Journal of Innovative Construction and Petrochemical Technologies, Vol.3, No.2 (December, 2026), 202-217

E-ISSN: 3007-9861 | DOI: https://doi.org/10.71285/icpt.v3i2.43

Journal of Innovative Construction and Petrochemical Technologies
https://icpt.ouk.kz/index.php/icpt/index

Research Article

Check for updates

Improving Arabic Text Classification Accuracy Using Lightweight NLP Techniques

Rasha Majid Hassoon
College of Physical Education and Sports Sciences for Women, University of Baghdad, Iraq
Email: Rasha.m@uobaghdad.edu.iq

Safana Mohammed Qasem
Department of Computer Engineering, College of Engineering, Al-Nahrain University, Baghdad, Iraq
Email: safana@nahrainuniv.edu.iq

Sarah Hikmat Khaled
Ministry of Higher Education and Scientific Research, Baghdad, Iraq
Email: sarah.hikmat@mohesr.edu.iq

Reiam Abd Al-Kareim Abd
College of Physical Education and Sports Sciences for Women, University of Baghdad, Iraq
Email: reiam.a.920@copew.uobaghdad.edu.iq

Rusul Hussein Hasan
University of Baghdad, Baghdad, Iraq
Email: russl@colaw.uobaghdad.edu.iq

Liqaa Mohammad Shoohi
College of Physical Education and Sports Sciences for Women, University of Baghdad, Iraq
Email: liqaa.m@uobaghdad.edu.iq

Abstract: Arabic text classification is a challenging task because of the complex morphology of the language, the existence of different writing forms and a multitude of dialects, which can result in sparser common text representations. While transformer models such as AraBERT have obtained superior results on many Arabic NLP tasks, their high computational requirements make them difficult to deploy in environments with limited hardware resources. In some cases this can also make the model less practical for researchers working with basic computer systems. This study focuses on a more practical issue: how much accuracy a simple classifier may lose when the amount of required computation is reduced. We use a combined TF-IDF representation based on both words and characters, then reduce the number of features using Chi-Square selection. The selected features are finally used with a linear SVM to create a model that is faster and more efficient, while still keeping good predictive performance. However, the method do not always provide the same level of accuracy as more complex models, especially with difficult text. The proposed lightweight method is tested against several other approaches, including word-based TF-IDF, character-based TF-IDF, Naive Bayes, Logistic Regression, BiLSTM, and AraBERT. All methods are tested using the same data distribution to make the comparison fair. The dataset used is the arbml/arabic_100k_reviews corpus, which originally contains 99,999 balanced reviews from three sentiment categories. After cleaning the data and removing very short reviews, the dataset was reduced to 99,759 samples. The training, validation, and testing sets includes 69,831, 9,976, and 19,952 reviews, respectively. This setup allows the different models to be compared under similar conditions and with the same evaluation process. After applying Chi-Square feature selection, the hybrid feature set was reduced from 250,000 to 50,000 dimensions, which represents an 80% reduction. That proposed method achieved 67.90% accuracy and a Macro-F1 score of 67.75%. Its training time was around 41 seconds, while the complete test set was processed in about 0.03 seconds. Among all the evaluated models, AraBERT achieved the best overall performance, reaching 74.08% accuracy and 74.25% Macro-F1. However, its computational requirements were much higher, with nearly 1,760 seconds needed for training and about 47 seconds for inference on the same test set. Depending on the processing stage, this makes AraBERT approximately 40 to 1,500 times slower than the proposed lightweight approach. The proposed method is therefore not presented as a replacement for AraBERT in terms of accuracy, since its accuracy is lower. Instead, its main advantage is the considerable and measurable reduction in computational cost. This trade-off can be useful in situations where GPU availability, memory capacity, or response time are limited.

Keywords: Arabic natural language processing; Arabic text classification; TF-IDF method; Chi-Square feature selection; linear SVM classifier; Logistic Regression; BiLSTM; AraBERT; efficient NLP models; accuracy and efficiency trade-off.

INTRODUCTION

The internet has become such a vast place and Arabic content is increasing so quickly that manual review is hard to achieve for most applications. Automated word processing systems are necessary for tasks like sentiment analysis, meaning extraction, content classification, ranking user comments, social media monitoring, and information retrieval. But there are a lot of difficulties for this systems with Arabic. It is morphologically complex with numerous prefixes, suffixes, word forms, and variations in spelling within and across writers and contexts. Besides, the Arabic language is written in Modern Standard Arabic and many various dialects, some of which may be quite different from the standard form. Due to these differences, a text representation model which is mainly concerned with the English language or less complex languages cannot be readily transferred to Arabic.

TF-IDF is popular because it's easy to understand, is easy to implement, is relatively fast to train and generally performs well. But, it is not very effective with Arabic text. Many words in Arabic can have different morphological components, such as roots, prefixes, and suffixes, that can be found in a variety of forms. These variations can be represented using both word-level and character-level TF-IDF, and provide better text coverage. The problem is that if you have a mix

of both, you can have a lot of features and the model will be large and expensive to run. Feature selection can help solves this issue by remove less useful features while keeping the information that is important for classification.

In recent years, there has been significant progress in the field of Arabic language processing using transformer language models. The Arabic language has seen impressive developments in transformer language models in recent years. AraBERT is one such well-known model, pre-trained with a large Arabic text corpus and that has shown good performance on various Arabic NLP tasks. But that's a high computational price that has to be paid as well. The model can be trained with more computation, memory and time than simpler linear classifiers that rely on a few number of features. These demands can be even more challenging if this system relies heavily on high-performance GPUs and resources or processing power are scarce.

This leads to a practical question that is often given less attention than simply comparing the accuracy of different models:

How a lightweight Arabic text content type pipeline can retain enough accuracy to be profitable while reducing computation cost by a large margin?

We study this question by using a hybrid TF-IDF representation that combines word-level and character-level features. The number of features is then reduced using Chi-Square selection, and a linear SVM is used for classification. We compare this approach with traditional methods, including Naive Bayes and Logistic Regression using word, character, and hybrid TF-IDF features. We also include a BiLSTM model and AraBERT in the comparison. All models are trained and tested using the same data splits, which makes the results more fair and easier to compare.

Contributions

1. A hybrid TF-IDF approach that combines word-level and character-level features for Arabic text classification using just one consistent processing pipeline.
2. The uses of Chi-Square feature selection to reduces the hybrid feature space by 80%, from 250,000 to 50,000 features, with a clear comparison before and after feature reduction.
3. A comparison of ten different models, including Naive Bayes, Logistic Regression, linear SVM, BiLSTM, and AraBERT, use the same evaluation measures and the same training, validation, and test sets.
4. An analysis of that balance between accuracy and computational efficiency, considering accuracy, F1-score, training time, and inference time for each model.
5. An ablation study that examines the separate effect of word features, character features, their combination, and feature selection on that final classification results.

Related Work

The field of Arabic Natural Language Processing (NLP) has seen significant growth since 2021. Much of that growth has been provided by pre-trained converter models like AraBERT, MARBERT or ARBERT. Meanwhile, traditional and lightweight TF-IDF-based and manually-designed features continue to be popular methods. The following part reviews some recent researches related to four topics: (1) Arabic sentiment analysis and aspect-based sentiment analysis, (2) Arabic text classification with TF-IDF and feature selection techniques, (3) Arabic named entity recognition and other general NLP tasks, and (4) lightweight Arabic models that aim to reduce the computational cost of their implementation while preserving efficient performance.

In Aspect-Based Sentiment Analysis (ABSA), Bensolton and Zaki used the AraBERT model and treated Aspect Class Discovery (ACD) for Arabic as a sentence-pair classification problem. Their approach achieved an F1 score of 84.1%, outperforming all previous sentence-embedding-based methods. Their results also highlighted the continuing gap in Arabic natural language processing research and the need for more effective, pre-trained Transformer models in future studies of Arabic texts [1], [2].

Other studies on deep learning have focused on Arabic language tasks related to emotions and opinions. Al-Hassan and Al-Dossari used deeps neural network techniques to identify hate speech on Arabic tweets [3]. Fadel et al. used contextual information to identify related terms in various contexts [4]. In another study, Al-Hassan et al. combined word-based information with deep learning technique to classify emotions on general Arabic texts. These studies demonstrate how deep learning can be use to analyze different types of Arabic texts [5].

In general, these studies show that deep learning models based on transformers and embeddings often achieve significantly better results than traditional models that rely primarily on words and

lexical features. However, dialectal variations in Arabic countries and differences between text domains remain common problems that may affect the performance of these models.

The classification of Arabic texts use TF-IDF features and methods for selecting different features is another important area. In this study, Masada et al. examined different data preprocessing techniques and text representation methods with the application of traditional machine learning classifiers. They concluded that the type of processing used and the type of the set of texts used are important factors to take into account when determining classification accuracy, particularly when short texts are compared to long documents [6]. Al-Zanin et al. employ a combination of genetic algorithm feature selection of useful features of the TF-IDF and BOW representations along with the use of clustering-based classifiers. Their approach yielded F1-scores of 80.88% and 68.76% for the Arabic sentiment analysis and multi-class classification tasks, respectively, indicating that the feature selection can boost the performance of traditional Arabic text classification methods [7]. Because of the complexity of the morphological structure of Classical Arabic other methods of feature selection have also been investigated. In previous work, Hadney and Hajjaj suggested a hybrid feature selection and classification algorithm using two algorithms: the Gray Wolf algorithm and the Firefly algorithm [8] and Building in earlier work, where they used a chaotic Firefly algorithm for feature selection [9]. Sabri and his team applied two methods, firstly TF-IDF along with Word2Vec embedding for dimensionality reduction without losing the semantic information of the Arabic words. They then compared the results with the use of a number of feature reduction and selection techniques such as principal component analysis (PCA), linear discrimination analysis (LDA), chi-squared test and cross-information analysis [10]. Based on the studies, it is obvious that the selection of suitable features and representation of the text is as vital as the selection of the classifier while working with the data set.

Besides text classification and sentiment analysis, other tasks dealing with Arabic natural language processing have been developed in great extent. Al-Anjari and Al-Jathmi examined the impact of the embedding and the preprocessing approach on the quality of the topic modelling. Their study used a large datasets of Arabic news articles and demonstrated the importance of the data preparation and selecting appropriate textual representations for topic modeling [11], Al-Bahli proposed a hybrid model that use feature fusion and mutual attention to identify named entities in Classical Arabic texts. The method was designed to tackle the highly structured morphology of the Arabic language and the issue of overlapping entities [12] and Al-Mousawi and Luqman discuss the evolution of NER from the Arabic perspective and the shift towards transformer-based approaches [13].

With the limited computing resources, that high efficiency and ease of use of Arabic natural language processing techniques came under focus recently. It is critical when it comes to Arabic language applications that don't have the power of very big models or powerful hardware. dzNLP's researchers, Leshuri et al, have demonstrated that by combining multiple Arabic language classifiers and different n-gram analyzers, weighted features derived from the TF-IDF algorithm can produce very good accuracy in the identification of multiple Arabic dialects. Their solution is an approach that does not require large transformational models [14]. In a recent research work, the Center for Computer Application Development (CVPD) team evaluated the performance of several lightweight Arabic language models like MARBERT, ArabicBERT and AraBERT, alongside with larger API-based Arabic language models on a challenging inference task. The MARBERT-based system attained an accuracy of 69.87%, whereas the best performing large Arabic language system attained an accuracy of 87.6%. Despite the fact that the bigger model had much higher accuracy, other considerations were taken into account including computational efficiency, ease of deployment across various devices, and privacy. That comparison was not only based on the model's accuracy but also its usability in real world. [15].

In general, the findings of previous studies agree with that of this work. The high accuracy of Arabic language models can be achieved with transformer models, but they have a high computational cost. Meanwhile, simpler methods based on TF-IDF and feature selection can yield reasonable results at a much lower computational cost. This is an accuracy and efficiency issue which this study aims to rectify.

Classical Approaches to Arabic Text Classification

Initially, the Arabic text classification systems were based on features that were manually extracted and classic machine learning approaches. The popularity of TF-IDF is due to the fact

that it assigns higher weight to words that are significant to the document rather than the entire set of documents and it doesn't require any extra classifying data other than that required to classify the documents. Another popular choice was naive Bayes since it does not require much computation and is simple. In text classification, the large number of features is frequently sparse, and logistic regression and support vector machine models are known to perform very well. The language of Arabic poses another problem since the same word can be found in several forms and meanings that vary according to the context of the sentence. This, in turn, could lead to the spread of vocabulary over this different forms, making data more variable and lowering statistical power of the single classifier.

Character-Level Representations

n-gram models are useful in the word segmentation. They work with words parts, not whole words, so they can identify prefixes, suffixes, word roots, and common spelling patterns regardless of word form. This is especially helpful in Arabic, which has a number of similar words with different forms of the same root that have the same meaning. In this study, it is tested at word level and letter level of TFIDF and then combined a results to understand the contribution of each model.

Deep Learning for Arabic NLP

In the field of classifying Arabic texts, traditional machine learning approaches have been replaced by more popular neural networks, such as convolutional neural networks (CNNs), long-term memory networks (LSTMs), and bidirectional long-term memory networks (BiLSTMs). So far, these models have yielded different outcomes, but they are still very successful in a host of different tasks. A BiLSTM reads the text both forward and backward from the start to the end of the text, allowing the network to infer relationships between words that may appear to be unrelated. This could make the BiLSTM model more advantageous than the simple "wordbag" methods that are based on word repetition. However, its performance can vary depending on factors such as vocabulary size, sequence length, embedding quality, and training settings. It is not necessarily superior to carefully-tuned sparse feature techniques. The study employs the intermediate model of the neural network as an intermediate model between the traditional machine learning methods and newer transformer_based models, namely BiLSTM.

Transformer-Based Arabic Language Models

Pre-trained models for Arabic NLP tasks have transformed the way that these tasks are tackled. These models are trained to extract contextual information from the massive amount of Arabic text, avoiding the need to manually develop task-specific properties. The AraBERT model, specifically trained for the Arabic, has achieved outstanding results across numerous Arabic natural language processing criteria. In the study, we take `aubmindlab/bert-base-arabertv02` as the base model for a transformer. With limited computing resources, this research is not primarily designed to prove that the lightweight model is significantly more accurate than AraBERT, but aims to see if a simpler, lighter model can produce useful results. That is, the emphasis is on achieving a realistic compromise between the model's accuracy and its expense.

Research Gap and Objectives

Recent studies in the Arabic text classification have largely focused on improving accuracy by using larger, more powerful, pre-trained Transformer models. While this approach performs well in the benchmark tests, it is not always suitable for practical situations where computing resources are a very limited. For example, small organizations, embedded systems, local servers, and large-scale data processing may suffer from limited access to GPUs, the memory, or latency. In such cases, using a large Transformer model may not be practical or even feasible. Few studies in a field of Arabic text classification address training time and inference alongside accuracy. Even fewer studies compare different types of models, from simple linear sparse methods to BiLSTM and finally AraBERT, using a same experimental setup. These makes it difficult to clearly understand a trade-off between model accuracy and computational cost.

This study focuses on bridging this gap by comparing the TF-IDF hybrid model with feature selection using the chi-square test, traditional machine learning models, the BiLSTM model, and the AraBERT model using the same dataset and evaluation criteria. This study also presents the prediction results and computational cost of each one of the approach. In ways, the balance between accuracy and efficiency are considered a fundamental part of this evaluation, not merely a secondary issue.

Research Questions

1. Q1: Does using both word-level and character-level TF-IDF features together improve classification results compared to using only one of them?
2. Q2: Does feature selection using this chi-square test improve classification results, or does it primarily reduce the number of features with a slight decrease in accuracy?
3. Q3: How does the performance of the proposed lightweight method compare with Naive Bayes, Logistic Regression, and Linear SVM?
4. Q4: What is the difference in accuracy between this proposed method and AraBERT, and is this difference in accuracy justified when compared to the computational costs?
5. Q5 : What is this trade-off between accuracy and efficiency when measured use actual numerical results?

Research Hypotheses

1. H1: Combining TF-IDF word- and character-level features is expecting to outperform either representation.
2. H2: Implementing feature selection use the chi-square test are expected to yield better results than using this full hybrid feature set.
3. H3: The proposed lightweight framework is expecting to require significantly less training and inference time than AraBERT.
4. H4: AraBERT is expected to achieve higher accuracy and higher Macro-F1 scores compared to lightweight TF-IDF-base models.

METHOD

Dataset

The experiments used the arbml/arabic_100k_reviews dataset. It contains 99,999 Arabic reviews that are evenly divided into three sentiment classes [16], as presented in Table 1.

Table 1. Class distribution of the original dataset.

Class	Number of Samples
Class 0	33,333
Class 1	33,333
Class 2	33,333
Total	**99,999**

Data Preprocessing

The initial reviews were cleaned use a standardized data preprocessing process. Missing or invalid records and duplicate reviews were removed first. The Arabic text was then standardized by unifying different character shapes and removing unnecessary symbols. Punctuation and extra spaces are also removed from the dataset. Finally, reviews that were too short were discarded because they likely lacked sufficient information for accurate emotion categorization. Only a small amount of the data was deleted during the process. The dataset size was reduced from 99,999 to 99,759 reviews, representing 99.76% of the original data. The table 2 illustrate the category distribution and provide an overview of the feature processing path.

Data Split

The purified data are divided into training, validation, and testing groups. A fixed random seed value of 42 was used to ensure the same division could be reproduced, as shown in Table 2 below.

Table 2. train, validation, test split.

Split	Samples	Percentage
Training	69,831	70%
Validation	9,976	10%

Split	Samples	Percentage
Test	19,952	20%
Total	**99,759**	**100%**

Proposed Framework

The proposed method first creates two separate TF-IDF representations, one word-based and the other character-based. These two representations are then combined into a single hybrid feature set. Feature selection using a chi-squared test is applied to this combined set to reduce its size before use a linear support vector machine for classification. The original hybrid representation contains 250,000 features, while the selection process reduces its size to 50,000 features. This means that the feature space is reduced by 80%, calculated as follows: (250,000 - 50,000) / 250,000 × 100.

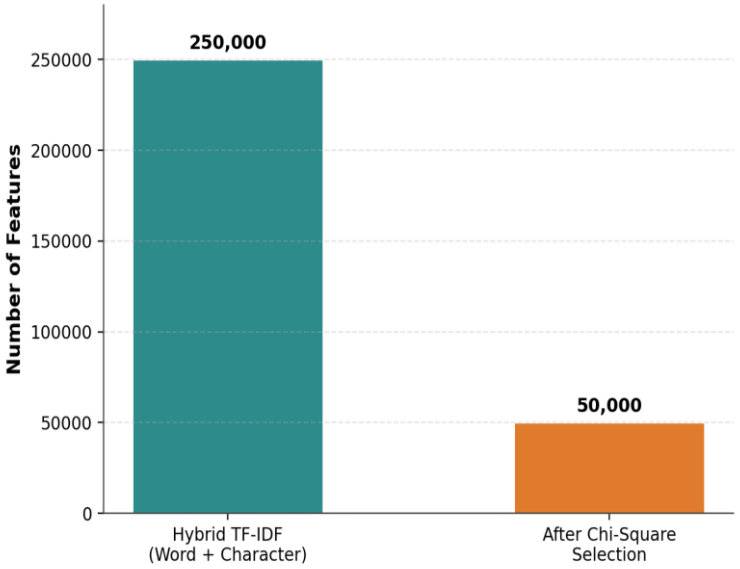

Figure 1. Feature space before and after Chi-Square selection.

Baseline Models

Ten different models are tested in this research. These included: word-level TF-IDF with Naive Bayes, logistic regression, and linear SVM; character-level TF-IDF with linear SVM; and hybrid TF-IDF with Naive Bayes, logistic regression, and linear SVM. The proposed method, combining hybrid TF-IDF, feature selection uses the chi-square test, and SVM, was also evaluated. Additionally, a two-way LSTM model and the AraBERT model were included in this experiments.

Evaluation Metrics

Models were evaluated using several metrics to measure algorithm performance, including accuracy, fine-tuning, and recall for each category, as well as this overall F1 score, the weighted F1 score, training time, and a time required to classify the entire test set. Because this dataset contains three balanced categories, accuracy and the overall F1 score are used as the two primary metrics. The overall F1 score is calculated by taking this average F1 score for all categories without assigning additional weight to any one category. This makes it useful for verifying that all three categories perform equally.

RESULT AND DISCUSSION

Result

Classical Machine-Learning Results

Table 3 presents the main results for all the classical model configurations. It includes several metrics for measuring algorithm performance on this given dataset, such as accuracy, overall F1 score, training time, inference time, and feature space size, including the proposed processing

path after the feature reduction using the chi-square test.

Table 3. Classical machine-learning results, ranked by accuracy.

Model	Accuracy	Macro-F1	Training Time (s)	Inference Time (s)	Features
Hybrid + Logistic Regression	69.13%	69.07%	119.75	0.120	250,000
Word TF-IDF + Logistic Regression	68.92%	68.81%	20.15	0.014	100,000
Proposed: Hybrid + Chi-Square + SVM	**67.90%**	**67.75%**	**40.28**	**0.029**	**50,000**
Character TF-IDF + SVM	66.92%	66.78%	25.72	0.071	150,000
Hybrid + Naive Bayes	66.68%	66.79%	0.33	0.086	250,000
Hybrid + SVM (no selection)	66.59%	66.48%	36.07	0.088	250,000
Word TF-IDF + Naive Bayes	66.41%	66.42%	0.034	0.008	100,000
Word TF-IDF + SVM	66.40%	66.26%	3.35	0.015	100,000

Among the traditional models we tested on this aforementioned data, the TF-IDF hybrid model with logistic regression achieved a best results, with an accuracy of 69.13% and a Macro-F1 value of 69.07%. Its performance outperformed this proposed model after feature selection using a chi-square test. This result are significant because reducing the number of features does not necessarily lead to higher accuracy. A main advantage of this proposed method is that it reduces the feature space before inputting features into the algorithm by 80% while maintaining the accuracy of 67.90%. This balance between a smaller model and acceptable performance are the central focus of this study.

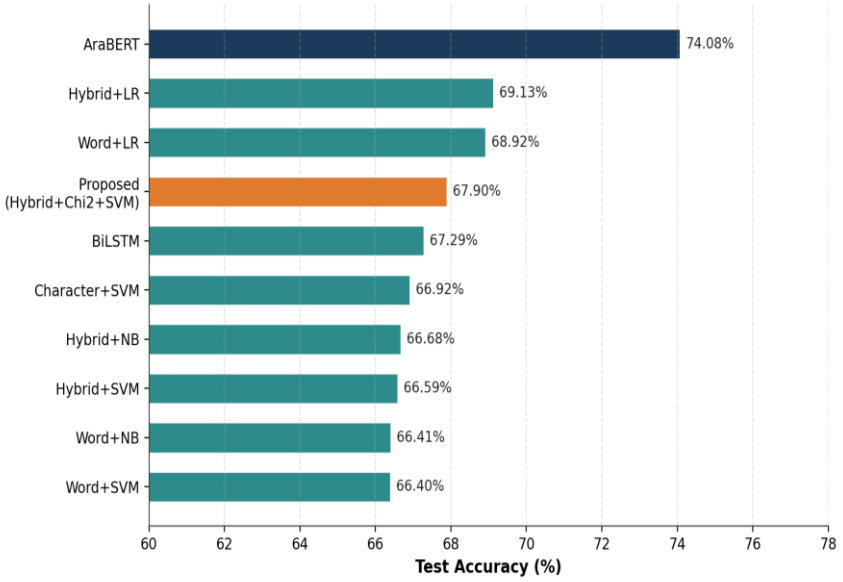

Figure 2. Test accuracy across all ten evaluated models.

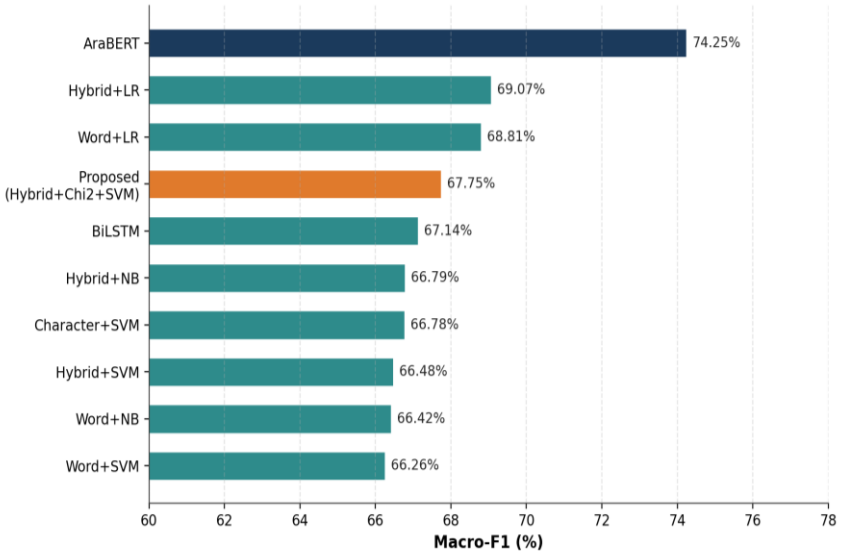

Figure 3. Macro-F1 across all ten evaluated models.

Ablation Study

Table 4 illustrates an effect of each feature design individually. It compares features containing only words, features containing only the letters, a combination of both, and hybrid features after applying chi-square testing.

Table 4. Ablation results.

Configuration	Accuracy	Macro-F1
Word TF-IDF only	66.40%	66.26%
Character TF-IDF only	66.92%	66.78%
Hybrid (word + character)	66.59%	66.48%
Hybrid + Chi-Square selection	**67.90%**	**67.75%**

Character-level properties perform slightly better than word-level properties when used alone. However, combining both types of the properties did not improve the result shown in a table 4 and a figure 4. In fact, the hybrid representation without property selection performed slightly worse than a character-only model. This may indicate that combining all properties creates some redundancy that a classifier must address. After applying chi-square selection, the hybrid representation became significantly more effective. The accuracy increase from 66.59% to 67.90%, an increase of 1.31 percentage points. Therefore, in this case, chi-square selection not only reduces the number of properties to be trained but also helps eliminate the least useful properties in the model and improves the classification result.

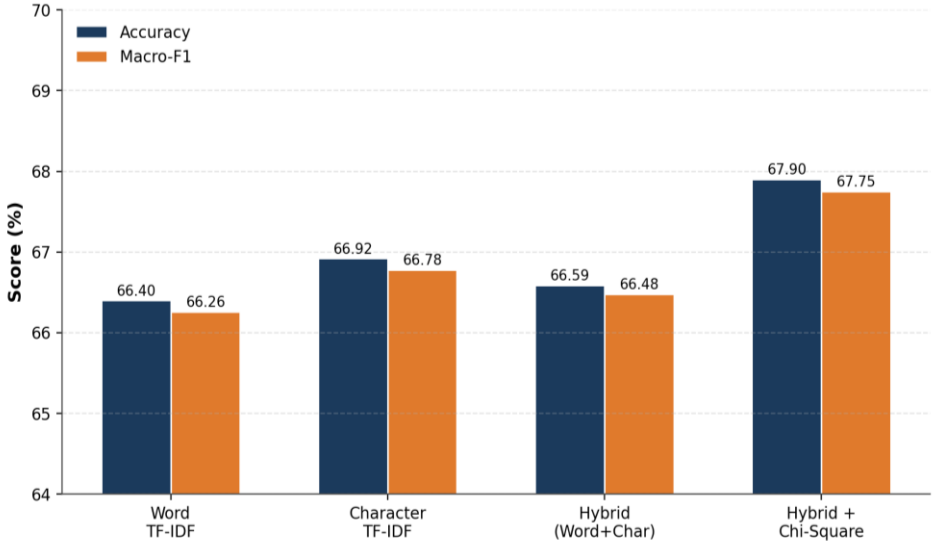

Figure 4. Ablation study: effect of representation and feature-selection choices.

Per-Class Performance of the Proposed Model

Table 5 shows the test results of the proposed model for each class separately.

Table 5. Classification report for the proposed model.

Class	Precision	Recall	F1-Score
Class 0	70.67%	74.00%	72.30%
Class 1	60.30%	56.42%	58.30%
Class 2	72.06%	73.29%	72.67%
Macro Average	**67.68%**	**67.91%**	**67.75%**

In table 5 shown the model performs very well in categories 0 and 2, with F1 values exceeding 72% in both. Category 1, however, is a weakest performer, with an F1 value of 58.30%. This is approximately 14 points lower than an other two categories and has the greatest impact on the overall F1 value. The difference is significant because it highlights the weakness in our model. Limited lexical features may not be able to accommodate an ambiguous and context-dependent language often found in a middle-emotion category of the dataset.

Error Analysis

The proposed model made incorrected predictions for 6,404 out of 19,952 test samples in a test dataset. Table 6 and Figure 5 illustrate this distribution of these errors across a three correct categories.

Table 6. Misclassified samples by true class.

True Class	Misclassified Samples	Share of Total Errors
Class 0	1,729	27.0%
Class 1	**2,900**	**45.3%**
Class 2	1,775	27.7%

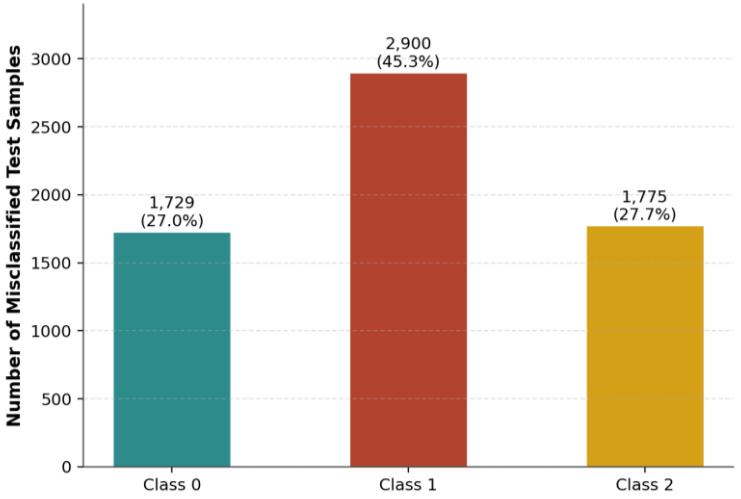

Figure 5. Distribution of misclassified samples by true class – Proposed Model (n= 6,404).

Figure 5 shows that a category 1 is responsible for approximately 45% of all classification errors. These are consistent with its low F1 value, as previously mentioned. Manual review of some of these errors indicates that negation, condensed words, mixed or weak emotions, and context-dependent phrases were common causes of these errors. These were indications the very limited lexical features, which do not take word order into account, may be difficult to detect it.

BiLSTM Results

Table 7. BiLSTM results

Metric	Result
Accuracy	67.29%
Macro Precision	67.04%
Macro Recall	67.30%
Macro-F1	67.14%
Weighted-F1	67.13%
Training Time	93.50 s
Inference Time	4.07 s
Vocabulary Size	30,000

The BiLSTM model did not outperform a most powerful classical TF-IDF models. Its results were very close to those of a lightweight model proposed in this study, without showing any clear improvement in accuracy. However, it took more than twice as long to train as other model. Based on a training settings used in this study, employing the more complex neural architecture did not provide clear advantage in classification accuracy.

AraBERT Results

Table 8. AraBERT results.

Metric	AraBERT
Accuracy	74.08%
Macro Precision	74.52%
Macro Recall	74.08%
Macro-F1	74.25%
Weighted-F1	74.25%

Metric	AraBERT
Training Time	1,760.45 s
Inference Time	46.84 s
Parameters	135.20 million

The AraBERT model achieved a best validation results in the second phase, with the accuracy of 74.57% and a Macro-F1 score of 74.70%. In the final test set, its accuracy decreased slightly to 74.08%. Overall, the AraBERT model performed best in a study in terms of prediction accuracy.

Overall Comparison

Table 9. All ten models ranked by accuracy.

Rank	Model	Accuracy	Macro-F1	Training Time
1	AraBERT	74.08%	74.25%	1,760.45 s
2	Hybrid + Logistic Regression	69.13%	69.07%	119.75 s
3	Word TF-IDF + Logistic Regression	68.92%	68.81%	20.15 s
4	**Proposed: Hybrid + Chi-Square + SVM**	**67.90%**	**67.75%**	**40.98 s**
5	BiLSTM	67.29%	67.14%	93.50 s
6	Character TF-IDF + SVM	66.92%	66.78%	25.72 s
7	Hybrid + Naive Bayes	66.68%	66.79%	0.33 s
8	Hybrid + SVM (no selection)	66.59%	66.48%	36.07 s
9	Word TF-IDF + Naive Bayes	66.41%	66.42%	0.034 s
10	Word TF-IDF + SVM	66.40%	66.26%	3.35 s

Accuracy–Efficiency Trade-off

The AraBERT model achieved 6.18 percentage points higher accuracy than the proposed lightweight framework (74.08% vs. 67.90%). However, the improvement comes at the significantly higher computational cost. Training takes approximately 1760 seconds instead of about 41 seconds, roughly 43 times longer (1760.45 / 40.98 ≈ 43.0). The difference becomes even more pronounced during inference. AraBERT requires approximately 46.84 seconds to process an entire test set, while the lightweight model takes only about 0.03 seconds. The represents the difference of approximately 1536 times (46.84 / 0.0305 ≈ 1536). Figures 6 and 7 shows these differences use a logarithmic scale, while Figure 8 compares between accuracy to training time and uses bubble size to represent inference time. Together, these figures facilitate understanding a trade-off between accuracy and efficiency across all ten models used on a dataset.

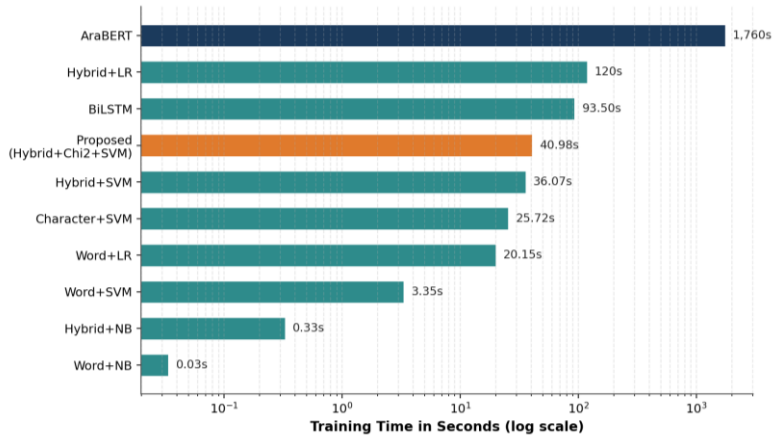

Figure 6. Training time comparison across models (log scale).

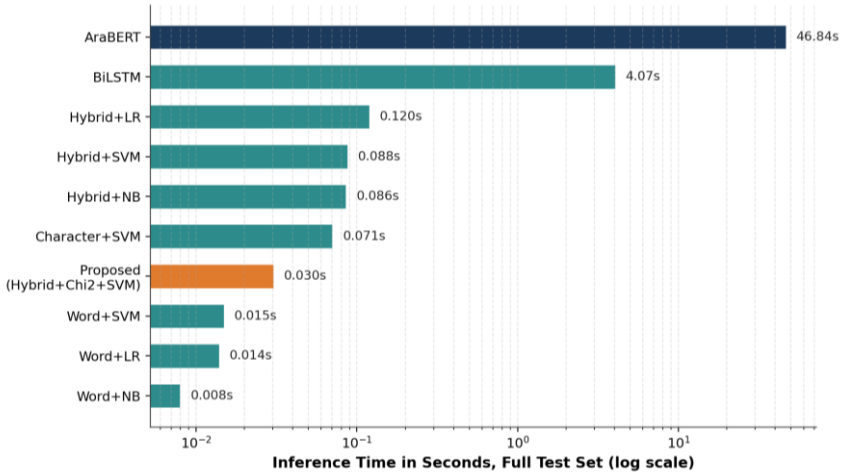

Figure 7. Inference time comparison across models, full test set (log scale).

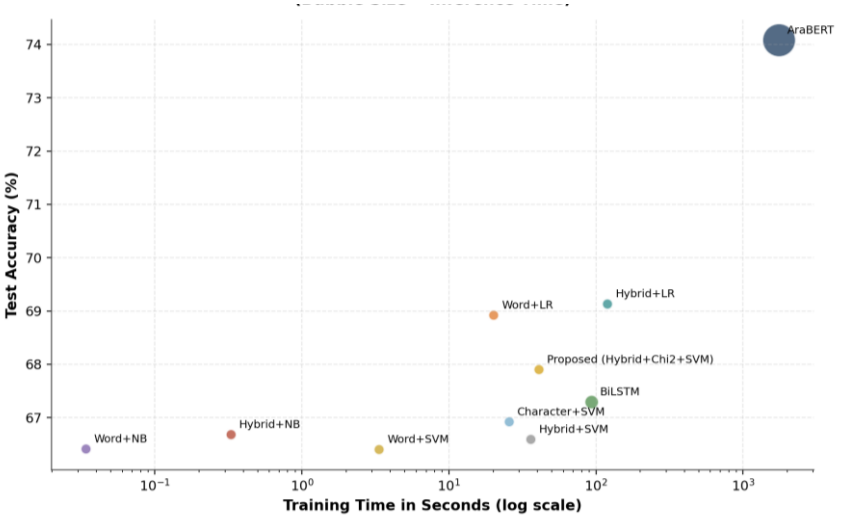

Figure 8. Accuracy–efficiency trade-off; bubble size reflects inference time.

Neither is there any one model which excels in every aspect. If accuracy is the most important one, then an AraBERT model is the best model to use. But, if training time or inference speed or feature space size is a significant constraint then, a proposed TF-IDF + Chi-Square + SVM Hybrid approach can be the viable (albeit less accurate) alternative. This can include things such as CPU-only systems, when memory is a concern, or large-scale batch processing. The approach can decrease the accuracy by around 6 percentage points, but can provide 10 to 3-fold speed-ups if either training or inference is a critical constraint.

Discussion

Three key points can be drawn from experiments we conducted in this study. First, the AraBERT model clearly outperform in terms of accuracy, overall F1 score, and weighted F1 score. Its contextual representations is more effective for this task than sparse lexical features. The result is not surprising, but it is worth confirming for this particular dataset. Second, this superior performance comes at the very high computational cost:

The AraBERT model takes approximately 29 minutes to train, while a simplified method completes training in under a minute. A difference is also evident during testing: AraBERT requires roughly 47 seconds to process an entire test set, compared to only about 0.03 seconds for a simplified method. This are the significant difference for applications requiring rapid response or short training times.

In additional, feature selection had a stronger impact than simply combining features. When TF-IDF data were combined at a word and character levels without removing any features, a Macro-F1 value were 66.48%, very close to the individual representations. After applying chi-square selection, a Macro-F1 value increased to 67.75%. Thus, removing 80% of a features actually improved the results. The suggests that many of a removed features were redundant or added clutter rather than useful information.

One of the interesting results are that a hybrid TF-IDF model with logistic regression outperformed the proposed Supporting Vector Machine (SVM)-based method, despite use five times as many features. This demonstrates that a quality of feature representation also depends on a classifier used. The choice of features and a choice of classifier can influence each other, so they should not be considered in isolation.

For this reason, we do not describe a proposed framework as the best or most accurate model in this study, as it is not. Its primary value lies in a significant reduction in computational cost while maintaining the moderate and noticeable decrease in the accuracy. Although this differs from simply achieving a highest accuracy, it remains a significant advantage for practical applications with very limited resources.

Why AraBERT Performs Better

TF-IDF (counts at word level and character level) considers a document as a bag of weighted words. It doesn't have a way of noting that 'جيد جداً')very good(and 'ليس جيداً')not good(have the same root but opposite meanings (one positive, the other negative). Unlike, the Transformer model builds representations that are sensitive to word order, negation and intensification that plays an important role in emotion categorization. The through morphological variation and the implicit or mixed emotions, both of which contextual models do better with than the overall "word pack" model, complicates Arabic further.

Why the Lightweight Model Still Has a Place

But it's not like the lightweight framework is of no use at all. Many real-world systems use only CPUs, have a relatively small amount of memory, demand a quick response, or a high number of the documents must be processed, so that the cost of running the data converter is significant. Typical environments are educational labs in Universities, on-premises corporate servers, peripheral or embedded devices. Here, it may be a practical and useful option to choose the model with an approximate accuracy of 68%, which can be trained in less than 1 minute, rather than a weaker alternative, as it will process the entire test set in about 0,03 seconds.

Limitations

1. Single Dataset: an experiments used only one dataset for reviews, so a generalizability of the results to other types of Arabic texts was not be tested.
2. Limited Scope: a dataset primarily consists off product and service reviews and may not include news, social media, medical, political, or dialectal Arabic texts.
3. TF-IDF Limitations: a divergent characteristics of TF-IDF have limited contextual understanding, making it difficult to fully match an AraBERT model.
4. No Statistical Testing: a differences mentioned are descriptive only. Statistical tests should be performed before presenting the final results.
5. Limited Adjustment: a SVM algorithm was tested use a small range of parameters; expanding a search scope may improve the results.
6.

Future Work

The framework needs to be tested in other Arabic dialects and domains to ensure its generalizability and to better account for the accuracy gap in future studies. Other techniques may help to achieve a more balanced level of accuracy and efficiency. It is also worthwhile to explore the multi-dataset tests, the clustering methods, an interpretation of the tests, statistical significance tests, and direct measurement of memory and power use in each model.

CONCLUSION

This study compared traditional machine learning, the BiLSTM network, and the AraBERT model for the classifying Arabic texts, with a particular focus on the lightweight processing path based on hybrid TF-IDF word-character data, feature selection using the chi-square test, and a linear SVM algorithm. This is the path reduced the hybrid feature space from 250,000 to 50,000 dimensions an 80% reduction achieving an accuracy of 67.90% and a Macro-F1 value of 67.75%. It required approximately 41 seconds to train and about 0.03 seconds to compute the results of the entire test set. The AraBERT model was a most powerful overall, with an accuracy of 74.08% and a Macro-F1 value of 74.25%, but it required approximately 1,760 seconds to train and about 47 seconds to infer the results of a same test set.

The practical conclusion was that there are trade-offs, not a single option: AraBERT is the optimal choice when the maximum accuracy is a priority when selecting a model, and the computing budget allows for it; a proposed lightweight framework is a reasonable option when speed, simplicity, and a small feature set are more important than achieving a highest accuracy. Lightweight Arabic language processing techniques still have their place alongside converter-based models, especially in very resource-constrained deployment scenarios, where the difference between 0.03 seconds and 47 seconds per test set is not a theoretical detail.

REFERENCES

[1] R. Bensoltane and T. Zaki, "Towards Arabic aspect-based sentiment analysis: A transfer learning-based approach," Social Network Analysis and Mining, vol. 12, no. 1, pp. 1–16, 2022, doi: 10.1007/s13278-021-00794-4.

[2] R. Bensoltane and T. Zaki, "Aspect-based sentiment analysis: An overview in the use of Arabic language," Artificial Intelligence Review, vol. 56, no. 3, pp. 2325–2363, 2023, doi: 10.1007/s10462-022-10215-3.

[3] A. Al-Hassan and H. Al-Dossari, "Detection of hate speech in Arabic tweets using deep learning," Multimedia Systems, vol. 28, no. 6, pp. 1963–1974, 2022, doi: 10.1007/s00530-020-00742-w.

[4] A. S. Fadel, M. E. Saleh, and O. A. Abulnaja, "Arabic aspect extraction based on stacked contextualized embedding with deep learning," IEEE Access, vol. 10, pp. 30526–30535, 2022, doi: 10.1109/ACCESS.2022.3159252.

[5] N. Elhassan, G. Varone, R. Ahmed, M. Gogate, K. Dashtipour, H. Almoamari, M. A. El-Affendi, B. N. Al-Tamimi, F. Albalwy, and A. Hussain, "Arabic sentiment analysis based on word embeddings and deep learning," Computers, vol. 12, no. 6, Art. no. 126, 2023, doi: 10.3390/computers12060126.

[6] M. Masadeh, A. Moustapha, B. Sharada, J. Hanumanthappa, K. Hemachandran, C. Chola, and A. Y. Muaad, "Investigating the impact of preprocessing techniques and representation models on Arabic text classification using machine learning," International Journal of Advanced Computer Science and Applications, vol. 15, no. 1, 2024, doi: 10.14569/IJACSA.2024.01501110.

[7] S. M. Alzanin, A. Gumaei, M. A. Haque, and A. Y. Muaad, "An optimized Arabic multilabel text classification approach using genetic algorithm and ensemble learning," Applied Sciences, vol. 13, no. 18, Art. no. 10264, 2023, doi: 10.3390/app131810264.

[8] M. Hadni and H. Hjiaj, "A new metaheuristic optimization technique for solving feature selection and classification problems for Arabic text," in Arabic Language Processing: From Theory to Practice, Commun. Comput. Inf. Sci., vol. 2340, pp. 221–235, Springer, 2024, doi: 10.1007/978-3-031-80438-0_17.

[9] M. Hadni and H. Hjiaj, "New model of feature selection based chaotic firefly algorithm for Arabic text categorization," The International Arab Journal of Information Technology, vol. 20, no. 3A, pp. 461–468, 2023, doi: 10.34028/iajit/20/3A/3.
[10] T. Sabri, S. Bahassine, O. El Beggar, and M. Kissi, "An improved Arabic text classification method using word embedding," International Journal of Electrical and Computer Engineering, vol. 14, no. 1, pp. 721–731, 2024, doi: 10.11591/ijece.v14i1.pp721-731.
[11] H. Alangari and N. Algethami, "Exploring the effects of pre-processing techniques on topic modeling of an Arabic news article data set," Applied Sciences, vol. 14, no. 23, Art. no. 11350, 2024, doi: 10.3390/app142311350.
[12] S. Albahli, "An advanced natural language processing framework for Arabic named entity recognition: A novel approach to handling morphological richness and nested entities," Applied Sciences, vol. 15, no. 6, Art. no. 3073, 2025, doi: 10.3390/app15063073.
[13] T. El Moussaoui and C. Loqman, "Advancements in Arabic named entity recognition: A comprehensive review," IEEE Access, vol. 12, pp. 180238–180266, 2024, doi: 10.1109/ACCESS.2024.3491897.
[14] M. Lichouri, K. Lounnas, Z. N. Zahaf, and A. Rabiai, "dzNLP at NADI 2024 shared task: Multi-classifier ensemble with weighted voting and TF-IDF features," in Proc. 2nd Arabic Natural Language Processing Conf. (ArabicNLP), 2024, pp. 754–757, doi: 10.18653/v1/2024.arabicnlp-1.84.
[15] S. E. Bekhouche, A. Z. Sellam, H. Telli, C. Distante, and A. Hadid, "CVPD at QIAS 2025 shared task: An efficient encoder-based approach for Islamic inheritance reasoning," arXiv preprint arXiv:2509.00457, 2025.
[16] ARBML, "Arabic 100k reviews," Hugging Face, Dataset. Online Available: https://huggingface.co/datasets/arbml/arabic_100k_reviews