suggests possibilities for further improvements.

In a related study, (Abdelhady, N., et al., 2022) introduced Stacked-CNN-BiLSTM-COVID mode, specifically designed for sentiment analysis of Arabic COVID-19 tweets. This model combines CNN and BiLSTM. To represent tweets as vectors, the authors employed word embedding models, including Aravec, FastText, and ArWordVec, to assess their impact on the model's performance. The proposed model was subjected to a comparative analysis alongside other DL models such as CNN, LSTM, and BiLSTM. The

experiments were carried out on three different Arabic datasets related to COVID-19 and vaccine-related discussions. The empirical results consistently demonstrated that the Stacked-CNN-BiLSTM-COVID model outperformed other models, achieving high F-measure scores of 76.76%, 87.25%, and 80.5% on the SenWave, AraCOVID19-SSD, and ArCovidVac datasets, respectively.

(Sorour, S. E. et al., 2022) introduced a system named AFND (Arabic Fake News Detection) designed for detecting fake news in Arabic by employing a hybrid DL model known as CNN-LSTM. The introduced model involves preprocessing the input dataset, incorporating word vectors as pre-trained vectors based on Arabic news, and utilizing the JSO optimization algorithm to determine the optimal structure for the CNN-LSTM model. Comparative experiments have unequivocally demonstrated that the proposed CNN-LSTM model achieved the highest performance with an impressive accuracy rate of 81.6%. The study highlights comprehensive improvements in Arabic fake news detection, showcasing the potential of the proposed methodology.

(Abu waik, K., et al., 2019) focused on sentiment analysis (SA) of Dialectal Arabic using deep learning methods. The authors have proposed a model that combines LSTM with CNN, achieving superior performance compared to two baseline models. Their model attains high accuracy, ranging from 81% to 93% for binary classification and 66% to 76% for three-way classification tasks. This model is considered state-of-the-art for applying deep learning to sentiment analysis in Dialectal Arabic. The authors initially experimented with a simple LSTM architecture but obtained subpar results. Subsequently, they adopted an existing sentiment analysis (SA) model that combines LSTM and CNN, which demonstrated improved performance. Ultimately, they introduced their model, a more advanced BiLSTM-CNN architecture with additional convolutional layers, which achieved state-of-the-art results on datasets where deep learning approaches had been previously applied.

The study's results are promising, but the authors have acknowledged room for further improvement, particularly in three-way classification tasks. They suggest exploring the use of task-specific word embeddings and employing more complex deep learning architectures, including those incorporating attention mechanisms. Additionally, they plan to expand the size of the ShamiSenti dataset for comparative analysis and investigate the impact of data size on model performance.

In a related study, (Abdullah, M., et al., 2018) have focused on sentiment and emotion detection in Arabic text, particularly in Arabic tweets. They introduced their system, SEDAT, which utilizes deep learning architectures to achieve accurate sentiment and emotion detection. This approach leverages word and document embeddings, along with semantic features, and applies CNN-LSTM and fully connected neural network architectures. The results of the SEDAT system exhibit significant improvements in Spearman correlation scores compared to baseline models. This approach has demonstrated promise in accurately detecting the intensity of emotions and sentiments in Arabic tweets. The authors employed various embedding models and feature vectors extracted from both Arabic and translated tweets, feeding them into various DL network architectures. SEDAT was rigorously evaluated on the SemEval-2018 Task 1 datasets and achieved competitive results, closely trailing the first-ranked model in the challenge. The system also showcased proficiency in detecting sentiments and emotions across different Arabic dialects and Modern Standard Arabic.

In a related context, (Zhou, C. et al., 2015) introduced a novel C-LSTM model that combines the strengths of CNN and LSTM for sentence representation and text classification. C-LSTM utilizes CNN to extract higher-level phrase representations and subsequently employs LSTM to obtain the overall sentence representation. This unique approach enables C-LSTM to capture both local phrase features and global temporal sentence semantics effectively. The proposed C-LSTM model underwent evaluation on sentiment classification and question classification tasks. The experimental results unequivocally demonstrate that C-LSTM outperforms both CNN and LSTM models, achieving remarkable performance in these tasks. The acquired semantic sentence representations prove to be highly effective for sentiment analysis and question classification.

In summary, the paper introduces C-LSTM as a unified model that effectively combines CNN and LSTM, facilitating the learning of phrase-level features and the capture of long-term dependencies. The model exhibits highly satisfactory performance in both sentiment classification and question-type classification tasks.

In the study conducted by (Abedalla, A. et al., 2019), the focus was on the detection of fake news using deep learning techniques. Leveraging the Fake News Challenge dataset, the authors developed various models that harnessed Convolutional Neural Network (CNN), Long Short-Term Memory network (LSTM), and Bidirectional LSTM to discern

fake news based on the relationship between the article headline and body. The experiments conducted in this research yielded an impressive accuracy rate of 71.2% on the official testing dataset, surpassing the results achieved in previous studies conducted on the same dataset.

The study underscores the widespread reliance on social media for news consumption and the subsequent rise in the dissemination of fake news. The proliferation of deliberately false information on online platforms has a detrimental impact on society. To address this issue, the research explores existing machine learning and deep learning approaches for detecting fake news and introduces four deep learning models utilizing CNN and LSTM networks. The best-performing model achieved a notably high accuracy score on the official test data, surpassing the results of previous studies.

Researchers (Ghourabi, A. et al., 2020) delved into the persistent issue of SMS spam, which continues to plague users despite the evolution of messaging services. Their study introduces a hybrid deep learning model for the detection of SMS spam messages, with a specific focus on mixed text messages written in Arabic or English. This model ingeniously combines Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) techniques to achieve effective detection.

In the experimental evaluation, the CNN-LSTM performance was meticulously compared to that of other well-known machine learning algorithms, including Support Vector Machine, K-Nearest Neighbors, Naive Bayes, Decision Tree, Logistic Regression, Random Forest, AdaBoost, Bagging classifier, and Extra Trees. The proposed model is versatile, designed to classify SMS spam across various contexts, encompassing mobile network, Facebook Messenger and WhatsApp messages. The evaluation dataset comprises real messages in both Arabic and English languages, ensuring the model's effectiveness in real-world scenarios. The model's impressive accuracy of 98.37%.

Meanwhile, (Hifny, Y. et al., 2020) have directed their focus towards emotion recognition from speech signals, with a specific emphasis on the Arabic language. To accomplish this, they have implemented two distinct neural architectures. The attention-based CNN-LSTM-DNN ad the deep CNN model. Experimental results have conclusively shown that the proposed attention-based CNN-LSTM-DNN model outperforms the deep CNN baseline system, resulting in improved accuracy in Arabic speech emotion recognition. It is essential to note, though, that the deep CNN models excel in terms of computational efficiency during both training and classification.

Table 2. The Summarization of The Related Works

#	Authors	Dataset	Train/Val/ Test %	Preprocessing			Embedding	Classification Model(s)
				Punctuation(**P**)/ Stopwords(**W**)/Non-Arabi Char Removal (**C**)	lemmatization(**L**)/ Stemming(**S**)/ Tokenization(**T**)	Removing Arabic Diacritics /Duplicate		
1	Truică, C. O., et al., 2023	FNC	80/20	**P** **W**	**L**		Word2Vec	CNN-3BiLSTM
		Kaggle FND					GloVe	
		GossipCop					Word2Vec	3BiLSTM
		Fakeddit						
2	Hawashin, B., et al., 2023	AFND	50/50	**W**	**T** **S**		TF-IDF	RF, NB
								SVM, LR
								AraBERT
								K-NN
3	Wotaifi, T. A. et al, 2023	AraNews	80/20		**T**		GloVe	Text-CNN
							BERT	LSTM
							Fast Text	Hybrid Network
4	Alawadh, H. M. et al, 2023	Original DS	90/10		**T**			DT, RF, NB
		Augmented DS	80/20					Linear SVM
								mini-BERT

Continued Table 2.

5	Alnabrisi, I. et al, 2023	Arabic-FakeNews	80/20	P W C	T	✓	Fast Text	BiLSTM
								BiGRU
								AraBERT v02
								Araelectra
								AraGPT2
							Word Embedding	QARiB
								AraBERT
								MARBERT
								Zero-Shot models
6	Awajan, A., 2023	New Arabic dataset	75/25		T			LSTM
								BERT
								Shallow Learning
7	Fouad, K. M., et al., 2022	3 datasets	80/20	P W	T S		Word Embedding	CNN , LSTM, BiLSTM
								CNN + BiLSTM/ LSTM
								Linear SVM
								MultinomialNB
								BernoulliNB
								SGD, SVM
								DT, RF, K-NN

Continued Table 2.

8	Nassif, A. B., et al., 2022	Two distinct datasets	80/20	P C	T		Word Enbedding	XLM-Roberta
								GigaBERTv4
								QaribBert ,MARBert
								Arabic-Bert, Araelectra
9	Raza, S., et al., 2022	NELA-GT-2019	75/10/15					FND-NS
		Fakeddit						
10	Sharifani, K., et al., 2022	18000 news articles	80/20					Passive Aggressive
								SVM, NB
11	Davoudi M., et al., 2022	PolitiFact		P W				DSS, SVM, GND
								RF, DT, LR
		GossipCop						XGBoost
12	Sorour, S. E., et al., 2022	ANS	80/10/10			✓	Word Enbedding	CNN- LSTM
								AraBERT v02
								QARiB, BERT
								AraGPT2, LSTM
13	Diwali, A. et al., 2022	ASTD		P W C	S	✓	TF	LR, SVM
		AJGT					TF-IDF	LSTM
							Word Embedding	BiLSTM
		ArTwitter					fastText	CNN

Continued Table 2.

14	Ouassil, M. A., et al., 2022	WELFake	80-20				Word2Vec CBOW Glove Word2Vec Skip-Word	CNN-BiLSTM
15	Harrag, F. et al., 2022	Arabic Fact-Checking Stance Detection Corpus	5-fold cross	W	T L		Word Embedding	CNN
16	Himdi, H. et al., 2022	Arabic fake news dataset based on crowd-sourcing	70/30	C	T	✓		Random (baseline) NB, RF, SVM
17	Ali, S. B. et al., 2022	Original data set SMOTE-ed data set	80/10/10					NB, RF, SVM LSTM CNN + LSTM
18	Khalil, A., et al., 2021	AFND	90/10 > multi-classification 90/10 > binary classification	C	S		Word Enbedding	Capsule networks DPCNN, DDQN Linear SVC ID-LSTM KMeans

Continued Table 2.

19	Jiang, T. A. O., et al., 2021	ISOT KDnugget PHEME	80/20	W		✓	GloVe	LR, DT, RF K-NN LSTM, GRU SVM, CNN
20	Kaliyar, R. K., et al., 2021	FN-COV PHEME	80/20				Word Embedding	CNN-LSTM
21	Alharbi, O., 2021	LABR OMCCA	70/15/15	P		✓	CBOW Skip-gram TF-IDF	linear SVM CNN CNN-BiLSTM
22	Antoun, W., et al., 2020	HARD ASTD ArSenTD-Lev LABR AJGT	80/20	P W C	T	✓	Word Embedding	ARABERT
23	Samih, Y., et al., 2021	8 polarized US-centric topics	80/20	C	T			FastText SVMRT SVMT EXT SVM, BERT
24	Farha, I. A., et al., 2020	SemEval 2020 Arabic	80/20	P C		✓	Skip-gram word2vec	BiLSTM CNN-BiLSTM MTL-S-N

Continued Table 2.

25	Shang, L. et al., 2020	Stanford's IMDb	80/20					CNN BLSTM CNN-BiLSTM-Attention depth LSTM LIWC CNN N- gram CNN
26	Huang, Y. F., et al.,2020	Cross-domain	80/20		T		Word2vec N-gram	Embedding LSTM depth LSTM LIWC CNN N- gram CNN
27	Hifny, Y. et al., 2020	Phase-mix	80/20				Word Embedding	CNN-LSTM-DNN CNN , LR
28	Jardaneh, G. et al., 2019	1862 tweets	80/20		S			DT, RF AdaBoost
29	Abu Kwaik, K., et al., 2019	LABR ASTD Shami-Senti	80/20	P W C			Aravec-CBOW CBOW	LSTM BiLSTM- CNN
30	Abedalla, A. et al., 2019	FNC-1	70/30	P W C	S	✓	GloVe	CNN, LSTM Bi-LSTM SVM
31	Sabbeh, S. F. et al, 2018	800 Arabic tweets			T			NB, DT Proposed System
32	Abdullah, M., et al., 2018	SemEval-2018 Task 1	80/20				Word Embedding	SEDAT CNN-LSTM

Continued Table 2.

33	Baniata L., et al., 2016	LARB	70/15/10	**P** **W**	**T**	✓		CNN-BiLSTM
34	Zhou, C. et al., 2015	SST	72/10/18			✓	N-gram	C-LSTM LSTM Bi-LSTM
35	Farha, I. A., et al., 2022	SenWave	70/30	**W**		✓	Aravec	Stacked-CNN-BiLSTM-COVID
		AraCOVID19-SSD					FastText	CNN
		ArCovidVac					ArWordVec	LSTM, BiLSTM
36	Alyoubi, S. et al., 2023	ArCOV19-Rumors	5-fold cross	**P**		✓	Word2vec	CNN
							Embedding Layer	MARBERT-CNN
		COVID-19 misinformation					FastText	CNN-RNN
							ARBERT and MARBERT	
37	Bahurmuz, N. O. et al., 2022	COVID-19 misinformation		**C**		✓		AraBERT
		DS of Mahlous, A. R. el al,2021						MARBERT
		DS of Alzanin, S. M. et al, 2019						

Continued Table 2.

38	Al-Yahya, M. et al., 2021	ArCOV19-Rumors	80/20	PWC	LT	✓	Word Enbedding	AraBERT
		AraNews						AraGPT2
		ANS						Linear (WL-W2V)
		Covid-19-fakes						CNN (WL-F)
								RNN (ChL-F)
								GRU (WL-W2V)
								AraELECTRA

2. METHODOLOGY

This chapter provides in-depth details regarding the dataset employed in our experiments, the experimental settings, preprocessing techniques, and the baseline method used for fake news detection. The methodology for detecting fake news encompasses a multifaceted approach that leverages various techniques from the domains of natural language processing (NLP), machine learning (ML), and data analysis.

The process typically commences with the loading and cleaning of the data. This crucial step involves curating a diverse and representative dataset containing both real and fake news articles. Subsequently, the data is fed into the model, output is generated, loss is calculated, gradients with respect to weights are computed, and the model is updated iteratively.

Feature engineering is another pivotal aspect of our methodology. It involves the extraction of relevant features from the text, encompassing elements such as word frequencies, n-grams, and semantic representations. These features enable the model to capture meaningful patterns within the data.

Our experimentation incorporates a wide array of machine learning models, including Support Vector Machines (SVMs), Random Forests, and advanced deep learning architectures such as Convolutional Neural Networks (CNNs). These models are meticulously trained on the features derived from the dataset, with the AFND dataset forming the foundation for this training process.

In addition to content-based features, metadata attributes (e.g., title, text, publication date) and social network analysis are integrated into the detection process, further enhancing its accuracy and effectiveness.

To evaluate the performance of our models and ensure their generalizability, we employ cross-validation techniques, notably k-fold cross-validation.

2.1. Arabic Fake News Corpuses Challenges

The domain of Arabic fake news detection has witnessed a growing interest in recent years, leading to the development of several dedicated datasets tailored for this purpose. These datasets serve the essential role of providing researchers and practitioners with labeled examples of both real and fake news in the Arabic language. In turn, this facilitates

the training and evaluation of machine learning models and algorithms designed to combat misinformation in Arabic media. While the emergence of Arabic fake news datasets is promising, it also brings forth challenges and limitations, as outlined below:

a. Scarcity of Large-Scale Datasets: A significant challenge lies in the scarcity of large-scale, publicly available labeled datasets specifically geared towards Arabic fake news. In contrast to languages like English, where numerous datasets are available, the limited availability of Arabic datasets hampers the development and evaluation of robust machine learning models and techniques.

b. Subjectivity and Contextual Nature: Labeling fake news in Arabic can be subjective and context-dependent. Achieving consensus among annotators can be challenging, as interpretations may vary.

c. Diverse Linguistic Landscape: Arabic encompasses a diverse range of dialects and writing styles. This linguistic diversity introduces complexities in both dataset creation and subsequent analysis.

d. Lack of Standardized Metrics: The absence of standardized evaluation metrics and benchmarks for Arabic fake news detection poses challenges for comparing and benchmarking different approaches. This hinders the establishment of best practices in the field.

e. Privacy, Legal, and Ethical Constraints: Privacy concerns, legal restrictions, and ethical considerations may limit access to certain types of data, particularly when dealing with sensitive information.

Despite the formidable challenges and limitations discussed, ongoing efforts are underway to expand the availability of Arabic fake news datasets and to address the unique linguistic and cultural aspects associated with misinformation in the Arab world. It is imperative to emphasize the importance of sustained collaboration, research endeavors, and data sharing to surmount these hurdles and propel the field of Arabic fake news detection and mitigation forward.

Arabic fake news datasets serve as invaluable resources for the study and combat of misinformation in the Arabic language. These datasets encompass collections of news articles, social media posts, or textual content meticulously labeled as either real or fake.

They are thoughtfully curated to encompass a diverse spectrum of topics, sources, and genres, effectively capturing the complexities and nuances prevalent in Arabic media.

Such datasets empower researchers and practitioners to craft, refine, and assess machine learning models, natural language processing techniques, and information retrieval systems tailored to the identification and mitigation of fake news in Arabic. Through the analysis of these datasets, experts gain profound insights into the distinct characteristics, patterns, and propagation dynamics of misinformation within the Arabic-speaking sphere. This knowledge is instrumental in the development of robust countermeasures, thus mitigating the adverse impact of fake news on individuals, societies, and democratic processes.

2.1.1. Arabic Fake News Detection (AFND) Dataset

In our study, we utilized the AFND dataset, proposed by (Khalil, A., et al., 2022). AFND stands as a substantial Arabic dataset, comprising a collection of 606,912 public news articles sourced from 134 public news websites representing 19 distinct Arab countries. These countries encompass the Levant countries (Jordan, Palestine, Lebanon, and Syria), Arab Gulf countries (United Arab Emirates, Saudi Arabia, Kuwait, Qatar, Bahrain, and Oman), Arab Maghreb countries (Algeria, Morocco, Tunisia, Libya, and Mauritania), as well as Yemen, Iraq, Egypt, and Sudan. Importantly, AFND is publicly accessible and serves as a valuable resource for research endeavors in the domain of Arabic news analysis.

Table 3 provides a snapshot of key statistical information regarding the corpus articles and the online sources encompassed within AFND. In our experiments, we tackled binary and multi-classification tasks. For binary classification, we worked with a dataset consisting of 83,616 articles for training and 250,850 articles for testing, with articles labeled as "undecided" being excluded from this binary classification task. In the context of multi-classification experiments, we leveraged a dataset comprising 388,405 training articles, 97,101 validation articles, and 121,377 testing articles.

The articles within the AFND dataset exhibit varying lengths, with the shortest article containing 51 characters and the longest stretching to 15,197 characters, reflecting an average article length of 232.14 characters.

To address classification tasks, we implemented distant labeling, whereby each article was labeled with the same label as the corresponding source, either credible or not credible. All pertinent information from the selected news sources was consolidated into a structured JSON file. Each online news source is represented by a key-value pair, where the key signifies the source name, and the value comprises a dictionary housing the source label, along with two lists. The first list encompasses RSS website links, while the second list contains links to local news pages. These JSON files were aggregated into a unified CSV and PKL file to facilitate machine modeling.

It's noteworthy that the AFND dataset exhibits imbalances, both in terms of the number of news websites and the distribution of articles within each class.

Table 3. AFND Dataset Statistic

Statistics	Credible	Not-Credible	Undecided
Source Count	52	51	31
Articles Count	207310	167233	232369
Average Number of Words in body text	230	217	254
Average Number of Words in headline	9	10	9

2.2. Data Preprocessing

Data preprocessing encompasses a diverse spectrum of tasks, encompassing data preparation, integration, cleaning, normalization, and data transformation. It also involves activities related to data reduction, including feature selection, instance filtering, and discretization. Furthermore, resampling methods are employed to effectively address challenges associated with imbalanced data (Luengo, J. et al., 2020).

During the learning stage, data preprocessing plays a pivotal role in conserving computational resources, both in terms of time and space. Moreover, it serves the crucial purpose of mitigating the impact of artifacts during the learning process. Through proper text preprocessing, noisy data is filtered, resulting in a logically structured representation of the data.

2.2.1. Data Preparation

The news within the AFND dataset is organized in JSON files, divided into 140 folders, each denoted as "Source_#," where "#" represents the folder number. These folders are categorized based on labels, including "credible," "not credible," and "undecided." Detailed definitions for these labels are provided in a separate file. For instance, the "Source_1" folder contains news articles categorized as "Creditable," and so forth.

Following the analysis and parsing of the JSON files, they are transformed into Comma-Separated Values (CSV) format. CSV files are a widely adopted format for storing tabular data, akin to spreadsheets. In this format, each line corresponds to a record, with multiple fields separated by commas. For instance, a record may resemble "Albert Einstein, March 14, 1879, Ulm, Federal Polytechnic School, Accomplishments in the Field of Gravitational Physics."

Subsequently, we employ the "Python Pandas Library" for loading and reading the CSV file (Zhang, A. et al., 2021). This file represents both fake and genuine news, where each row pertains to a distinct news article. The columns encompass the title of the news (Title), the news text (Text), and the publication date of each news article (Publish Date). Additionally, we introduce a new column, "iscredible," to denote the label of each news article. This label is determined based on the source file type, as outlined in the "sources.json" file, as illustrated in Figure 2.

Finally, we organize the news articles into two separate CSV files: "credibale.csv" and "not credible.csv," based on their respective labels.


```
  "source_1": "credible",
  "source_2": "undecided",
  "source_3": "credible",
  "source_4": "credible",
  "source_5": "undecided",
  "source_6": "credible",
  "source_7": "not credible",
  "source_8": "not credible",
  "source_9": "not credible",
  "source_10": "not credible",
  "source_11": "not credible",
  "source_12": "credible",
  "source_13": "not credible",
```


Figure 2. An example of the contents of the JSON file named “sources.json”.

Once the news has been converted and categorized into CSV files based on their labels, we proceed to merge the "credible" and "not credible" news articles using the

"Python Pandas Library." This step is crucial because pandas automatically replaces all CSV entries with a special "NaN" (not a number) value when it encounters missing or empty entries, such as those found in news articles with very few characters, often referred to as "meaningless news." An example of this scenario is the case of the rows with news less than 50 characters as illustrated in Figure 3.

These empty entries are commonly known as missing values and represent a significant challenge in data science, akin to dealing with "bed bugs" in a physical space. Handling missing values requires careful attention and appropriate imputation techniques to ensure the integrity and accuracy of the data.

Upon merging the data, as depicted in Figure 4, we have a total of 374,543 rows, with 207,310 belonging to the "credible" class and 167,233 to the "not credible" class.

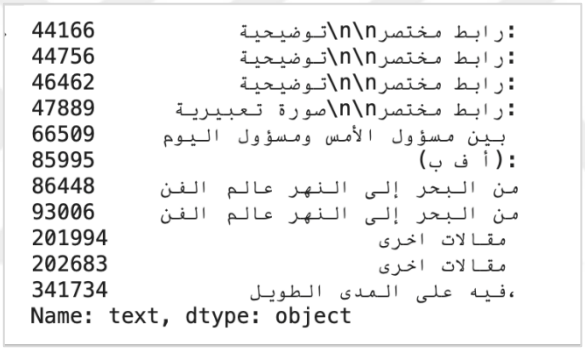


Figure 3. An example of the meaningless data in AFND article news.

Upon thorough examination of the AFND dataset, we have identified an imbalance in the distribution of news websites and articles among different classes. Specifically, the "Undecided" class exhibits a lower percentage of public websites when compared to the "credible" and "not credible" classes. Remarkably, the "Undecided" class contains the largest number of public articles, as illustrated in Figure 8.

This intriguing observation presents a compelling research opportunity to delve into the realm of natural language processing techniques. By mining the textual content of articles, researchers can explore correlations and semantic similarities that have the potential to automate the detection of public Arabic fake news.

	title	text	published date	iscredible
0	...خطط عون الجديدة: طاولة حوار لإحراج الحريري واس	...تجاوز مجلس النواب الغرق في تفسير الدستور وتأوي	2021-05-23T00:00:00	1
1	...برّي لتعطيل الاشتباك السنّي الماروني.. وجنبلاط	...تحت كل هذه العناوين، يعمل برّي على إبرام تسوية	2021-05-22T00:00:00	1
2	...انتخابات" الأسد: الوضاعة اللبنانية والمهانة ا"	...واللبنانيون أيضاً مشكلتهم أنهم يستقوون على الض	2021-05-21T00:00:00	1
3	...الأحزاب تنسف انتخابات المهندسين: تابت يبقّ الب	...أزاح نقيب المهندسين، جاد تابت، عن صدره كل أداء	2021-05-20T00:00:00	1
4	فيديو "المدن": الروح والدم في صناديق بشار الأسد	...في يوم التمديد\n\n\n\n"الفيديو من إنتاج "المدن	2021-05-20T00:00:00	1
...	...	...	...	...
374538	...اخبار المغرب اليوم ندوة تنادي بتحسين المرافعات	... دعا المشاركون في ندوة نظمت، السبت بمراكش، إلى	2021-06-13T03:47:02+02:00	0
374539	...اخبار المغرب اليوم إصدار جديد يرصد أبرز معيقات	...موازاة مع التحضير لتفعيل النموذج التنموي الجدي	2021-06-13T03:52:02+02:00	0
374540	...حصريا - المخرج أمير رمسيس لـ «المص	...اخبار اليوم حصريا - المخرج أمير رمسيس لـ «المص	2021-06-13T06:21:07+02:00	0
374541	...اخبار العالم اليوم - مقتل العشرات في العديد من	...اخبار العالم اليوم - مقتل العشرات في العديد من	2021-06-13T08:44:04+02:00	0
374542	صحافة نت	...\nقبل 1 ساعة و 5 دقيقة\n\n\nقبل 1 ساعة و 5 دقيقة	2021-06-12T00:00:00	0

374543 rows × 4 columns

Figure 4. An example of the contents of AFND article news.

2.2.2. Data Cleaning

In our study, the news articles sourced from the AFND dataset were collected from a wide array of news resources. This diversity underscores the importance of thorough cleaning and preprocessing of the text for each article. This crucial step ensures that the text is presented in a consistent and standardized format, devoid of any unnecessary noise or artifacts that could potentially impact the subsequent detection process.

During this preprocessing phase, raw data is transformed into a format suitable for the ensuing processing steps. To address various challenges encountered during the model training, such as overfitting and vanishing issues, we employed the following preprocessing techniques.

Algorithm 1. Preprocessing

Input: Arabic text (row of article).
Output: A list of cleaned data derived from the provided Arabic text.
Step 1: Tokenize the input text using the word_tokenize function from the NLTK library.
Step 2: Apply regular expressions to remove punctuation, special characters, numbers, and non-Arabic characters.
Step 3: If the word is found in the list of stop-words, skip it.
Step 4: If the length of the word is less than one character, skip it.
Step 5: Remove Arabic diacritics using the strip_diacritics function from the PyArabic library.
Step 6: Perform stemming on the word.

a. Arabic stop word removal: involves the elimination of words that lack a meaningful semantic connection to their surrounding context. These words are exceedingly common and frequently used, but they do not significantly contribute to the selection of documents that align with a user's requirements (Alajmi, A. et al, 2012). To achieve this, we initially utilized the "Arabic-Stopwords 0.4.3" library provided by (Zerrouki, T., 2020). However, it was not entirely accurate in some cases, as demonstrated in Table 4 where the word "**شهر**" was considered a stop word. In reality, "**شهر**" translates to "month" and should not belong to the Arabic stop words list. To ensure the removal of unwanted words without affecting meaningful ones, we turned to the stop word list from NLTK. The NLTK library contains 754 words in its Arabic stop word collection. Notably, frequently occurring words like "**الذي**" "**التي**," and "**حيث**" were successfully removed from the sentences.

Table 4. Comparison of the stop words removal libraries over a sample sentence

Original Text	Output with Arabic-Stopwords 0.4.3 library	Output with stopwords from nltk library
كيف عاد الفيروس بعد شهر من "صفر إصابات"؟	['فرس', 'صفر', 'صبو']	فرس', 'شهر', 'صفر', 'صبو"]

b. Stemming and Lemmatization: involve reducing variant forms of a word into a common root representation. In our experiments, we assessed three different libraries to determine the most accurate and effective one:

   i. ISRIStemmer, also known as the "Improved Stemming Algorithm," is specifically designed to reduce words to their root form or stem. This algorithm is renowned for its simplicity and efficiency, making it well-suited for applications where speed is paramount, such as information retrieval and text mining. Utilizing isri.stem(token), we obtain the Arabic root for the given token.

   ii. ArabicLightStemmer from the Tashaphyne library serves as an Arabic light stemmer and segmentor. It primarily focuses on light stemming, which involves removing prefixes and suffixes, while also providing all possible

segmentations. This approach employs a modified finite state Automaton that enables the generation of all segmentations (Zerrouki, T., 2012).

iii. Farasa (Darwish, K., et al., 2016) is a comprehensive Arabic Natural Language Processing (NLP) toolkit specially developed to tackle the challenges associated with processing Arabic text. It offers a wide array of linguistic tools and resources that facilitate various NLP tasks, including tokenization, morphological analysis, named entity recognition, and sentiment analysis. Farasa incorporates state-of-the-art techniques and linguistic resources to ensure accurate and efficient processing of Arabic text. It leverages a deep learning-based architecture and employs advanced models for tasks such as part-of-speech tagging and syntactic parsing. With its rich functionality and robust performance, Farasa serves as a valuable resource for researchers, developers, and language processing enthusiasts working with Arabic text. Its availability as an open-source toolkit encourages collaboration and further advancements in Arabic NLP research.

Table 5 shows an example executed with the above libraries; which provide the best result with Farasa library that we have applied in our experiments.

Table 5. Comparison of the Stemming/ lemmatization libraries over a sample sentence

Original Text	Output with ISRIStemmer	Output with Farasa	Output with ArListem
أسباب جوهرية تجعلك تعشق عادة يومية.. ما هي؟	['سبب', 'جوهر', 'جعل', "عشق", 'عدة', 'يوم']	['سبب', 'جوهري', 'جعل', 'عشق', 'عادة', "يومي"]	['سباب', 'جوهر', 'جعل', 'تعشق', "عاد", 'يوم']

c. Normalization using PyArabic (Zerrouki, T., 2010), a Python library designed specifically for Arabic natural language processing (NLP) tasks. This library offers a wide range of functionalities and tools for handling Arabic text, simplifying the work of developers and researchers dealing with Arabic language data. PyArabic provides robust support for various Arabic text processing tasks, including text normalization, tokenization, morphological analysis, and more. It incorporates linguistic resources like dictionaries and lexicons, enabling users to access essential

linguistic information for accurate analysis. With its modular design and user-friendly API, PyArabic can be seamlessly integrated into existing NLP pipelines. The library's comprehensive documentation and active community support ensure that users can effectively explore its features and resolve any potential issues. PyArabic stands as a valuable resource for individuals working with Arabic text, empowering them to perform diverse NLP tasks with efficiency and precision.

d. Replacing punctuation, removing special characters, numbers and non-Arabic characters.
e. Removing duplications, null rows, words with 2 or less characters and news with less than 60 words (which was 854 rows) because of the meaningless issue. The description of the remaining data after applying this step as in Figure 5.
f. Shuffling the training set using sklearn library, this step offers benefits such as reducing bias and overfitting, improving generalization, and enhancing optimization in machine learning models. It helps avoid introducing patterns, allows better exploration of the loss landscape, and ensures representative mini batches in stochastic gradient descent.

	iscredible	text_len
count	362821.000000	362821.000000
mean	0.557950	233.990136
std	0.496631	235.064200
min	0.000000	2.000000
25%	0.000000	104.000000
50%	1.000000	173.000000
75%	1.000000	281.000000
max	1.000000	17348.000000

Figure 5. The description of AFND Data

g. To process each collected article, we perform tokenization using the Tokenizer class from the Keras library. We set the parameter 'num_words' to 199,422, which corresponds to the total number of words in the corpus. Tokenization involves converting text into sequences of integers, where each word is assigned a unique integer index based on its frequency and appearance.
h. For training, validation, and test datasets, we apply post-padding to sequences of data. This process is achieved using the 'pad_sequences' function from the Keras

library. Post-padding is a common practice to ensure that all sequences have the same length, a requirement for feeding data into a neural network with a fixed input size.

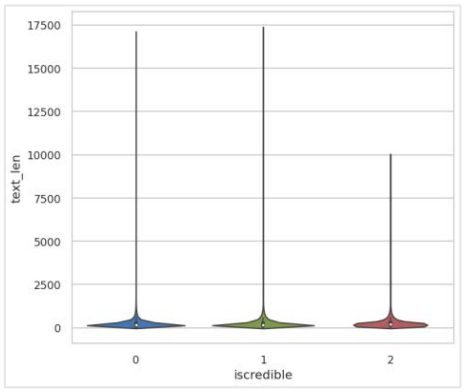


Figure 6. Distribution of texts length in the dataset AFND

To determine the ideal sequence length, we analyzed the distribution of review lengths depicted in Figures 5 and 6. Subsequently, we calculated the input size, 'maxlen,' using the following padding equation. This 'maxlen' value serves as the reference for padding all sentences to a uniform length, equivalent to the maximum length observed in the dataset.

$$\times_{\ell} \; + \; v_{n-\ell} \tag{1}$$

Where + is the concatenating operation, v is the zero vector, ℓ is the text length, and n is the required input dimension. The post padding operation essentially adds zero vectors to the ending of the word embeddings until it reaches length n, which is the maxlen parameter as illustrated in Figure 7; maxlen is changed through the experiments by 256 and 800. All sequences have been truncated or padded to have a length of maxlen value. Post-sequence truncation refers to the process of trimming all sequences at their ends to match the length of the shortest sequence or the selected 'maxlen.' If 'maxlen' exceeds the length of the smallest sequence, the smallest sequence is padded to reach the required length, as described in Dwarampudi M. et al. (2019).

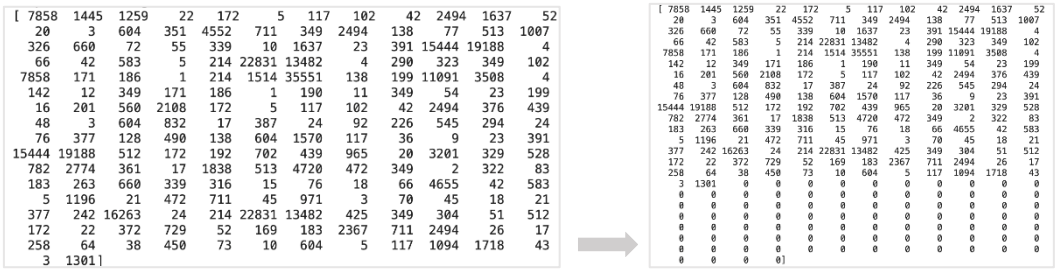


Figure 7. Post Padding Example, the zeros are added after padding.

Table 6. Samples from AFND before and after preprocessing

Before preprocessing	After preprocessing
أخبار يونيو 2, 2021 42 يمكنك مشاركة الملفات والصور بين مختلف الأجهزة تحت نظام #HarmonyOS كمثال يمكنك الاتصال بالحاسوب ومشاركة احد الصور على الجهاز الى أحد محادثاتك في الهاتف الذكي ???? #LiveSmartWithHuawei	خبر مكن شرك ملف صور خلف جهز نظم كمثال يمك تصل حسب شار احد صور جهز الى محادثاتك هتف ذكي
أقل من دقيقة أقل من دقيقة 39 التصويت بالاجماع على سحب الثقة من الأمينة العامة لحزب العمال لويزة حنون تم اليوم بفندق مزافران خلال اجتماع تصحيح لحزب العمال سحب الثقة من لويزة حنون الزعيمة الحالية لحزب العمال	دقيقة صوت جمع سحب ثقة مين عمة حزب عمل لوز حنن تم اليوم ندق زفر خلل جمع صحح حزب عمل سحب ثقة لوز حنن زعم حلي حزب عمل
اصيب ظهر اليوم الخميس، عامل (50 عاما) بجراح متوسطة خلال عمله على الة قص في شارع ابراهيم دافيد بمدينة القدس. وقامت طواقم الاسعاف التابعة لنجمة داوود الحمراء بنقل المصاب الى مستشفى هداسا عين كارم بالقدس للعلاج.	اصب ظهر اليوم خمس عمل عما جرح توسط خلل عمل الة قص شرع راهيم دافيد بمد قدس وقم طقم سعف ابع نجم دود حمراء نقل صاب الى شفى هدس كرم قدس علج
رجل يقول إنه نجا من الموت 18 مرة.. حادث سيارة خطير و17 نوعا من السرطان 05/18 12:05 يقول الأمريكي جيمس غريشام إنه نجا من حادث سيارة العام الماضي وكذلك من 17 نوبة من السرطان. تقرير من محطة WTVR التابعة لـ CNN.	رجل يقل نجا موت مرة حدث سير خطر نوع سرط يقل امر جيمس غريشام نجا حدث سير عام اضي كذل نوب سرط قرر حطة ابع
الصحة في غزة: إرتفاع عدد الشهداء إلى 109 بينهم 28 طفلاً و15 سيدة و621 إصابة بجراح مختلفة	صحة غزة رفع عدد شهداء بين طفل سيد صاب جرح خلف

2.3. Data Balancing

The length of articles in our news corpus varies, as illustrated in Figure 8. After applying the preprocessing function, the total number of articles in each class is as follows: 16,723 for 'not-credible,' 207,310 for 'credible,' and 232,340 for 'undecided.' To address this data imbalance and prevent model bias toward the majority class, we took the following steps:

a. Imbalance Ratio Definition: We established imbalance ratios for the training, validation, and test datasets. Each dataset was reduced to one-tenth (1/10) of its original size, resulting in new datasets that are only 10% the size of the originals.
b. Class-Based Split: The training, validation, and test datasets were split into two subsets based on class labels. One subset contains samples from the 'credible' class, and the other contains samples from the 'not credible' class.
c. Subsampling the 'Not Credible' Class: Within each subset (credible and not credible), we subsampled the 'not credible' class by selecting a portion of the 'not credible' samples. The size of the subsample was determined by the specified imbalance ratio (e.g., 1/10 of the original 'not credible' class size). This step was crucial for balancing the class distribution.
d. Concatenation and Shuffling: Following the subsampling of the 'not credible' class, we concatenated the 'credible' and 'not credible' samples within each dataset. This operation effectively balanced the class distribution within the datasets.

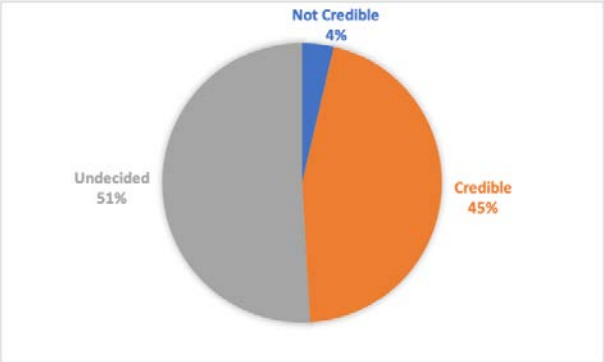


Figure 8. Percentage of articles for each class after preprocessing

2.4. Experimental Settings

The experiments were conducted using NVIDIA V100 and A100 Tensor Core GPUs with Python3 on the Google Compute Engine backend, which was equipped with a GPU. The machine boasted 52 GB of RAM. For classification tasks, distant labeling was employed, wherein each article received the same label as its corresponding source, either 'credible' or 'not credible.' The selection of the number of articles was carefully executed to balance the dataset across classes. Specifically, there were 374,542 articles for the binary classification task and 606,883 articles for the multi-class classification task.

2.5. Feature Extraction

In the realm of text classification, the utilization of embedding and feature extraction techniques assumes a pivotal role in transforming textual data into a format comprehensible to machine learning and deep learning models. In our study, we employed the following methods for embedding and feature extraction in text classification:

a. Word Embeddings: This technique represents words as dense, low-dimensional vectors within a continuous space. Such embeddings capture both semantic and syntactic relationships among words, providing a more expressive representation compared to conventional methods like the bag-of-words model (Li, S. et al., 2021).

   i. Bag-of-Words (BoW): This method, considered one of the fundamental and simplest approaches, represents each document as a vector of word counts. It focuses solely on the frequency of words within the document, disregarding their order or context. This technique is often referred to as vectorization. In our study, we employed the CountVectorizer class from scikit-learn, which includes the filtering of stopwords. This approach was applied in conjunction with the Multinomial Naive Bayes, Random Forest, XGBoost, and Logistic Regression models. The primary objective of this step is to create a document-term matrix, where each row corresponds to an article from the AFND dataset, and each column represents an individual word. The values within this matrix denote the word count for each term in each document. Subsequently, this matrix serves as input for subsequent stages in the pipeline, including TF-IDF transformation, feature selection, and classification.

ii. Term Frequency-Inverse Document Frequency (TF-IDF): TF-IDF is a numerical statistic that measures the significance of a word within a document relative to the entire collection or corpus. It plays a crucial role as a weighting factor in tasks related to information retrieval and text mining (Christian, H. et al., 2016). TF-IDF combines term frequency (indicating how often a word appears in a document) with inverse document frequency (representing the importance of a word across the entire corpus). This approach assigns higher weights to words that are more discriminative within a specific document. In our experiments, we applied TF-IDF to models such as Multinomial Naive Bayes, Random Forest, XGBoost, and Logistic Regression. Following the transformation of text data into word counts by the CountVectorizer class, the TfidfTransformer class calculates TF-IDF scores for each term within each document. The equations for TF-IDF are provided below.

$$TF = \frac{number\ of\ times\ the\ term\ appears\ in\ a\ document}{total\ number\ of\ words\ in\ the\ document} \quad (2)$$

$$IDF = \log\left(\frac{number\ of\ the\ document\ in\ the\ coprus}{number\ of\ document\ in\ the\ coprus\ contain\ the\ term}\right) \quad (3)$$

The TF-IDF of a term is calculated by multiplying TF and IDF scores:

$$TF - IDF = TF \times IDF \quad (4)$$

b. N-gram Features: An N-gram represents a contiguous sequence of 'n' words extracted from a given document or text segment (Zhang, Y. et al., 2020). N-grams offer insights into word order and local context. Typically, N-grams come in various forms, including unigrams (single words), bigrams (two-word sequences), and trigrams (three-word sequences).

c. Word2vec: Word2vec is a Word Embedding approach introduced by (Mikolov, T. et al., 2013) to address the need for more efficient neural network-based word embedding training. Word2vec offers two training modes: the Continuous Bag-of-Words Model (CBOW) and the Continuous Skip-gram Mode (Skip-gram). To

illustrate these concepts, please refer to Figure 9 (Yue, W. et al., 2020). Word2vec calculates continuous vector representations of words by considering a local window of context. In our deep learning experiments, we employed Word2vec to train the input data and obtain the necessary word vectors.

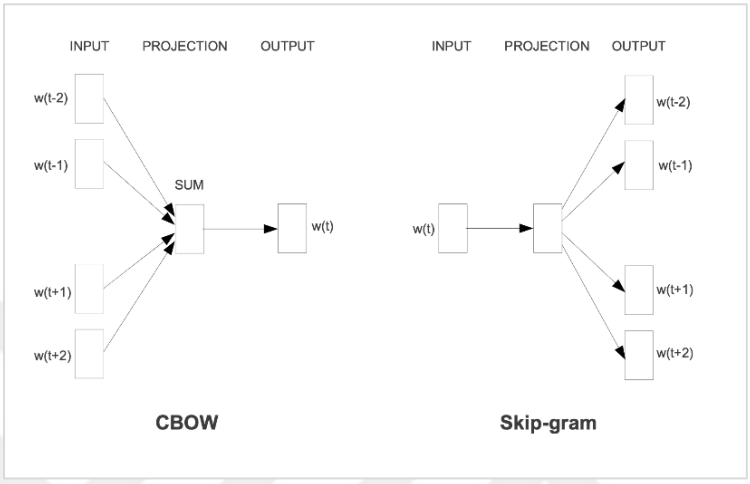


Figure 9. Word2vec structure chart (Mikolov, T. et al, 2013).

2.6. Feature Selection

Following feature extraction, we conducted feature selection to identify the most relevant features or terms. Feature selection is a widely used approach aimed at reducing data dimensionality. Its primary goal is to pinpoint a concise subset of significant features from the original feature set, guided by specific relevance evaluation criteria. This process typically offers several advantages, including enhanced learning performance (such as improved classification accuracy), reduced computational requirements, and heightened model interpretability (Tang, J. et al., 2014). This step plays a vital role in diminishing dimensionality while enhancing model efficiency and performance. Below, we outline the technique for feature selection applied in our experiments for the machine learning algorithms:

2.6.1. Filter-based Feature Selection

Filter-based feature selection proves to be an invaluable technique for dimensionality reduction, improving model efficiency, and enhancing predictive performance by focusing on the most informative aspects of the data. It is a vital tool in a data scientist's toolkit for

optimizing feature sets and streamlining the modeling process. In our experiments, we applied the SelectKBest method for feature selection. This method is available in the scikit-learn library and assesses and ranks features using a specified scoring function. For our study, we utilized the chi-squared (chi2) scoring function, as implemented in the SelectKBest class, to identify the most crucial features within our datasets (Pathan, M. S., 2022).

The SelectKBest method allows for the selection of the top 'k' most informative features from a dataset, where 'k' is a user-defined parameter. Feature selection depends on the evaluation of the scoring function, with features chosen based on their respective scores. It's worth noting that various scoring functions can be applied, with common options including chi-squared (chi2), mutual information, and the F-statistic, among others. In the case of chi-squared, it is a univariate filter that utilizes the X2 statistic as its foundation, assessing each feature independently in relation to the classes. Greater chi-squared values indicate increased relevance of a feature to a class (Anukrishna, P. R., 2017). Therefore, the formula for the Chi-square statistic (Thaseen, I. S., 2019) is defined as follows:

$$x^2(a_i\,,\,y_i) = \frac{N.(TZ - YX)^2}{(T+X)(T+Z)(X+Z)(Y+Z)} \tag{5}$$

Where T=frequency of feature a_i and class label y_j in the dataset.

X = frequency of a_i appearing without y_j.

Y = frequency of y_j appearing without a_i.

Z = frequency of neither y_j nor a_i appearing together in the dataset.

N = total number of records, i = 1...41 features j = 1, −1 (class labels).

2.7. Classification Models

Classification models play a pivotal role in text classification tasks, where the goal is to categorize text documents into predefined classes or categories. These models harness machine learning algorithms and techniques to discern patterns and relationships within textual data. They typically employ methods such as bag-of-words or word embeddings to represent text and utilize algorithms such as logistic regression, Naive Bayes, support vector machines (SVM), or deep learning models like recurrent neural networks (RNNs) and transformers for text classification. These models undergo training on labeled datasets, where each document is associated with a known class, enabling them to learn underlying patterns and subsequently make predictions on new, unseen text data.

Accurate classification of text documents enables automation of information retrieval, enhances search engine performance, facilitates content recommendation systems, and aids in various text-based decision-making processes. The versatility and effectiveness of classification models in text classification have rendered them indispensable tools in natural language processing and text analysis. In particular, the application of classification models in fake news detection holds the potential to safeguard individuals from misinformation, uphold the integrity of journalism, and contribute to a more informed and discerning society.

To ascertain the most effective approach for our specific text classification task, we conducted a comparative analysis of neural network models against traditional algorithms. Below, we provide details for each model used in our study, including its configuration.

2.7.1. Logistic Regression

Logistic regression is a supervised classification algorithm that has gained substantial importance in recent times, with its application expanding extensively. This algorithm is employed to classify individuals into categories based on a logistic function. In practical scenarios, it is common to encounter situations where a perfect linear relationship does not precisely fit all the data points. Logistic regression serves as a statistical method for assessing the significance of each independent variable in relation to the probability of a specific outcome. It proves particularly potent for modeling binomial outcomes, such as predicting whether an individual is at risk of suffering from a specific condition

(represented as 0 or 1) based on one or multiple explanatory variables. Logistic regression is well-suited for classification tasks, primarily when the outcome variable is binary (Shah, K. et al., 2020).

Logistic Regression is a discriminative model utilized to calculate P(c|x) by distinguishing among different possible values of the class y based on the provided input x. The equation that represents this process is as follows:

$$P(c|x|) = \sum_{i=1}^{N} w_i \, . f_i \tag{5}$$

Directly calculating the value of P(c|x) using the previous formula is not practical, as it can yield values ranging from -∞ to ∞, which fall outside the desired range of 0 to 1. To constrain the output within the specified range, an exponential (exp) function is applied, enabling the generation of values between 0 and 1, as demonstrated below (Indra, S. T. et al., 2016):

$$P(c|x|) = \frac{1}{z} \exp \sum_{i_i^w} f_i \tag{6}$$

We initiate the creation of a pipeline using the `Pipeline` class from scikit-learn. A pipeline is a structured approach to organizing a sequence of data processing steps, where the output of one step serves as the input to the next. This pipeline begins with raw text data as input, and we meticulously fine-tune the following components and hyperparameters for this model:

a. Vectorizer Step: In this step, we employ the `CountVectorizer` class from scikit-learn to transform text data into numerical feature vectors. The `CountVectorizer` converts the text into a matrix where each row represents a document (or text sample), and each column represents a unique word found within the entire corpus of documents. The parameters passed to `CountVectorizer` are detailed in Table 7, excluding ngrams.

b. TfidfTransformer Step: Following the conversion of text into word counts, we apply the `TfidfTransformer`. This step calculates the Term Frequency-Inverse Document Frequency (TF-IDF) representation of the word counts. TF-IDF provides

a numerical representation of the significance of each word in a document relative to the entire corpus. The parameter `norm=None` indicates that no normalization is applied to the TF-IDF values.

c. Feature Selection Step: Feature selection is performed using the `SelectKBest` class, which selects the top `k` features (words) based on a statistical test. In this instance, we utilize the `chi2` test to select the top 1,000 features. The `chi2` statistic measures the dependence between categorical variables (words, in this case) and a categorical target variable (class labels for classification).

d. Classifier Step: The final step in the pipeline involves classification, wherein we employ logistic regression. We utilize the `LogisticRegression` class from scikit-learn, specifying the hyperparameters `solver='liblinear'` and `random_state=0`.

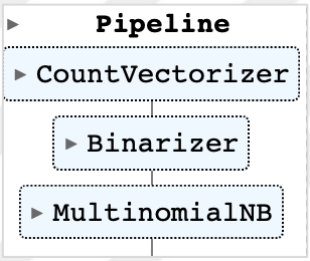


Figure 10. Multinomial Naive Bayes model Architecture

Additionally, two more pipelines have been employed, each utilizing Logistic Regression (LR) and other machine learning (ML) algorithms mentioned in the subsequent subsections, including Linear Support Vector Machine (LSVM), Naïve Bayes (NB), and XGBoost. The structural overview of these models is elucidated in Figure 10.

In all these pipelines, the initial step involves a vectorizer, which is responsible for converting text data into numerical features. Specifically, the CountVectorizer is used with specific settings outlined in Table 7. Following vectorization, a normalization step is incorporated using the Binarizer. This step transforms the numerical features into binary values (0s and 1s) utilizing the default threshold, whereby news labeled as not credible are scaled to 0, and credible news to 1, regardless of their frequency.

Subsequently, classification is carried out by the respective classifiers: Logistic Regression with the solver parameter set to 'liblinear,' Multinomial Naïve Bayes, Linear Support Vector Classifier (Linear SVC), and XGBoost. In one of these models, the

SelectKBest Feature Selector with the chi-squared (chi2) statistical test as the scoring function is applied to select the top 100 features based on their chi-squared scores. These selected features are stored in the variable 'filtered.'

These pipelines offer a streamlined approach to sequentially apply the necessary steps during both the training and prediction processes.

Table 7. The parameters of the Count Vectorizer function

Parameter	Value
analyzer	word
tokenizer	Tokenizer (custom function)
preprocessor	Preprocess function in Algorithm 1
stop_words	Arabic stopwords_list
ngram_range	(1,3)

2.7.2. Naïve Bayes

The Naive Bayes classifier stands out as both the simplest and one of the most extensively employed classification methods. This classifier constructs a probabilistic model to represent the distribution of documents within each class. It operates under the assumption of term independence, meaning that the distribution of various terms is considered unrelated to one another. Despite acknowledging the inherent oversimplification of this "naive Bayes" assumption in numerous real-world scenarios, the classifier demonstrates a remarkable level of effectiveness (Allahyari, M. et al., 2017).

There are several variations of Naïve Bayes, with the most common one being Multinomial Naïve Bayes. Multinomial Naïve Bayes is typically used for text classification tasks and is particularly suitable for discrete data, such as word counts. Equation 7 illustrates the Multinomial Naive Bayes (MNB) model, which is one of the parametric models frequently applied in text classification:

$$P(c|d|) = \frac{P(c)\Pi_{i=1}^{n} P(w_i|c|)^{f_i}}{p(d)} \quad (7)$$

Where 'f_i' represents the frequency of word 'w_i' occurring in a document 'd,' while 'P(w_i|c)' denotes the conditional probability of word 'w_i' appearing in document 'd' given

the class 'c.' The variable 'n' stands for the total number of unique words within document 'd.' Additionally, 'P(c)' signifies the prior probability of a document with class label 'c' occurring within the collection of documents (Su, J. et al, 2011).

2.7.3. Support Vector Machine (SVM)

The Support Vector Machine (SVM), developed by Vapnik and Chervonenkis, is a kernel-based machine learning model primarily designed for classification tasks, rather than regression (Kowsari, K. et al., 2019). It employs a straightforward mathematical model represented as $y = wx' + \gamma$ and adapts it to facilitate the division of the input domain into linear regions. SVMs come in two main varieties: linear and nonlinear models, as illustrated in Figure 11.

When the data domain can be separated into linear regions, such as a straight line or a hyperplane, to distinguish between different classes in the original feature space, it is termed a "Linear Support Vector Machine." Conversely, when the data domain cannot be linearly separated in its original space but can be transformed into a different space called the "feature space," where linear separation becomes achievable, it is referred to as a "Nonlinear Support Vector Machine." In the feature space, the SVM becomes capable of effectively classifying the data into distinct categories (Suthaharan, S. et al., 2016).

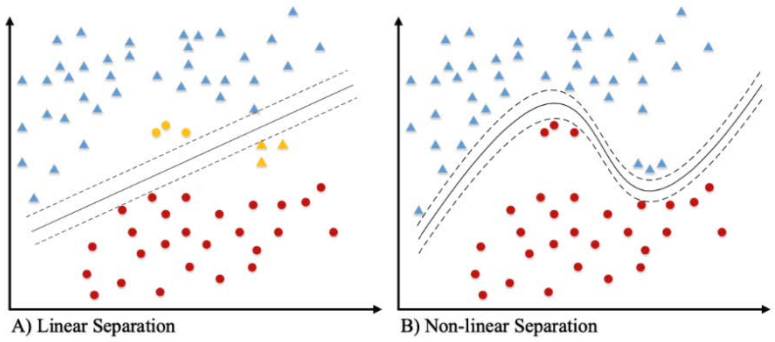


Figure 11. The linear and non-linear Support Vector Machine (SVM) for a 2D data set. The red is class 1, the blue color is class 2 and yellow color is miss-classified data points (Suthaharan, S. et al, 2016).

2.7.4. X-Gradient Boost (XGBoost)

The XGBoost classifier, proposed by Chen, T. in 2016, represents a scalable machine learning system for tree boosting and is widely acclaimed for its superior performance. It operates as an ensemble method primarily based on decision trees and is specifically engineered for exceptional scalability.

Much like traditional gradient boosting, XGBoost constructs an iterative expansion of the objective function by minimizing a loss function. However, XGBoost distinguishes itself by exclusively employing decision trees as its base classifiers. Furthermore, it utilizes a modified variation of the loss function to govern the complexity of these trees (Bentéjac, C. et al., 2021).

$$L_{xgb} = \sum_{i=1}^{N} L\big(y_i,\, F(x_i)\big) \;+\; \sum_{m=1}^{M} \Omega(h_m) \tag{8}$$

where

$$\Omega(h) \;=\; \gamma T \;+\; \frac{1}{2}\lambda\big|\big|w\big|\big|^2$$

In Equation 8, 'L' represents a differentiable convex loss function that quantifies the disparity between the prediction $F(x_i)$ and the target y_i. The term Ω pertains to the model's complexity, specifically the complexity of the regression tree functions. In this context, 'T' denotes the number of leaves within the tree, and 'w' signifies the output scores associated with these leaves. The role of the loss function is pivotal in this equation. It can be seamlessly integrated into the decision tree's split criterion, effectively leading to a proactive pruning strategy.

Notably, higher values of the parameter 'γ' yield simpler trees, granting precise control over the minimum loss reduction required to justify splitting an internal node. Additionally, 'lambda' represents L2 regularization. Therefore, as elucidated by the authors (Chen, T., 2016), the tree ensemble model in Equation 8 cannot be optimized using traditional optimization methods in Euclidean space. XGBoost employs a Second-Order Taylor Approximation for the summation of squared errors, as delineated in Equation 9, to rapidly optimize the objective in the general setting.

(9)

$$L_{xgb} = \sum_{i=1}^{N} \left[g_i \, F_t(x_i) \; + \frac{1}{2} h_i f_t^2(x_i) \right] \Omega(f_t)$$

In addition to its core functionality, XGBoost introduces an additional regularization hyper-parameter known as "shrinkage." This hyper-parameter serves to reduce the step size within the additive expansion process, contributing to the model's stability and convergence. Moreover, XGBoost provides mechanisms to control the complexity of the trees, such as constraining tree depth. This complexity reduction not only enhances model interpretability but also accelerates training and reduces storage requirements.

To address overfitting and expedite training, XGBoost incorporates randomization techniques. These techniques include random subsampling for training individual trees and column subsampling, both at the tree and tree node levels. Furthermore, XGBoost offers the flexibility to extend its functionality to accommodate user-defined loss functions. This can be achieved by defining a function capable of producing both the gradient and the Hessian (second-order gradient) and subsequently passing it through the "objective" hyper-parameter (Bentéjac, C. et al., 2021).

The XGBoost model is initialized using the XGBClassifier class, specifically designed for gradient boosting tasks. During initialization, several critical hyperparameters are configured to fine-tune the XGBoost model's behavior.

2.7.5. Long Short-Term Memory (LSTM)

Recurrent neural networks (RNNs) are powerful models employed for various tasks, including natural language understanding, language generation, and video processing. These models operate by processing a sequence of symbols as input and applying a simple neural network (RNN unit) to each symbol in the sequence, taking into account the network's output from the previous time step.

However, a primary challenge associated with RNNs is their tendency to overfit quickly, especially when dealing with small datasets. This susceptibility to overfitting arises because RNN models lack built-in regularization, making it challenging to handle noise and variability commonly found in small datasets. Researchers often employ strategies like early stopping or opt for smaller and less complex RNN models to mitigate

the risk of overfitting, thereby enhancing model accuracy and reliability (Gal, Y. et al., 2016).

Over the years, various RNN architectures have been developed to address the limitations of the standard RNN architecture. One notable example is LSTM, which stands for Long Short-Term Memory. LSTM is an enhanced neural network architecture derived from recurrent neural networks (RNNs). LSTMs excel in tasks involving sequential data because they possess the ability to selectively retain or forget information over time. This adaptability makes them highly effective for analyzing the temporal structure of news articles and capturing linguistic patterns indicative of fake news, as illustrated in Figure 12.

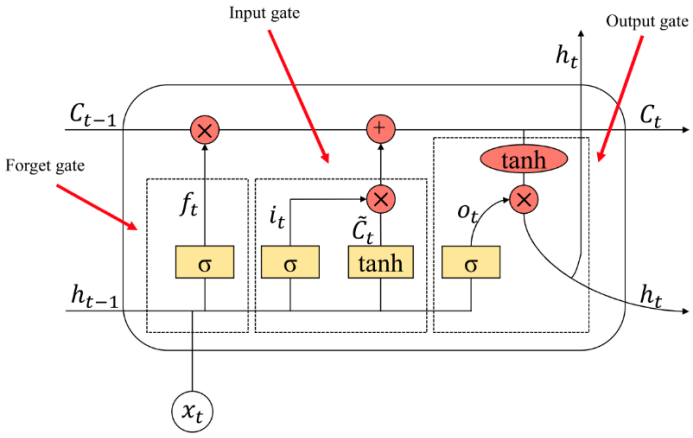


Figure 12. Structure of LSTM (Jin, N. et al, 2020)

LSTM introduces memory cells and three control gates to effectively preserve historical information within long sequences. This innovation addresses challenges such as historical information loss and gradient vanishing that can occur during the training of deep RNNs with multiple layers (Zhang, J. et al., 2018). The three gates are the forget gate (f_t), input gate (i_t), and output gate (o_t), below are details of each gate (Jin, N. et al, 2020):

a. The forget gate determines which information should be discarded from the cell state, the calculation is shown in equation 10.

$$f_t = \sigma\left(W_f.[h_{t-1}x_t] + b_f\right) \quad (10)$$

b. The input gate decides what information to incorporate into the cell state. Once these decisions are made, the cell state is updated accordingly. function of this gate as in equation 11.

$$i_t = \sigma(W_i.[h_{t-1}x_t] + b_i)$$
$$C_t = \tanh(W_c.[h_{t-1}x_t] + b_c) \quad (11)$$

c. The output gate determines the ultimate output of the network, it combines the current input and cell state. The calculations are shown in equation 12.

$$o_t = \sigma(W_o[h_{t-1}x_t] + b_o)$$
$$h_t = o_t \cdot \tanh(C_t) \quad (12)$$

In our experimental model, we implemented the LSTM architecture with different input data, as depicted in Figure 13. The model comprises a sequential structure with an embedding layer serving as the initial layer. This embedding layer maps each word in the input sequence to a dense vector representation of size 800. To mitigate overfitting, a dropout layer is applied after the embedding layer (Kowsher, M., 2021), as explained in the subsequent subsection.

The output shape of this layer remains consistent with the input shape, which is (None, 800, 800). Following the embedding layer, we introduced an LSTM layer with 100 units. This LSTM layer is specifically designed to process sequential data, and it produces an output with a shape of (None, 100). To further address overfitting concerns, we applied another dropout layer after the LSTM layer, maintaining the output shape at (None, 100).

Finally, a dense layer with a single unit is added to the model. This layer alters the output shape to (None, 1) and is responsible for binary classification, as evidenced by the presence of a single output unit.

Layers in the deep learning model can be considered as the architecture of the model. There can be various types of layers that can be used in the models. All of these different layers have their own importance based on their features. Here is an explanation on the additional layers we applied in our experiments.

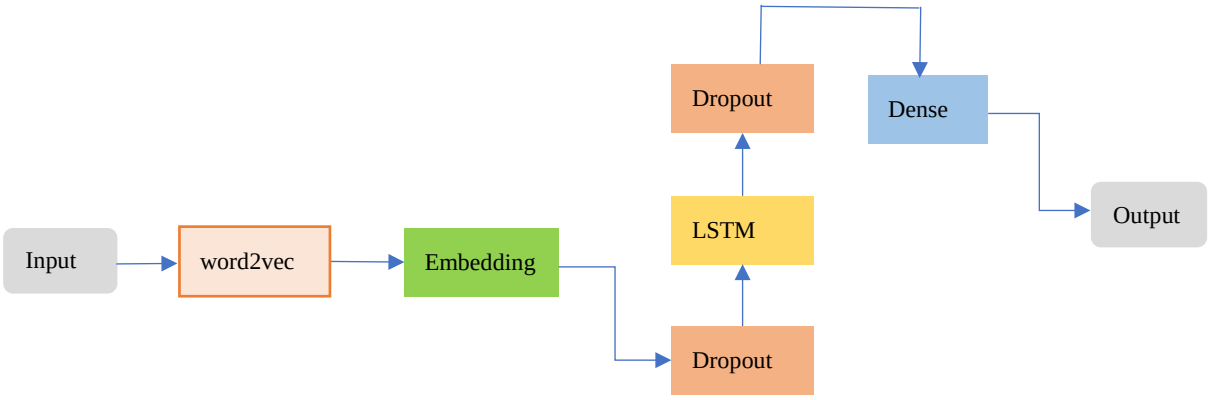


Figure 13. The block diagram of the LSTM/ BiLSTM experiments

- Embedding Layer

At the outset of our deep learning models, we introduced the embedding layer, a critical component in neural networks. Positioned at the neural network's input, this layer is trained in a supervised manner using the Backpropagation algorithm. Its primary role is to establish associations between one-hot encoded words and their corresponding word vectors, a process tailored for the specific task of fake news detection (Bogale Gereme, F. et al, 2020).

The embedding layer's function is to generate word embeddings by mapping the unique words present in the text data to dense vectors of a predefined length. The size of the vocabulary determines the number of unique words that the embedding layer can accommodate. Additionally, the input length parameter specifies the expected input sequence lengths for this layer, which we adjusted as part of our experimentation.

- Dropout Layer

Deep neural networks have emerged as potent machine learning systems, capable of tackling intricate problems. Nevertheless, these networks are susceptible to overfitting, a phenomenon that can significantly impair their performance. Overfitting arises when a network becomes overly attuned to the training data, resulting in poor generalization to new, unseen data. This issue is particularly problematic in large networks, which possess numerous parameters and tend to operate slowly.

To combat overfitting, a technique known as dropout was introduced by (Srivastava et al., 2014). Dropout entails the random omission of units (along with their connections) from the neural network during training, as depicted in Figure 14. This strategy prevents units from excessively co-adapting and encourages the network to learn more robust

features. During training, dropout effectively samples from an exponential number of different "thinned" networks, mitigating overfitting. At test time, the impact of averaging the predictions from these thinned networks can be approximated by utilizing a single unthinned network with reduced weights. This approach significantly alleviates overfitting and yields substantial improvements over other regularization methods.

Numerous studies have demonstrated the advantages of dropout, showcasing its ability to enhance the performance of deep neural networks across various tasks. For instance, (Srivastava et al., 2014) reported that dropout boosted the accuracy of a network in a speech recognition task by over 10%. In essence, dropout stands as a potent technique for mitigating overfitting in deep neural networks. By randomly dropping units during training, dropout encourages the network to learn more resilient features and guards against excessive specialization to the training data. This can result in noteworthy performance enhancements and has proven effective across diverse tasks.

In the word-based model, dropout entails the random removal of word types from the sentence. This can be interpreted as a means to prevent the model from excessively relying on individual words for its task (Gal, Y. et al, 2016).

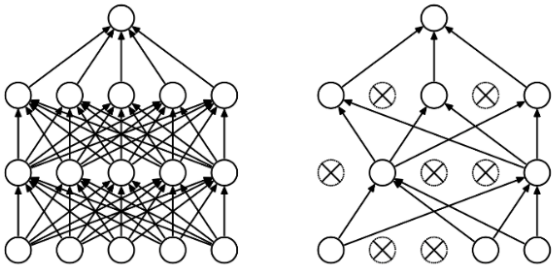


Figure 14. On the left, we have a standard neural network with 2 hidden layers. On the right, we present an example of a simplified network achieved by implementing dropout in the original network. The crossed-out units indicate the neurons that have been deactivated through dropout (Srivastava et al., 2014).

We incorporated dropout layers both before and between LSTM model layers. In Figure 15, dropout layers are represented by black circles, denoting the neurons selected for deactivation. The red and green LSTM layers are depicted by horizontal arrows, indicating their bidirectional functionality. Within this illustration, two distinct approaches are highlighted.

- Initial Approach: This approach involves applying dropout to the same set of LSTM inputs for both directions.

- Alternative Approach: Alternatively, you can selectively choose distinct inputs to drop for the left-to-right LSTM and the right-to-left LSTM (Bluche, T. et al, 2015).

The application scenarios for dropout are as follows:

a. Before the input values of LSTM.
b. Within the recurrence loop on the output values of LSTM.
c. After all the recurrence iterations have been completed on the output values of LSTM without modifying the internal representation of the LSTM layer.

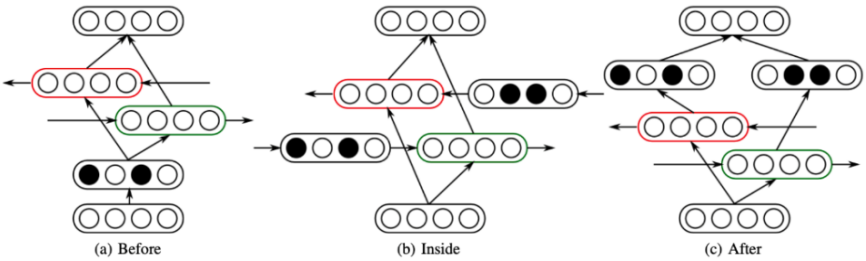


Figure 15. Dropout Place in RNNs with respect to the LSTM layer (Bluche, T. et al, 2015).

In the results of experiments conducted by Bluche, T. et al. (2015), they demonstrated that the relative placement of dropout can have a significant impact on performance. Merely applying dropout either after LSTM layers or between every layer may not consistently yield optimal results. When considering RNNs in isolation, it appears favorable to apply dropout before recurrent connections. However, within the context of complete recognition systems, it becomes evident that applying dropout after the final LSTM layer is crucial for achieving improvements in Word Error Rate (WER) without significantly compromising the standalone performance of the RNN. It is evident that dropout is most effective when positioned in proximity to the network's inputs and outputs.

- Dense Layer

The dense layer is a widely used component in neural networks, known for its interconnected nodes. It operates as a hidden layer connected to every node in the subsequent layer, performing the following operations.

$$\text{Output} = \text{activation}(\text{dot}(\text{input}, \text{Kernal}) + \text{bias}) \quad (13)$$

Hidden dense layers play a crucial role in neural networks. In a study conducted by Josephine, V. H. et al. (2021), the authors investigated the impact of hidden dense layers

within Convolutional Neural Networks to improve the performance of a classification model. They employed three different deep network models for classification. The results showed that as the model's depth increased, with more dense layers containing an increased number of neurons in each hidden layer, the number of hidden dense layers decreased. Furthermore, enhanced classification performance was observed when the network had more hidden layers and a deeper structure.

2.7.6. Bidirectional LSTM (BiLSTM)

Utilizing the LSTM model enhances its capacity to capture extended dependencies within sequences. This improvement is attributed to LSTM's ability to discern relevant information while filtering out extraneous details during training. However, an inherent limitation arises when using LSTM to model sentences, as it lacks the capability to encode information bidirectionally, meaning it cannot effectively incorporate context from both preceding and succeeding elements of a sequence. In contrast, the BiLSTM model provides a more effective solution for capturing bidirectional semantic dependencies, enabling it to consider context from both directions within a sequence, thereby enhancing its ability to comprehend the temporal dynamics of the data (Yue, W. et al., 2020).

The BiLSTM model commences with an embedding layer, mapping each word in the input sequence to a dense vector representation of size 800. Following the embedding layer, a dropout layer is applied, maintaining the same output shape as the input (None, 800, 800). A bidirectional layer is then added, enveloping the underlying LSTM layer and processing the input sequence in both forward and backward directions, resulting in an output shape of (None, 200). Another dropout layer is subsequently applied, keeping the output shape unchanged (None, 200). Finally, a dense layer with a single unit is introduced, with the output shape becoming (None, 1). This layer serves the purpose of binary classification, as indicated by the single output unit.

It's worth noting that we observed a decrease in performance when altering the input embedding layer.

The model structure, depicted as shown in Figure 13, with the substitution of the LSTM block for a BiLSTM block, comprises a total of 157,841,001 trainable parameters for the balanced data experiment and 24,614,001 for the imbalanced one. Notably, all of these parameters are trainable, and there are no non-trainable parameters in this model..

2.7.7. CNN-BiLSTM + XGBoost

In this study, the combination of CNN-BiLSTM and XGBoost has been employed alongside three distinct sequential neural network models. Sequential models represent a prevalent deep learning architecture, frequently used for tasks such as classification, where layers are arranged sequentially, with each layer's output serving as the input for the subsequent layer. Let's now examine each component within the entire model depicted in Figure 16:

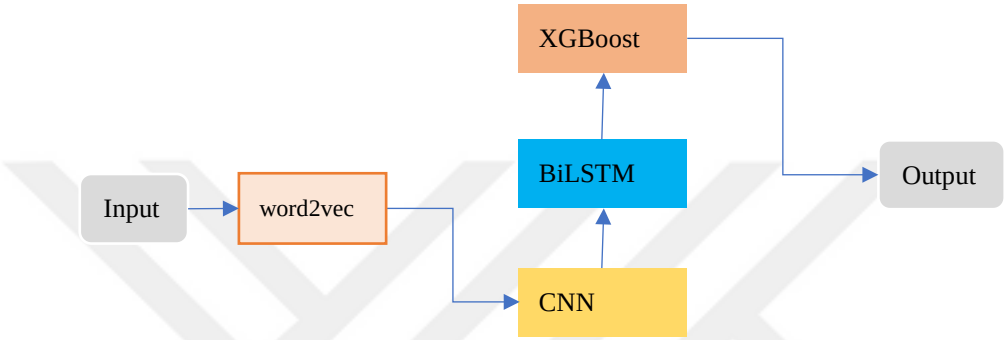


Figure 16. The block diagram of CNN-BiLSTM + XGBoost experiment

a. CNN Block: The first block commences with a sequential linear layer, followed by three 1D Convolutional Layers (`Conv1D`). These layers are designed to capture local patterns and features within the input data. The first `Conv1D` layer is configured with 64 filters, a kernel size of 3, and employs the ReLU activation function. It takes an input shape of (100, 1). After each `Conv1D` layer, there is a MaxPooling1D layer to down-sample the spatial dimensions. The last `Conv1D` layer is succeeded by a GlobalMaxPooling1D layer, which selects the most significant features globally during computation.
b. BiLSTM Block: In this block, we start with a linear stack of layers, followed by Bidirectional LSTM layers (`Bidirectional (LSTM) `). These Bidirectional LSTM layers process input data in both forward and backward directions, thereby capturing long-range dependencies in the data. After each BiLSTM layer, Dropout layers with a dropout rate of 0.2 are added to prevent overfitting. It's important to note that this model is defined but not separately summarized, as it is integrated into the final model.
c. XGBoost Model: In this step, we define an XGBoost classifier with specific hyperparameters such as the number of estimators, maximum depth, learning rate,

and subsample ratio. Finally, we combine these models into the final model, which includes the `cnn_model`. Following the `cnn_model`, there is a Dense layer with a single output unit utilizing the sigmoid activation function, a common choice for binary classification tasks. This composite model is employed for binary classification, harnessing the feature extraction capabilities of the CNN and BiLSTM models while leveraging the predictive power of XGBoost.

2.7.8. CNN-BiLSTM

In this experimental model, we applied the same structural configuration to different input data, both balanced and imbalanced. The CNN-BiLSTM model, illustrated in Figure 17, begins with an embedding layer. This layer takes an input sequence and maps each word to a dense vector representation, the size of which can vary depending on our experiments. Following the embedding layer, a dropout layer is applied, maintaining the output shape as (None, 800, 800).

Subsequently, a one-dimensional convolutional layer is introduced with 32 filters and a kernel size of 5. This layer operates on the input sequence, employing the convolution operation to capture local patterns, resulting in an output shape of (None, 796, 32). After the first convolutional layer, a max pooling layer is applied, performing down-sampling by selecting the maximum value within each window of size 2, thus altering the output shape to (None, 398, 32).

Another convolutional layer is added with 32 filters and a kernel size of 5, resulting in an output shape of (None, 394, 32). Following this, another max pooling layer is applied, leading to an output shape of (None, 197, 32).

A bidirectional layer is subsequently introduced, wrapping the underlying LSTM layer and processing the input sequence in both forward and backward directions. This step transforms the output shape to (None, 200). Another dropout layer is applied after the bidirectional layer, maintaining the output shape as (None, 200).

Finally, a dense layer with a single unit is added, resulting in an output shape of (None, 1). This particular layer is responsible for binary classification, as indicated by the presence of a single output unit.

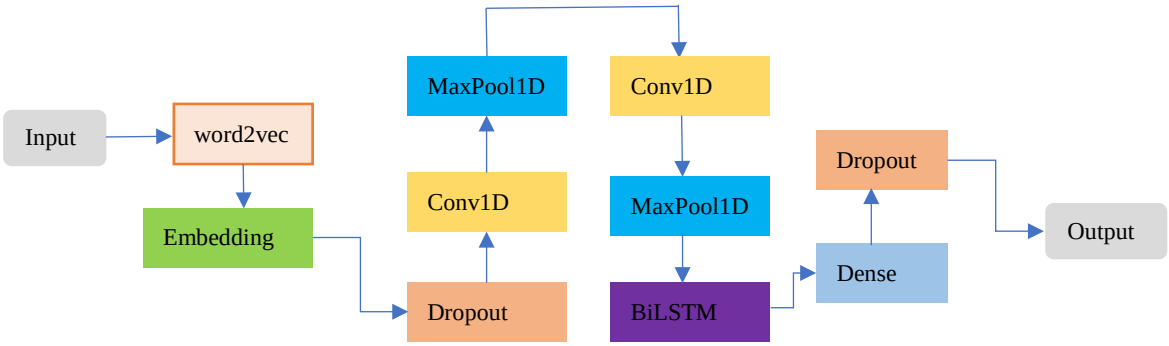


Figure 17. The block diagram of CNN-BiLSTM experiment

2.7.9. Model Evaluation

Model evaluation is a critical process in which the performance and effectiveness of a trained model are assessed through the application of diverse metrics and techniques. The primary objective of model evaluation is to gauge how effectively a model generalizes to novel, unseen data and whether it meets the specific performance criteria established for the given task.

In the realm of machine learning, once a model has been trained on a designated dataset, it becomes imperative to gauge its potential performance on real-world data. Model evaluation serves as a means to comprehend the model's strengths and weaknesses, as well as its capacity to provide accurate predictions or classifications. This evaluation process encompasses the utilization of various evaluation metrics and techniques, including:

a. Accuracy: This metric represents the percentage of correctly classified test set instances by the classifier.

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \quad (14)$$

b. Precision: This metric calculates the ratio of predicted positive examples that are genuinely positive.

$$\text{Pr}\, e\, cision = \frac{TP}{TP + FP} \quad (15)$$

c. Recall or sensitivity: This metric quantifies the classifier's ability to identify positive examples.

$$Recall = \frac{TP}{TP + FN} \quad (16)$$

d. F1-Score: This metric combines precision and recall into a single measure.

$$F = \frac{2 \times precision \times Recall}{precision + Recall} \quad (17)$$

e. Cohens kappa Score: Cohen's Kappa is a coefficient used to measure inter-judge agreement for nominal scales, as presented by Cohen, J. in 1960. The formula for computing Cohen's Kappa is expressed in Equation 18.

$$k = \frac{p_o - p_e}{1 - p_e} \quad (18)$$

Where p_o represents percent agreement, defined as the proportion of subjects on which the raters agree, and p_e signifies chance agreement, defined as the proportion of agreement that would be expected by chance. The maximum value of k is 1.00, achievable only when both raters achieve perfect agreement, indicating that their marginal distributions are exactly identical. Conversely, the minimum value of k falls within the range of 0 to -1.00, and this minimum value is influenced by the extent to which the raters' marginal distributions vary or disperse (Sun, S., 2011).

f. Confusion Matrix: This is a matrix that displays the counts of true positive, true negative, false positive, and false negative predictions, offering a comprehensive overview of the model's performance. Figure 18 illustrates the output of the confusion matrix in our experiment.

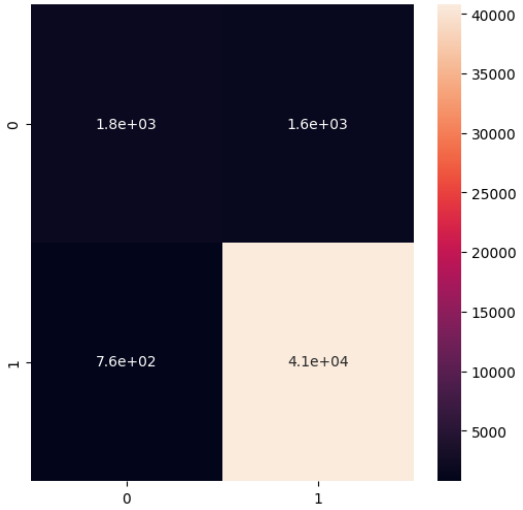


Figure 18. Confusion Matrix of NB Model

g. Cross-Validation: It is a technique in which the dataset is divided into multiple subsets for the iterative training and testing of a model. Its purpose is to evaluate the model's performance across different data subsets while mitigating the risk of overfitting. The fundamental concept of cross-validation is to simulate the process of training and testing a model on various data subsets, thereby reducing the chance of obtaining overly optimistic or pessimistic performance results due to random data variations. One of the most prevalent forms of cross-validation is k-fold cross-validation, wherein the dataset is partitioned into 'k' subsets, often referred to as folds. In our machine learning experiments (LG and MNB), we utilized Stratified K-Fold cross-validation with 5 splits. This approach helps ensure a robust evaluation of the model's performance.

h. Data Splitting: Typically, datasets are divided into two or more subsets: one for training the model and one or more for evaluating its performance. In our deep learning (DL) experiments, we employed the Train/Validation/Test Split approach to partition a dataset into three distinct subsets: a training set, a validation set, and a test set. This division serves the purpose of assessing and optimizing the performance of machine learning models by enabling independent evaluation on unseen data.

The validation set consists of a series of examples used to fine-tune the model's parameters. Meanwhile, a set of examples reserved exclusively for assessing the performance of a fully specified model is known as the test set.

The initial split involves creating a combined training and validation set by randomly selecting 64%, 70%, or 60% of the data. The remaining data is then allocated to the test set by excluding the indices of samples present in the training and validation sets.

Subsequently, the training and validation set is further subdivided into two subsets: a new training set and a validation set. Similar to the previous split, this is achieved by randomly sampling a portion of the data from the combined training and validation set.

To ensure consistent sequential indexing and prevent any inconsistencies, the indices of the training, validation, and test sets are reset using the `reset_index(drop=True) ` function. This operation resets the index of each dataset without adding a new column for the index data, which is why we specify the `drop=True` parameter.

3. EXPERIMENTS

Within the context of my research on the FND task, we conducted two experiments using the AFND dataset. These experiments were organized into two distinct phases.

In the initial phase, we employed well-established ML algorithms for classification, employing different feature extraction methods.

The second phase of our experiments focused on the utilization of DL models for classification. Subsequently, we compared the results obtained from these deep learning models with those achieved by the machine learning algorithms employed in the initial phase.

In the subsections that follow, we provide a summary of these experiments. These experiments involve the evaluation of several text classification techniques using a variety of text representation methods, feature selection techniques, and performance metrics. The performance metrics utilized to assess these models encompass accuracy, precision, recall, F1-score, and Cohen's Kappa score.

3.1. Experiments of Logistic Regression

Logistic Regression is utilized in this study for comparison purposes with the results obtained in the study conducted by Wang, W. Y. in 2017. Table 8 below provides a summary of the results.

Table 8. Results from the Logistic Regression experiment using various text representation and feature selection methods

Text Representation	**Feature Selection (FS)**	**Performance Metrics**				
		Accuracy	**Precision**	**Recall**	**F1-score**	**Kappa score**
TF-IDF	Without FS	0.9271	0.93	0.99	0.96	0.18
	SelectKBest	0.9305	0.93	1.00	0.96	0.12
Binarizer	SelectKBest	0.9309	0.93	**1.00**	**0.96**	0.16
Binarizer (ngram 1-3)		**0.9404**	**0.93**	0.94	0.93	**0.50**

From the table above, it is evident that all three approaches delivered impressive accuracy, signifying their effectiveness in text classification. Furthermore, these models consistently demonstrated high precision and recall, indicating their proficiency in avoiding false positives while capturing true positives. Notably, the third approach, which involved the utilization of an n-gram range of 1-3 and Binarizer normalization, achieved the highest accuracy. A significant observation is the Cohen's Kappa score, which stands at 0.50, indicating substantial agreement between the model's predictions and the actual labels. This outcome suggests that this particular approach strikes an optimal balance between precision and recall, rendering it especially promising for text classification tasks.

3.2. Experiments of Naïve Bayes

In this subsection, Table 9 showcases the performance of various text classification models using different combinations of text representation, feature selection techniques, and balanced data. Let's delve into these results:

a. The first approach: Utilizing a simple Binarizer in conjunction with SelectKBest yielded a notably high accuracy of 0.92, signifying overall correctness. Precision was also impressive at 0.93, suggesting a robust ability to avoid false positives. However, the exceptionally high recall of 0.99 raises the possibility of overprediction in the positive class. This is mirrored in the F1-Score, which stands at 0.96, slightly lower than precision, indicating a minor imbalance between precision and recall.
b. The second approach: This approach, which incorporates an n-gram range of 1-3 alongside the Binarizer and SelectKBest, exhibits a lower accuracy of 0.86, hinting at a decrease in overall correctness. Nevertheless, precision remains high at 0.95, underlining a strong capacity to minimize false positives. Recall (0.90) and the F1-Score (0.92) are slightly lower, indicating a trade-off between precision and recall, likely attributed to the introduction of character n-grams.

Based on the values presented in the table below, it is evident that the first approach outperforms in terms of overall correctness (accuracy) and recall, whereas the second approach consistently maintains a higher precision. Additionally, it can be concluded that the model employing the SelectKBest feature selection technique achieved the highest accuracy.

Table 9. Results of Naïve Bayes experiment with different text representation and feature selection methods

Text Representation	**Feature Selection**	**Performance Metrics**				
		Accuracy	**Precision**	**Recall**	**F1-score**	**Kappa score**
Binarizer	SelectKBest	**0.9243**	0.93	**0.99**	**0.96**	0.14
Binarizer (ngram 1-3)		0.8613	**0.95**	0.89	0.92	0.24

3.3. Experiments of XGBoost

Table 10 below provides an overview of the results obtained with the XGBoost algorithm across various evaluation metrics, considering different text representation and feature selection techniques. In this study, the following hyperparameters were fine-tuned for XGBoost:

a. 'max_depth’: This parameter governs the maximum depth of individual decision trees, influencing their complexity and serving as a means to mitigate overfitting.
b. 'n_estimators': Determining the number of decision trees to be assembled in the ensemble, this parameter enables the model to learn progressively from multiple weak learners.
c. 'learning_rate': This parameter defines the step size taken during each iteration while constructing the ensemble. It significantly impacts the convergence speed and generalization capability of the model.

Table 10. Results of XGBoost experiment with different text representation and feature selection methods

Text Representation	**Feature Selection**	**Performance Metrics**				
		Accuracy	**Precision**	**Recall**	**F1-score**	**Kappa score**
Binarizer	SelectKBest	0.9331	0.93	1.00	0.96	0.14
Binarizer (ngram 1-3)		**0.9401**	**0.94**	**1.00**	**0.97**	**0.16**

Both approaches exhibit strong performance, but the second approach stands out by achieving perfect recall and slightly higher accuracy. Here's an interpretation of the results:

a. The first approach: This approach, which incorporates a Binarizer with SelectKBest, performs admirably with a high accuracy of 0.93, indicative of robust overall correctness. Precision is also commendably high at 0.93, implying a noteworthy capacity to minimize false positives. Importantly, recall is perfect, resulting in a high F1-Score of 0.96, which signifies a well-balanced trade-off between precision and recall.
b. The second approach: In this case, the use of an n-grams range of 1-3 alongside the Binarizer and SelectKBest yields an even higher accuracy of 0.94. Precision remains impressively high at 0.94, underlining its effectiveness in preventing false positives. Notably, recall achieves perfection at 1.00, indicating that it captures all positive instances. Consequently, the F1-Score reaches an impressive 0.97, reflecting a robust balance between precision and recall.

3.4. Experiments of SVM

The table below presents performance metrics for two distinct approaches, each employing different text representation and feature selection techniques. Notably, the second approach (utilizing the Binarizer with character n-grams and SelectKBest) consistently outperforms the first approach (employing the Binarizer with SelectKBest) across the majority of metrics. The second approach attains higher accuracy, precision, and recall, highlighting its efficacy in correctly classifying instances and capturing positive cases. Interestingly, the F1-Scores for both approaches exhibit proximity, suggesting a comparable balance between precision and recall.

Table 11. Results of SVM experiment with different text representation and feature selection methods

Text Representation	**Feature Selection**	**Performance Metrics**				
		Accuracy	**Precision**	**Recall**	**F1-score**	**Kappa score**
Binarizer	SelectKBest	0.8001	**0.82**	0.84	0.83	0.12
Binarizer (ngram 1-3)		**0.8131**	0.81	**0.85**	**0.84**	**0.12**

3.5. Experiments of LSTM

In classification tasks, a primary objective is often to minimize the error rate, which represents the proportion of instances where our predictions deviate from the ground truth. To optimize our deep learning (DL) model's parameters, we minimize the loss calculated on the AFND dataset, which encompasses numerous training examples. However, achieving strong performance on the training data doesn't guarantee similar performance on new, unseen data (Zhang, A. et al, 2021). Therefore, we divide the available data into three segments: the training dataset (or training set), utilized for learning model parameters; the validation dataset (or validation set); and the test dataset (or test set), reserved for evaluation purposes.

Table 12 below showcases the results of the LSTM experiment, which encompasses various configurations and performance metrics. Each configuration is characterized by a splitting ratio, learning rate (LR), padding max length (PML), batch size, and whether the dataset is imbalanced or balanced. The model achieves moderate performance, with accuracy ranging from 0.56 to 0.57 and F1-scores hovering around 0.72. Notably, the third configuration, featuring a larger PML of 800 tokens, produces significantly superior results with an accuracy of 0.87 and an F1-score of 0.89.

Of particular significance is the observation that the balanced dataset configuration consistently outperforms the imbalanced ones, with accuracy reaching 0.92 and an impressive F1-score of 0.96. This underscores the importance of addressing class imbalance and conducting careful hyperparameter tuning, including the selection of PML, in LSTM-based natural language processing tasks.

Table 12. Results of LSTM experiment with different hyperparameters

Splitting Ratio %	LR	PML	Batch Size	Data	Performance Metrics				
					Accuracy	Precision	Recall	F1-score	Kappa score
70/15/15	0.001	256	64	Imbalanced	0.5601	0.56	1.00	0.72	0.08
60/20/20	5e-4	256	34	Imbalanced	0.5743	0.56	0.99	0.72	0.02
64/16/20	0.00001	800	64	Imbalanced	0.8721	0.88	0.90	0.89	0.69
				Balanced	**0. 9251**	**0.92**	**1.00**	**0.96**	**0.70**

3.6. Experiments of BiLSTM

The configurations in this experiment are defined by splitting ratios, learning rates, maximum padding length, batch sizes, and the nature of the dataset (whether it is imbalanced or balanced). The reported performance metrics include accuracy, precision, recall, F1-score, and Kappa score.

In Table 13:

- Configuration **1**: With a 70/15/15 splitting ratio, LR of 0.001, PML of 256, and a batch size of 64, the model performs reasonably well, achieving an accuracy of 0.75 and an F1-score of 0.81, indicative of good overall performance. Precision, recall, and the Kappa score also exhibit strong predictive capabilities.
- Configuration **2**: Featuring a 60/20/20 splitting ratio, a slightly lower LR of 5e-4, and a batch size of 34, this configuration maintains competitive performance, with an accuracy of 0.71 and an F1-score of 0.79. While precision, recall, and the Kappa score are slightly lower, they still suggest a well-fitting model.
- Configuration **3**: Distinguished by a 64/16/20 splitting ratio, an extremely low LR of 0.00001, a larger PML of 800, and a batch size of 64, this configuration stands out with impressive results. It attains an accuracy of 0.88 and an F1-score of 0.90, showcasing the positive impact of hyperparameter tuning and a more balanced dataset split.

- Balanced Configuration: Presumed to address class imbalance, this configuration excels with the highest reported metrics across the board. It achieves an accuracy of 0.94 and an outstanding F1-score of 0.97, highlighting the importance of balanced data for enhanced model performance. Precision, recall, and the Kappa score also indicate exceptional predictive capabilities.

In summary, these results emphasize the significance of meticulous hyperparameter selection and dataset balance in optimizing our BiLSTM model for diverse tasks.

Table 13. Results of BiLSTM experiment with different hyperparameters

Splitting Ratio %	**LR**	**PML**	**Batch Size**	**Data**	**Performance Metrics**				
					Accuracy	**Precision**	**Recall**	**F1-score**	**Kappa score**
70/15	0.001	256	64	Imbalanced	0.7504	0.69	0.98	0.81	0.49
60/20	5e-4	256	34	Imbalanced	0.7113	0.67	0.98	0.79	0.38
80/20	0.00001	800	64	Imbalanced	0.88	0.87	0.92	0.90	0.65
				Balanced	**0. 9441**	**0.96**	**0.98**	**0.97**	**0.81**

While the BiLSTM model demonstrates impressive accuracy, an early manifestation of underfitting becomes evident at the first epoch. The accuracy swiftly climbs to 94% across all subdivisions of the test set. However, in the case of the training set, the accuracy hovers around 72.8%, as depicted in Figure 23.

The primary objective during training is to minimize the Loss (L) with respect to the model parameters. Consequently, we engaged in an extensive fine-tuning process, experimenting with multiple optimizers, learning rates, and epochs in our pursuit of optimal model performance.

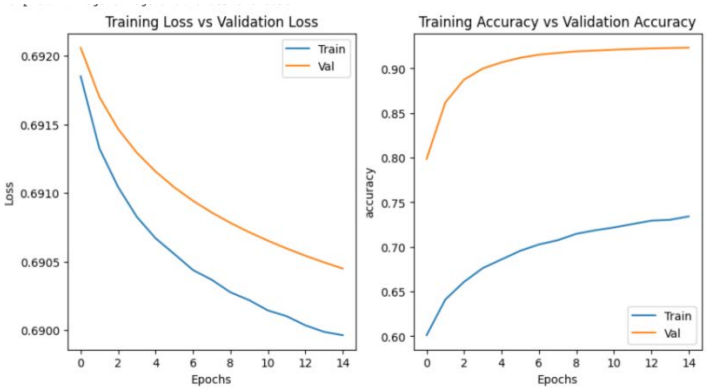


Figure 19. Underfitting issue in BiLSTM experiment.

3.7. Experiments of CNN-BiLSTM

The outcomes of the CNN-BiLSTM experiment are detailed in Table 14, presenting various configurations along with their corresponding performance metrics. Each configuration is characterized by splitting ratios, learning rates, maximum padding length, batch sizes, and the dataset's balance status. The performance metrics encompass accuracy, precision, recall, F1-score, and Kappa score.

The results underscore the substantial impact of hyperparameter selection and data balance on model performance. The "Balanced" configuration stands out with remarkable results, achieving a high accuracy of 0.95 and an exceptional F1-score of 0.98. This emphasizes the significance of addressing class balance and optimizing hyperparameters for the CNN-BiLSTM-based task.

Table 14. Results of CNN-BiLSTM experiment with different hyperparameters

Splitting Ratio %	LR	PML	Batch Size	Data	Performance Metrics				
					Accuracy	Precision	Recall	F1-score	Kappa score
70/15	0.001	256	64	Imbalanced	0.56	0.56	1.00	0.72	0.11
60/20	5e-4	256	34		0.74	0.69	0.98	0.81	0.44
80/20	0.0001	800	64		0.87	0.88	0.89	0.89	0.64
				Balanced	**0.95**	**0.96**	**0.99**	**0.98**	**0.9**

In the analysis of the results from various neural network models in Tables 12, 13, and 14, we discern their performance under two distinct scenarios: one with imbalanced data and the other with balanced data.

For the LSTM model dealing with imbalanced data, it achieves an accuracy of 0.87. However, it's worth noting that it exhibits a lower precision of 0.88, implying that while it correctly identifies a significant portion of instances, it may generate some false positives. The recall value is 0.90, indicating its effective capture of actual positive cases. Consequently, the F1-score stands at 0.89, signifying a balanced trade-off between precision and recall in this scenario.

Conversely, when applied to balanced data, the LSTM model's performance experiences a significant improvement. The accuracy soars to 0.92, showcasing its capability to make more accurate predictions across both classes. The precision remains high at 0.92, and the recall reaches a perfect score of 1.00, indicating its ability to correctly identify all actual positive cases. This leads to an impressive F1-score of 0.96, highlighting the model's excellent overall performance in the balanced data scenario.

Turning to the BiLSTM model, it exhibits a similar pattern. In the imbalanced data setting, it achieves an accuracy of 0.88, maintaining a decent balance between precision (0.87) and recall (0.92), resulting in an F1-score of 0.90. However, when faced with balanced data, the BiLSTM model's performance undergoes a noticeable boost. The accuracy climbs to 0.94, and both precision and recall achieve impressive scores of 0.96

and 0.98, respectively. This harmonious blend of precision and recall results in a remarkable F1-score of 0.97, emphasizing its strength in this particular context.

Lastly, the CNN-BiLSTM model follows a similar pattern. In the imbalanced data scenario, it attains an accuracy of 0.87, with a balance between precision (0.88) and recall (0.89), resulting in an F1-score of 0.89. However, when transitioning to balanced data, the CNN-BiLSTM model's performance experiences a significant surge. Its accuracy reaches 0.95, and both precision and recall maintain robust values of 0.96 and 0.99, respectively. This leads to an exceptional F1-score of 0.98, highlighting its proficiency in handling balanced datasets.

Additionally, we experimented with the opposite approach, the BiLSTM-CNN model combined with XGBoost, which achieved an accuracy of 0.49 and resulted in an F1-score of 0.66.

In summary, all three neural network models—LSTM, BiLSTM, and CNN-BiLSTM—demonstrate improved performance when applied to balanced data as compared to imbalanced data. The balanced data setting enables these models to attain higher accuracy and strike a more harmonious balance between precision and recall, ultimately resulting in superior F1-scores.

3.8. Experiment of Multi-Classification Task

In the multi-classification experiment, we utilized the CNN-BiLSTM model to classify text data into three classes: credible, not credible, and undecided. The results presented in Table 15 pertain to the performance evaluation of the model, which employed pre-trained word embeddings to convert text data into numerical vectors for analysis. It's worth noting that the dataset used for training and testing the model is perfectly balanced.

Regarding model performance, the Accuracy stands at 45%, indicating that the model correctly predicted 45% of the examples in the dataset. The Precision score of 46% implies that 46% of the model's positive predictions were indeed true positives. Notably, the Recall is exceptional at 100%, but the F1-score is 0.63, representing a balanced trade-off between precision and recall.

In conclusion, this model demonstrates outstanding recall but somewhat lower precision, with the F1-score striking a balance between these two performance metrics.

Table 15. Result of Multi-Classification Task

Model	Hyperparameters		Performance Metrics			
	Splitting Ratio %	Batch Size	Accuracy	Precision	Recall	F1-score
CNN-BiLSTM	63/16/20	64	0.4545	0.46	1.00	0.63

4. DISCUSSION

For the ML Algorithms in the course of conducting our research on text classification methodologies, We have conducted experiments with various approaches aimed at efficiently preprocessing and modeling textual data. Each of these approaches represents distinct strategies within the field, and we would like to outline them here.

First and foremost, we implemented a comprehensive pipeline that begins with the utilization of a `CountVectorizer`. This initial step transforms textual data into a matrix of word frequencies. To delve deeper into feature representation, we introduced a `TfidfTransformer` into the pipeline. This step converts word frequencies into TF-IDF (Term Frequency-Inverse Document Frequency) values, which are instrumental in highlighting the significance of words in their respective contexts. Furthermore, we integrated feature selection into the pipeline by incorporating the `SelectKBest` method with the chi-squared statistic. This allows for the selection of the top 1000 features that are most informative for the classification task. Lastly, we employed a logistic regression classifier equipped with the 'liblinear' solver for the classification phase.

Moving on to the second approach, we retained the use of a `CountVectorizer` for the initial text preprocessing step. However, a significant departure lies in the subsequent normalization stage. In this case, we opted for a `Binarizer` transformation, which directly converts word frequencies into binary values (0 or 1). This simplifies the feature representation significantly. As in the first approach, we employed a logistic regression classifier with the 'liblinear' solver for classification.

The third approach ventured into a different facet of text classification by incorporating character-level n-grams into the feature representation. To accomplish this, we employed a `CountVectorizer` with an n-gram range of (1,3). This allowed our model to capture not only word-level information but also delve into character-level patterns present in the text. Similar to the second approach, normalization involved the application of a `Binarizer` transformation. Classification remained consistent with the use of a logistic regression model with the 'liblinear' solver.

In summary, these distinct approaches have provided me with a range of options for constructing text classification pipelines. The choice among these strategies depends on the unique characteristics of the dataset and the specific research objectives at hand. Considerations include the importance of feature selection, the level of granularity in

feature representation (word-level or character-level), and the balance between computational efficiency and classification performance. Such considerations are crucial in developing robust and contextually appropriate text classification models for my research.

The models appear to perform very well on the classification task, with the third approach (Binarizer with character n-grams and SelectKBest) showing a slightly better F1-score, indicating a good balance between precision and recall. Figure 21 provides a comparison of the ML Models in terms of classification accuracy, where XGBoost achieved the highest classification accuracy of 94.4 percent.

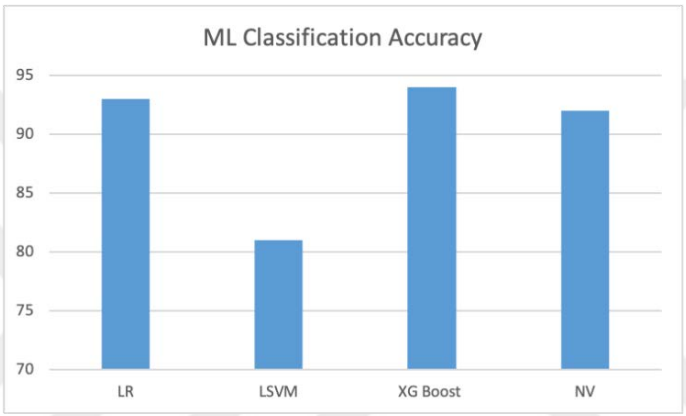


Figure 20. Comparison results of classification accuracy for the ML Classifiers

Since the FND task is essentially a classification task, leveraging deep learning (DL) approaches becomes an appealing solution. The word embeddings obtained from the feature extraction step were used to train various deep learning models, including CNN and RNN, in order to compare them with machine learning models. In all the deep learning experiments, categorical cross-entropy was employed as the loss function, and we used both ADAM and SGD as optimizers. Additionally, we experimented with different batch sizes, ranging from 34 to 256, and each model was trained for a varying number of epochs, ranging from 20 to 1000.

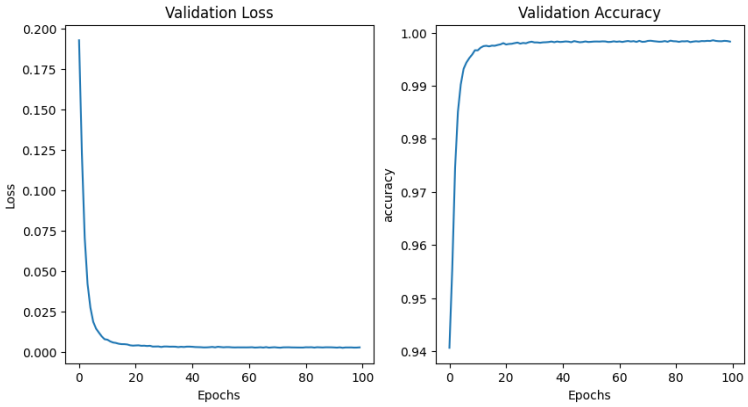


Figure 21. Validation Loss and Accuracy Output of CNN-BiLSTM exp.

The optimization of deep learning models often relies on the stochastic gradient descent (SGD) algorithm. However, fine-tuning the learning rate of SGD can be quite challenging, especially when dealing with parameters of varying magnitudes. This challenge necessitates continuous adjustments during the training phase (Zhang, Z., 2018).

In our experiments, we configured the models to use the Stochastic Gradient Descent (SGD) optimizer, as described in equation 19. SGD is a gradient descent algorithm that optimizes a differentiable objective function by iteratively adjusting model parameters in the direction of the steepest descent of the objective function. Its stochastic nature introduces some noise into the training process, which can help prevent overfitting and enhance generalization performance.

Furthermore, we updated the learning rate value to be calculated based on the loss value, as shown in the equation below, where β is set to 0.001 and "loss" represents the loss value of the test set:

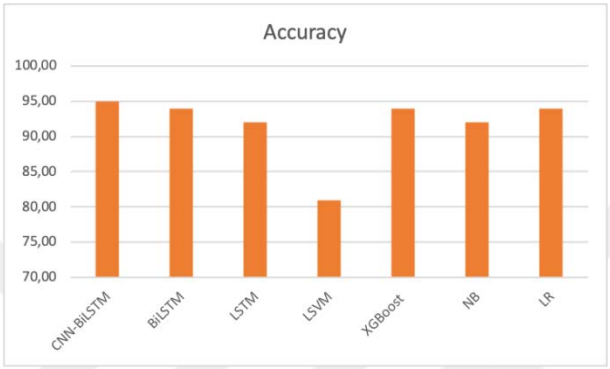
$$\alpha = \frac{\beta}{1 + e^{-loss}} \tag{19}$$

Callbacks in deep learning are functions or objects used to perform specific actions during the training process, such as stopping training early under certain conditions. One commonly used callback is the EarlyStopping callback, which we configured to monitor the validation loss throughout our model training.

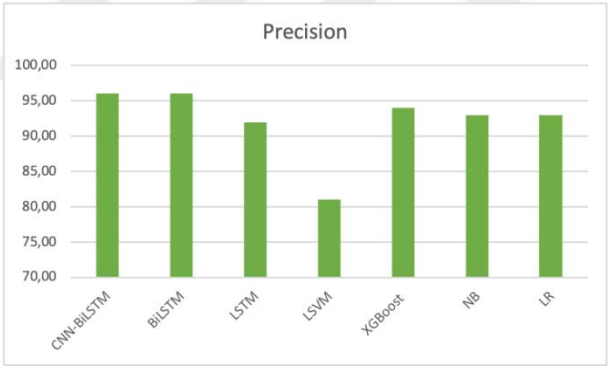
The EarlyStopping callback allows training to continue as long as the validation loss either keeps improving or remains constant for up to 10 consecutive epochs. However, if the validation loss increases for 10 consecutive epochs, the training process is halted. This

technique is particularly useful for preventing overfitting and conserving time during model training. It automatically stops training when further iterations do not result in improved performance on the validation data.

In our experiments, we also applied the "stop accuracy" method. This method enabled us to stop the training process when the model reached its highest accuracy, thus avoiding unnecessary time and storage consumption.



a) Accuracy comparison between ML & DL Models



b) Precision comparison between ML & DL Models

c) Recall comparison between ML & DL Models

d) F1-Score comparison between ML & DL Models

Figure 22. Comparison between ML & DL experiments with respect to a) Accuracy b) Precision c) Recall and d) F1-Score.

5. RESULTS

This chapter summarizes the results and lessons learned from the implementation and presents an analysis of the ML and DL modules with AFND task.

5.1. Comparison of All the Machine Learning Models

In this section, we conduct a comparative analysis of four machine learning classification models. Table 16 presents the Accuracy (Acc) scores and F1-scores of these classifier models. During this comparative analysis of classification models, we observed variations in their performance metrics. The Linear Support Vector Machine (LSVM) achieves a reasonable accuracy of 0.81 but shows a slightly lower F1-score of 0.84, indicating potential room for improvement in balancing precision and recall. Logistic Regression stands out with impressive results, boasting an accuracy of 0.93 and an F1-score of 0.97, showcasing both strong overall correctness and a remarkable balance between precision and recall. Naïve Bayes also performs well, with an accuracy of 0.92 and an F1-score of 0.96, indicating reliable predictions and a good trade-off between precision and recall. X-Gradient Boost surpasses all in terms of accuracy, achieving 0.94, and maintains an exceptional balance between precision and recall with an F1-score of 0.97.

Overall, Logistic Regression and X-Gradient Boost demonstrate superior performance, making them compelling choices for this classification task, depending on specific task requirements and objectives.

Table 16. Comparison of the performance of the machine learning models

Classification Model	Accuracy	F1-Score
LSVM	0.81	0.84
Logistic Regression	0.93	0.97
Naïve Bayes	0.92	0.96
X-Gradient Boost	0.94	0.97

5.2. Comparison With Another Studies

Compared to both deep learning and traditional machine learning (ML) models, the CNN-BiLSTM model exhibits the highest accuracy for both tasks on the test and validation sets. Table 17 provides a summary of various studies conducted in the context of Fake News Detection, outlining the models employed, datasets utilized, languages of the text data, and the highest achieved accuracy in each study.

In the first study (Khalil, A. et al., 2021), several models, including the Capsule Network, DPCNN, DDQN, and others, were applied to the same dataset used in our study, yielding a best accuracy of 79.0%. In the study by Wang, W. Y. (2017), models such as Bi-LSTM, CNN, SVM, LR (Logistic Regression), and LIAR were applied to an English dataset, resulting in an accuracy of 27%.

In our research, a variety of models, including LSVM, XGBoost, NV, LR, LSTM, BiLSTM, and CNN-BiLSTM, were applied to an Arabic dataset for Fake News Detection, achieving a best accuracy of 95.0%. When compared to reinforcement learning and traditional ML models, as well as the models employed in Khalil, A. et al. (2021), our model demonstrated the highest accuracy.

Table 17. Performance Comparison of our study to the referenced ones

Work	Dataset	Lang.	Applied Model	Accuracy
Khalil, A. et al., 2021	AFND	Arabic	**Capsule Network**	**79.0%**
			DPCNN	77.5%
			DDQN	73.4%
			SVC	57.6%
			Linear SVC	54.5%
			ID-LSTM	50.4%
			KMeans	49.5%
			Bayesian Gaussian	50.2%
Wang, W. Y., 2017	LIAR	English	Bi-LSTM	23.3%
			CNN	**27.0%**
			LSVM	20.44%
			LR	21.37%
			XGBoost	25.42%
			NB	24.18%

Continued Table 17.

Our Study	AFND	Arabic	LSVM	81%
			XGBoost	94%
			NV	92%
			LR	93%
			LSTM	92%
			BiLSTM	94%
			CNN-BiLSTM	**95.0%**

6. CONCLUSION

As the use of news sources like social media and various online platforms continues to rise, the task of distinguishing authentic information from fake news has grown increasingly challenging and essential. This challenge is especially pronounced in the case of Arabic language news due to the various writing styles and formats unique to this language. We've identified a limited amount of research in this area, particularly concerning Arabic language posts, and a previous lack of focus on optimizing the feature selection process, which is essential for improving detection accuracy.

In this study, we employed word embedding techniques and a CNN-BiLSTM model to develop a fake news detection model using machine learning and deep learning techniques. Our machine learning models showed promising performance in our experiments, especially when utilizing n-grams and selectkbest, which resulted in a slightly better F1-score. This indicates a well-balanced trade-off between precision and recall. When comparing various machine learning models in terms of classification accuracy, it's evident that XGBoost achieved the highest accuracy, reaching 94 percent.

The results of this research have demonstrated that the CNN-BiLSTM model achieved the highest accuracy for binary classification tasks when using the first extensive and diverse Arabic fact-check news corpus, AFND. However, it is important to note that while training various models, we encountered challenges related to underfitting and overfitting. To address these challenges and ensure the model's generalization, we found it necessary to implement an early stopping mechanism during training.

These observations underscore the formidable nature of the Arabic corpus used in our study. Its richness and diversity in content can potentially introduce noise into the dataset, emphasizing the need for models to meticulously distinguish between relevant information and extraneous details.

The employed model exhibits the capability to accurately identify fake news based solely on the content of a news article. It offers state-of-the-art performance on the tested datasets, making it an efficient tool for Arabic news detection.

7. FUTURE WORK

Looking ahead, our future objectives involve implementing enhancements to the model to achieve optimal results with the AFND dataset. We plan to optimize Arabic word processing, particularly with the Farasa library, which currently consumes a significant amount of processing time. We aim to introduce a solution that enables us to call this library via an API for big data processing, streamlining our workflow.

While this research has primarily concentrated on binary classification, our next steps will involve expanding our focus to multi-class classification using the same AFND dataset, considering it as the latest version. Additionally, we intend to incorporate transfer learning into our testing process to provide a more comprehensive analysis.

8. REFERENCES

Su, Q., Wan, M., Liu, X., & Huang, C. R. (2020). Motivations, methods and metrics of misinformation detection: an NLP perspective. Natural Language Processing Research,1(1-2), 1-13.

Paka, W. S., Bansal, R., Kaushik, A., Sengupta, S., & Chakraborty, T. (2021). Cross SEAN: A cross-stitch semi-supervised neural attention model for COVID-19 fake news detection. Applied Soft Computing, 107, 107393.

Zhou, X., & Zafarani, R. (2020). A survey of fake news: Fundamental theories, detection methods, and opportunities. ACM Computing Surveys (CSUR), 53(5), 1-40.

Gayathri, K., & Marimuthu, A. (2013, January). Text document pre-processing with the KNN for classification using the SVM. In 2013 7th International Conference on Intelligent Systems and Control (ISCO) (pp. 453-457). IEEE.

Du, J., Xu, R., He, Y., & Gui, L. (2017, August). Stance classification with target-specific neural attention networks. International Joint Conferences on Artificial Intelligence.

Küçük, D., & Can, F. (2020). Stance detection: A survey. ACM Computing Surveys (CSUR), 53(1), 1-37.

Aker, A., Derczynski, L., & Bontcheva, K. (2017). Simple open stance classification for rumour analysis. arXiv preprint arXiv:1708.05286.

Kuyumcu, B., Aksakalli, C., & Delil, S. (2019, June). An automated new approach in fast text classification (fastText) A case study for Turkish text classification without pre-processing. In Proceedings of the 2019 3rd International Conference on Natural Language Processing and Information Retrieval (pp. 1-4).

Singh, P., Srivastava, R., Rana, K. P. S., & Kumar, V. (2022). SEMI-FND: Stacked Ensemble Based Multimodal Inference For Faster Fake News Detection. arXiv preprint arXiv:2205.08159.

Khalil, A., Jarrah, M., Aldwairi, M., & Jaradat, M. (2022). AFND: Arabic fake news dataset for the detection and classification of articles credibility. Data in Brief, 42, 108141.

Seyam, A., Bou Nassif, A., Abu Talib, M., Nasir, Q., & Al Blooshi, B. (2021, August). Deep Learning Models to Detect Online False Information: a Systematic Literature Review. In The 7th Annual International Conference on Arab Women in Computing in conjunction with the 2nd Forum of Women in Research (pp. 1-5).

Patil, R., Boit, S., Gudivada, V., & Nandigam, J. (2023). A Survey of Text Representation and Embedding Techniques in NLP. IEEE Access.

Mridha, M. F., Keya, A. J., Hamid, M. A., Monowar, M. M., & Rahman, M. S. (2021). A comprehensive review on fake news detection with deep learning. *IEEE Access*, 9, 156151-156170.

Rahab, H., Zitouni, A., & Djoudi, M. (2021, November). Arabic Fake News and Spam Handling: Methods, Resources and Opportunities. In *2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP)* (pp. 1-7). IEEE.

Wang, W. Y. (2017). "liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*.

Khalil, A., Jarrah, M., Aldwairi, M., & Jararweh, Y. (2021, November). Detecting Arabic fake news using machine learning. In *2021 Second International Conference on Intelligent Data Science Technologies and Applications (IDSTA)* (pp. 171-177). IEEE.

Nabil, M., Aly, M., & Atiya, A. (2015, September). Astd: Arabic sentiment tweets dataset. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 2515-2519).

Salem, F. K. A., Al Feel, R., Elbassuoni, S., Jaber, M., & Farah, M. (2019, July). Fa-kes: A fake news dataset around the Syrian war. In *Proceedings of the international AAAI conference on web and social media* (Vol. 13, pp. 573-582).

Ali, Z. S., Mansour, W., Elsayed, T., & Al-Ali, A. (2021, April). AraFacts: the first large Arabic dataset of naturally occurring claims. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop* (pp. 231-236).

Alkhair, M., Meftouh, K., Smaïli, K., & Othman, N. (2019). An Arabic corpus of fake news: Collection, analysis and classification. In *Arabic Language Processing: From Theory to Practice: 7th International Conference, ICALP 2019, Nancy, France, October 16–17, 2019, Proceedings 7* (pp. 292-302). Springer International Publishing.

Alhindi, T., Alabdulkarim, A., Alshehri, A., Abdul-Mageed, M., & Nakov, P. (2021). Arastance: A multi-country and multi-domain dataset of Arabic stance detection for fact checking. *arXiv preprint arXiv:2104.13559*.

Saadany, H., Mohamed, E., & Orasan, C. (2020). Fake or real? A study of Arabic satirical fake news. *arXiv preprint arXiv:2011.00452*.

Himdi, H., Weir, G., Assiri, F., & Al-Barhamtoshy, H. (2022). Arabic fake news detection based on textual analysis. *Arabian Journal for Science and Engineering*, 47(8), 10453-10469.

Hawashin, B., AlZu'bi, S., Althunibat, A., Kanan, T., & Sharrab, Y. Improving Arabic Fake News Detection Using Optimized Feature Selection.

Hammad, M., & Hemayed, E. (2013). Automating credibility assessment of Arabic news. In *Social Informatics: 5th International Conference, SocInfo 2013, Kyoto, Japan, November 25-27, 2013, Proceedings 5* (pp. 139-152). Springer International Publishing.

Antoun, W., Baly, F., & Hajj, H. (2020). Arabert: Transformer-based model for Arabic language understanding. *arXiv preprint arXiv:2003.00104*.

Al-Yahya, M., Al-Khalifa, H., Al-Baity, H., AlSaeed, D., & Essam, A. (2021). Arabic fake news detection: comparative study of neural networks and transformer-based approaches. *Complexity*, 2021, 1-10.

Jardaneh, G., Abdelhaq, H., Buzz, M., & Johnson, D. (2019, April). Classifying Arabic tweets based on credibility using content and user features. In *2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT)* (pp. 596-601). IEEE.

Nassif, A. B., Elnagar, A., Elgendy, O., & Afadar, Y. (2022). Arabic fake news detection based on deep contextualized embedding models. *Neural Computing and Applications*, 34(18), 16019-16032.

Sharifani, K., Amini, M., Akbari, Y., & Aghajanzadeh Godarzi, J. (2022). Operating Machine Learning across Natural Language Processing Techniques for Improvement of Fabricated News Model. *International Journal of Science and Information System Research*, 12(9), 20-44.

Raza, S., & Ding, C. (2022). Fake news detection based on news content and social contexts: a transformer-based approach. *International Journal of Data Science and Analytics*, 13(4), 335-362.

Davoudi, M., Moosavi, M. R., & Sadreddini, M. H. (2022). DSS: A hybrid deep model for fake news detection using propagation tree and stance network. *Expert Systems with Applications*, 198, 116635.

Najadat, H., Tawalbeh, M., & Awawdeh, R. (2022). Fake news detection for Arabic headlines-articles news data using deep learning. *International Journal of Electrical & Computer Engineering (2088-8708)*, 12(4).

Abd Elaziz, M., Dahou, A., Orabi, D. A., Alshathri, S., Soliman, E. M., & Ewees, A. A. (2023). A Hybrid Multitask Learning Framework with a Fire Hawk Optimizer for Arabic Fake News Detection. *Mathematics*, 11(2), 258.

Samih, Y., & Darwish, K. (2021, April). A few topical tweets are enough for effective user stance detection. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume* (pp. 2637-2646).

Safaya, A., Abdullatif, M., & Yuret, D. (2020). Kuisail at semeval-2020 task 12: Bert-cnn for offensive speech identification in social media. *arXiv preprint arXiv:2007.13184*.

Huang, Y. F., & Chen, P. H. (2020). Fake news detection using an ensemble learning model based on self-adaptive harmony search algorithms. *Expert Systems with Applications*, 159, 113584.

Jiang, T. A. O., Li, J. P., Haq, A. U., Saboor, A., & Ali, A. (2021). A novel stacking approach for accurate detection of fake news. *IEEE Access*, 9, 22626-22639.

Harrag, F., & Djahli, M. K. (2022). Arabic fake news detection: A fact checking based deep learning approach. *Transactions on Asian and Low-Resource Language Information Processing*, 21(4), 1-34.

Baniata, L. H., & Park, S. B. (2016). Sentence representation network for Arabic sentiment analysis. *한국정보과학회 학술발표논문집*, 470-472.

Ouassil, M. A., Cherradi, B., Hamida, S., Errami, M., El Gannour, O., & Raihani, A. (2022). A Fake News Detection System based on Combination of Word Embedded Techniques and Hybrid Deep Learning Model. *International Journal of Advanced Computer Science and Applications*, 13(10).

Kaliyar, R. K., Goswami, A., & Narang, P. (2021, January). A Hybrid Model for Effective Fake News Detection with a Novel COVID-19 Dataset. In *ICAART (2)* (pp. 1066-1072).

Alharbi, O. (2021). A deep learning approach combining CNN and Bi-LSTM with SVM classifier for Arabic sentiment analysis. *International Journal of Advanced Computer Science and Applications*, 12(6).

Farha, I. A., & Magdy, W. (2020, May). Multitask learning for Arabic offensive language and hate-speech detection. In *Proceedings of the 4th workshop on open-source Arabic corpora and processing tools, with a shared task on offensive language detection* (pp. 86-90).

Truică, C. O., Apostol, E. S., Nicolescu, R. C., & Karras, P. (2023). MCWDST: a Minimum-Cost Weighted Directed Spanning Tree Algorithm for Real-Time Fake News Mitigation in Social Media. *arXiv preprint arXiv:2302.12190*.

Shang, L., Sui, L., Wang, S., & Zhang, D. (2020, May). Sentiment analysis of film reviews based on CNN-BLSTM-Attention. In *Journal of Physics: Conference Series* (Vol. 1550, No. 3, p. 032056). IOP Publishing.

Abdelhady, N., Soliman, T. H. A., & Farghally, M. F. (2022). Stacked-CNN-BiLSTM-COVID: An Effective Stacked Ensemble Deep Learning Framework for Sentiment Analysis of Arabic COVID-19 Tweets.

Sorour, S. E., & Abdelkader, H. E. (2022). AFND: Arabic fake news detection with an ensemble deep CNN-LSTM model. *J. Theor. Appl. Inf. Technol.*, 100(14), 5072-5086.

Abu Kwaik, K., Saad, M., Chatzikyriakidis, S., & Dobnik, S. (2019). LSTM-CNN deep learning model for sentiment analysis of dialectal Arabic. In *Arabic Language Processing: From Theory to Practice: 7th International Conference, ICALP 2019, Nancy, France, October 16–17, 2019, Proceedings 7* (pp. 108-121). Springer International Publishing.

Abdullah, M., Hadzikadicy, M., & Shaikhz, S. (2018, December). SEDAT: sentiment and emotion detection in Arabic text using CNN-LSTM deep learning. In *2018 17th IEEE international conference on machine learning and applications (ICMLA)* (pp. 835-840). IEEE.

Zhou, C., Sun, C., Liu, Z., & Lau, F. (2015). A C-LSTM neural network for text classification. *arXiv preprint arXiv:1511.08630*.

Abedalla, A., Al-Sadi, A., & Abdullah, M. (2019, October). A closer look at fake news detection: A deep learning perspective. In *Proceedings of the 3rd International Conference on Advances in Artificial Intelligence* (pp. 24-28).

Ghourabi, A., Mahmood, M. A., & Alzubi, Q. M. (2020). A hybrid CNN-LSTM model for SMS spam detection in Arabic and English messages. *Future Internet*, 12(9), 156.

Hifny, Y., & Ali, A. (2019, May). Efficient Arabic emotion recognition using deep neural networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6710-6714). IEEE.

Mubarak, H. (2017). Build fast and accurate lemmatization for Arabic. *arXiv preprint arXiv:1710.06700*.

Zeroual, I., & Lakhouaja, A. (2017, April). Arabic information retrieval: Stemming or lemmatization?. In *2017 Intelligent Systems and Computer Vision (ISCV)* (pp. 1-6). IEEE.

Wotaifi, T. A., & Dhannoon, B. N. (2023). An Effective Hybrid Deep Neural Network for Arabic Fake News Detection. *Baghdad Science Journal*.

Alawadh, H. M., Alabrah, A., Meraj, T., & Rauf, H. T. (2023). Attention-Enriched Mini-BERT Fake News Analyzer Using the Arabic Language. *Future Internet*, 15(2), 44.

Alnabrisi, I., & Saad, M. Detect Arabic Fake News Through Deep Learning Models and Transformers. Available at SSRN 4341610.

Awajan, A. (2023). Enhancing Arabic Fake News Detection for Twitters Social Media Platform Using Shallow Learning Techniques. *Journal of Theoretical and Applied Information Technology*, 101(5).

Fouad, K. M., Sabbeh, S. F., & Medhat, W. (2022). Arabic Fake News Detection Using Deep Learning. *Computers, Materials & Continua*, 71(2).

Diwali, A., Dashtipour, K., Saeedi, K., Gogate, M., Cambria, E., & Hussain, A. (2022). Arabic sentiment analysis using dependency-based rules and deep neural networks. *Applied Soft Computing*, 127, 109377.

Farha, I. A., Oprea, S. V., Wilson, S., & Magdy, W. (2022, July). SemEval-2022 task 6: iSarcasmEval, intended sarcasm detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)* (pp. 802-814).

Alyoubi, S., Kalkatawi, M., & Abukhodair, F. (2023). The Detection of Fake News in Arabic Tweets Using Deep Learning. *Applied Sciences*, 13(14), 8209.

Bahurmuz, N. O., Amoudi, G. A., Baothman, F. A., Jamal, A. T., Alghamdi, H. S., & Alhothali, A. M. (2022). Arabic Rumor Detection Using Contextual Deep Bidirectional Language Modeling. *IEEE Access*, 10, 114907-114918.

Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2021). Dive into deep learning. *arXiv preprint arXiv:2106.11342*.

Tang, J., Alelyani, S., & Liu, H. (2014). Feature selection for classification: A review. *Data classification: Algorithms and applications*, 37.

Ahuja, N., & Kumar, S. (2020, June). S-HAN: Hierarchical attention networks with stacked gated recurrent unit for fake news detection. In *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)* (pp. 873-877). IEEE.

Mahlous, A. R., & Al-Laith, A. (2021). Fake news detection in Arabic tweets during the COVID-19 pandemic. *International Journal of Advanced Computer Science and Applications*, 12(6), 778-788.

Alzanin, S. M., & Azmi, A. M. (2019). Rumor detection in Arabic tweets using semi-supervised and unsupervised expectation–maximization. *Knowledge-Based Systems*, 185, 104945.

Sabbeh, S. F., & Baatwah, S. Y. (2018). ARABIC NEWS CREDIBILITY ON TWITTER: AN ENHANCED MODEL USING HYBRID FEATURES. *journal of theoretical & applied information technology*, 96(8).

Alajmi, A., Saad, E. M., & Darwish, R. R. (2012). Toward an ARABIC stop-words list generation. *International Journal of Computer Applications*, 46(8), 8-13

Zerrouki, T. (2020). Towards an open platform for Arabic language processing.

Darwish, K., & Mubarak, H. (2016, May). Farasa: A new fast and accurate Arabic word segmenter. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)* (pp. 1070-1074).

Baly, R., Khaddaj, A., Hajj, H., El-Hajj, W., & Shaban, K. B. (2019). Arsentd-lev: A multi-topic corpus for target-based sentiment analysis in Arabic Levantine tweets. *arXiv preprint arXiv:1906.01830*.

Ameur, M. S. H., & Aliane, H. (2021). AraCOVID19-MFH: Arabic COVID-19 multi-label fake news & hate speech detection dataset. *Procedia Computer Science*, 189, 232-241.

Haouari, F., Hasanain, M., Suwaileh, R., & Elsayed, T. (2020). ArCOV19-rumors: Arabic COVID-19 Twitter dataset for misinformation detection. *arXiv preprint arXiv:2010.08768*.

Assaf, R., & Saheb, M. (2021, October). Dataset for Arabic fake news. In *2021 IEEE 15th International Conference on Application of Information and Communication Technologies (AICT)* (pp. 1-4). IEEE.

Luengo, J., García-Gil, D., Ramírez-Gallego, S., García, S., & Herrera, F. (2020). *Big data preprocessing*. Cham: Springer.

Zerrouki, T. (2023, April 02). Tashaphyne, Arabic light stemmer, Accessed through https://pypi.python.org/pypi/Tashaphyne/0.2

Zerrouki, T. (2023, April 02). *Pyarabic, An Arabic language library for Python* , Accessed through https://pypi.python.org/pypi/pyarabic/, 2010.

Dwarampudi, M., & Reddy, N. V. (2019). Effects of padding on LSTMs and CNNs. *arXiv preprint arXiv:1903.07288*.

Lopez-del Rio, A., Nonell-Canals, A., Vidal, D., & Perera-Lluna, A. (2019). Evaluation of cross-validation strategies in sequence-based binding prediction using deep learning. *Journal of chemical information and modeling*, 59(4), 1645-1657.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1), 1929-1958.

Li, S., & Gong, B. (2021). Word embedding and text classification based on deep learning methods. In *MATEC Web of Conferences* (Vol. 336, p. 06022). EDP Sciences.

Christian, H., Agus, M. P., & Suhartono, D. (2016). Single document automatic text summarization using term frequency-inverse document frequency (TF-IDF). *ComTech: Computer, Mathematics and Engineering Applications*, 7(4), 285-294.

Zhang, Y., & Rao, Z. (2020, June). n-BiLSTM: BiLSTM with n-gram Features for Text Classification. In *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)* (pp. 1056-1059). IEEE.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Yue, W., & Li, L. (2020, December). Sentiment analysis using Word2vec-CNN-BiLSTM classification. In *2020 seventh international conference on social networks analysis, management and security (SNAMS)* (pp. 1-5). IEEE.

Bogale Gereme, F., & Zhu, W. (2020, November). Fighting fake news using deep learning: Pre-trained word embeddings and the embedding layer investigated. In *Proceedings of the 2020 3rd International Conference on Computational Intelligence and Intelligent Systems* (pp. 24-29).

Pathan, M. S., Nag, A., Pathan, M. M., & Dev, S. (2022). Analyzing the impact of feature selection on the accuracy of heart disease prediction.*Healthcare Analytics*, 2, 100060.

Anukrishna, P. R., & Paul, V. (2017, January). A review on feature selection for high dimensional data. In *2017 International Conference on Inventive Systems and Control (ICISC)* (pp. 1-4). IEEE.

Thaseen, I. S., Kumar, C. A., & Ahmad, A. (2019). Integrated intrusion detection model using chi-square feature selection and ensemble of classifiers. *Arabian Journal for Science and Engineering*, 44, 3357-3368.

Shah, K., Patel, H., Sanghvi, D., & Shah, M. (2020). A comparative analysis of logistic regression, random forest and KNN models for the text classification. *Augmented Human Research*, 5, 1-16.

Indra, S. T., Wikarsa, L., & Turang, R. (2016, October). Using logistic regression method to classify tweets into the selected topics. In *2016 international conference on advanced computer science and information systems (icacsis)* (pp. 385-390). IEEE.

Gal, Y., & Ghahramani, Z. (2016). A theoretically grounded application of dropout in recurrent neural networks. *Advances in neural information processing systems*, 29.

Zhang, J., Li, Y., Tian, J., & Li, T. (2018, October). LSTM-CNN hybrid model for text classification. In *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)* (pp. 1675-1680). IEEE.

Jin, N., Wu, J., Ma, X., Yan, K., & Mo, Y. (2020). Multi-task learning model based on multi-scale CNN and LSTM for sentiment classification. *IEEE Access*, 8, 77060-77072.

Kowsher, M., Tahabilder, A., Sanjid, M. Z. I., Prottasha, N. J., Uddin, M. S., Hossain, M. A., & Jilani, M. A. K. (2021). LSTM-ANN & BiLSTM-ANN: Hybrid deep learning models for enhanced classification accuracy. *Procedia Computer Science*, 193, 131-140.

Bluche, T., Kermorvant, C., & Louradour, J. (2015, August). Where to apply dropout in recurrent neural networks for handwriting recognition?. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)* (pp. 681-685). IEEE.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1), 37-46.

Sun, S. (2011). Meta-analysis of Cohen's kappa. *Health Services and Outcomes Research Methodology*, 11, 145-163.

Josephine, V. H., Nirmala, A. P., & Alluri, V. L. (2021, April). Impact of hidden dense layers in convolutional neural network to enhance performance of classification model. In *IOP Conference Series: Materials Science and Engineering* (Vol. 1131, No. 1, p. 012007). IOP Publishing.

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv preprint arXiv:1707.02919*.

Su, J., Shirab, J. S., & Matwin, S. (2011). Large scale text classification using semi-supervised multinomial naive bayes. In *Proceedings of the 28th international conference on machine learning (ICML-11)* (pp. 97-104).

Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. *Information*, 10(4), 150.

Suthaharan, S., & Suthaharan, S. (2016). Support vector machine. *Machine learning models and algorithms for big data classification: thinking with examples for effective learning*, 207-235.

Khanam, Z., Alwasel, B. N., Sirafi, H., & Rashid, M. (2021, March). Fake news detection using machine learning approaches. In *IOP conference series: materials science and engineering* (Vol. 1099, No. 1, p. 012040). IOP Publishing.

Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785-794).

Bentéjac, C., Csörgő, A., & Martínez-Muñoz, G. (2021). A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, 54, 1937-1967.

Zhang, Z. (2018, June). Improved adam optimizer for deep neural networks. In *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)* (pp. 1-2). IEEE.

CURRICULUM VITAE

Maysaa ALSAFADI, completed her high school education in 2008 before pursuing undergraduate studies in Computer Engineering at the Islamic University of Gaza, Faculty of Engineering. Graduating in 2013, she embarked on a dynamic career trajectory. In 2019, she initiated a thesis-based master's program in Statistics and Computer Sciences at Karadeniz Technical University, completing her master's thesis, titled "Stance Classification for Fake News Detection in Social Media," in 2023.

Since 2012, she has demonstrated her versatility across various sectors and countries. Initially starting as a Game Developer until 2014, she transitioned to the role of a Delphi Developer in the government sector from 2014 to 2017. 2017-2018, she served as a Web Development Team Lead at LZEEZA, providing pivotal support to development teams and contributing to a 30% increase in efficiency through the design and implementation of a web portal. 2018-2020 served as an ERP Team Leader at Newline, where she constructed an HR System based on the open-source application ERPNext. Transitioning to the role of an ERP System Consultant at SIR from 2020 to 2022, she led the development of an ERP system implemented for companies in Saudi Arabia and Kuwait. 2020-2023 served as a Product Manager at STME, offering hands-on leadership and analysis for over 300 programming requests. Additionally, from July 2022 to January 2023, she contributed as an ERP Technical Consultant at PwC, providing valuable insights in the development and structuring of an ERP system for the King Salman Bin Abdulaziz Royal Reserve Development Authority in Saudi Arabia. She excels in English, is very good in Turkish, have a good command of French, and is a native Arabic speaker.

ALSAFADI, M. (2023). Stance Classification for Fake News Detection with Machine Learning. *The Eurasia Proceedings of Science Technology Engineering and Mathematics, 22, 191-198*. https://doi.org/10.55549/epstem.1344457.