Journal of Theoretical and Applied Information Technology
15th February 2025. Vol.103. No.3
© Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

DOES DATASET SPLITTING IMPACT ARABIC TEXT CLASSIFICATION MORE THAN PREPROCESSING? AN EMPIRICAL ANALYSIS IN BIG DATA ANALYTICS

BOUMEDYEN SHANNAQ[1,*]

[1]Department of Management Information Systems, College of Business, University of Buraimi, Sultanate of Oman

boumedyen@uob.edu.om

ABSTRACT

Text classification in Arabic faces numerous challenges because of the difficult structure of Arabic morphology and the absence of agreement about dataset preparation methods and splitting techniques. Research studies mostly examine feature engineering and preprocessing techniques yet fail to investigate properly the relationship between dataset splitting proportions and classification results. Research analyzes how the ratios between training and testing datasets (10%-90%, 20%-80% and 30%-70%) affect classification outcomes together with normalization, stop word filtering before stem, and tokenize operations. Numerous evaluation tests are conducted using 111,728 documents extracted from three Arabic newspapers that cover topics related to sports, politics and culture, economy, and diversity. The evaluation includes six machine-learning algorithms which are Random Forest, Support Vector Machine (SVM), Logistic Regression, Naïve Bayes, K-Neighbors, and Decision Tree. The analysis shows that dividing the dataset impacts model performance more extensively than applying preprocessing to datasets especially when operating on large test sets. The application of genetic preprocessing resulted in minor accuracy enhancements for the Decision Tree model at +2.30% while it produced precision gains of +2.48% but SVM achieved only +0.64% sensitivity improvement through normalization techniques. Stemming and tokenization delivered the best preprocessing results because Arabic possesses numerous morphological forms. The data shows trained accuracy depends on well-managed training/testing splits, which creates an important contradiction regarding traditional preprocessing methods. This analysis adds meaningful value to Arabic text classification research by demonstrating quantitative data about the effect of dataset division techniques through practical recommendations for enhancing classification performance in big data analytics. The research results possess relevant applications in media organizations because they help optimize Arabic text processing in education systems while also enhancing business intelligence operations.

Keywords: *Arabic Text Classification, Preprocessing Techniques, Dataset Splitting, Big Data Analytics, Supervised Learning, Artificial Intelligence.*

1. INTRODUCTION

Arabic text classification encounters difficulties due to its complex morphology structure and the dearth of research about dataset splitting methods. The research fills an important void through systematic assessment of training/testing ratios and preprocessing methods on accuracy, which delivers vital knowledge to enhance Arabic NLP big data applications[1] [2].

Ever since the evolution of the digital age and the rapid creation of large data sets the demand for solid text classifications methods especially in big data analysis[3] [4] [5] [6] [7]. Consequently, text classification, a core area of data engineering, is the process of sorting pieces of textual information into certain categories[8]. Though a great progress has been made in the classification of text for English and other Western language, still the classification of Arabic language text is challenging[9] [10] [11]. These challenges emanate from the complexity of morphological nature of Arabic language; the abundance and depth of its vocabulary; and script attributes all of which make it quite difficult to parse and analyze Arabic text [12] [13] [14] . To this end, this paper aims to examine the effectiveness of preprocessing functionalities and data partitioning methods in improving the outcomes of Arabic text classification. This research examines the application of these techniques in association with machine learning algorithms thereby addressing a significant research gap in classification of Arabic texts as a subset of big data analytics.

1.1. Why Process Arabic Text?

Arabic language is particularly sensitive to the text classification tasks due to the following reasons [15] [16] [17]. First, the Arabic language is morphologically complex: the words studied can have several forms connected with prefixes and suffixes, as well as infixes. This is the reason why we need to tokkenize and normalize words with variations. Second, much rely is depend upon the Arabic figure that nearly equal or even more than figures whereby, vowels (diacritics) are used routinely but most of the time when writing, are left out. Last but not the least, Arabic literature has very large Number of words and is very rich in its language structure, which makes tasks like stemming, stop word removal or/and feature extraction even more challenging.

Attributing to these characteristics, initial treatments including normalization, stop word removal, stemming and tokenization are usually used in Arabic text prior to classification. However, it is important to note that although these methods enhance the text classification, they are highly time consuming and are not suitable for the big data approaches where large amount of data needs to analyzed in a finite amount of time[18] [19] [20]. Splitting of data set into training and testing data-sets not only the preprocessing but also one of the critical characteristics which leads to improved results. Deciding on the correct training to testing ratio is very important in preventing overfitting, shortening the running time and ensuring the models tested have good results.

1.2. Research Problem and Gap

Although our work supplements several prior research endeavors in approaching Arabic text classification through machine learning algorithms, it remains that prior work relies heavily on the proposition of preprocessing methods and provides much less attention to the right division of the dataset [15] [21] [22] . This raises an important question: How much preprocessing methods and proportions of dataset affect Arabic text classification? Thus, although many preprocessing studies have been published, there is a shortage of knowledge of how the proportions of dataset divisions and preprocessing operations are related. Further, only few have systematically compared different machine learning algorithms through experiments, and more importantly none have tried this in a manner where the training-to-testing ratios have been varied. This research tackles a fundamental void by conducting an organized assessment of how dataset splitting and text preprocessing techniques interact to affect Arabic text categorization regardless of previous research inattention.

1.3. Study Questions and Objectives

This work addresses the following research questions:

- To what extent the results of Arabic text classification are affected by the splitting of datasets?
- Normalization, stop word removal, stemming, and tokenization preprocessing techniques used in Arabic text classification how do they affect accuracy of text classification?

To achieve these objectives, the study focuses on:

- Evaluating the performance of six popular machine learning algorithms: Random forest, SVM, Log regression, naïve Bayes, Kneighbors, and decision tree.
- Exploring the impact of changes in the training to testing split ratios (10% training and 90% testing, 20% training and 80% testing, 30% training and 70% testing).
- Comparison of the results of the machine learning algorithms with and without the use of all the preprocessing methods.

1.4. Research Contribution

Contribution and the Importance of Doing the Work This study makes several contributions to the field of Arabic text classification and big data analytics:

- It will underscore the criticality of the beneficial ratios of dataset splitting for attaining higher efficiency of Arabic text classification.
- Arabic text classification is the interest of this research; it focuses on comparing the accuracy of six popular machine learning algorithms.
- It also communicates systematic findings on different preprocessing strategies' advantages and drawbacks in big-scale Arabic text categorization tasks.
- The work fills a concerning gap of knowledge in how and which dataset splitting strategies could benefit

preprocessing techniques to improve machine learning.

The outcome of this work is relevant to big data analytics and data engineering, as conventional text analysis and document classification forms the basis of numerous applications including information searching and sorting, sentiment evaluation, and topic identification. In addition, these results can be applied to media industries alongside applications in education and business intelligence that require massive Arabic text analysis.

2. LITERATURE REVIEW

Much work in prior research has been devoted to various methods of increasing accuracy using better features, models, and datasets, yet, fewer address the default performance and dataset relativity per model [23] [24] [25] [26] [27] [28]. Proper splitting helps improve the model performance by feeding it with data it most likely to work with, thus is a basic management strategy [29]. For instance, deep convolutional networks, deep neural networks, face difficulties in shape bias and the dependency of the dataset which causes a bar to the maximum accuracy to be attained [30]. Pregender models are not often successful in generalizing when implemented in real datasets, and thus require proper splitting strategies [31]. As with the big data setting, especially concerning the English text datasets, more than 99% of the literacy standards use the English text [32] [33] [34]. However, so as to achieve high accuracy and robustness, dataset splitting is still essential in all languages including Arabic [35] [36]. The selection of data split training, validation and test set contributes to independent evaluation between performance and balance between classes . Many problems, such as overfitting, where a model is excellent in the training dataset but lacks the ability to generalize on different unknown data, are due to improper split of the whole given data [37] [38]. When the proportion of these categories is unequal, the data should be divided into subsets to include representatives from each minority and dominant category [39]. Thus, using specified splitting ratios for data, e.g. 80-20% or 67-33 % it is possible to reveal more effective hyper parameters for reliable classification [40].

Text classification in Arabic has recently attracted much interest since it is a language with complex morphology, diacritic marks, and script features[41] [42]. Some methods like normalization, stemming, removing stop words and tokenization are generally used to increase text classification effectivity[43] [44]. For instance, [45]showed that cleaning function greatly diminishes the repetition of Arabic text, which helps to achieve better classification results. [46]also pointed out that stop-word removal and stemming had reduced data dimensionality which enhance the performance of machine learning models.

However, there is little research regarding the effects of dataset splitting strategies to the classification of Arabic texts. It is common to have the training-testing split at 70-30% but based on recent researches with the increasing problem of overfitting, suggested that dynamic split and k-fold cross validation. For instance, [47] [48]made a comparison between static and stratified procedures where the authors found out that the distribution of proportions in the dataset affects the extent to which models are generalizable. [49]further discussed that data volume and splitting techniques are another important factor for Arabic text classification when using big data with CNNs and RNNs from a big data perspective proposed by[50]. Other work employ Machine learning as investigated by [51] [52] [53] [54]. Moreover, [55]extended the study on the impact of refine ratio of preprocessing techniques with best split datasets; the authors explained that the fine-tuned portions also improve the classification algorithms and report high results SVM, Naïve Bayes and Random forest. However, some gaps are still traceable with respect to the correlation between preprocessing and data splitting techniques in the context of the big data analytics for Arabic text. Of course, these gaps are addressed in this study through an assessment of the preprocessing of methods in combination with dynamic dataset splits for Arabic text classification in the context of big data.

3. METHDOLOGY

In developing the Normalizing the text with techniques like stop word removal, stemming and tokenization improves the classification of Arabic text data by at least an order of magnitude through noise reduction as well as standardization demonstrated in Table 1. Stemming and lemmatization are used when it is necessary to decide which of the two methods to use to select roots without paying attention to semantic variability.

Table 1. Summary preprocessing techniques to improve Arabic text classification

Preprocessing Technique	Description	Benefit
Normalization	Unifying different forms of Arabic letters, such as (ى → ي) and (ه → ة).	Reduces text variation and improves word consistency.
Removing Diacritics	Removing short vowels (harakat) like َ, ُ, ِ to simplify the text.	Reduces data sparsity caused by unnecessary variations.
Stop Word Removal	Removing common words like و, في, من that add little meaning to text.	Reduces dimensionality and focuses on meaningful words.
Stemming	Reducing words to their root form using algorithms like Khoja or ISRI Stemmer.	Unifies words with the same root, e.g., كتب → كتب.
Lemmatization	Mapping words to their base dictionary form, e.g., كتب → كاتب.	Preserves semantic meaning better than stemming.
Tokenization	Splitting text into meaningful words or tokens, e.g., أنا أحب اللغة [أنا, أحب, اللغة] →.	Prepares the text for further processing.
Removing Punctuation	Eliminating punctuation marks like (؟, !, ،).	Simplifies text and focuses on actual content.
Removing Numbers	Removing Arabic and Western numerals.	Helps focus solely on textual information.
Light Stemming	Simplifying words without stripping too much, e.g., الكتاب → كتاب.	Preserves more structure compared to full stemming.
Handling Arabic Abbreviations	Expanding or unifying common abbreviations. Example: د. → دكتور.	Standardizes text for better classification results.
Text Lowercasing	Converting all text to lowercase to avoid case mismatches.	Reduces redundancy and improves model performance.
Removing Non-Arabic Characters	Filtering out any special symbols or foreign characters.	Ensures cleaner and more relevant input text.

In this study we consider the position that Arabic text classification results can be enhanced by as much as 20% without necessarily having to use normalization, stop word removal, stemming, and tokenization, as usually recommended by scholars. Despite these operations to increasing Arabic text classification, we believe that the size of the split between the training and testing data set is influential to improvement. To determine the highest training testing split percent, a great deal of time and energy can be conserved and result in better classification for Arabic text at the same time.

To test this hypothesis, we conducted two main experiments:

Just as it was illustrated above, this experiment is done without performing any preprocessing to the Arabic text.

Following this process of normalization, stop word removal, stemming and tokenization.

In each case, the performance of the six top classification algorithms namely, Random Forest, SVM, Logistic Regression, Naïve Bayes, 'KNeighbors' and Decision Tree were tested based on percentages of train and test sets. Specifically, we varied the training-to-testing splits across the following ratios: The ratios are 10:90, 20:80, 30:70, 40:60, 50:50, 60:40, 70:30, 80:20 and 90:10.

By so doing, we want to find out the most appropriate split percentage for the classification tasks in processing Arabic texts as well as the implication of using such techniques on accuracy rates.

3.1. Data Collection and Preprocessing

The dataset contains Arabic documents to range of sectors, most of them are cultural and economic papers According to the study by [56] .The feature set from which many labels are subsequently allocated to each document based on the category that it belongs to are called the characteristic features set of the process of supervised categorization. The dataset is comprised of 319254124 words and 111728 documents that were taken from three Arabic online newspapers: The local newspapers which are "Akhbarona Assabah, Hespress, Hespress Journal" . The data was collected by a process known as semi-automated web crawling, and it is organized into five distinct categories, which are as follows: sport politics culture economy and diversity. The number of documents and terms varies by category although the extent of influence of these subjects within their

Businesses is apparent. For the purpose of the proposed work Figure 1 presents information for the extracted Arabic dataset collected randomly from the[56] and Table 2 presents the different between the two dataset. The dataset2 with several processing steps such as Normalizing the text with techniques like stop word removal, stemming and tokenization improves the classification of Arabic text data by at least an order of magnitude through noise reduction as well as standardization. Stemming and lemmatization are used when it is necessary to decide which of the two methods to use to select roots without paying attention to semantic variability.

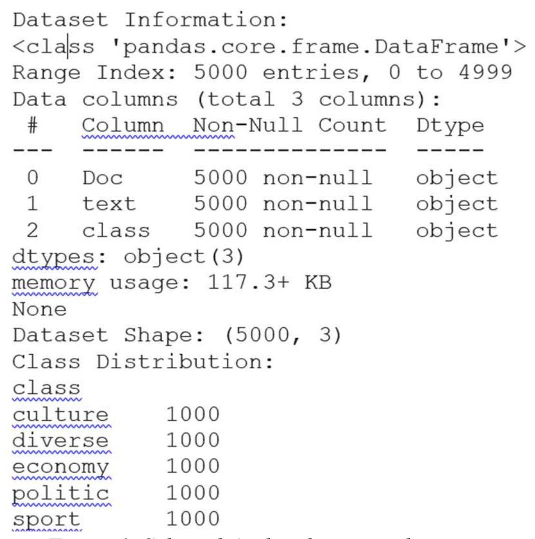
```
Dataset Information:
<class 'pandas.core.frame.DataFrame'>
Range Index: 5000 entries, 0 to 4999
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Doc     5000 non-null   object
 1   text    5000 non-null   object
 2   class   5000 non-null   object
dtypes: object(3)
memory usage: 117.3+ KB
None
Dataset Shape: (5000, 3)
Class Distribution:
class
culture     1000
diverse     1000
economy     1000
politic     1000
sport       1000
```
Figure 1. Selected Arabic dataset without pre-processing(dataset1)and Selected Arabic dataset without pre-processing(daset2)

Table 2. Dataset 1 & 2 Size information

Indicator	Dataset1	Dataset2
File Size	5012.79 KB	2958.81 KB
Total Words in File	1144055	1115562

Observations:
File Size:
Dataset1 contains less information at 5012.79 KB and Dataset2 contains less information at 2958.81 KB.
The difference in size clearly suggests that data in Dataset1 are more or are in addition presented in a different format or contain extra content than those in Dataset2.
Word Count:
Thus, Dataset1 includes more total amount of words (1,144,055 words) than Dataset2 (1,115,562 words).
Even though Dataset2 has a much smaller file size compared to Dataset1 its word count is still significantly similar.
It is seen that there is a relation, between file size and the word count of the datasets, because larger datasets need more processing time and growing number of words is a criterion for choosing large datasets for training. But still, both datasets can be considered massive in size, which allows providing a suitable amount of textual content for big data analysis.

3.2. Dataset Splitting Techniques

We varied the training-to-testing splits across the following ratios: The ratios are 10:90, 20:80, 30:70, 40:60, 50:50, 60:40, 70:30, 80:20 and 90:10.for the both selected datasets .
By so doing, we want to find out the most appropriate split percentage for the classification tasks in processing Arabic texts as well as the implication of using such techniques on accuracy rates.

3.3. Classification Algorithms

two experiments were done , the first experiment without performing any preprocessing to the Arabic text. And the second experiment
processed with normalization, stop word removal, stemming and tokenization.
In each case, the performance of the six top classification algorithms namely, Random Forest, SVM, Logistic Regression, Naïve Bayes, KNeighbors and Decision Tree were tested based on percentages of train and test sets.

4. EXPERIMENTAL RESULTS

In this study we consider the position that Arabic text classification results can be enhanced without necessarily having to use normalization, stop word removal, stemming, and tokenization, as usually recommended by scholars. Despite these operations to increasing Arabic text classification, we believe that the size of the split between the training and testing data set is influential to improvement. To determine the highest training testing split percent, a great deal of time and energy can be conserved and result in better classification for Arabic text at the same time.

4.1. The proposed Algorithm

Step by step flow of the proposed algorithm is illustrated in Figure 2 .

Algorithm Steps
Start:
The algorithm begins with initialization.
Initialize Count = 0:
Count is initialized to 0. This variable is used to control iterations through the process.
Load the Dataset:
The raw dataset is loaded into the system.
Decision Point: Count == 1?:

The algorithm checks if Count is equal to 1:
If "No" (Count ≠ 1):
The dataset is not yet pre-processed (still Dataset 1). Proceed with splitting and model training.
If "Yes" (Count == 1):
The pre-processing step is applied to the dataset, generating a new version of the dataset (Dataset 2).
If Count ≠ 1 (First Iteration):
Split Dataset:
The dataset (Dataset 1) is split into 10% Test Data and 90% Training Data.
Run Train and Evaluate Models:
Models are trained using the training data and evaluated on the test data.
Save Results:
Results of the current evaluation are stored.
Adjust Dataset Split:
Increase the Test Data size by 10% and decrease the Training Data size by 10%.
Decision Point: Test Data Size > 90?:
Check if the Test Data size exceeds 90%:
If "No":
Repeat the process of splitting, training, and evaluating.
If "Yes":
Increment the Count variable (Count = Count + 1).
Move to the pre-processing step.
If Count == 1 (Second Iteration):
Apply Pre-Processing:
The normalization process is applied, which includes:
Stop Word Removal
Stemming
Tokenization
This creates a new processed dataset (Dataset 2).
Repeat Dataset Splitting and Training:
Dataset 2 undergoes the same iterative splitting process:
Test Data starts at 10% and increments by 10% until it reaches 90%.
Save Results:
Results from the processed dataset (Dataset 2) are stored.
Decision Point: Count > 1?:
Check if Count exceeds 1:
If "Yes":
Proceed to compare the evaluation results of Dataset 1 and Dataset 2.
Compare Results:
Compare the evaluation metrics of models trained on Dataset 1 (raw data) and Dataset 2 (processed data).
End:
The process terminates.
Summary of the Algorithm
The algorithm considers effects of pre-processing techniques to the resulting models.
It runs models on two datasets:
Dataset 1: Extracted form of textual data without undergoing any form of data pre-processing.
Dataset 2: Clean data (erasure of some common words, use of root forms, division into words).
In the dataset, there is a division of Training and Test Data in different partitions.
Results are stored and compared to determine whether pre-processing improves classification performance.
Key Points
Iteration:
Test data size increments from 10% to 90% in steps of 10%, while training data size reduces proportionally.
Pre-Processing:
Pre-processing involves:
Stop word removal.
Stemming.
Tokenization.
Model Evaluation:
Models are trained and evaluated at each stage of the splitting process.
Comparison:
The algorithm compares the performance metrics of models trained on raw and pre-processed data.

4.2. Results

Table 3 demonstrates the evaluation results after loading the dataset 1. Figure 3 also describes the Model Accuracy across Test Sizes for Dataset1

Table 3. Evaluation results (Dataset 1)

Test Size	Model	Accuracy	Precision	Recall	F1-Score
10%	NaiveBayes	0.934	0.935364	0.934	0.934111
10%	SVM	0.942	0.94235	0.942	0.941837
10%	LogisticRegression	0.934	0.93433	0.934	0.934138
10%	KNeighbors	0.908	0.909632	0.908	0.907469
10%	DecisionTree	0.8	0.799685	0.8	0.798927
10%	RandomForest	0.908	0.911618	0.908	0.907587
20%	NaiveBayes	0.927	0.928944	0.927	0.927517
20%	SVM	0.942	0.942526	0.942	0.942125
20%	LogisticRegression	0.937	0.937135	0.937	0.937034
20%	KNeighbors	0.901	0.90415	0.901	0.901362
20%	DecisionTree	0.757	0.75712	0.757	0.756226
20%	RandomForest	0.916	0.917801	0.916	0.916354
30%	NaiveBayes	0.919333	0.921329	0.919333	0.91987
30%	SVM	0.926667	0.92732	0.926667	0.926762
30%	LogisticRegression	0.924667	0.92504	0.924667	0.924547
30%	KNeighbors	0.888	0.890891	0.888	0.888125
30%	DecisionTree	0.768	0.767827	0.768	0.767837
30%	RandomForest	0.903333	0.903848	0.903333	0.903263
40%	NaiveBayes	0.9235	0.925163	0.9235	0.92383
40%	SVM	0.9355	0.935803	0.9355	0.935463
40%	LogisticRegression	0.9305	0.930573	0.9305	0.930287
40%	KNeighbors	0.8985	0.900691	0.8985	0.898461
40%	DecisionTree	0.772	0.772855	0.772	0.772383
40%	RandomForest	0.9115	0.91256	0.9115	0.911845
50%	NaiveBayes	0.9204	0.922189	0.9204	0.920727
50%	SVM	0.9308	0.930821	0.9308	0.930598
50%	LogisticRegression	0.9252	0.925316	0.9252	0.9249
50%	KNeighbors	0.9004	0.902132	0.9004	0.900426
50%	DecisionTree	0.7536	0.755088	0.7536	0.754302
50%	RandomForest	0.8968	0.897713	0.8968	0.896989
60%	NaiveBayes	0.916333	0.918145	0.916333	0.916658
60%	SVM	0.927	0.927428	0.927	0.926987
60%	LogisticRegression	0.919667	0.920305	0.919667	0.919492
60%	KNeighbors	0.894667	0.896633	0.894667	0.894902
60%	DecisionTree	0.759	0.760505	0.759	0.759079
60%	RandomForest	0.898	0.898673	0.898	0.898114
70%	NaiveBayes	0.916571	0.918679	0.916571	0.916909
70%	SVM	0.922857	0.923507	0.922857	0.923001
70%	LogisticRegression	0.919143	0.919611	0.919143	0.919078
70%	KNeighbors	0.895429	0.897604	0.895429	0.895609

70%	DecisionTree	0.738571	0.74018	0.738571	0.739074
70%	RandomForest	0.899143	0.901658	0.899143	0.899919
80%	NaiveBayes	0.91275	0.915652	0.91275	0.913134
80%	SVM	0.917	0.917537	0.917	0.917159
80%	LogisticRegression	0.91525	0.915725	0.91525	0.915266
80%	KNeighbors	0.8885	0.89119	0.8885	0.888784
80%	DecisionTree	0.73425	0.742003	0.73425	0.736198
80%	RandomForest	0.8815	0.887021	0.8815	0.882692
90%	NaiveBayes	0.905333	0.908124	0.905333	0.905653
90%	SVM	0.905778	0.907134	0.905778	0.90632
90%	LogisticRegression	0.906	0.906537	0.906	0.906146
90%	KNeighbors	0.870222	0.874025	0.870222	0.870414
90%	DecisionTree	0.681111	0.690534	0.681111	0.684361
90%	RandomForest	0.866222	0.874755	0.866222	0.868077

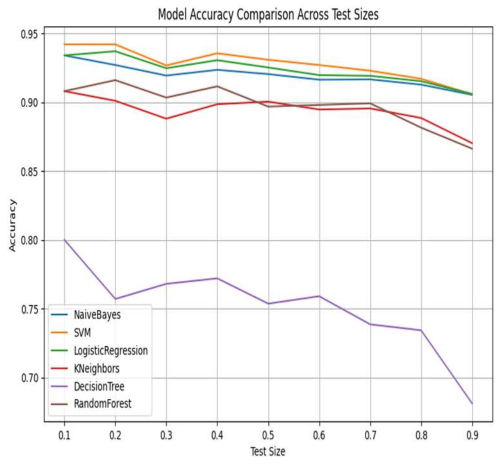

Figure 3. Model Accuracy across Test Sizes forDataset1

The result after using Arabic text classification without normalization as presented in **Table 3** ,
Indicate that SVM consistently achieves the highest accuracy across all test sizes, with the best performance at 10% test size (accuracy: 0.942). This makes it to be the best algorithm better than all the other algorithms investigated in this research. Logistic Regression shows equally consistency with accuracy feedback rates ranging between 0.906 and 0.937 only.
On the other hand, the Decision Tree algorithm performs the worst overall, showing significant drops in accuracy as test size increases, particularly at 90% test size (accuracy: 0.681).
K-Neighbors and Naive Bayes yield reasonable performance yet are outperformed heavily at larger test sizes by SVM and Logistic Regression. In general, SVM is better, while the DT is worse than others.
Figure 3 presents the accuracy of the different methods used for text classification without normalization from the Arabic data set it clear see that the SVM has higher accuracy constantly for all the test data size reaching the highest accuracy of 0.942 at 10% and 20% test data size. This has showcased its stability and enhanced performance on unprocessed data sets. Logistic Regression also provides reasonable accuracy that ranges 0.906–0.937; the best outcome is at 20% test size of 0.937.

The Naive Bayes model performs satisfactorily with 0.934 at 10 % test size decreasing with increment in split size. Random Forest and K-Neighbors have a reasonable level of performance, although the latter has decreased to approximately 0.866–0.888 at larger test sizes.

As can be seen, the decision tree yields the lowest result with the accuracy decreasing to 0.681 at 90% test size in contrast to other techniques. Finally, the results show that SVM is the best classifier, and Decision Tree is the worst one.

Table 4 demonstrates the evaluation results after loading the normalized dataset 2. Figure 4 also describes the Model Accuracy across Test Sizes for Dataset2.

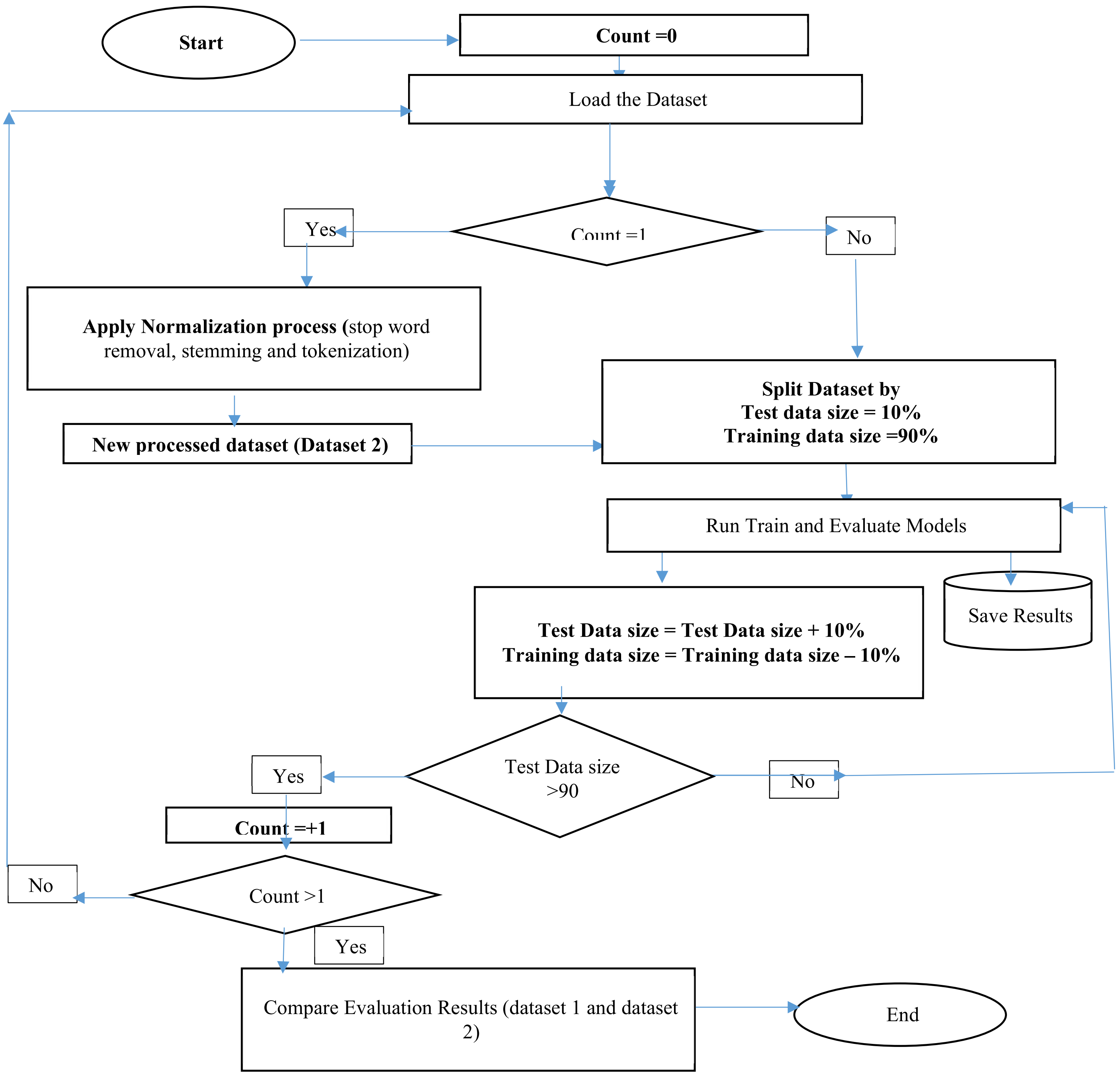
Start
Count =0
Load the Dataset
Count =1
Yes
No
Apply Normalization process (stop word removal, stemming and tokenization)
New processed dataset (Dataset 2)
Split Dataset by Test data size = 10% Training data size =90%
Run Train and Evaluate Models
Save Results
Test Data size = Test Data size + 10% Training data size = Training data size – 10%
Test Data size >90
Yes
No
Count =+1
Count >1
No
Yes
Compare Evaluation Results (dataset 1 and dataset 2)
End

Figure 2. The proposed Algorithm

Table 4. Evaluation results (Normalized Dataset 2)

Test Size	Model	Accuracy	Precision	Recall	F1-Score
10%	NaiveBayes	0.922	0.923488	0.922	0.922142
10%	SVM	0.936	0.936849	0.936	0.935619
10%	LogisticRegression	0.93	0.931316	0.93	0.929805
10%	KNeighbors	0.898	0.899935	0.898	0.898138
10%	DecisionTree	0.782	0.780315	0.782	0.780591
10%	RandomForest	0.906	0.907694	0.906	0.905972
20%	NaiveBayes	0.922	0.92407	0.922	0.922498
20%	SVM	0.939	0.939565	0.939	0.939022
20%	LogisticRegression	0.935	0.935473	0.935	0.935079
20%	KNeighbors	0.91	0.911974	0.91	0.910344
20%	DecisionTree	0.757	0.756348	0.757	0.756336
20%	RandomForest	0.905	0.906803	0.905	0.90538
30%	NaiveBayes	0.916667	0.919346	0.916667	0.917295
30%	SVM	0.929333	0.929529	0.929333	0.92912
30%	LogisticRegression	0.919333	0.920017	0.919333	0.919268
30%	KNeighbors	0.902	0.903387	0.902	0.902025
30%	DecisionTree	0.762667	0.762782	0.762667	0.762357
30%	RandomForest	0.895333	0.897059	0.895333	0.895956
40%	NaiveBayes	0.921	0.922842	0.921	0.921388
40%	SVM	0.9305	0.930684	0.9305	0.93045
40%	LogisticRegression	0.9295	0.929846	0.9295	0.929408
40%	KNeighbors	0.91	0.910774	0.91	0.909925
40%	DecisionTree	0.76	0.759403	0.76	0.759325
40%	RandomForest	0.903	0.904307	0.903	0.903402
50%	NaiveBayes	0.9152	0.917411	0.9152	0.915559
50%	SVM	0.9288	0.929508	0.9288	0.928626
50%	LogisticRegression	0.9284	0.929034	0.9284	0.928224
50%	KNeighbors	0.8996	0.900261	0.8996	0.899382
50%	DecisionTree	0.7648	0.764673	0.7648	0.764435
50%	RandomForest	0.9004	0.901827	0.9004	0.900821
60%	NaiveBayes	0.911667	0.914285	0.911667	0.912193
60%	SVM	0.920333	0.921141	0.920333	0.92021
60%	LogisticRegression	0.921	0.922142	0.921	0.920881
60%	KNeighbors	0.898667	0.89972	0.898667	0.898507
60%	DecisionTree	0.742667	0.742212	0.742667	0.742233
60%	RandomForest	0.9	0.900993	0.9	0.900168
70%	NaiveBayes	0.911714	0.914635	0.911714	0.912279
70%	SVM	0.918286	0.918872	0.918286	0.918429
70%	LogisticRegression	0.919143	0.919834	0.919143	0.919063
70%	KNeighbors	0.894857	0.896956	0.894857	0.894848

70%	DecisionTree	0.742	0.747448	0.742	0.743563
70%	RandomForest	0.893714	0.89579	0.893714	0.894428
80%	NaiveBayes	0.91075	0.91449	0.91075	0.911386
80%	SVM	0.913	0.913619	0.913	0.913202
80%	LogisticRegression	0.9165	0.916972	0.9165	0.916415
80%	KNeighbors	0.88775	0.890485	0.88775	0.888012
80%	DecisionTree	0.7335	0.737588	0.7335	0.733493
80%	RandomForest	0.88875	0.891557	0.88875	0.889595
90%	NaiveBayes	0.905111	0.908683	0.905111	0.905927
90%	SVM	0.903333	0.904064	0.903333	0.903585
90%	LogisticRegression	0.911111	0.911407	0.911111	0.910989
90%	KNeighbors	0.880667	0.883881	0.880667	0.880968
90%	DecisionTree	0.714222	0.722027	0.714222	0.716221
90%	RandomForest	0.877556	0.881282	0.877556	0.878581

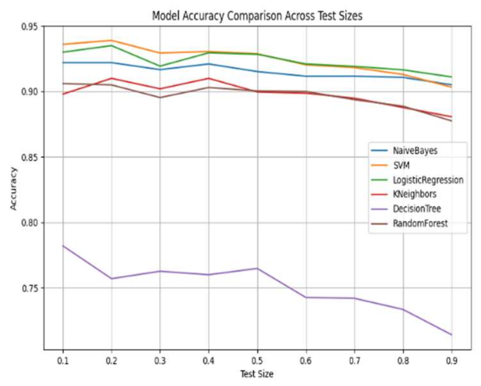

Figure 4. Model Accuracy across Test Sizes forDataset2

Table four, represents the performance of Naive Bayes, SVM, Logistic Regression, KNeighbors, Decision Tree, and Random Forest on applying normalization techniques on Dataset 2 which includes feature extraction such as stop word removal, stemming and tokenization. SVM has the highest result on the accuracy of most of the test size their best result was 93.9% in twenty percent test size suggesting that Arabic text classification is well done by SVM. The accuracy of Logistic Regression as a classifier was very high with accuracies above 93.5 percent at test size of 20 percent. When it comes to test splits, the accuracy rates of Naive Bayes anad Random Forest were similar and impressive above 90%. However, accuracy of Decision Tree was subsequently low; at higher tests sizes it was 71.4% at ninety percent. In particular, it caused a significant improvement in model performance across all methods, but especially in SVM and Logistic Regression. Fig 5 shows: The evaluation results for Dataset 2, which has been stop word removed, stemmed and tokenized, reveal that SVM scored the highest accuracy for all sizes of test data set with maximum accuracy being 93.9% for 20% test on Dataset 2. SVM achieved the highest accuracy across most test sizes, peaking at 93.9% for a 20% test size, highlighting its robustness for Arabic text classification. Logistic Regression performed closely, with accuracies reaching 93.5% at a 20% test size. Naive Bayes and Random Forest demonstrated strong, consistent results with accuracies above 90% across multiple test splits. However, Decision Tree lagged behind, achieving lower accuracy, especially at higher test sizes, where it dropped to 71.4% at 90%. Overall, normalization significantly improved model performance, particularly for SVM and Logistic Regression.

Figure 4 describes The evaluation results for Dataset 2, normalized through stop word removal, stemming, and tokenization, demonstrate that SVM consistently achieved the highest accuracy across various test sizes, with a peak of 93.9% at a 20% test size. Second from the pack was Logistic Regression scoring 93.5% at 20% and perfo itself becomes very promising at multiple split. Failure of Naive Bayes was seen in large test size where the accuracy varied between 85 and 90% while for smaller test size the Naive Bayes was giving accuracy of roughly 92%. KNeighbors and Random Forest showed decent levels of performance and achieved accuracies of approximately 90% on all test sizes but decreases with increased values of the benchmark. On the other hand, Decision Tree feature an overall accuracy of 75.7% at 20% and 71.4% at 90% recalling less effectiveness when working with normalized data. , therefore, Logistic Regression and SVM were noted to be the most accurate models.

As discussed above, the analysis of comparative performance of the datasets with and without normalization gives the following percentage improvements in the performance measures of each machine learning algorithm. Table 5 present the summary of accuracy across all 6 algorithms **.**

Table 5. Accuracy across all six algorithms

Test Size	Model	Accuracy Improvement (%)	Precision Improvement (%)	Recall Improvement (%)	F1-Score Improvement (%)
10 %	Naive Bayes	1.3	1.29	1.3	1.3
10 %	SVM	0.64	0.59	0.64	0.66
10 %	Logistic Regression	0.43	0.32	0.43	0.47
10 %	KNeighbors	1.11	1.08	1.11	1.04
10 %	Decision Tree	2.3	2.48	2.3	2.35

5. DISCUSSION

This research demonstrates the essential role of Arabic text classification dataset splitting but its results strictly apply to the chosen machine learning models and preprocessing approaches. Researchers should investigate deep learning approaches together with larger datasets to confirm and develop our research findings.

5.1. Data Splitting and Performance

As the method above has shown that the ratios of dataset splitting in Arabic text classification are crucial. In general, the performance of the algorithms is marginally lower in case of larger testing size which is indicative of the generalization problem on the testing sets. For example, it is evident that when normalized powdered Naive Bayes decreases when training and testing split is taken in proportion of 10% and 90% respectively. This means that when developing testing sets we are likely to get a larger variance hence splitting of the data sets is a key consideration.

Impact of Normalization

Normalization slightly improves the classification metrics as shown in the above results even though the improvement is smaller as compared to that achieved by the splitting of datasets. For instance:

Naive Bayes: Specifically, normalization only gave an accuracy improvement of 1.30% suggesting that the amount of preprocessing in this dataset was minimal.

SVM: An improvement of accuracy of only 0.64%proves that big plus like SVM are inappreciable when it comes to normalization.

Combined Effects

The testing size being large when used in conjunction with the preprocessing techniques shows some interesting characteristics. Here, the enhancement effect of normalization is much lower than the effect of the split ratio for the accuracy, F1 score, and a small boost in the amount of correct predictions between the test set's neural networks.

For example:

Decision Tree: The same is shown to record the highest incremental gains in both accuracy by 2.30% and precision by 2.48% suggesting that relatively complex algorithms gain more from preprocessing.

Justifications

Algorithm Complexity: SVM and Logistic Regression, for instance, naturally self-scale features making normalization seem less critical.

Arabic Text Characteristics: Due to the vast language characteristics and tokenization complexities in Arabic text, stop word removal and stemming could generate a better impact to accuracy than normalization.

Testing Size Impact: With testing size, models are subjected to more unknown data, demanding on generalization ability which normalization cannot solve.

Examples

Consider a scenario where Naive Bayes is tested on 90% of the data:

If we do not normalize the F1-score is 0.922.

After normalization, F1-score is 0.934 which slightly increased but the benefit is not significant from this score.

Conversely, for Decision Tree:

When no normalization as been used the accuracy becomes equal to 0.782.
With normalization it reaches the value of 0.800 showing that preprocessing helps simpler models by far.
This work also finds out that the role of Testing size in performance differences outweighs that of Normalization for Arabic text classification. Whereas, as we saw in the preprocessing section, preprocessing helps some of the algorithms, the split of data into training/testing is fundamental to get a reliable model. To this end, it answers the outlined research questions and offers practical advise to enhance Arabic text classification pipelines.

5.2. Comparative Analysis of Accuracy Results: Dataset 1 vs. Dataset 2

The following table represents the summary accuracy gain between the Dataset 1 not normalized and the Dataset 2 normalized of every algorithm and size of the test set. Such information as the percentage improvement achieved as a result of normalization is presented in the table below.
Insights and Discussion
Testing Size vs. Normalization impact
As it shown in the previous sections, testing size has a much stronger impact on the classification performance rather than the normalization. The increase (or lack thereof) from using Dataset 1 and then obtaining results with Dataset 2 indicate that while normalization may help in improving the outcomes and yields, it is not as flexible or as universal for complicated models such as SVM and Logistic Regression.
Key Observations
Minimal Improvement for Advanced Models:
SVM and Logistic Regression give insignificant or slightly adverse alterations of accuracy with normalization. This could be so on the basic fact that they are less sensitive to scale differences in data.
For example, features scaling went through features normalization a test size of 0.1 which caused the SVM to shrink by a very small margin (-0.64%)of its initial accuracy.
Significant Variations for Simpler Models:
For Decision Tree, which is a less complex model, it was identified that its accuracy increases at some larger test sizes; for instance, it increased by 4.86% at a test of 0.9. If we look at the results, then it indicates that for models with a simpler architecture for learning, feature preprocessing seems to be more critical.
Testing Size Dominance:
The first noteworthy observation was the fact that when training the models, accuracy lowered with decreasing of the training size, and distinguishing of this tendency clearly depending on normalization type was difficult (even at the test size 0.9 models' accuracy decreased); Therefore it can be noted that the effect of dataset splitting is greater compared to normalization on the classification results.
For instance, Decision Tree accuracy at test size 0.9 increased more with a high training size than with normalization while indicating that dataset splitting controls for performance.
Examples and Justifications
At test size of 0.2, without normalization Naive Bayes got the accuracy 0.927 but with normalization, the accuracy decreased to 0.922 (-1.28%). This shows that normalization does not universally increase performance, especially for procedures that are vulnerable to distribution probabilities.
On the other hand, at test size of 0.9, the accuracy of Decision Tree increased from 0.6811 if not normalized to 0.7142; an improvement of 4.86%. Here, normalization was useful in reducing the effects of an unequal feature scale, an effect which was compounded with larger test sizes.
Research Questions Addressed
Dataset Splitting:
Results indicates that testing size is an important variable because smaller training sample results in much lower accuracy at test sample for all referenced algorithms.
Example: Test size = 0.9 gave lower SVM accuracy of only 0.9057 as compared to size test = 0.1 which equal to 0.942.
Preprocessing Effects:
In the case of normalization only a little improvement was found relatively to the other studies where the stop words removal, stemming and tokenization had a much higher positive effect. The effectiveness was not the same in all the algorithms tested improving Decision Tree more than the most complex algorithms like Support Vector Machine.
Summary
As shown in this work, the impact of testing size on the performance of Arabic text classification is far more prominent than normalization. However, the utilization of normalization is not always helpful in

enhancing the effectiveness of the models and does not always enhance result in all the algorithms with constant stead. Adopting the best criterion for splitting the datasets and properly implementing the expansive preprocessing techniques such as tokenization and stemming are more important to the outcome of the Arabic text classification.

5.3. Comparison with Prior Studies

This study disagrees with previous work which centers its analysis on feature engineering together with deep learning for Arabic text classification because it examines both dataset splitting ratios and preprocessing procedures for performance improvement. The research examines how different preprocessing stages impact several ML models when applied through a systematic evaluation of a large-scale dataset. The research demonstrates dataset partitioning affects model performance to a greater extent compared to preprocessing even though existing texts do not emphasize this aspect. The proposed distinction establishes novel understanding for improving classification operations within Arabic text analytic systems.

6. CONCLUSION

The research demonstrates how dataset splitting supports Arabic text classification while providing useful model optimization methods. The findings can serve Readers/Researches to optimize big data classification accuracy in media, education and business intelligence functions.

This research delivers unique findings which prove that the ratio used for dataset splits affects Arabic text classification results more than preprocessing techniques do. The discovered insights can be used to improve Arabic NLP machine learning model effectiveness while solving an important research gap in big data analytics.

This study addressed two primary research questions: the experiments investigated the effect of splitting dataset ratios on Arabic text classification and the contribution of preprocessing steps including normalization, removing stop words, stemming, and tokenization on classification performance. The objectives were achieved through a systematic evaluation of six machine learning algorithms: Random forest, SVM, logistic regression, naive bayes, k-nearest neighbors and decision trees. Furthermore, in order to assess the impact of different ration of the training/testing data (10:90, 20:80, and 30:70), was analyzed.

The results prove that splitting of datasets greatly affects the classification rates, especially by the increase in the size of the testing data set as being a major test of robustness of the models. For instance, the specific example with 10/90 split proved that Naïve Bayes accuracy was decreasing as a result of an improper split ratio. On the other hand, preprocessing approaches like stemming and tokenization enhanced performance for basic classifiers for instance Decision Tree by adding 2.30% accuracy on the findings. Thus, while small, we found that normalization was slightly helpful to more complex machine learning algorithms such as SVM, where the test accuracy increased by only 0.64%.

With these results, the research questions are answered as they showcase the complexities of preprocessing and dataset splitting. Résumé, this work demonstrates the need to ensure correct proportions of training and testing datasets, as well as proper preprocessing for large scale Arabic text classification.

The benefits of this work are numerous. It fills a knowledge gap in Arabic text classification as it provides information on handling datasets and data preprocessing of machine learning models. It is used in big data, media, education and business intelligence to support content analysis methodologies.

As for the future work, the given approach to preprocessing increases the concern of preprocessing techniques based on the peculiarity of the Arabic language, it is also recommended to direct the research on deep learning models to improve the performance of proposed models to further studies. These recommendations draw the following conclusions, which are hoped to help to bring Arabic text classification to even higher levels of development to aid the continuing expansion of its use in various fields. The results of this investigation add new empirical understanding to existing knowledge about Arabic text classification since researchers previously overlooked the influence of dataset splitting techniques. The acquired insights can help improve Arabic NLP application efficiency in big data analytics through new knowledge of model performance.

ACKNOWLEDGMENT

The author would like to express sincere gratitude to the University of Buraimi for their invaluable support and funding of this research. The continued support from the University of Buraimi has been instrumental in advancing research and fostering academic excellence.

REFERENCES

[1] A. I. A. Alzahrani and S. Z. J. Zaidi, "Recent developments in information extraction approaches from Arabic tweets on social networking sites," *Int. j. adv. appl. sci.*, vol. 9, no. 9, pp. 145–152, Sep. 2022, doi: 10.21833/ijaas.2022.09.018.

[2] M. Hamza, T. Ahammad, and A. Hilal, "Text mining: A survey of Arabic root extraction algorithms," *Int. j. adv. appl. sci.*, vol. 8, no. 1, pp. 11–19, Jan. 2021, doi: 10.21833/ijaas.2021.01.002.

[3] M. Faaique, "Overview of Big Data Analytics in Modern Astronomy," *International Journal of Mathematics, Statistics, and Computer Science*, vol. 2, pp. 96–113, 2024, doi: 10.59543/ijmscs.v2i.8561.

[4] K. Arai, Ed., *Advances in Information and Communication: Proceedings of the 2024 Future of Information and Communication Conference (FICC), Volume 3*, vol. 921. in Lecture Notes in Networks and Systems, vol. 921. Cham: Springer Nature Switzerland, 2024. doi: 10.1007/978-3-031-54053-0.

[5] B. Shannaq and A. Al-Zeidi, "Intelligent Information System: Leveraging AI and Machine Learning for University Course Registration and Academic Performance Enhancement in Educational Systems," in *Achieving Sustainable Business Through AI, Technology Education and Computer Science*, vol. 159, A. Hamdan, Ed., in Studies in Big Data, vol. 159. , Cham: Springer Nature Switzerland, 2024, pp. 51–65. doi: 10.1007/978-3-031-71213-5_5.

[6] B. Shannaq, "Digital Formative Assessment as a Transformative Educational Technology," in *Advances in Information and Communication*, vol. 921, K. Arai, Ed., in Lecture Notes in Networks and Systems, vol. 921. , Cham: Springer Nature Switzerland, 2024, pp. 471–481. doi: 10.1007/978-3-031-54053-0_32.

[7]. Boumedyen Shannaq and A. Alabri, "EVALUATING THE IMPACT OF CHATGPT IMPLEMENTATION ON HR PERFORMANCE SKILLS: A CASE STUDY OF ACADEMIC INSTITUTIONS IN OMAN," *BJMSR*, pp. 11–21, Feb. 2025, doi: 10.46281/bjmsr.v10i1.2281.

[8] K. Taha, P. D. Yoo, C. Yeun, and A. Taha, "A Comprehensive Survey of Text Classification Techniques and Their Research Applications: Observational and Experimental Insights," 2024, *arXiv*. doi: 10.48550/ARXIV.2401.12982.

[9] M. A. AlAfnan, "Artificial Intelligence and Language: Bridging Arabic and English with Technology," *Journal of Ecohumanism*, vol. 4, no. 1, Art. no. 1, 2025, doi: 10.62754/joe.v4i1.4961.

[10] A. Abdelali *et al.*, "LAraBench: Benchmarking Arabic AI with Large Language Models," in *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, Y. Graham and M. Purver, Eds., St. Julian's, Malta: Association for Computational Linguistics, Mar. 2024, pp. 487–520. Accessed: Dec. 17, 2024. [Online]. Available: https://aclanthology.org/2024.eacl-long.30

[11] B. Shannaq, "Enhancing Human-Computer Interaction: An Interactive and Automotive Web Application - Digital Associative Tool for Improving Formulating Search Queries," in *Advances in Information and Communication*, vol. 921, K. Arai, Ed., in Lecture Notes in Networks and Systems, vol. 921. , Cham: Springer Nature Switzerland, 2024, pp. 511–523. doi: 10.1007/978-3-031-54053-0_35.

[12] S. I. Abudalfa, F. J. Abdu, and M. M. Alowaifeer, "Arabic Text Formality Modification: A Review and Future Research Directions," *IEEE Access*, pp. 1–1, 2024, doi: 10.1109/ACCESS.2024.3511661.

[13] A. A. Aladeemy *et al.*, "Advancements and challenges in Arabic sentiment analysis: A decade of methodologies, applications, and resource development," *Heliyon*, vol. 10, no. 21, Nov. 2024, doi: 10.1016/j.heliyon.2024.e39786.

[14] Y. H. Farhan, M. Shakir, M. A. Tareq, and B. Shannaq, "Incorporating Deep Median Networks for Arabic Document Retrieval Using Word Embeddings-Based Query Expansion," *Journal of Information Science Theory and Practice*, vol. 12, no. 3, pp. 36–48, Sep. 2024, doi: 10.1633/JISTAP.2024.12.3.3.

[15] A. Wahdan, M. Al-Emran, and K. Shaalan, "A systematic review of Arabic text classification: areas, applications, and future directions," *Soft*

Comput, vol. 28, no. 2, pp. 1545–1566, Jan. 2024, doi: 10.1007/s00500-023-08384-6.

[16]. Boumedyen Shannaq*, "Investigating the Distribution of Arabic and English Keywords and Their Progress Over Different Text File Formats," *American Journal of Computing Research Repository*, vol. 1, no. 1, Art. no. 1, 2013, doi: 10.12691/ajcrr-1-1-1.

[17]. Boumedyen Shannaq* and R. Adebiaye, "Analytic-Synthetic Processing Of Information as Smart-Based Environment for Text Summarization," *International Journal of Innovative Research in Science, Engineering and Technology*, vol. 4, no. 1, Art. no. 1, 2015, doi: 10.15680/IJIRSET.2015.0401055.

[18]. Boumedyen Shannaq, "DEVELOPMENT OF A NEW ENCODING ALGORITHM USING VIRTUAL KEYPAD LETTER SUBSTITUTIONS FOR IMPROVED TEXT CLASSIFICATION," *JATIT*, vol. 103, no. 2, Jan. 2025, [Online]. Available: https://www.jatit.org/volumes/Vol103No2/27Vol103No2.pdf

[19] Y. H. Farhan *et al.*, "UTILIZING WORD EMBEDDING'S FOR AUTOMATED QUERY EXPANSION IN ARABIC INFORMATION RETRIEVAL: A BLENDED METHODOLOGY," *JATIT*, vol. 103, no. 2, Jan. 2025, [Online]. Available: https://www.jatit.org/volumes/Vol103No2/33Vol103No2.pdf

[20]. Boumedyen Shannaq, "IMPROVING SECURITY IN INTELLIGENT SYSTEMS: HOW EFFECTIVE ARE MACHINE LEARNING MODELS WITH TFIDF VECTORIZATION FOR PASSWORD-BASED USER CLASSIFICATION," *JATIT*, vol. 102, no. 22, Nov. 2024, [Online]. Available: https://www.jatit.org/volumes/Vol102No22/27Vol102No22.pdf

[21] H. H. Alfartosy and H. K. Khafaji, "A hybrid technique for arabic text classification using semi-supervised learning," presented at the FIFTH INTERNATIONAL CONFERENCE ON APPLIED SCIENCES: ICAS2023, Baghdad, Iraq, 2024, p. 050003. doi: 10.1063/5.0213293.

[22]. Boumedyen Shannaq* and F. AlAzzawi, "On the Development of Novel Arabic Documents Classifier," *International Journal of Advanced Science and Technology*, vol. 29, no. 3, Art. no. 3, 2020.

[23] T. Le Quy, A. Roy, V. Iosifidis, W. Zhang, and E. Ntoutsi, "A survey on datasets for fairness-aware machine learning," *WIREs Data Min & Knowl*, vol. 12, no. 3, p. e1452, May 2022, doi: 10.1002/widm.1452.

[24] E. Shafie, "An empirical study of extracting embedded text from digital images," *Int. j. adv. appl. sci.*, vol. 10, no. 6, pp. 48–53, Jun. 2023, doi: 10.21833/ijaas.2023.06.006.

[25] S. Bardab, T. Ahmed, and T. Mohammed, "Data mining classification algorithms: An overview," *Int. j. adv. appl. sci.*, vol. 8, no. 2, pp. 1–5, Feb. 2021, doi: 10.21833/ijaas.2021.02.001.

[26] S. Promboonruang and T. Boonrod, "Deep transfer learning CNN based for classification quality of organic vegetables," *Int. j. adv. appl. sci.*, vol. 10, no. 12, pp. 203–210, Dec. 2023, doi: 10.21833/ijaas.2023.12.022.

[27] H. D. Campos and M. V. Caya, "Utilizing convolutional neural networks for the classification and preservation of Kalinga textile patterns," *Int. j. adv. appl. sci.*, vol. 11, no. 6, pp. 229–236, Jun. 2024, doi: 10.21833/ijaas.2024.06.024.

[28] A. A. Alzahrani, A. Bramantoro, and A. Permana, "Multi-label text classification on unbalanced Twitter with monolingual model and hyperparameter optimization for hate speech and abusive language detection," *Int. j. adv. appl. sci.*, vol. 11, no. 5, pp. 177–185, May 2024, doi: 10.21833/ijaas.2024.05.019.

[29] Q. H. Nguyen *et al.*, "Influence of Data Splitting on Performance of Machine Learning Models in Prediction of Shear Strength of Soil," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–15, Feb. 2021, doi: 10.1155/2021/4832864.

[30] A. Mahmoud and A. Mohammed, "Leveraging Hybrid Deep Learning Models for Enhanced Multivariate Time Series Forecasting," *Neural Process Lett*, vol. 56, no. 5, p. 223, Aug. 2024, doi: 10.1007/s11063-024-11656-3.

[31] D. Apostolo, M. S. Santos, A. C. Lorena, N. Japkowicz, and P. H. Abreu, "Exploring the Interplay of Dataset Shift and Imbalance Strategies on Classification Performance," 2024. doi: 10.2139/ssrn.4790488.

[32] H. Taherdoost, "A systematic review of big data innovations in smart grids," *Results in Engineering*, vol. 22, p. 102132, Jun. 2024, doi: 10.1016/j.rineng.2024.102132.

[33] A. Ahmed, R. Xi, M. Hou, S. A. Shah, and S. Hameed, "Harnessing Big Data Analytics for Healthcare: A Comprehensive Review of Frameworks, Implications, Applications, and Impacts," *IEEE Access*, vol. 11, pp. 112891–

112928, 2023, doi: 10.1109/ACCESS.2023.3323574.

[34] A. Kondraganti, G. Narayanamurthy, and H. Sharifi, "A systematic literature review on the use of big data analytics in humanitarian and disaster operations," *Ann Oper Res*, vol. 335, no. 3, pp. 1015–1052, Apr. 2024, doi: 10.1007/s10479-022-04904-z.

[35] A. A. Nafea *et al.*, "A Brief Review on Preprocessing Text in Arabic Language Dataset: Techniques and Challenges," *Babylonian Journal of Artificial Intelligence*, vol. 2024, pp. 46–53, May 2024, doi: 10.58496/BJAI/2024/007.

[36] M. Aloui, H. Chouikhi, G. Chaabane, H. Kchaou, and C. Dhaouadi, "101 Billion Arabic Words Dataset," 2024, *arXiv*. doi: 10.48550/ARXIV.2405.01590.

[37] C. Aliferis and G. Simon, "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI," in *Artificial Intelligence and Machine Learning in Health Care and Medical Sciences*, G. J. Simon and C. Aliferis, Eds., in Health Informatics. , Cham: Springer International Publishing, 2024, pp. 477–524. doi: 10.1007/978-3-031-39355-6_10.

[38] U. Michelucci, "Model Validation and Selection," in *Fundamental Mathematical Concepts for Machine Learning in Science*, Cham: Springer International Publishing, 2024, pp. 153–184. doi: 10.1007/978-3-031-56431-4_7.

[39] G. Ignacz *et al.*, "Machine learning for the advancement of membrane science and technology: A critical review," *Journal of Membrane Science*, vol. 713, p. 123256, Jan. 2025, doi: 10.1016/j.memsci.2024.123256.

[40] S. Szeghalmy and A. Fazekas, "A Comparative Study of the Use of Stratified Cross-Validation and Distribution-Balanced Stratified Cross-Validation in Imbalanced Learning," *Sensors*, vol. 23, no. 4, p. 2333, Feb. 2023, doi: 10.3390/s23042333.

[41] M. Almanea, "Deep Learning in Written Arabic Linguistic Studies: A Comprehensive Survey," *IEEE Access*, vol. 12, pp. 172196–172233, 2024, doi: 10.1109/ACCESS.2024.3488553.

[42] W. Nazih, A. Fashwan, A. El-Gendy, and Y. Hifny, "Ibn-Ginni: An Improved Morphological Analyzer for Arabic," *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 23, no. 2, pp. 1–22, Feb. 2024, doi: 10.1145/3639050.

[43] M. Siino, I. Tinnirello, and M. La Cascia, "Is text preprocessing still worth the time? A comparative survey on the influence of popular preprocessing methods on Transformers and traditional classifiers," *Information Systems*, vol. 121, p. 102342, Mar. 2024, doi: 10.1016/j.is.2023.102342.

[44] "Analyzing the Impact of Preprocessing Techniques on Arabic Document Classification: Comparative Study," *IJATCSE*, vol. 13, no. 6, pp. 228–240, Dec. 2024, doi: 10.30534/ijatcse/2024/041362024.

[45] "The effects of Pre-Processing Techniques on Arabic Text Classification," *IJATCSE*, vol. 10, no. 1, pp. 41–48, Feb. 2021, doi: 10.30534/ijatcse/2021/061012021.

[46] I. Youb, S. Ventura, and M. Hamlich, "Exploring the impact of preprocessing and feature extraction on deep learning-based sentiment analysis for big data in apache spark," *Prog Artif Intell*, Nov. 2024, doi: 10.1007/s13748-024-00355-8.

[47] J. Wang *et al.*, "Statistical Modeling of Spatially Stratified Heterogeneous Data," *Annals of the American Association of Geographers*, vol. 114, no. 3, pp. 499–519, Mar. 2024, doi: 10.1080/24694452.2023.2289982.

[48] G. Aguiar, B. Krawczyk, and A. Cano, "A survey on learning from imbalanced data streams: taxonomy, challenges, empirical study, and reproducible experimental framework," *Mach Learn*, vol. 113, no. 7, pp. 4165–4243, Jul. 2024, doi: 10.1007/s10994-023-06353-6.

[49] S. O. Alhumoud, "CNN-BiGRU-Focus: A Hybrid Deep Learning Classifier for Sentiment and Hate Speech Analysis of Ashura-Arabic Content for Policy Makers," *ijacsa*, vol. 15, no. 11, 2024, doi: 10.14569/IJACSA.2024.0151199.

[50] E.-Y. Daraghmi, S. Qadan, Y.-A. Daraghmi, R. Yousuf, O. Cheikhrouhou, and M. Baz, "From Text to Insight: An Integrated CNN-BiLSTM-GRU Model for Arabic Cyberbullying Detection," *IEEE Access*, vol. 12, pp. 103504–103519, 2024, doi: 10.1109/ACCESS.2024.3431939.

[51] B. Shannaq, "Unveiling the Nexus: Exploring TAM Components Influencing Professors' Satisfaction With Smartphone Integration in Lectures: A Case Study From Oman," *TEM Journal*, pp. 2365–2375, Aug. 2024, doi: 10.18421/TEM133-63.

[52] B. Shannaq, I. Al Shamsi, and S. Abdul Majeed, "Management Information System for Predicting Quantity Martial's," *TEM Journal*, vol. 8, pp. 1143–1149, Dec. 2019, doi: 10.18421/TEM84-06.

[53] M. T. Shakir and B. Shannaq, "Enhancing Security through Multi-Factor User Behavior Identification: Moving Beyond the Use of the Longest Common Subsequence (LCS)," *IJCAI*, vol. 48, no. 19, Nov. 2024, doi: 10.31449/inf.v48i19.6270.

[54] B. Shannaq, "The Role of AI in University Course Registration in the Middle East: AI and Machine Learning Approaches to Improve Academic Performance," in *2024 2nd International Conference on Computing and Data Analytics (ICCDA)*, Shinas, Oman: IEEE, Nov. 2024, pp. 1–6. doi: 10.1109/ICCDA64887.2024.10867316.

[55] J. O. Ukpata, D. E. Ewa, N. G. Success, G. U. Alaneme, O. N. Otu, and B. C. Olaiya, "Effects of aggregate sizes on the performance of laterized concrete," *Sci Rep*, vol. 14, no. 1, p. 448, Jan. 2024, doi: 10.1038/s41598-023-50998-1.

[56] B. Mohamed, "DataSet for Arabic Classification." Mendeley, Jul. 30, 2018. doi: 10.17632/V524P5DHPJ.2.