Investigating the Impact of Preprocessing Techniques and Representation Models on Arabic Text Classification using Machine Learning

Mahmoud Masadeh[1¶], Moustapha.A[2¶], Sharada B[3], Hanumanthappa J.[4], Hemachandran K[5], Channabasava Chola[*6], Abdullah Y. Muaad[*7]
Computer Engineering Department, Yarmouk University, Irbid 21163, Jordan[1]
Department of Studies in Computer Science, Mysore University, Manasagangothri, Mysore 570006, India[2,3,4,7]
Department of Business Analytics, School of Business, Woxsen University, Hyderabad, India 502345[5]
Department of Electronics and Information Convergence Engineering, College of Electronics and Information, Kyung Hee University, Suwon-si 17104, Republic of Korea[6]
[¶]The authors have an equal contribution

***Abstract*—Arabic Text Classification (ATC) is a crucial step for various Natural Language Processing (NLP) applications. It emerged as a response to the exponential growth of online content like social posts and review comments. In this study, *preprocessing techniques* and *representation models* are used to evaluate the effectiveness of ATC using Machine Learning (ML). Generally, the ATC operation depends on various factors, such as stemming in preprocessing, feature extraction and selection, and the nature of the dataset. To enhance the overall classification performance, preprocessing methodologies are primarily employed to transform each Arabic term into its root form and reduce the dimensionality of representation. In the representation of Arabic text, feature extraction and selection processes are imperative, as they significantly enhance the performance of ATC. This study implements the chosen classifiers using various feature selection algorithms. The comprehensive assessment of classification outcomes is conducted by comparing various classifiers, including Multinomial Naive Bayes (MNB), Bernoulli Naive Bayes (BNB), Stochastic Gradient Descent (SGD), Support Vector Classifier (SVC), Logistic Regression (LR), and linear Support Vector Classifier (LSVC). These ML classifiers are assessed utilizing short and long Arabic text benchmark datasets called BBC Arabic corpus and the COVID-19 dataset. The assessment findings indicate that the efficacy of classification is significantly influenced by the preprocessing methods, representation model, classification algorithm, and the datasets' characteristics. In most cases, the SGDC and LSVC have consistently surpassed other classifiers for the datasets under consideration when significant features are chosen.**

Keywords*—Arabic Text Classification (ATC); Text Mining (TM); Machine Learning (ML); preprocessing methods; representation models; Feature Extraction (FE); Feature Selection (FS)

I. Introduction

Text Mining (TM), i.e., knowledge discovery, which entails extracting meaningful information from text, has gained significant attention in recent years. With the exponentially generated textual data on many social media sites, understanding and analyzing text data become increasingly complex [1]. TM is a discipline that requests the support of other scientific subjects such as statistics, machine learning (ML), natural language processing (NLP), and linear algebra [2].

One of the vital topics in TM is text classification (TC), i.e., categorization, which is a challenging computational task in the TM field [3]. TC has become a big data problem as textual data's volume, variety, and velocity increase rapidly. Furthermore, only a few models and tools can help understand and classify Arabic Text (AT). Therefore, it is compulsory to design efficient models to address AT's challenges and support decision-makers in making the right decisions in many real-life domains such as healthcare, economics, social media, and financial markets [4].

Over 477 million people speak Arabic as their first language. Furthermore, a significant percentage speak Arabic as a second language [5]. Arabic is the official language in 22 nations and the original script for Persian and Urdu [6]. The Arabic language consists of 29 letters written from right to left. The Arabic language distinguishes itself by position-dependent letter forms and shapes [7]. Arabic, unlike other languages, is founded on roots, which contributes to its complexity. Aside from that, Arabic is divided into three formats: Classical Arabic (CA), Modern Standard Arabic (MSA), and Dialectal Arabic (DA). Other AT-related issues include phonology, orthography, and morphology. So, working with AT is more complicated than other languages, such as English [7]. Thus, there is an urgent need to handle Arabic Text Classification (ATC).

Arabic Text Classification starts with preprocessing to make the text ready for further processing. Then, a group of *representation learning methods* are applied. They make the text understandable by machines and automatically find the patterns that will help to discriminate classes and achieve a classification task [2]. Once the representation for a given text collection is created, an optimal set of features is required. Therefore, Feature Extraction (FE) and Feature Selection (FS) techniques must be applied to extract and select the best features to decrease the dimension of the representation process. Subsequently, the classifier is trained to learn the pattern in the training phase (offline learning) and classify the text into different classes in the testing phase (online learning) [8].

*Corresponding authors

Text Classification (TC) is a process in which the system will assign a suitable label for each test document based on recognizing what has already been learned from the training phase [9]. The complexity of textual data made classification a challenging task. Therefore, over the past few decades, TC have been extensively researched and addressed in different applications such as mail spam filtering, document classification, web searching, and web page classification. However, today's TC information has become challenging and more significant, mainly with detection tasks such as hate speech and rumours detection, especially in the Arabic language [10].

This study examines the influence of preprocessing techniques and representation models on Arabic text classification, where six ML-based classifiers are utilized for the classification task. Therefore, this research allows developers in the profession to choose a powerful ML-based approach for robust ATC applications. The primary contributions of this study are outlined as follows:

- We investigate the effect of preprocessing, representation and feature selection techniques on Arabic Text Classification (ATC).
- We employ and evaluate many classification models with various preprocessing algorithms to ascertain their efficacy in ATC.
- We demonstrate the efficacy of preprocessing and representation methods; all ML-based classifiers are assessed using two benchmark datasets for Arabic text, specifically BBC Arabic and COVID-19.
- We verify the effectiveness of both long and short Arabic text datasets for ATC tasks.

The rest of this article is presented in the following manner. Section II explains preliminaries essential to comprehend the context of the work. Section III exposes some literature review on preprocessing and representation for ATC. Section IV demonstrates the proposed model of ATC strategy. SectionV explores experimental results, examines a comparative analysis, and deliberates on the outcomes of the experiments. The conclusion of this work is presented in Section VII.

II. Preliminaries

A. Natural Language Processing (NLP)

It is an important aspect of Artificial Intelligence (AI). It allows programs and machines to analyze and understand human language, allowing them to carry out repetitious exercises without human intervention. Several domains have developed the basics of NLP, such as artificial intelligence, computer and information sciences, linguistics, electronic and electrical engineering, robotics, math, and psychology. Appliances can analyze and learn human language through a technique known as NLP [11]. NLP-based techniques manipulate a substantial portion of data to fetch valuable information/knowledge. Therefore, various data mining and machine learning strategies are used. Thus, text pre-processing should be involved to prepare the text for other processing, e.g., representation features engineering that is mandated to extract features and hand it to ML techniques [12]. For example, pre-processing could incorporate text tokenization and stop-word removal. Recently, Arabic NLP has emerged as a nascent research domain. It encompasses the evolution of approaches and tools using the Arabic language. However, it faces multiple complex concerns associated with the form and nature of the Arabic language.

B. Machine Learning (ML) Algorithms

ML is integrated into various areas, including healthcare [13], hardware design [14] [15], quality control [16], and NLP [12], with this study focusing on the latter. Information is a systematic collection of discrete facts that suppresses the complete range of typical patterns. The primary purpose of machines is to find rituals that remind them of a specific occasion. If the system recognizes these patterns, machine learning has occurred. The authors of [17] stated that advances in machine learning, particularly deep learning, allow us to develop algorithms that utilize real-world data to produce conclusions that look subjective. There are several approaches for preparing text for subsequent processing, as demonstrated in Section IV-C. Text tokenization, also known as text segmentation or linguistic analysis, divides the text into smaller units called tokens, which can be words, characters, or subwords. The most typical method of creating tokens is based on space. Articles (e.g., a, an, the), conjunctions (e.g., and, but, if), and prepositions (e.g., in, at, on) [18] are stop words that do not communicate a clear meaning. As a result, they must be removed. In ML, features are numerical properties. However, in certain cases, such as sentiment analysis, the data may not include numerical qualities. Thus, many forms of features (e.g., word and character) are translated into numerical features, and selecting from them to make ML operate adequately is referred to as feature engineering (*feature selection and feature extraction*).

C. Text Classification

The number of available complex text documents and the size of the text have just grown exponentially. This mandates a more profound understanding of machine learning techniques to accurately categorize texts in various applications. ML models achieve successful results in NLP since they rely on their ability to comprehend complicated prototypes and non-linear associations within data.

The authors of [19] evaluated the effect of the preprocessing schemes on classification success, considering e-mail and news domains, for Turkish and English. They discovered that selecting suitable combinations of preprocessing tasks, rather than including or excluding them all, may supply substantial enhancement in classification accuracy depending on the target domain and language. The authors of [20] discussed a short recap of text classification algorithms including text feature extractions, dimensionality compaction techniques, existing algorithms, and evaluation methods. The authors of [21] indicated that DL–based models have exceeded classical ML-based techniques in different text classification studies. They introduced an exhaustive study of more than 150 DL–based models for text classification and examined their technical contributions, similarities, and strengths.

D. Evaluation Metrics

The efficiency of the proposed models is assessed in terms of accuracy, precision, recall, and F1-Score. As presented in

Eq. 1, **accuracy** represents the number of correctly categorized data instances over the whole number of data samples. For an unbalanced dataset, the positive and negative classes have a diverse number of samples. Thus, the accuracy is not appropriate for assessing the model and other metrics are required. **Precision**, i.e., positive predictive value, describes how many of the precisely foreseen cases were positive. As represented in Eq. 2, it should be 1 for an ideal classifier where FP is zero. **Recall**, i.e., sensitivity or true positive rate, is represented in Eq. 3.

Recall for a label is represented as the number of true positives divided by the total number of confirmed positives. For an excellent classifier, recall should be 1 where FN is zero. For a perfect classifier, both precision and recall are 1. **F1-score** is a measure that relies on both precision and recall as represented in Eq. 4. F1-score is 1 when both precision and recall are 1. So, the F1-score is the harmonic mean of precision and recall and it is a more reasonable measure than accuracy [22].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall/Sensitivity = \frac{TP}{TP + FN} \tag{3}$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

The above equations can be used with only the binary classification problem. For Multi-Class, the earlier formulas for Precision and Recall might not seamlessly apply. We have to calculate the per-class values of precision, recall, and F1 score, where we have seven classes for the BBC dataset and three classes for the COVID-19 dataset. However, rather than having multiple per-class precision, recall, and F1-score, it would be more suitable to average them to get a single number to represent overall performance. For that, we will use macro average and weighted average.

Macro averaging is the most direct among the considerable averaging methods. The macro-averaged of a specific score is calculated by taking the arithmetic mean of all the per-class scores. Thus, treats all classes equally regardless of their support values. Support indicates the number of real occurrences of the class in the dataset. For example, the BBC dataset contains 2,356 documents of class *Middle East News* while containing only 49 documents of class *International Press*. The *weighted-average* of a specific metric, e.g., F1-score, is computed by considering the mean of all per-class F1 scores while taking each class's support. The *weight* indicates the proportion of each class's contribution comparable to the sum of all support values.

III. Literature Review

This section emphasizes related works on preprocessing and representation in Arabic text classification using machine learning classifiers. The first step in developing ATCs is preprocessing, which includes cleaning text, simplifying computations, and preparing the dataset for further processing [10] [23]. Text preprocessing is essential for preparing it for representation, given that text data is inherently unorganized. Such unstructured text is not amenable to subsequent analysis without a pre-established data model. As a result, specialized preprocessing techniques are necessary to condition the text for further use. Reducing the size of the text and removing extraneous elements could either enhance or detract from its performance [24]. As a second step, converting unstructured text into structured data is called the representation process since machines can only process structured information, whereas AT is unstructured [9] [2].

Reducing the dimensionality of the data is not a compulsory step in ATC but it's often essential. This is due to the enormous number of text features that can generate large and sparse matrices that adversely affect the efficiency of ATCs. Some of these features may add noise, making it difficult to classify different categories accurately. Various methods can be used at this stage, such as Feature Extraction (FE), Feature Selection (FS), and Principal Component Analysis (PCA), among other multivariate techniques that improve performance. Dimensionality can also be lowered in preprocessing through approaches like stemming and lemmatization. It's worth noting that many scholars have used Term Frequency-Inverse Document Frequency (TF-IDF) and Bag-of-Words (BoW) methods, which have three main drawbacks: they create sparse matrices, lose the sequence of words, and neglect semantic context. Thus, leading to identical representations for different sentences. To overcome these constraints, various feature selection methods like Information Gain (IG), Mutual Information (MI), and Chi-square (CHI) have been suggested to identify the best features and reduce dimensionality [10] [25].

IV. Proposed Methodology

Fig. 1 introduces an Arabic text classification framework using varied preprocessing and representation methods like BoW and TF-IDF. Diverse classifiers are applied, including Multinomial NB, Bernoulli NB, LR, Stochastic Gradient Descent (SGD) Classifier, SVC, and Linear SVC. The study's outcomes highlight the impact of these techniques on enhancing ATC system performance. As depicted in Fig. 1, the system's workflow involves pre-processing, representation, feature selection, and classification algorithms. Initial steps include stop word removal, normalization, and stemming to reduce dimensions. The resulting text undergoes TF-IDF and BoW processes, producing a matrix as input. Machine learning employs this matrix, where the data is divided into 80% for training and 20% for testing.

A. The Proposed Architecture

The proposed method is divided into four phases. In the first stage, dataset preparation and splitting are carried out. In these data, preprocessing steps, such as normalization,

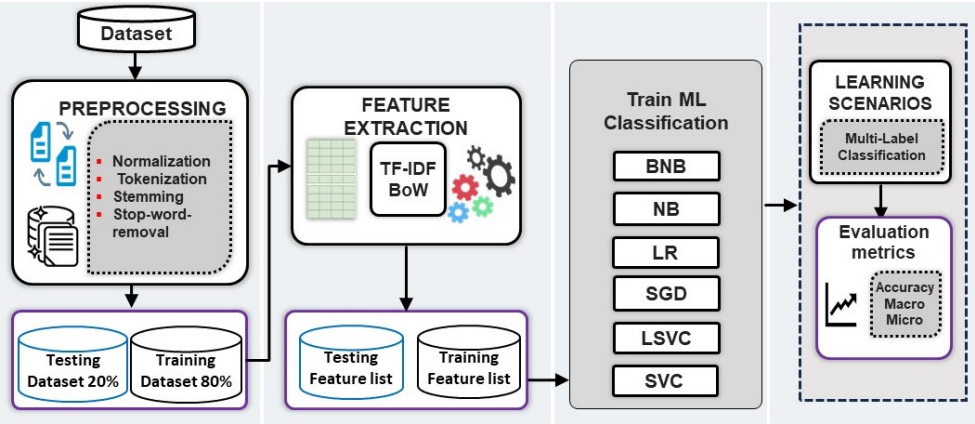

Fig. 1. Schematic representation of the proposed method.

scaling, and missing values handling, are applied to ensure the data can be used for machine learning. Representation is the second phase, where raw data is transformed into a format that ML algorithms can process. Then, classification, several classifiers are used to categorize text into one or more different classes. Finally, assessment metrics are used to examine the performance and efficacy of a statistical or machine-learning model using quantitative measurements. Fig. 1 depicts the suggested method's design.

B. DataSet

We selected two Arabic datasets, BBC Arabic and COVID-19, to experimentally evaluate six classifiers based on preprocessing and representation. Performance metrics are analyzed to assess classifier effectiveness [26]. The Arabic BBC corpus dataset comprises 4,763 Arabic documents classified into seven classes. Document distribution per class is Middle East News (2,356), world news (1,489), business and economics (296), sports (219), international press (49), science and technology (232), and art and culture (122) [26]. The corpus contains around 1.8 million words and 106,733 distinct keywords, summarised in Table I. Arabic comments related to COVID-19 are classified using an additional dataset of short texts. These comments are analyzed, and the data distribution of this COVID-19 dataset is presented in Table II which summarizes the details of each benchmark dataset. These datasets belong to two applications (long text and short text). We divided each dataset into training and validation test sets with a ratio of 80% and 20%, according to the Pareto principle [27].

TABLE I. BBC DATASET DISTRIBUTION

No.	Class Type	No. of Documents
1	Middle East News	2,356
2	Science and Technology	232
3	International Press	49
4	Art and Culture	122
5	Sports	219
6	Business and Economy	296
7	World News	1,489

TABLE II. COVID-19 DATASET DISTRIBUTION

No.	Class Type	No. of Documents
1	Positive	7,962
2	Negative	635
3	Natural	1,391

C. Preprocessing

Preprocessing involves transforming raw data into a suitable input format for ML models. It qualifies text by converting it into a convenient form for document classification, reducing complexity. This phase removes non-significant characters, stop words, and punctuation in Arabic text. Preprocessing steps include tokenization, normalization, stop word removal, and stemming.

1) Tokenization: Tokenization involves breaking text into tokens and converting words into numbers. The acquired segments can be single items (1-gram) or a sequence of n words (n-gram). In Arabic, sentences are divided using signals like commas, quotes, and spaces. Tokens can be individual words, irrespective of meaning or relationships

2) Normalization: Normalization involves converting text letters to a canonical form or removing diacritics, such as changing ة ت هـ [23].

3) Stop Word Elimination: For a work that targets Arabic text, the first pre-processing step is the elimination of non-Arabic text. Thus, a whitespace character is used instead of each non-Arabic character. Then, we deduct *stopwords*, which occur often in the text and are insignificant for text classification, e.g., هو ، أو ، منذ ، على ، هناك. A list of the most often used Arabic stop words is accessible at [18]. Stopwords account for around 20%-30% of a document's exhaustive words. These terms can be deleted since they are repetitive [28]. The basic approach for extracting stopwords is static, meaning it uses a pre-filled list of all words that are semantically irrelevant to a specific language. For the dynamic approach, stopwords are recognized online rather than previously established, and attributes are given depending on their relevance. In this effort, analogous to the removal of

stopwords, we removed punctuation and numerals from the Arabic text.

4) Stemming: Stemming involves removing prefixes, suffixes, and definite articles from words to reach their root form. This process, like root, light, and hybrid stemming, aims to simplify words for analysis. As in the shown example, the words in the sentence are given in their root form.

D. Text Representation/Feature Engineering

Text representation involves transforming unstructured text into manageable forms. Various text feature representation methods exist, each with distinct traits. The initial dataset includes a group of documents with many classes as expressed in Eq. 5. The frequency (TF) representation technique is the average occurrence within a specific topic divided by the occurrence rate.

$$AD = d_1, d_2, d_3, ..., d_n \tag{5}$$

$$tf(t,d) = \frac{f_d(t)}{Max_{w \in d} \ \ f_d(w)} \tag{6}$$

$$idf(t,D) = Ln[\frac{|D|}{|\{d \in D; t \in d\}|}] \tag{7}$$

$$tf - idf(t,d,D) = tf(t,d).idf(t,D) \tag{8}$$

Where $f_d(t)$is the rate of the term '*t*' in the document '*d*', and '*w*' is a set of words in the document '*d*' while '*D*' is the corpus of documents. The TF-IDF model combines Term Frequency (TF) with Inverse Document Frequency (IDF) to measure the importance of a term in a document relative to a corpus. IDF indicates how common or rare a term is across all documents and is calculated using a logarithmic formula, as shown in Eq. 7. The final TF-IDF model is presented in Eq. 8.

In Text Classification (TC), representation and feature extraction processes often lead to many terms, causing issues like the "*curse of dimensionality*" This is due to the inherent characteristics of textual data, like noise, redundancy, and sparseness. To address this, selecting an efficient Feature Selection (FS) technique is crucial. The Chi-Square (C.H.I.) method is commonly used to choose relevant features in various works.

E. Arabic Text Classification

TC is the task of deciding whether a piece of text belongs to prescribed classes based on understanding and discriminating the pattern of the text [29]. It is a significant task involved in the TC system [3] [30]. TC is one of the most challenging computational tasks in the ML community. Only a few researchers worked in ATC, and numerous classifier learning algorithms have been used, such as Naive Bayes (NB) [31], Support Vector Machine (SVM) [4] and Artificial Neural Network (ANN) [32].

1) Bag of Words (BoW): BoW is a textual presentation style suitable for classification models in which the text is treated as a collection of words, regardless of syntax. BoW shows whether or not a certain word occurs in the document, with no regard for word order. The performance of the various ML algorithms we developed using BoW was unsatisfactory owing to the loss of semantic and syntactic information between phrases. To boost performance, we used different representation schemes that can accept semantic and syntactic differences.

2) Term Frequency—Inverse Document Frequency (TFIDF): Term Frequency (TF) is a popular textual presentation approach that is equivalent to the BoW technique. The term's recurrence in a given text is what determines TF, whereas its existence determines BoW. Eq. 9 represents the frequency of any word in the supplied document. However, typical terms included in all documents, such as articles, conjunctions, and prepositions, receive a low rating since they add little to the content. Thus, we employ Inverse Document Frequency (IDF) to reduce the value of terms that appear frequently in the document collection while increasing the significance of phrases that appear infrequently. IDF is consistent across corpora and determines the proportion of papers that have a certain phrase. Eq. 10 describes how it is assessed. TF-IDF is a statistical criterion for determining how closely a phrase is associated with a document in a collection of manuscripts, known as a corpus. TF-IDF is calculated by multiplying TF and IDF. TFIDF is a straightforward technique to text categorization. Thus, the TF-IDF is created during model training and subsequently applied to the test set.

$$TF = \frac{Number\ of\ times\ a\ term\ appear\ in\ the\ document}{Total\ number\ of\ terms\ in\ the\ document} \tag{9}$$

$$IDF = Log_{10} \frac{Total\ number\ of\ Documents}{Number\ of\ documents\ that\ includes\ the\ term} \tag{10}$$

F. Building of Classification Models (Learning)

1) Multinomial Naive Bayes (MNB): The naive Bayes classifier is based on Bayes Theorem, which operates on conditional probability. The conditional probability is the likelihood that something will happen if something else has already happened [15]. Eq. 11 provides the formula for computing conditional probability, with *A* representing the hypothesis and *B* representing the evidence.

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)} \tag{11}$$

NB computes numerous model parameters for a given set of labeled training data, including the likelihood of each class label happening. Then, estimate the class for each set of test data based on its chance of being assigned to different classes. The expected class is determined by the largest likelihood [33]. Multinomial Naive Bayes (MNB) is a prominent supervised learning classification approach used to analyze categorical text data. It is a probabilistic learning strategy that is widely used

in natural language processing (NLP). To anticipate a text's tag, the algorithm applies the Bayes principle. It calculates the likelihood of each tag for a given sample and presents the tag with the highest probability as output.

2) Bernoulli Naive Bayes (BNB): It is a subset of the Naive Bayes Algorithm that is used to classify binary features such as '1' or '0' that are independent of one another. Bernoulli Naive Bayes is used to identify spam, classify text, do sentiment analysis, and determine whether a given word occurs in a document. Eq. 12 expresses the decision rule of Bernoulli NB, where $P(x_i|y)$ is the conditional probability of x_i happening if y has occurred, i is the event, and x_i has a binary value of 0 or 1.

$$P(x_i|y) = P(x_i = 1|y)x_i + (1 - P(x_i = 1|y))(1 - x_i) \quad (12)$$

3) Stochastic Gradient Descent (SGD): Gradient descent is an optimization approach for training machine learning models and neural networks. The training data allows these models to learn over time, and the cost function in gradient descent quantifies accuracy with each parameter update, i.e., direction and learning rate. The model will keep modifying its parameters to get the minimal feasible error. Stochastic gradient descent (SGD) handles one training epoch per example and changes its parameters one at a time. SGDs are easy to retain in memory since they only require one training sample. Its regular updates assist in avoiding the local minimum and discovering the global one.

4) Support Vector Classifier (SVC): It is a specific implementation of the Support Vector Machine (SVM) algorithm developed for classification. It aims to discover the n-dimensional hyperplanes that sufficiently divide the data instances into various types.

5) Logistic Regression (LR): Regression modeling is a well-known and reliable statistical approach for analyzing and describing the relationship between a dependent variable and a group of independent predictors. Logistic regression is a specific case in regression modeling in which the result is binary. A logistic function, also known as a *sigmoid function*, may be used to explain logistic regression. This function receives any actual input x and returns a probability value between 0 and 1 as defined in Eq. 13.

$$f(x) = \frac{1}{1 + e^{-x}} \quad (13)$$

6) Linear Support Vector Classifier (LSVC): The Linear Support Vector Classifier (SVC) approach uses a linear kernel function to complete classification and it is more suitable than SVC for large datasets.

V. Experimental Results and Comparative Analysis

The machine learning algorithms have been implemented using Python version 3.8.0 in the Anaconda environment, specifically within the Jupyter Notebook. Python machine-learning libraries such as NLTK, pandas, and sci-kit-learn are utilized to evaluate the proposed methods' performance. The findings and discussions related to the various methods employed are delineated in the following sections. We used the sci-kit-learn library, which contains ML algorithms for the experimentation. To evaluate the effect of the preprocessing and representation on classification, we conducted several experiments on the BBC Arabic dataset and COVID-19 and analyzed the performance results of various classifiers. Additionally, the proposed is evaluated using classifications generated from the baseline model. We summarise all experimental results and compare the proposed method with other methods.

In general, the works provide a comprehensive summary of the performance results for two datasets with preprocessing and without feature selection and vice versa, allowing for comparing different transformation methods based on various evaluation metrics. We found that pre-processing, representation of AT, and feature engineering techniques played an essential role in enhancing the performance of ATC. In the beginning, pre-processing can affect ATC's performance. In addition, there were other techniques of representation, such as BoW and TF-IDF, which have some drawbacks, like missing the order of the words and losing the meaning of the words. Pre-processing techniques, such as stop word removal, can be used to reduce the dimension, but in some cases, pre-processing can positively or negatively affect the performance. Finally, we know accuracy is insufficient to evaluate ATCs, so we extend our investigation using different evaluation metrics. We summarise our findings in the conclusion section after the results and discussions.

A. Results on COVID-19 Dataset

This section discusses the experimental result of the COVID-19 dataset. It evaluates the model's performance and studies its effects on ATC regarding Accuracy (ACC), Precision (PR), Recall (RE), and F1-score. We discuss and evaluate the experimental result based on different methods.

First, we applied six classification algorithms on the COVID-19 dataset for ATC without pre-processing and without feature selection. Table III shows the evaluation metrics for this scenario, where the accuracy ranges from 81% to 83%. Regarding the macro metrics, the precision ranges from 52% to 69% while the recall ranges from 34% to 53%. F1-score ranges from 31% to 57%. Generally, the weighted metrics show a better performance since the contribution of each class is considered where the precision ranges from 75% to 80% while the recall ranges from 81% to 83%. Weighted F1-score ranges from 73% to 80%. Generally, SGDC and LSVC have the best performance among the used classifiers.

TABLE III. Evaluation metrics for COVID-19 Dataset **without** Preprocessing and **without** Feature Selection

Transformation Method	Acc	Macro			Weighted		
		PR	RE	F1-score	PR	RE	F1-score
NBC	81	52	34	31	75	81	73
BNBC	81	52	34	31	75	81	73
LRC	82	**69**	43	47	79	82	78
SGDC	**83**	68	51	55	**80**	**83**	**80**
SVC	82	67	42	45	78	82	77
LSVC	82	66	**53**	**57**	**80**	82	**80**

Table IV shows the evaluation metrics for classification algorithms on the COVID-19 dataset with pre-processing but without feature selection. The LSVC has the highest accuracy of 83%. The NBC and BNBC have the lowest accuracy of

81%. Also, they have the lowest metrics with a precision of 52%, recall of 34%, and F1-Score of 31% for macro metrics. Their weighted metrics are minimal, i.e., precision, recall, and F1-Score are 75%, 81%, and 73%, respectively. The LSCV almost achieves the best performance indicated by both macro and weighted metrics.

TABLE IV. EVALUATION METRICS FOR COVID-19 DATASET **WITH** PREPROCESSING AND **WITHOUT** FEATURE SELECTION

Transformation Method	Acc	Macro			Weighted		
		PR	RE	F1-score	PR	RE	F1-score
NBC	81	52	34	31	75	81	73
BNBC	81	52	34	31	75	81	73
LRC	82	**70**	43	47	79	82	77
SGDC	82	68	50	55	**80**	82	80
SVC	82	67	42	45	78	82	77
LSVC	**83**	67	**52**	**57**	**80**	**83**	**81**

As shown in Table V, five classifiers have an 82% accuracy when the COVID-19 dataset is evaluated without preprocessing but with feature selection. Regarding the macro metrics, the precision ranges from 59% to 77% while the recall ranges from 23% to 52%. F1-score ranges from 35% to 54%. For the weighted metrics, the precision ranges from 77% to 81% while the recall is either 78% or 82%. Weighted F1-score ranges from 74% to 78%. There is no single classifier that is the best in most metrics, where the best classifiers are BNBC, SGDC, and LSVC. As with the previous two scenarios, the weighted metrics are higher than the macro ones because the contribution/weight of each class is considered.

TABLE V. EVALUATION METRICS FOR COVID-19 DATASET **WITHOUT** PREPROCESSING AND **WITH** FEATURE SELECTION

Transformation Method	Acc	Macro			Weighted		
		PR	RE	F1-score	PR	RE	F1-score
NBC	**82**	74	23	35	79	**82**	74
BNBC	78	59	**52**	**54**	79	78	**78**
LRC	**82**	71	42	44	79	**82**	77
SGDC	**82**	**77**	41	43	**81**	**82**	76
SVC	**82**	60	45	48	77	**82**	77
LSVC	**82**	66	46	51	78	**82**	**78**

The last scenario is when the COVID-19 dataset is classified and the metrics evaluated with data preprocessing and with feature selection as given in Table VI. The accuracy ranges from 79% to 82%. Regarding the macro metrics, the precision ranges from 55% to 81% while the recall ranges between 37% and 52%. The minimum F1-score is 36% and the maximum is 53%. When the metrics are evaluated with the weight of classes, the precision ranges between 77% and 81% while the recall is either 78% or 82%. The minimum F1-score is 74% and the maximum is 78%. The best classifiers are BNBC, SGDC, and LSVC.

From the Tables III, IV, V, and VI we notice that the weighted metrics are always better than the macro metrics. Also, there is no specific classifier that is the best in the seven metrics, i.e., accuracy, macro and weighted (precision, recall, and F1-score). In most of the cases, SGDC and LSVC were the best.

B. Results on BBC Dataset

We extended our experimentations in this section by studying the performance of long ATC using the BBC dataset. The study evaluates six classification algorithms on an extended BBC Arabic dataset for Arabic text classification (ATC).

TABLE VI. EVALUATION METRICS FOR COVID-19 DATASET **WITH** PREPROCESSING AND **WITH** FEATURE SELECTION

Transformation Method	Acc	Macro			Weighted		
		PR	RE	F1-score	PR	RE	F1-score
NBC	**82**	**81**	37	36	**81**	**82**	74
BNBC	79	55	**52**	**53**	77	79	**78**
LRC	**82**	78	39	41	80	**82**	76
SGDC	**82**	78	38	39	**81**	**82**	75
SVC	81	63	40	42	77	81	76
LSVC	**82**	68	42	45	79	**82**	77

Table VII shows the evaluation metrics for the classification of the BBC dataset for ATC without pre-processing and without feature selection. The accuracy ranges from 68% to 93%, where the accuracy of SGDC is 93% and the accuracy of LSVC is 92%. Regarding the macro metrics, the precision ranges from 78% to 94% where SVC has the maximum value and both SGDC and LSVC have a value of 93%. The maximum recall is 89% while the minimum is 34%. The F1-score ranges from 38% and 91% where the SGDC has the maximum value while LSVC has a score of 90%. For the weighted metrics, the SGDC has the best precision, recall, and F1-score with a value of 93% for all of them, while LSVC has a value of 92% for the three metrics. Thus, the best classifier is the SGDC followed by LSVC while both NBC and BNBC show the minimum metrics.

TABLE VII. EVALUATION METRICS FOR BBC DATASET **WITHOUT** PREPROCESSING AND **WITHOUT** FEATURE SELECTION

Transformation Method	Acc	Macro			Weighted		
		PR	RE	F1-score	PR	RE	F1-score
NBC	68	78	34	38	76	68	63
BNBC	68	78	34	38	76	68	63
LRC	86	80	62	68	86	86	85
SGDC	**93**	93	**89**	**91**	**93**	**93**	**93**
SVC	87	**94**	68	76	88	87	86
LSVC	92	93	88	90	92	92	92

The metrics for ATC for the BBC dataset with preprocessing and without feature selection are shown in Table VIII. There is a great similarity with the values reported in Table VII. The best classifier is the SGDC followed by LSVC, while both NBC and BNBC show the minimum metrics.

TABLE VIII. EVALUATION METRICS FOR BBC DATASET **WITH** PREPROCESSING AND **WITHOUT** FEATURE SELECTION

Transformation Method	Acc	Macro			Weighted		
		PR	RE	F1-score	PR	RE	F1-score
NBC	69	78	35	39	77	69	64
BNBC	69	78	35	39	77	69	64
LRC	86	80	61	67	86	86	85
SGDC	**93**	**94**	**89**	**91**	**93**	**93**	**93**
SVC	87	**94**	69	77	88	87	86
LSVC	92	93	88	90	92	92	92

The results of classification without preprocessing but with feature selection are shown in Table IX. NBC has a minimal accuracy of 57% while the SVC has a maximum accuracy of 97%. Both BNBC and SGDC have an accuracy of 96% while the LSVC has a 95% accuracy. For the metrics that are evaluated at the macro level, the precision ranges from 50%

to 96%, where both SGDC and LSVC have a value of 96% and the SVC has a precision of 95%. The macro recall ranges from 23% to 96%, while the F1-score is between 24% and 95%. The weighted precision ranges from 66% to 98%. The SVC has the maximum precision while both BNBC and SGDC have a precision of 96%. The weighted recall ranges from 57% and 97%. Both BNBC and SGDC have a precision of 96% and NBC has the minimal value. The range of the F1-score is similar to the precision where the SVC has the maximum of 97% followed by 96% for both BNBC and SGDC, while NBC has the lowest F1-score. The worst classifier regarding all metrics is NBC while the best classifier is the SVC, followed by SGDC and LSVC.

TABLE IX. EVALUATION METRICS FOR BBC DATASET **WITHOUT** PREPROCESSING AND **WITH** FEATURE SELECTION

Transformation Method	Acc	Macro			Weighted		
		PR	RE	F1-score	PR	RE	F1-score
NBC	57	50	23	24	66	57	47
BNBC	96	94	88	88	96	96	96
LRC	71	78	42	48	78	71	67
SGDC	96	**96**	91	93	96	96	96
SVC	**97**	95	**96**	**95**	**98**	**97**	**97**
LSVC	95	**96**	89	92	95	95	95

The metrics for BBC dataset classification with preprocessing and with feature selection are shown in Table X. Similar to Table IX, the best classifier is SVC followed by SGDC, LSVC, and BNBC. The minimal metrics are obtained by NBC.

We notice that without feature selection, as shown in Table VII and VIII, the best classifier is SGDC followed by LSVC. However, with feature selection, as shown in Table IX and X, the best classifier is SVC followed by SGDC.

TABLE X. EVALUATION METRICS FOR BBC DATASET **WITH** PREPROCESSING AND **WITH** FEATURE SELECTION

Transformation Method	Acc	Macro			Weighted		
		PR	RE	F1-score	PR	RE	F1-score
NBC	49	07	14	09	24	49	32
BNBC	95	92	87	87	96	95	95
LRC	78	38	34	35	71	78	72
SGDC	96	**97**	89	92	96	96	96
SVC	**98**	96	**96**	**96**	**99**	**98**	**98**
LSVC	94	**97**	84	88	95	94	94

VI. DISCUSSION AND RESULTS

This section discusses the effectiveness of preprocessing and feature selection on short and long Arabic Text classification. The authors of [34] evaluated representation and preprocessing on a short text dataset and indicated that the effectiveness of preprocessing is positive in the case with Bow and negative in others with TFID. Still, LRC and SVC generally offered the best performance across most metrics. The authors found that as the number of characteristics increased, so did the execution time. In the meanwhile, the classifiers' performance remains unchanged. After increasing the number of characteristics from 7,000 to 10,000, all classifiers except SVC maintained their accuracy.

The experimentation evaluates the performance of various models on short data for ATC without any preprocessing or feature selection. We observed under these conditions that all models achieved accuracy scores between 81% and 83%. Among them, LRC had the best macro accuracy, while LSVC had the highest macro recall and F1-score. When weighted criteria like precision, recall, and F1-Score were considered, SGDC and LSVC emerged as standouts.

When introducing preprocessing, but still without feature selection, LSVC consistently outperformed the other models across almost all metrics. On the other hand, NBC and BNBC demonstrated the lowest performance.

The experimental outcomes underscore that preprocessing had mixed effects on model results while it improved performance for some models, it hindered the performance of others. LSVC and SGDC were the top-performing models across various scenarios, while NBC and BNBC lagged. When we continued our experimentation on the BBC dataset, in the initial experiment, which focused on the dataset without preprocessing and feature selection, the SGDc and LSVC with TF-IDF were the top performers. Specifically, SGDc and LSVC achieved the highest accuracy, with 93% and 92%, respectively. In terms of macro-precision, both algorithms scored 93%. For macro-recall and macro-F-score, the top scores were 89% and 91%, respectively. In weighted metrics, SGDC outperformed with 93% across all metrics, followed closely by LSVC at 92%.

The subsequent experiment, which incorporated preprocessing but left out feature selection, affirmed the superiority of the SGDC and LSVC classifiers for the extended BBC dataset. The third experiment did not involve preprocessing while including feature selection. The SVC stood out by almost securing the highest values. Notably, the NBC and LRS underperformed compared to the others in this context. The final experiment, including preprocessing and feature selection, yielded comparable results to the third experiment. The variations among metrics were minimal, between 1% and 3%. SVC was a standout again. Many models performed exceptionally well in weighted metrics, such as SGDC and LSVC. However, as in the third experiment, NBC and LRC lagged the rest.

In summary, when working with the BBC dataset for Arabic text classification, the SGDC and LSVC consistently demonstrate high performance, especially without preprocessing and feature selection. However, with preprocessing and feature selection, SVC tends to be the best performer, while NBC and LRC trail behind their counterparts. The experiment's result demonstrated that the preprocessing and the feature selection impact the text classification performance, and the dataset type (short/long) also weighs heavily on the performance.

VII. CONCLUSION

In categorizing Arabic text, a complete study investigation was undertaken to demonstrate the usefulness of pre-processing, feature extraction, feature selection, and the dataset's features. Numerous ML models have been introduced to underscore the efficacy of diverse techniques in classifying Arabic text. The study's results indicate that many models influence the accuracy of system performance in ATC. The experimentation indicates that representation, encompassing feature extraction and feature selection, is essential in ATC.

Simultaneously, the preprocessing, classification algorithm and the dataset's characteristics influence the classification performance's efficacy. The findings clearly illustrate the benefits of the feature representation method and its impact on text classification efficacy. Based on the analysis presented in this article, which is limited to two datasets, several outstanding issues remain for future research, such as the absence of benchmark datasets, the shortage of lexicons, and the challenge of identifying techniques that address the contextual significance of ATC. The study highlights the effectiveness of pre-processing and feature representation on text classification performance. Challenges remain, like a lack of benchmark datasets and context-aware techniques for ATC; opportunities exist for enhancing tools like data augmentation and preprocessing techniques, mainly stemming.

REFERENCES

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

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] F. Sebastiani, "Machine learning in automated text categorization," *ACM computing surveys (CSUR)*, vol. 34, no. 1, pp. 1–47, 2002.

[4] A. Moh'd A Mesleh, "Chi square feature extraction based svms arabic language text categorization system," *Journal of Computer Science*, vol. 3, no. 6, pp. 430–435, 2007.

[5] H. Alsayadi, A. Abdelhamid, I. Hegazy, and Z. Taha, "Data augmentation for arabic speech recognition based on end-to-end deep learning," *International Journal of Intelligent Computing and Information Sciences*, vol. 21, no. 2, pp. 50–64, 2021.

[6] F. Sadat, F. Kazemi, and A. Farzindar, "Automatic identification of arabic dialects in social media," in *Proceedings of the first international workshop on Social media retrieval and analysis*, 2014, pp. 35–40.

[7] N. Y. Habash, *Introduction to Arabic natural language processing*. Springer Nature, 2022.

[8] H. Tavasoli, B. J. Oommen, and A. Yazidi, "On utilizing weak estimators to achieve the online classification of data streams," *Engineering Applications of Artificial Intelligence*, vol. 86, pp. 11–31, 2019.

[9] M. Suhil, "Representation and classification of text data," 2019.

[10] A. Ayedh, G. Tan, K. Alwesabi, and H. Rajeh, "The effect of preprocessing on arabic document categorization," *Algorithms*, vol. 9, no. 2, p. 27, 2016.

[11] I. Guellil, H. Saâdane, F. Azouaou, B. Gueni, and D. Nouvel, "Arabic natural language processing: An overview," *Journal of King Saud University-Computer and Information Sciences*, vol. 33, no. 5, pp. 497–507, 2021.

[12] S. L. Marie-Sainte, N. Alalyani, S. Alotaibi, S. Ghouzali, and I. Abunadi, "Arabic natural language processing and machine learning-based systems," *IEEE Access*, vol. 7, pp. 7011–7020, 2018.

[13] M. Masadeh, A. Masadeh, O. Alshorman, F. Khasawneh, and M. Masadeh, "An efficient machine learning-based covid-19 identification utilizing chest x-ray images," *IAES International Journal of Artificial Intelligence*, pp. 356–366, 2022.

[14] M. Masadeh, O. Hasan, and S. Tahar, "Machine-learning-based self-tunable design of approximate computing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 4, pp. 800–813, 2021.

[15] M. Masadeh, A. Aoun, O. Hasan, and S. Tahar, "Decision tree-based adaptive approximate accelerators for enhanced quality," in *International Systems Conference (SysCon)*. IEEE, 2020, pp. 1–5.

[16] M. Masadeh, O. Hasan, and S. Tahar, "Machine learning-based self-compensating approximate computing," in *2020 IEEE International Systems Conference (SysCon)*. IEEE, pp. 1–6.

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[18] R. NL, "Stopword lists," 2024, https://www.ranks.nl/stopwords/arabic [Accessed: 17.01.2024].

[19] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Information processing & management*, vol. 50, no. 1, pp. 104–112, 2014.

[20] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," *Information*, vol. 10, no. 4, p. 150, 2019.

[21] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning–based text classification: a comprehensive review," *ACM computing surveys (CSUR)*, vol. 54, no. 3, pp. 1–40, 2021.

[22] M. Masadeh, H. J. Davanager, and A. Y. Muaad, "A novel machine learning-based framework for detecting religious arabic hatred speech in social networks," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 9, 2022.

[23] A. Y. Muaad, H. Jayappa, M. A. Al-antari, and S. Lee, "Arcar: a novel deep learning computer-aided recognition for character-level arabic text representation and recognition," *Algorithms*, vol. 14, no. 7, p. 216, 2021.

[24] Y. Albalawi, J. Buckley, and N. S. Nikolov, "Investigating the impact of pre-processing techniques and pre-trained word embeddings in detecting arabic health information on social media," *Journal of big Data*, vol. 8, no. 1, p. 95, 2021.

[25] Y. A. Alhaj, J. Xiang, D. Zhao, M. A. Al-Qaness, M. Abd Elaziz, and A. Dahou, "A study of the effects of stemming strategies on arabic document classification," *IEEE access*, vol. 7, pp. 32 664–32 671, 2019.

[26] M. K. Saad and W. Ashour, "Osac: Open source arabic corpora," in *6th ArchEng Int. Symposiums, EEECS*, vol. 10, 2010, p. 55.

[27] H. Benjamin and S. Sotardi, "The pareto principle," *J Am College Radiol*, vol. 15, no. 6, p. 931, 2018.

[28] T. Kanan, B. Hawashin, S. Alzubi, E. Almaita, A. Alkhatib, K. A. Maria, and M. Elbes, "Improving arabic text classification using p-stemmer," *Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science)*, vol. 15, no. 3, pp. 404–411, 2022.

[29] G. Kanaan, R. Al-Shalabi, S. Ghwanmeh, and H. Al-Ma'adeed, "A comparison of text-classification techniques applied to arabic text," *Journal of the American society for information science and Technology*, vol. 60, no. 9, pp. 1836–1844, 2009.

[30] A. Hotho, A. Nürnberger, and G. Paaß, "A brief survey of text mining," *Journal for Language Technology and Computational Linguistics*, vol. 20, no. 1, pp. 19–62, 2005.

[31] S. Ganjavi, P. Georgiou, and S. Narayanan, "A transcription scheme for languages employing the arabic script motivated by speech processing applications," in *Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages*, 2004, pp. 59–65.

[32] F. Harrag and E. El-Qawasmah, "Neural network for arabic text classification," in *Second International Conference on the Applications of Digital Information and Web Technologies*. IEEE, 2009, pp. 778–783.

[33] M. K. A. Aljero and N. Dimililer, "A Novel Stacked Ensemble for Hate Speech Recognition," *Applied Sciences*, vol. 11, no. 24, p. 11684, 2021.

[34] A. Y. Muaad, H. J. Davanagere, D. Guru, J. B. Benifa, C. Chola, H. AlSalman, A. H. Gumaei, and M. A. Al-antari, "Arabic document classification: performance investigation of preprocessing and representation techniques," *Mathematical Problems in Engineering*, vol. 2022, pp. 1–16, 2022.