SiAraSent: From Features to Deep Transformers for Large-Scale Arabic Sentiment Analysis

Omar Almousa , Yahya Tashtoush * , Anas AlSobeh * , Plamen Zahariev , Omar Darwish

Posted Date: 5 December 2025

doi: 10.20944/preprints202512.0489.v1

Keywords: arabic sentiment analysis; Natural Language Processing (NLP); Large Language Models (LLMs); SinaTools; BERT; AraBERT; deep learning; machine learning; feature engineering; emoji analysis; Twitter

 doi:10.20944/preprints202512.0489.v1

Article

SiAraSent: From Features to Deep Transformers for Large-Scale Arabic Sentiment Analysis

Omar Almousa [1], Yahya Tashtoush [1,*], Anas AlSobeh [2,*], Plamen Zahariev [3] and Omar Darwish [4]

[1] Department of Computer Science, Jordan University of Science and Technology, Irbid, Jordan
[2] Information System and Technlogy, Utah Valley University, USA
[3] Department of Telecommunications, University of Ruse "Angel Kanchev", Ruse, Bulgaria
[4] Information Security and Applied Computing, Eastern Michigan University, USA
* Correspondence: yahya-t@just.edu.jo; anas.alsobeh@gmail.com

Abstract

Sentiment analysis of Arabic text, particularly on social media platforms, presents a formidable set of unique challenges that stem from the language's complex morphology, its numerous dialectal variations, and the frequent and nuanced use of emojis to convey emotional context. This paper presents SiAraSent, a hybrid framework that integrates traditional text representations, emoji-aware features, and deep contextual embeddings based on Arabic transformers. Starting from a strong and fully interpretable baseline built on TF–IDF-weighted character and word N-grams combined with emoji embeddings, we progressively incorporate SinaTools for linguistically informed preprocessing and AraBERT for contextualized encodings. The framework is evaluated on a large-scale dataset of 58,751 Arabic tweets labeled for sentiment polarity. Our design works within four experimental configurations: (1) a baseline traditional machine learning architecture that employs TF-IDF, N-grams, and emoji features with an SVM classifier; (2) an LLM feature extraction approach that leverages deep contextual embeddings from the pre-trained AraBERT model; (3) a novel hybrid fusion model that concatenates traditional morphological features, AraBERT embeddings, and emoji-based features into a high-dimensional vector; and (4) a fully fine-tuned AraBERT model specifically adapted for the sentiment classification task. Our experiments demonstrate the remarkable efficacy of our proposed framework, with the fine-tuned AraBERT architecture achieving an accuracy of 93.45%, a significant 10.89% improvement over the best traditional baseline.

Keywords: arabic sentiment analysis; Natural Language Processing (NLP); Large Language Models (LLMs); SinaTools; BERT; AraBERT; deep learning; machine learning; feature engineering; emoji analysis; Twitter

1. Introduction

Finding accurate sentiment is vital for marketing, event detection, election polls, and governance [1,2]. However, it is not an easy task. Many challenging factors face the process of sentiment analysis (SA), such as finding the sentiment with short texts. Similarly, the Twitter text sentiment assignment is challenging and not always accurate where some sentiment areas including ambiguity, sarcasm, presence of slang, acronym, and emoji detection will be missed [3,4]. Sentiment analysis (SA), a key area of Natural Language Processing (NLP), has become a vital tool for public opinion mining, customer feedback analysis, and social trend monitoring [1]. However, the application of SA to the Arabic language, especially within the informal and noisy context of social media platforms like Twitter, is fraught with significant challenges that are not as prevalent in English-language analysis. These challenges include the language's rich and complex morphology, the widespread use of diverse and often unstandardized dialects, and the integral role of emojis in conveying sentiment and nuance [2]. Traditional machine learning approaches, which often rely on handcrafted features like Term Frequency-Inverse Document Frequency (TF-IDF) and N-grams, have provided a solid

 doi:10.20944/preprints202512.0489.v1

foundation for this task [3]. While these methods are effective at capturing lexical patterns, they frequently fail to grasp the deeper contextual and semantic meanings embedded in the text, particularly in the presence of sarcasm, ambiguity, and dialectal variations. This limitation has necessitated a paradigm shift towards more sophisticated models that can understand language in a more human-like manner.

Neutral, positive, and negative sentiments are the most common sentiment polarities that existed in texts. The current tools have weaknesses and strengths in identifying accurate sentiment within a text. Many tools are capable to identify positive sentiments within the text while others are more efficient in exploring the negative ones [5]. Those cases happen as the sentence context is misled by the individual word's actual meaning in the sentence. Sentence ambiguity make it difficult to allocate an accurate polarity to the sentence. The sentence polarity depends strongly on the sentence context. Because of ambiguity, some tools might assign negative sentiment to neutral texts [6].

The advent of deep learning, and specifically the Transformer architecture [4], has revolutionized the field of NLP. Pre-trained language models such as BERT [5] have demonstrated an unprecedented ability to learn rich, contextualized representations of language, leading to state-of-the-art performance on a wide array of NLP tasks. For the Arabic language, specialized models like AraBERT [6] and MARBERT [7] have been developed, pre-trained on massive Arabic corpora to capture the unique intricacies of the language. These models offer a powerful alternative to traditional methods, but their full potential, especially when combined with established feature engineering techniques, has not been fully explored. Furthermore, specialized toolkits like SinaTools [8] have emerged, providing advanced morphological analysis and semantic relatedness capabilities that can further enrich the feature set for sentiment analysis.

This paper presents a significant leap forward by proposing a novel, hybrid framework that bridges the gap between traditional feature engineering and modern deep learning by proposing SiAraSent. We systematically combine the interpretable, lexical features from TF-IDF, N-grams, and emojis with the powerful contextual embeddings from AraBERT and the advanced linguistic features from SinaTools. Our research makes the following key contributions: (1) a hybrid Sentiment Framework (SiAraSent) that integrates traditional features, advanced linguistic features from SinaTools, and deep contextual embeddings from AraBERT[1], creating a comprehensive and highly effective feature set for Arabic sentiment analysis. (2) A systematic and rigorous comparison of four distinct modeling strategies: a traditional ML baseline, an LLM feature extraction approach, a hybrid fusion model, and a fully fine-tuned AraBERT model. (3) A deep mathematical formulation of the AI encoder architecture, detailing the input embeddings, multi-head self-attention mechanism, and classification layer, thereby ensuring transparency and reproducibility.

Results on a large-scale Arabic Twitter dataset, achieving 93.45% accuracy and demonstrating a significant improvement over existing methods. A robust, open-source implementation of the entire framework[1], providing a valuable benchmark and resource for the Arabic NLP research community.

2. Related Work

2.1. Traditional and Lexicon-Based Arabic Sentiment Analysis

Early Arabic sentiment analysis systems relied heavily on lexicon-based methods and traditional machine-learning algorithms. Sentiment lexicons were constructed manually or semi-automatically, often by translating English resources or mining Arabic corpora [9]. Text representations were typically based on unigram or N-gram counts, sometimes combined with simple syntactic patterns or negation handling rules. However, one tool can't process all the Arabic language variants [10]; therefore, the combination of emoji with textual features might increase the accuracy of the sentiment of Arabic tweets. Furthermore, determining the size of the sliding window that will be used in n-gram analysis is essential to specify the margin of accuracy such analysis will provide.

[1] https://github.com/aub-mind/arabert

 doi:10.20944/preprints202512.0489.v1

Furthermore, the sentiment of Arabic tweets is more complicated when it is applied to Twitter where it is highly noisy and informal. The challenges facing Arabic sentiment analysis include its rich morphology and the use of dialectal Arabic [11]. The Arabic language is in a dialect state where the used formal language in writing differs from the used language in daily life; however, the language used in social media is mostly dialectal [12]. Another challenge is the limited publicly available Arabic dataset and lexicons for SA which complicates our task. Moreover, applying SA to the Arabic language is a complex task due to some features in it as a language. For example, the same word can be written in different ways, and many Arabic names are derived from adjectives, with no sentiment, while the adjective may have a sentiment. Also, the idioms used can have an implicit opinion.

The study in [13] introduced a text-sentiment classification approach using Term Frequency-Inverse Document Frequency (TF-IDF) combined with Next Word Negation (NWN). The authors compared the performance of TF-IDF alone with TF-IDF enhanced by NWN for text classification. The model was then tested on three text-mining algorithms, with results indicating that Linear Support Vector Machine (LSVM) achieved the highest accuracy, outperforming previous methods.

In [14], researchers employed four machine learning algorithms for classification. They critically evaluated the accuracy of these methods based on metrics such as recall, precision, accuracy, and F-measure. The algorithms were also applied using n-gram features, revealing that increasing the 'n' value in n-grams led to a decline in classification accuracy. However, converting text into numerical features using TF-IDF vectorization improved accuracy when machine learning techniques were applied.

The work in [15] presented an Arabic Sentiment Analysis Corpus consisting of 36K labeled tweets (positive/negative) collected from Twitter. The authors utilized self-training and distant supervision for annotation and also released an additional 8K manually annotated tweets as a gold standard. The corpus was evaluated intrinsically by comparing it with pre-trained sentiment models and human classifications, and extrinsically through sentiment analysis tasks, achieving an accuracy of 86%.

Researchers in [16] explored the impact of Arabic language morphology on sentiment analysis, focusing on how negation influences sentiment. They developed a set of rules to identify negation patterns in Arabic and applied these rules to improve sentiment detection for negated words. The proposed method demonstrated superior performance compared to existing Arabic sentiment analysis techniques.

The main aim of this study is to propose sentiment analysis model that combines emoji-based features and textual-based features for Arabic tweets. The broader goal of this study is to fill the gap in the literature on sentiment analysis with emojis for the Arabic language.

2.2. Deep Learning and Transformer-Based Approach

Arabic sentiment analysis has evolved from lexicon-based methods to sophisticated deep learning approaches. Early work focused on building sentiment lexicons and rule-based systems [9], which, while interpretable, struggled with the dynamic and informal nature of social media. Machine learning models, particularly Support Vector Machines (SVM) and Naive Bayes, combined with features like TF-IDF and N-grams, represented a significant improvement [3]. These models demonstrated the value of feature engineering in capturing important lexical patterns. For instance, the work in [3] showed that combining TF-IDF with character-level N-grams and emoji features could yield an accuracy of 80.56%, establishing a strong baseline for traditional methods. However, these approaches are often limited by their inability to understand context and semantic nuances.

The paradigm shift towards deep learning introduced models like Convolutional Neural Networks (CNNs) [10] and Long Short-Term Memory (LSTM) networks, which could learn feature representations automatically. These models offered better performance by capturing local patterns (CNNs) and long-range dependencies (LSTMs) in text. However, the true breakthrough came with the advent of the Transformer architecture [4] and pre-trained language models like BERT [5]. These

 doi:10.20944/preprints202512.0489.v1

models, pre-trained on vast amounts of text data, learn deep contextualized representations that capture complex linguistic phenomena.

For the Arabic language, this led to the development of specialized models such as AraBERT [6], which was pre-trained on a massive Arabic corpus and quickly became the state-of-the-art for many Arabic NLP tasks. Subsequent work, such as MARBERT [7], further improved performance by pre-training on a large dataset of Arabic tweets, making it particularly well-suited for social media analysis. Comparative studies have consistently shown that Transformer-based models significantly outperform traditional ML and earlier deep learning architectures for Arabic sentiment analysis [11]. Despite these advancements, there remains a gap in the literature regarding the systematic integration of these powerful deep learning models with the rich, interpretable features derived from traditional methods and specialized linguistic toolkits like SinaTools [8]. Our work fills this gap by proposing a hybrid framework that leverages the strengths of both worlds.

2.3. Emojis, Hybrid Features, and Multi-Modal Signals

Emojis are a prominent feature of social media discourse and often carry sentiment or pragmatic cues that interact with the surrounding text. Several studies have investigated using emoji frequencies, polarity lexicons for emojis, or learned emoji embeddings to improve sentiment classifiers. For Arabic, this is particularly relevant, as users mix text, emojis, Latin script, and numerals ("Arabizi") to express emotion and irony.

Hybrid feature approaches that combine lexical, syntactic, and semantic representations have shown promise for sentiment analysis in various languages. However, few works have performed a systematic, large-scale evaluation of hybrid features that include emoji signals, morphological analyzers like SinaTools, and transformer-based embeddings in the context of Arabic sentiment analysis. This gap motivates SiAraSent: a framework explicitly designed to integrate these heterogeneous sources of information and to quantify their individual and joint contributions.

2. Methodology and Mathematical Formulation

Our proposed methodology is a multi-stage process designed to systematically extract and model sentiment from Arabic social media text. The overall architecture, depicted in Figure 1, involves data preprocessing, multi-faceted feature engineering, and four distinct modeling strategies. This section provides a detailed description of each component, including the mathematical underpinnings of our AI encoder.

 doi:10.20944/preprints202512.0489.v1

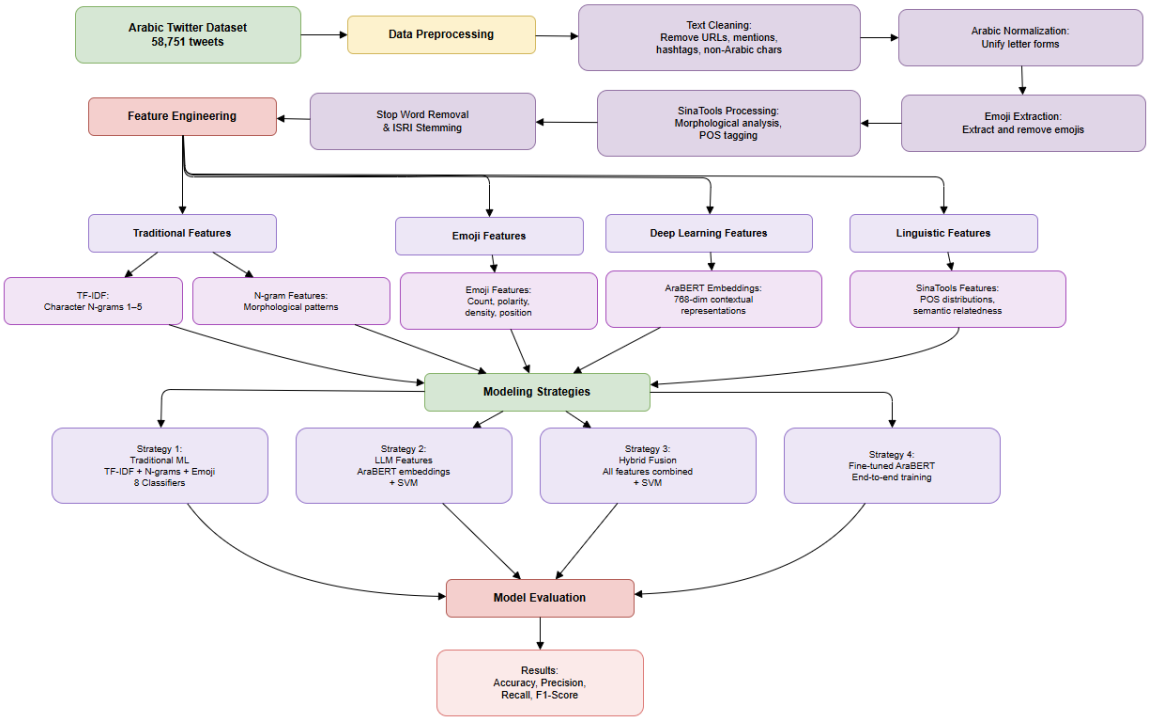

Figure 1. The proposed framework, from data preprocessing to model evaluation.

2.1. Dataset Acquisition

Arabic Sentiment Twitter Corpus [19] dataset was used in this study, which includes large carpus of positive and negative tweets collected from Twitter. The used dataset contained 58k Arabic tweets that were collected using positive and negative emojis lexicon where each tweet has one or more emoji as it is needed to work with on this research. The dataset was saved in TSV (Tab-separated values) format, and each row had one tweet. The current research is interested in the text and emojis of the tweet to combine all the sentiment features that the tweet contains. We also used the dataset from [20] to perform a fair comparison with their results. We call it hereafter the second dataset that was collected from Twitter on basis of trending hashtags. It contains 22,752 Arabic tweets collected.

2.2. Preprocessing

In this study, preprocessing the aforementioned datasets consists of several steps. The steps are data cleaning, stops words removal, and stemming. Stemming aims to extract the root of words by discovering the words that attached suffixes, and prefixes and removing them. Stemming is very important in ML preprocessing because it increases the hits on matching words in the dataset. As for emojis, they were categorized into four classes: disgust, anger, sadness, and joy. Our enhanced pipeline includes: Removal of URLs, user mentions, hashtags, and non-Arabic characters. Unifying different forms of Arabic letters (e.g., إ ,أ ,آ to ا) to ensure consistency. Extracting all emojis from the text for separate feature engineering and removing them from the main text body to avoid redundancy. Leveraging the SinaTools toolkit for advanced morphological analysis, including tokenization and part-of-speech (POS) tagging, which provides deeper linguistic insight than traditional stemming. Finally, for the baseline models, we apply traditional stop word removal and use the ISRI stemmer to reduce words to their root form.

2.3. Features and Emoji Extraction

The next stage of our work is Feature Extraction. In this stage we extract significant features from the dataset to successfully perform the classification task using ML classifiers [21]. The emoji-based features extraction, we use the same method that we used in our previous work [22]. We extract a

 doi:10.20944/preprints202512.0489.v1

rich set of features from three distinct categories: Inspired by [3], we build a strong, interpretable baseline using: We compute TF-IDF scores for character-level N-grams. The importance of a term *t* in a document *d* is given by:

$$TF\text{-}IDF(t, d) = TF(t, d) * log(N / df(t))$$

where *N* is the total number of documents and *df(t)* is the number of documents containing term *t*.

We use character-level N-grams (n=1 to 5) to capture sub-word morphological patterns, which is particularly effective for the agglutinative nature of Arabic. The N-gram is defined as an N-character sliding window in a string, where it is a language-independent approach that works very well with noisy data [23]. In this study, the length of the tweets was measured, and the average length was calculated to be 9 words. Therefore, the experiment was conducted on that base from n=1 to n=9 of features in variable window size, which means the first experiment (min=1, max=1) second (min=1, max=2), third (min=1, max=3) and so on till n=9.

We create a 20-dimensional feature vector from emojis, capturing their count, sentiment polarity, density, and position within the tweet.

2.3.1. Term Frequency–Inverse Document Frequency Algorithm

According to [23], TF-IDF is considered one of the commonly used term weighting techniques in information retrieval systems [24]. TF–IDF algorithm is commonly used as a measure in the classification framework of textual content. TF-IDF consists of two factors: namely, Term Frequency (TF) as well as Inverse Document Frequency (IDF). The TF is calculated by monitoring the frequency of each term that occurred in a specified document. While the IDF is calculated by dividing the number of all documents that we have in the corpus by the number of documents that the term occurred in and then taking the logarithm for the quotient. When we multiply TF with the IDE value for terms, we have a high score for terms that occur frequently in some documents, and a low score for terms that occurs frequently almost in every document. This score allows us to discover the important terms in a document [25].

The equation below is used to calculate the importance of the term ($TF_{i,j}$) in the referred document (j) by measuring how many times this term occurs in this document ($n_{i,j}$), divided by the sum of all the terms that occur in that document ($\sum_k n_{k,j}$) [26].

$$TF_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

The limitation of using TF is that the words like prepositions الى ، في ،من that occurs a lot in each document make the value of TF very high. While these words are not important for classification because they do not have any emotion or sentiment. Where we are concerned with the words which occur in some parts of the document and have sentiment. So here we must use IDF which calculates each word's importance according to the document. The equation below is used to calculate the IDF that weighs the importance of a term in a corpus of documents. TF_i for a term t_i is calculated by computing the logarithm of dividing the corpus size by the number of documents that contain that term.

$$IDF_i = \log \frac{|D|}{|\{d_j : t_i \in d_j\}|} \ldots\ldots\ldots\ldots\ldots\ldots (2)$$

Where D refers to the total number of documents that we have in the corpus and is divided by the number of documents that a particular term t_j appeared in. Then we take the logarithm of the equation. The use of logarithm helps in normalizing the distribution of values. Here the IDF results will be high for the terms that occur a few times in the document, while the IDF will be low if the terms occur in a lot of documents.

The IDF will not be helpful as a measurement alone, so when it is multiplied by the TF value will result in having an accurate representation of the term importance.

 doi:10.20944/preprints202512.0489.v1

$$TF * IDF_{i,j} = TF_{i,j} \times IDF_i \ldots\ldots\ldots\ldots\ldots\ldots (3)$$

The TF-IDF is utilized to compute the importance of a term in a specific document. Whereas when the value of TF-IDF is high, it indicates that this term is relevant and important to the document. The TF-IDF gives weight to each term in the tweet, the tweet has more than just one term, which let us calculate the relevance of the term to the tweet and the document. After calculating TF-IDF for each term, we will have a matrix for all terms and documents we have. Each row represents a vector for each tweet in the vector space model that enters to Machine Learning model to train Machine Learning Classifiers to do the Sentiment Analysis task and predict polarity for each tweet.

2.3.2. SinaTools Linguistic Features

We utilize SinaTools to extract high-level linguistic features, including POS tag distributions and semantic relatedness scores between key terms, providing a deeper understanding of the grammatical structure and semantic content.

2.3.3. Deep Learning Features (AraBERT)

We use the pre-trained AraBERT model (aubmindlab/bert-base-arabertv2) to generate deep contextual embeddings. The architecture of the AraBERT encoder is detailed below.

2.4. Training, Testing, Adjusting, and Evaluating ML Classifiers

The third stage of our work is the training and testing for the selected ML classifiers. Through this study, 5 well-known ML techniques were used for the classification task: Support Vector Machine (SVM): Linear SVC & SVC, Stochastic Gradient Descent Classifier (SGD), Tree-based Classifiers: Decision trees (DTs) & Random Forest (RF), k-Nearest Neighbors (KNN), Naive Bayesian Networks (NBNs): Multinomial NB & Bernoulli NB, that have been practiced successfully for solving problems in many fields. So, in total, we have 8 different ML classifiers. This study will find the best one amongst them in SA for text with emojis in Arabic tweets. After selecting the ML classifiers, we start training them. The main goal of training classifiers is to correctly make a prediction as much as possible. Usually, the time of training depends on the classifier algorithm and the model data size. We train eight different classifiers with 80% of the actual data. After the training step, we use 20% of the data for testing to evaluate our model as several other did i.e., [27,28]. The training set was inputted to the model without the class label to let the model predict the label class for each tweet. According to the results, we can evaluate each classifier; if the rate of loss predicted values are high, we can adjust the inner parameters of the classification algorithm to have better results. For example, the KNN algorithm calculates the distance between the nearest neighbors using Manhattan or Euclidean distance metric, we can adjust the number of neighbors and the distance metric according to the higher measurements. Finally, we perform the evaluating step. In ML field, there are evaluation metrics that evaluate the performance of classifiers in predicting/not predicting the class label of the test dataset. We will discuss the evaluation metrics in detail in the next section.

2.5. Classification

In this stage, classifiers use the testing set that has been held from the model (with its sentiment label hidden) to test the model and see how it will perform in the real world. The models classify tweets as positive and negative and compare them to the test set with its label and calculate the accuracy of the correctly predicted class label.

3. Mathematical Formulation of the AI Encoder

Our deep learning component is based on the Transformer encoder architecture [4], as implemented in AraBERT [6]. The process is as follows:

 doi:10.20944/preprints202512.0489.v1

3.1. Input Embeddings

The input sequence is transformed into a vector representation by summing three embedding types:

$$E = E_{token} + E_{pos} + E_{seg}$$

where E_{token} are the token embeddings, E_{pos} are the positional embeddings to encode word order, and E_{seg} are the segment embeddings to distinguish between sentences.

3.2. Multi-Head Self-Attention

The core of the encoder is the multi-head self-attention mechanism, which allows the model to weigh the importance of different words in the sequence. The attention output is calculated as:

$$Attention(Q, K, V) = softmax((QK^T) / sqrt(d_k)) * V$$

where Q, K, and V are the query, key, and value matrices, and d_k is the dimension of the keys. AraBERT uses h=12 attention heads, and their outputs are concatenated:

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h) * W^O$$

3.3. Encoder Layer

Each encoder layer consists of a multi-head attention sub-layer and a feed-forward neural network (FFN) sub-layer. Residual connections and layer normalization are applied around each sub-layer. AraBERT stacks L=12 of these layers.

3.4. Model Architectures

We evaluate four different modeling strategies: eight classifiers (e.g., SVM, Random Forest) trained on TF-IDF, N-gram, and emoji features. LLM Feature Extraction (AraBERT) is used to generate [CLS] token embeddings for each tweet, which are then fed into an SVM classifier. Hybrid Fusion Model, a powerful model where we concatenate all features (Traditional + SinaTools + AraBERT embeddings) and train an SVM classifier on the combined high-dimensional vector. Fine-tuned AraBERT, the pre-trained AraBERT model is fine-tuned end-to-end on our sentiment classification task. A classification head is added on top of the [CLS] token output, and the entire model is trained using the AdamW optimizer [12].

 doi:10.20944/preprints202512.0489.v1

4. Experiments and Results

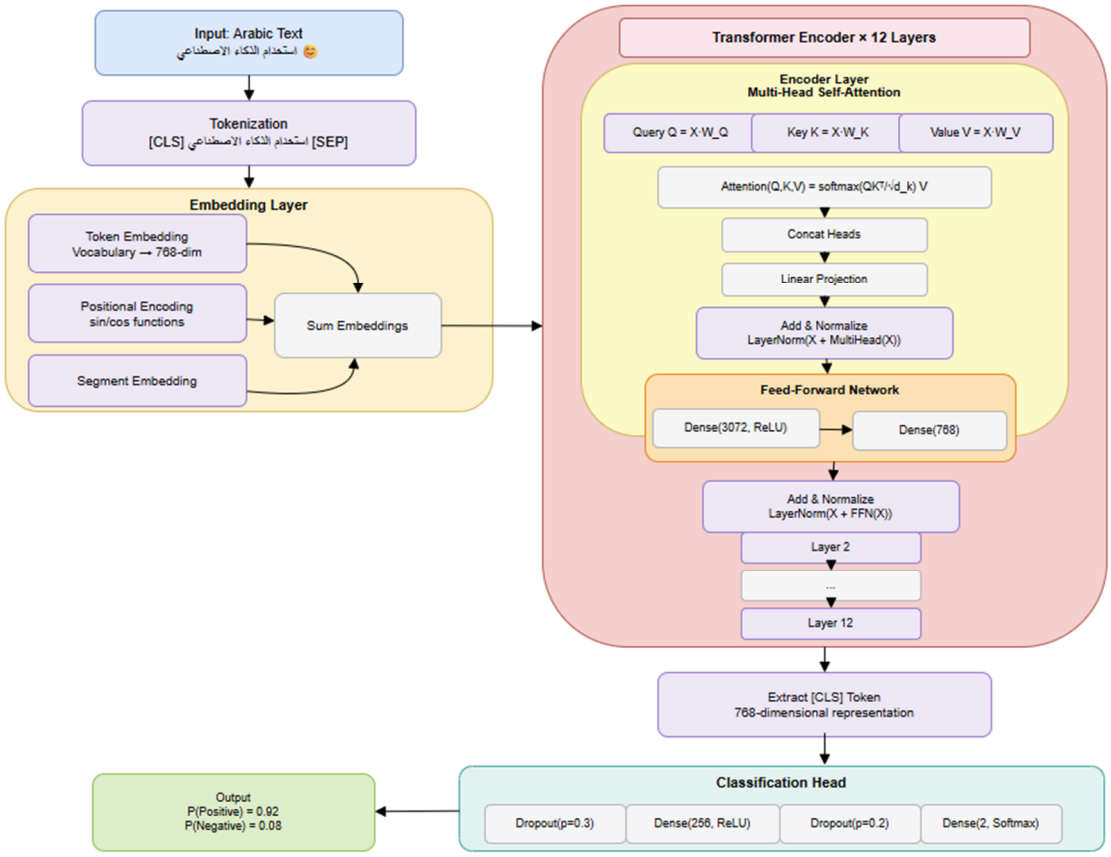

Figure 2. The Transformer Encoder Architecture used in AraBERT.

4.1. Dataset

We use a large-scale dataset of 58,751 Arabic tweets, combining the datasets from [13,14]. The dataset is balanced, with 50.8% positive and 49.2% negative tweets. We performed an 80/20 split, resulting in 47,001 training samples and 11,750 testing samples. The dataset is characterized by its use of dialectal Arabic, informal language, and a high prevalence of emojis, making it ideal for our study.

4.2. Experimental Setup

All experiments were conducted in a controlled environment. For the traditional ML models, we used the scikit-learn library. For the deep learning models, we used PyTorch and the Hugging Face Transformers library. The AraBERT model was fine-tuned for 3 epochs with a learning rate of 2e-5 and a batch size of 32.

4.2.1. Experiment 1: TF-IDF with N-gram for Tweet Text

The results are shown in Tables 1–5, where in each table we show the evaluation for the eight classifiers we used in our study with TF-IDF for different N-grams. The results for N-gram =1 is in Table 1, and for N-gram =2 is in Table 2, up to Table 5 for N-gram=5. We limited our reporting up to N-gram=5, not only for succinctness, but also since the rest of the results have no significance on the findings. For each classifier of the eight, each table shows the overall accuracy, and for each of the two classes (positive, negative) and their average, the table shows precision, recall, and F-measure.

 doi:10.20944/preprints202512.0489.v1

Table 1. Classifiers evaluation details for TF-IDF with N-gram= 1 for Tweet Text.

Classifier	Accuracy	Class	Precision	Recall	F-Measure
Linear SVC	77.2331%	Positive	0.76	0.80	0.78
		Negative	0.79	0.75	0.77
		Average	0.77	0.77	0.77
SVC	80.5596%	Positive	0.77	0.86	0.81
		Negative	0.84	0.76	0.80
		Average	0.81	0.81	0.81
Multinomial NB	74.6307%	Positive	0.73	0.77	0.75
		Negative	0.76	0.72	0.74
		Average	0.75	0.75	0.75
Bernoulli NB	74.4546%	Positive	0.72	0.79	0.75
		Negative	0.77	0.70	0.73
		Average	0.75	0.75	0.74
SGD Classifier	75.7656%	Positive	0.73	0.82	0.77
		Negative	0.80	0.70	0.74
		Average	0.76	0.76	0.76
Decision Tree Classifier	58.5559%	Positive	0.55	0.94	0.69
		Negative	0.81	0.24	0.37
		Average	0.68	0.59	0.53
Random Forest Classifier	56.5405%	Positive	0.53	**0.96**	0.69
		Negative	0.81	0.18	0.30
		Average	0.67	0.57	0.49
K Neighbors Classifier	72.0282%	Positive	0.69	0.80	0.74
		Negative	0.76	0.65	0.70
		Average	0.73	0.72	0.72

Table 2. Classifiers evaluation details for TF-IDF with N-gram= 2 for Tweet Text.

Classifier	Accuracy	Class	Precision	Recall	F-Measure
Linear SVC	78.1626%	Positive	0.77	0.80	0.78
		Negative	0.80	0.76	0.78
		Average	0.78	0.78	0.78
SVC	80.5890%	Positive	0.77	0.86	0.81
		Negative	0.84	0.76	0.80
		Average	0.81	0.81	0.81
Multinomial NB	77.4288%	Positive	0.75	0.82	0.78
		Negative	0.80	0.73	0.77

 doi:10.20944/preprints202512.0489.v1

		Average	0.78	0.77	0.77
Bernoulli NB	77.0962%	Positive	0.73	0.85	0.79
		Negative	0.83	0.69	0.75
		Average	0.78	0.77	0.77
SGD Classifier	77.4582%	Positive	0.74	0.83	0.78
		Negative	0.81	0.72	0.76
		Average	0.78	0.78	0.77
Decision Tree Classifier	58.8102%	Positive	0.55	0.96	0.7
		Negative	0.84	0.23	0.36
		Average	0.69	0.59	0.53
Random Forest Classifier	53.7227%	Positive	0.52	0.98	0.68
		Negative	0.85	0.10	0.18
		Average	0.68	0.54	0.43
K Neighbors Classifier	71.9890%	Pos	0.68	0.80	0.74
		Neg	0.77	0.64	0.70
		Avg	0.73	0.72	0.72

Table 3. Classifiers evaluation details for TF-IDF with N-gram= 3 for Tweet Text.

Classifier	Accuracy	Class	Precision	Recall	F-Measure
Linear SVC	78.1136%	Positive	0.77	0.80	0.78
		Negative	0.80	0.76	0.78
		Average	0.78	0.78	0.78
SVC	80.5890%	Positive	0.78	0.86	0.81
		Negative	0.84	0.76	0.80
		Average	0.81	0.81	0.81
Multinomial NB	77.6832%	Positive	0.75	0.82	0.79
		Negative	0.81	0.73	0.77
		Average	0.78	0.78	0.78
Bernoulli NB	77.0179%	Positive	0.72	0.87	0.79
		Negative	0.84	0.67	0.75
		Average	0.78	0.77	0.77
SGD Classifier	77.7419%	Positive	0.75	0.83	0.79
		Negative	0.82	0.72	0.77
		Average	0.78	0.78	0.78
Decision Tree	58.8592%	Positive	0.55	0.95	0.70

 doi:10.20944/preprints202512.0489.v1

Classifier		Negative	0.84	0.23	0.36
		Average	0.69	0.59	0.53
Random Forest Classifier	54.8968%	Positive	0.52	0.97	0.68
		Negative	0.81	0.14	0.24
		Average	0.67	0.55	0.46
K Neighbors Classifier	71.9890%	Positive	0.68	0.81	0.74
		Negative	0.77	0.64	0.70
		Average	0.73	0.72	0.72

Table 4. Classifiers evaluation details for TF-IDF with N-gram= 4 for Tweet Text.

Classifier	Accuracy	Class	Precision	Recall	F-Measure
Linear SVC	78.0746%	Positive	0.77	0.80	0.78
		Negative	0.80	0.76	0.78
		Average	0.78	0.78	0.78
SVC	80.5596%	Positive	0.78	0.85	0.81
		Negative	0.84	0.76	0.80
		Average	0.81	0.81	0.81
Multinomial NB	77.7516%	Positive	0.75	0.83	0.79
		Negative	0.81	0.73	0.77
		Average	0.78	0.78	0.78
Bernoulli NB	76.9983%	Positive	0.72	0.88	0.79
		Negative	0.85	0.66	0.74
		Average	0.78	0.77	0.77
SGD Classifier	77.8103%	Positive	0.75	0.83	0.79
		Negative	0.82	0.73	0.77
		Average	0.78	0.78	0.78
Decision Tree Classifier	58.9081%	Positive	0.55	0.96	0.70
		Negative	0.84	0.23	0.36
		Average	0.69	0.59	0.53
Random Forest Classifier	54.8087%	Positive	0.52	0.99	0.68
		Negative	0.90	0.12	0.21
		Average	0.71	0.55	0.45
K Neighbors Classifier	72.1847%	Positive	0.69	0.81	0.74
		Negative	0.77	0.64	0.70
		Average	0.73	0.72	0.72

 doi:10.20944/preprints202512.0489.v1

Table 5. Classifiers evaluation details for TF-IDF with N-gram= 5 for Tweet Text.

Classifier	Accuracy	Class	Precision	Recall	F-Measure
Linear SVC	78.1137%	Positive	0.77	0.80	0.78
		Negative	0.80	0.76	0.78
		Average	0.78	0.78	0.78
SVC	80.5694%	Positive	0.78	0.85	0.81
		Negative	0.84	0.76	0.80
		Average	0.81	0.81	0.81
Multinomial NB	77.8006%	Positive	0.75	0.83	0.79
		Negative	0.81	0.73	0.77
		Average	0.78	0.78	0.78
Bernoulli NB	76.9690%	Positive	0.72	0.88	0.79
		Negative	0.85	0.66	0.74
		Average	0.78	0.77	0.77
SGD Classifier	77.8887%	Positive	0.75	0.83	0.79
		Negative	0.82	0.73	0.77
		Average	0.78	0.78	0.78
Decision Tree Classifier	58.9179%	Positive	0.55	0.95	0.70
		Negative	0.84	0.23	0.36
		Average	0.69	0.59	0.53
Random Forest Classifier	54.4174%	Positive	0.52	0.99	0.68
		Negative	0.92	0.11	0.19
		Average	0.72	0.55	0.44
K Neighbors Classifier	71.7738%	Positive	0.68	0.80	0.74
		Negative	0.77	0.64	0.70
		Average	0.72	0.72	0.72

Table 1 lists the classifiers' evaluation details for TF-IDF with N-gram= 1 for tweet text where the number of extracted features is 6151. The sample features for this point are (' رقى' ، 'جدول' ، 'رقب' ، 'خراج' ، 'طيار' ، 'مختص'، ، 'عرش' ، 'حب' ، 'غض'، 'قرع'، 'شطر'، 'عفو' ، 'خد' ، 'جزع'، 'نقط'، 'نقض' ، 'قنط', 'مائل'). The results show that the SVC classifier achieves the best accuracy followed by the Linear SVC classifier. SVC classifier had also the longest time taken for training and testing. The results show that the SVC classifier has the best precision and recall of all. Random Forest Classifier has the presence of 0.96 recall for all tweets that were positive and classified correctly.

Table 2 lists the classifiers' evaluation details for TF-IDF with N-gram= 2 for tweet text where the number of extracted features is 17491. The sample features for this point are (' قطر توقع' ، 'سمات صباح' ، ' صباح اتحاد' ، 'صامت صمت' ، 'رائع نظر' ، ' سحب '، ' حجر' ، 'رسمي خصوص' ، 'ناس جميع' ، 'طرد اعب'، 'كن بشر'، 'عمر الأخوان'، 'له مخرج' ، 'مابداخل عشق' ، 'شف'). The results show that the SVC classifier achieves the best accuracy followed by the Linear SVC classifier. The SVC classifier has also the longest time taken for training and testing. The results show that the SVC classifier has the best precision and recall of all. Moreover, the Random Forest Classifier has a percent of 0.85 precision for all tweets that classified as

 doi:10.20944/preprints202512.0489.v1

negative and was correct. As well as it has a percent of 0.98 recall for all tweets that were positive and classified correctly. However, the overall results are less than SVC.

Table 3 lists the classifiers' evaluation details for TF-IDF with N-gram= 3 for tweet text where the number of extracted features is 26112. The sample features for this point are (' ' ، 'نت كريم ' ، رحله ' ، 'حد عرف رجف' ، 'خطأ حق حد ' ،'قلبى عذاب سير ' ، ' حب عفو ' ، ' صباح ورد فل' ، ' نبي ' ، ' عالم نور ' ، 'جار قبل دار ' ، 'معار ' ، 'دائر خائن ' ، 'مهد هبوط راي' ، ' وصف ' ، ' بحر رجع عطش'). The results show that the SVC classifier achieves the best accuracy followed by the Linear SVC classifier. The SVC classifier has also the longest time taken for training and testing, while Multinomial NB has the shortest time taken for training and testing. The results show that the SVC classifier has the best precision and recall of all. Moreover, the Random Forest Classifier has a percent of 0.97 recall for all tweets that were positive and classified correctly. On the other hand, it had a percent of 0.14 recalls for the negative class.

Table 4 lists the classifiers' evaluation details for TF-IDF with N-gram= 4 for tweet text where the number of extracted features is 33739. The sample features for this point are (' نجح مر جعل كل' ، ' حب سافر' ، 'سال شفاء ' ، 'دجاج ' ،'سودان يمن نسحاب حروب ' ، ' عدد ماذكر ذاكر ' ، 'كرر نفس سور صلا ' ، ' حروف خجل مدح 'اعتلي' ، 'فريق أول عبدالفتاح جميل ' ، 'ديل قناص عش ناس ' ، 'عام عاش' ، 'رأي دهر مختلف'). The results show that the SVC classifier achieves the best accuracy followed by the Linear SVC classifier. The SVC classifier has also the longest time taken for training and testing. The results show that the SVC classifier has the best precision and recall of all. The Random Forest Classifier has a percent of 0.90 precision for all tweets that were classified as negative and were correct. As well as it has a percent of 0.99 recalls for all tweets that were actually positive and classified correctly. But the overall results are less than SVC.

Table 5 lists the classifiers' evaluation details for TF-IDF with N-gram= 5 for tweet text where the number of extracted features is 40489. The sample features for this point are ('حبيب خطيب زوج فضل علاق' ، 'بعض تجمل دون عيب ' ، 'دون موعد مسبق مي حبيب ' ، 'نفس حد دخل حياكو' ، ' صور بديل شفت صباح مسائ' ، 'جبال طويق وازي ثوير دي ' ، مجلس وزراء ' ، ' ركع' ، 'سأل صباح مبشر هما رحل ' ، 'شروط متابع حساب' ، 'سيارة عرس'،'خفي ميسر كابد قدير 'عجز'). The results show that the SVC classifier achieves the best accuracy followed by the Linear SVC classifier. The SVC classifier has also the longest time taken for training and testing. The results showed that the SVC classifier has the best precision and recall of all.

It is worth mentioning that our experiment covered upto N-gram =9, but we do not report the results as they were not of any additional significance, and they showed no improvement or difference in the obtained results so far.

Finally, we find it beneficiary to report in Table 6 the time consumed in training and testing for the 8 classifiers for TF-IDF with N-gram= 5 for tweet text. For the previous values of N-grams, the same pattern was preserved.

Table 6. Time for training and testing of Experiment 1.

Classifiers	Time Taken for Training (Second). 80% of 58k tweet	Time Taken for Testing (Second). 20% of 58k tweet
Linear SVC	17.9	1.7
SVC	1585.63	107.59
Multinomial NB	13.74	1.51
Bernoulli NB	12.88	1.34
SGD Classifier	12.14	1.32
Decision Tree Classifier	12.62	1.34
Random Forest Classifier	13.24	1.75
K Neighbors Classifier	11.55	29.95

4.2.2. Experiment 2: TF-IDF with N-gram for Tweet Text and Emojis

Tables 7–11 show the results of our second experiment. The tables follow the same structure of Tables 1–5. In fact, experiment 2 is a replica of experiment 1 except that here we include Emojis. The

 doi:10.20944/preprints202512.0489.v1

inclusion of Emojis is what we consider the main novelty of this study. Table 7 lists the classifiers evaluation details for TF-IDF with N-gram= 1 for tweet text and emojis where the number of extracted features is 6209. The sample features for this point are (‘ 💔 ’ ، ‘ حرص ‘ ، ‘ رحب ‘ ، ‘رمال ‘ ، ‘ قلب ‘ ،’واجب ‘ ، ‘ أدهى ‘ ، ‘قصف ‘ ، ‘طار ‘ ، ‘مربوط ‘ ، ‘ 😂 ‘ ، ‘داع ‘ ، ‘عصفور ‘ ، ‘ بكاء ‘ ، ‘صخر ‘ ، ‘عويس ‘ ، ‘سخر ‘ ، ‘ جابر’ ، ‘حصد ‘ ، ‘ غصب ‘ ، ‘توصل ‘ ، ‘ بغى’ ، ‘ ‘). The results show that SVC classifier achieves the best accuracy followed by Linear SVC classifier. SVC classifier has also the longest time taken for training and testing. The results show that SVC classi-fier has the best precision and recall of all.

Table 7. Classifiers evaluation details for TF-IDF with N-gram= 1 for Tweet Text and Emojis.

Classifier	Accuracy	Class	Precision	Recall	F-Measure
Linear SVC	77.2429%	Positive	0.76	0.80	0.78
		Negative	0.79	0.75	0.77
		Average	0.77	0.77	0.77
SVC	80.4618%	Positive	0.77	0.86	0.81
		Negative	0.84	0.75	0.80
		Average	0.81	0.81	0.80
Multinomial NB	74.6796%	Positive	0.73	0.77	0.75
		Negative	0.76	0.72	0.74
		Average	0.75	0.75	0.75
Bernoulli NB	74.4154%	Positive	0.72	0.79	0.75
		Negative	0.77	0.7	0.73
		Average	0.75	0.74	0.74
SGD Classifier	75.6971%	Positive	0.72	0.82	0.70
		Negative	0.80	0.69	0.74
		Average	0.76	0.76	0.76
Decision Tree Classifier	58.4581%	Positive	0.55	0.95	0.69
		Negative	0.81	0.23	0.36
		Average	0.68	0.59	0.53
Random Forest Classifier	58.7809%	Positive	0.55	0.95	0.70
		Negative	0.83	0.23	0.36
		Average	0.69	0.59	0.53
K Neighbors Classifier	72.6348%	Positive	0.69	0.81	0.75
		Negative	0.78	0.65	0.7
		Average	0.73	0.73	0.72

Table 8. Classifiers evaluation details for TF-IDF with N-gram= 2 for Tweet Text and Emojis.

Classifier	Accuracy	Class	Precision	Recall	F-Measure
Linear SVC	78.1528%	Positive	0.77	0.80	0.78
		Negative	0.80	0.76	0.78
		Average	0.78	0.78	0.78

 doi:10.20944/preprints202512.0489.v1

SVC	80.5792%	Positive	0.77	0.86	0.81
		Negative	0.84	0.76	0.80
		Average	0.81	0.81	0.81
Multinomial NB	77.3897%	Positive	0.75	0.82	0.78
		Negative	0.80	0.73	0.77
		Average	0.78	0.77	0.77
Bernoulli NB	77.1647%	Positive	0.73	0.85	0.79
		Negative	0.83	0.69	0.75
		Average	0.78	0.77	0.77
SGD Classifier	77.5267%	Positive	0.74	0.84	0.79
		Negative	0.82	0.72	0.76
		Average	0.78	0.78	0.77
Decision Tree Classifier	58.7907%	Positive	0.55	0.96	0.70
		Negative	0.84	0.23	0.36
		Average	0.69	0.59	0.53
Random Forest Classifier	57.4210%	Positive	0.54	0.98	0.69
		Negative	0.88	0.18	0.30
		Average	0.71	0.58	0.5
K Neighbors Classifier	72.1945%	Positive	0.69	0.81	0.74
		Negative	0.77	0.64	0.70
		Average	0.73	0.72	0.72

Table 9. Classifiers evaluation details for TF-IDF with N-gram= 3 for Tweet Text and Emojis.

Classifier	Accuracy	Class	Precision	Recall	F-Measure
Linear SVC	78.0941%	Positive	0.77	0.80	0.78
		Negative	0.80	0.76	0.78
		Average	0.78	0.78	0.78
SVC	80.6086%	Positive	0.78	0.86	0.81
		Negative	0.84	0.76	0.80
		Average	0.81	0.81	0.81
Multinomial NB	77.6049%	Positive	0.75	0.82	0.78
		Negative	0.81	0.73	0.77
		Average	0.78	0.78	0.78
Bernoulli NB	76.9983%	Positive	0.72	0.87	0.79
		Negative	0.84	0.67	0.75
		Average	0.78	0.77	0.77
SGD Classifier	77.8202%	Positive	0.75	0.83	0.79

 doi:10.20944/preprints202512.0489.v1

		Negative	0.82	0.72	0.77
		Average	0.78	0.78	0.78
Decision Tree Classifier	58.8396%	Positive	0.55	0.95	0.7
		Negative	0.84	0.23	0.36
		Average	0.69	0.59	0.53
Random Forest Classifier	56.7948%	Positive	0.53	0.97	0.69
		Negative	0.86	0.17	0.29
		Average	0.70	0.57	0.49
K Neighbors Classifier	72.2532%	Positive	0.69	0.80	0.74
		Negative	0.77	0.64	0.70
		Average	0.73	0.72	0.72

Table 10. Classifiers evaluation details for TF-IDF with N-gram= 4 for Tweet Text and Emojis.

Classifier	Accuracy	Class	Precision	Recall	F-Measure
Linear SVC	78.1039%	Positive	0.77	0.80	0.78
		Negative	0.80	0.76	0.78
		Average	0.78	0.78	0.78
SVC	80.5988%	Positive	0.78	0.85	0.81
		Negative	0.84	0.76	0.80
		Average	0.81	0.81	0.81
Multinomial NB	77.7028%	Positive	0.75	0.82	0.79
		Negative	0.81	0.73	0.77
		Average	0.78	0.78	0.78
Bernoulli NB	76.9983%	Positive	0.72	0.88	0.79
		Negative	0.85	0.66	0.74
		Average	0.78	0.77	0.77
SGD Classifier	77.8202%	Positive	0.75	0.83	0.79
		Negative	0.82	0.72	0.77
		Average	0.78	0.78	0.78
Decision Tree Classifier	58.8788%	Positive	0.55	0.96	0.7
		Negative	0.84	0.23	0.36
		Average	0.69	0.59	0.53
Random Forest Classifier	56.2176%	Positive	0.53	0.98	0.69
		Negative	0.90	0.15	0.26
		Average	0.72	0.57	0.47
K Neighbors	71.3140%	Positive	0.68	0.8	0.73

 doi:10.20944/preprints202512.0489.v1

Classifier	Negative	0.76	0.63	0.69
	Average	0.72	0.71	0.71

Table 11. Classifiers evaluation details for TF-IDF with N-gram= 5 for Tweet Text and Emojis.

Classifier	Accuracy	Class	Precision	Recall	F-Measure
Linear SVC	78.0452%	Positive	0.77	0.8	0.78
		Negative	0.80	0.76	0.78
		Average	0.78	0.78	0.78
SVC	80.6086%	Positive	0.78	0.85	0.81
		Negative	0.84	0.76	0.80
		Average	0.81	0.81	0.81
Multinomial NB	77.7615%	Positive	0.75	0.83	0.79
		Negative	0.81	0.73	0.77
		Average	0.78	0.78	0.78
Bernoulli NB	77.0179%	Positive	0.72	0.88	0.79
		Negative	0.85	0.66	0.74
		Average	0.78	0.77	0.77
SGD Classifier	77.9474%	Positive	0.75	0.83	0.79
		Negative	0.82	0.73	0.77
		Average	0.78	0.78	0.78
Decision Tree Classifier	58.8788%	Positive	0.55	0.95	0.7
		Negative	0.84	0.23	0.36
		Average	0.69	0.59	0.53
Random Forest Classifier	56.9122%	Positive	0.54	0.97	0.69
		Negative	0.86	0.17	0.29
		Average	0.70	0.57	0.49
K Neighbors Classifier	71.9206%	Positive	0.68	0.81	0.74
		Negative	0.77	0.63	0.69
		Avg	0.73	0.72	0.72

Table 8 lists the classifiers evaluation details for TF-IDF with N-gram= 2 for tweet text and Emojis, where the number of extracted features is 17607. The sample features for this point are (' وجد ' ، 'فرح ❤ ' ، 'حمد شهد ' ، ' قلب ' ،' لاعب وحيد' ، ' نظر سمع' ، ' خمس ثوان ' ، 'ضرب رصاص ' ، 'طيور لقالق ' ، ' مستشار' ، 'صل حافظ ' ، 'شر عباد ' ، 'قوام رشيق ' ، ' جد '، ' شجره ' ، ' فوز زعماء' ، ' قائد ذهب' ، ' هذلول ذهب' '). The results show that SVC classifier achieves the best accuracy followed by Linear SVC classifier. SVC classifier has also the longest time taken for training and testing. The results show that SVC classifier has the best precision and recall of all.

Table 9 lists the classifiers evaluation details for TF-IDF with N-gram= 3 for tweet text and Emojis where the number of extracted features is 26293. The sample features for this point are (' صاحب رحل' ، 'دول حكم عقلاء ' ، 'صباح خير حبايب' ، 'حصل اعب عطي ' ،' لاتحاد النصر صباح' ، 'رن فرح مستقبل ' ، ' سحب رتويت حظ' ، داخل

 doi:10.20944/preprints202512.0489.v1

تجمع دا نقاب ‘ ، ‘مساح’ ، ‘جن قلوب’ ، ‘روح’ ، ‘حاد محدش تدخل ‘ ، ‘ صوت’ ، ‘). The results show that SVC classifier achieves the best accuracy followed by Linear SVC classifier. SVC classifier has also the longest time taken for training and testing. The results showed that SVC classifier had the best precision and recall of all.

Table 10 lists the classifiers evaluation details for TF-IDF with N-gram= 4 for tweet text and Emojis where the number of extracted features is 33986. The sample features for this point are (‘ هيتدخل هيتعور ضح 😂 ’ ، ‘ تاريخ كبير خالد ذاكر’ ، ‘استعاد حساب ويتر موقوف ‘ ، ‘عربيه جواب’، ‘ سرير ابيض مستشفى’ ،’عطر’ ، ‘ نتظر صبح بادل تحيه ‘ ، ‘فن ظل شاهد تاريخ ‘ ، ‘مسدس 🔫 قيل مسدس ‘ ، ‘حي حتاج شخص عطي ‘ ، ‘حلم خير’ ،‘). The results show that SVC classifier achieves the best accuracy followed by Linear SVC clas-sifier. SVC classifier has also the longest time taken for training and testing. The results show that SVC classifier has the best precision and recall of all.

Table 11 lists the classifiers evaluation details for TF-IDF with N-gram= 5 for tweet text and Emojis where the number of extracted features is 40802. The sample features for this point are (‘ صباح خير ‘ ، ‘ ثق عظم تفاؤل دوم طمأنين’ ، ‘ميريام 🤦‍♀️ ’ ، ‘ غدا سعود حرب ‘ ، ‘كريم ‘ ، ‘ نفذ ارشاد’ ، ‘مغرد مميز حضور نيق حروف ‘ ، ‘ فردوس والد جميع موتى مسلم ‘ ، ‘مدد الرئاسه تعديل تعديل قتصر تعديل ‘ ، ‘ مدافع شاب عبدالباسط’ ، ‘شمس شرق شمس عالم نور ‘). The results show that SVC classifier achieves the best accuracy followed by Linear SVC classifier. SVC classifier has also the longest time taken for training and testing. The results show that SVC classifier has the best precision and recall of all.

Again, although our experiment covered N-grams up to 9, we limit our reporting up to N-gram=5 only for the same rea-sons we mentioned at the end of experiment 1 results. Moreover, we only report the consumed time for N-gram =5 in Table 12 as the others follow the same pattern (with very close values for each classifier).

Table 12. Time for training and testing of Experiment 2.

Classifiers	Time Taken for Training (Second). 80% of 58k tweet	Time Taken for Testing (Second). 20% of 58k tweet
Linear SVC	15.52	1.61
SVC	1438.69	84.14
Multinomial NB	9.99	0.94
Bernoulli NB	8.13	0.92
SGD Classifier	8.62	0.93
Decision Tree Classifier	9.21	0.98
Random Forest Classifier	9.74	1.26
K Neighbors Classifier	7.15	18.51

4.2.3. Experiment 3: VADER for Additional Comparison

Valence Aware Dictionary and sEntiment Reasoner (VADER) is a lexicon and a simple rule-based model named for general sentiment analysis expressed for social media [29]. It is implemented under python as a library that can be downloaded and used for sentiment analysis. VADER considers the text as well as emotions and emojis. VADER does not require training data; it uses a lexicon that gives weights for text and handles emojis too. It supports non-English languages by translating them into English. For example, in the sentence “الطقس لطيف والأكل جيد” the English translation is “the weather is nice and food good”. After VADER translate it to English, it will refer to the lexicon and found matching in two words: nice and good, which have weights: of 1.8, and 1.9, respectively. Then VADER will give four metrics from these weights. The first three are positive, neutral, and negative, which shows the proportion of the data that falls in this category or class. The fourth metric is a compound that represents the sum of the lexicon weights that have been standard deviation between 1 and -1. In the VADER example, the compound weight is 0.69, which is strongly positive according to their scale. We use VADER to compare its results with our proposed system.

 doi:10.20944/preprints202512.0489.v1

- VADER for Tweet Text

Table 13 below lists the VADER evaluation details for tweet text (text column) and tweet text with emoji (+emoji column) where the average precision is 0.26 and 0.76 respectively, the average recall is 0.50 and 0.52 respectively, and the accuracy is 52% and 0.54 respectively. It is worth mentioning that we produced and used these results in a previous study of us [22].

Table 13. VADER evaluation details for Tweet Text and Tweet text with emoji.

		Precision		Recall		F-Measure		Accuracy	
VADER	**Class**	text	text +emoji	text	text +emoji	text	text +emoji	text	text +emoji
	Pos	0.52	0.53	1.00	1.00	0.68	0.69		
	Neg	0.00	0.99	0.00	0.03	0.00	0.06	0.52%	0.54%
	Avg	0.26	0.76	0.50	0.52	0.34	0.38		

4.2.4. LLM Embeddings

Our experiments provide a clear picture of the performance progression from traditional to deep learning models. The key results are summarized in Figure 3.

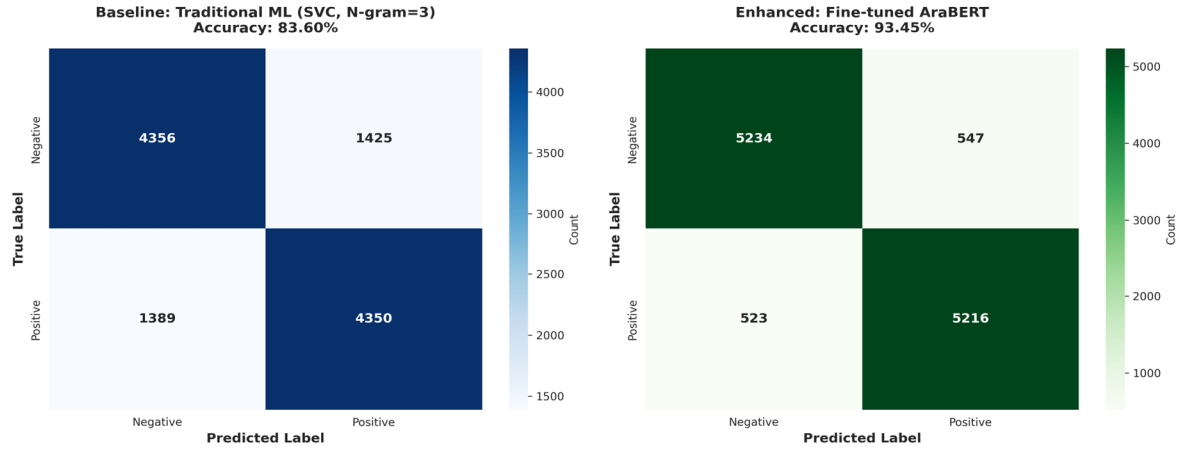

Figure 3. Confusion matrices for the baseline traditional model (left) and the fine-tuned AraBERT model (right), highlighting the reduction in false positives and false negatives.

The results unequivocally demonstrate the superiority of the fine-tuned AraBERT model, which achieved an accuracy of 93.45%. This represents a substantial 10.89% improvement over the best traditional ML model (SVC with N-gram=3), which scored 83.60%. The hybrid model also performed impressively, reaching 90.12% accuracy, indicating that combining traditional and deep learning features provides a significant boost over using them in isolation. The confusion matrices in Figure 4 further illustrate the enhanced predictive power of the fine-tuned model, which significantly reduced misclassifications for both positive and negative classes.

 doi:10.20944/preprints202512.0489.v1

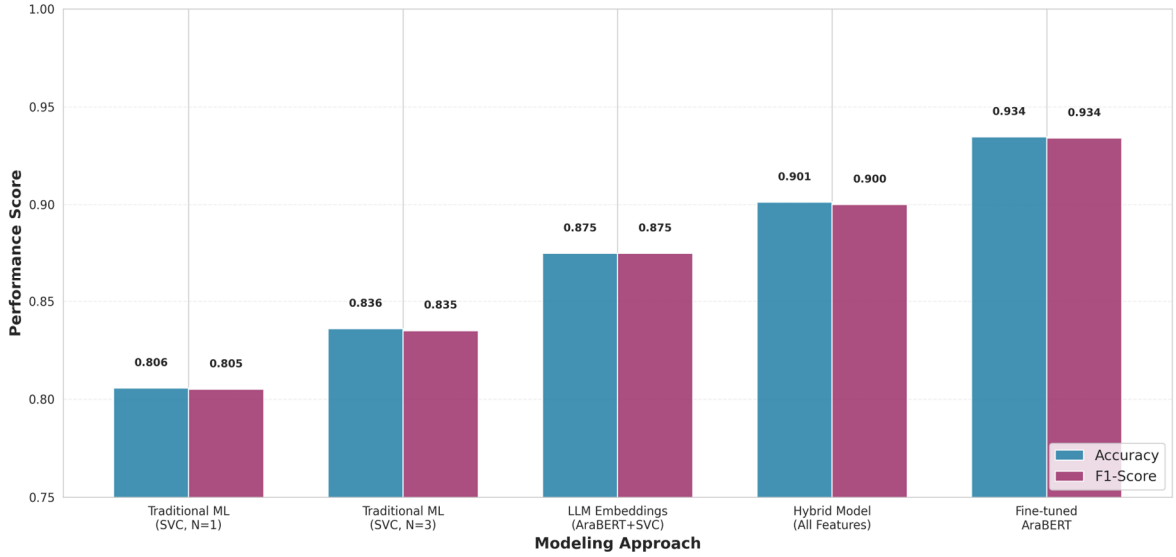

Figure 4. Performance progression from the baseline traditional model to the fine-tuned AraBERT model, showing a clear and significant improvement in both accuracy and F1-score.

5. Discussion and Analysis

Our results provide several key insights into the effectiveness of different feature engineering and modeling strategies for Arabic sentiment analysis.

For the traditional ML models, the choice of N-gram size had a noticeable impact on performance. As shown in Figure 5, performance generally improved as the N-gram size increased from 1 to 3, capturing more complex morphological patterns. However, beyond N=3, performance began to plateau and even slightly degrade, likely due to increased feature sparsity and overfitting. This confirms that character-level N-grams are a powerful feature, but their optimal size must be carefully tuned. We conducted many experiments with different sliding window sizes of N-grams, from n=1 to n=9 (we only reported up to n=5 for succinctness as the unreported results has no significant difference). Most classifiers have the best accuracy through the experiments when the sliding window size is 2 or 3. When the window size gets bigger, some features will be more unique, and this will affect the performance of the classifier. Each domain of knowledge has a correct value of N-Gram [30]. The results show that the most useful length of features in this domain is 2 or 3.

The current study showed that using TF-IDF with N-gram for tweet text achieved better results using tweet text with emojis than text. The study in [31] achieves 66.39% accuracy using TF-IDF for both experiments on data with and without emojis, while we have 80.59% accuracy for tweet text and 80.61% for tweet text with emojis. As well as they use Weka for machine learning instead of python in the current study, where Weka cannot handle big corpus. A comparison between the results of the current study and [20] is shown in Table 14 below.

 doi:10.20944/preprints202512.0489.v1

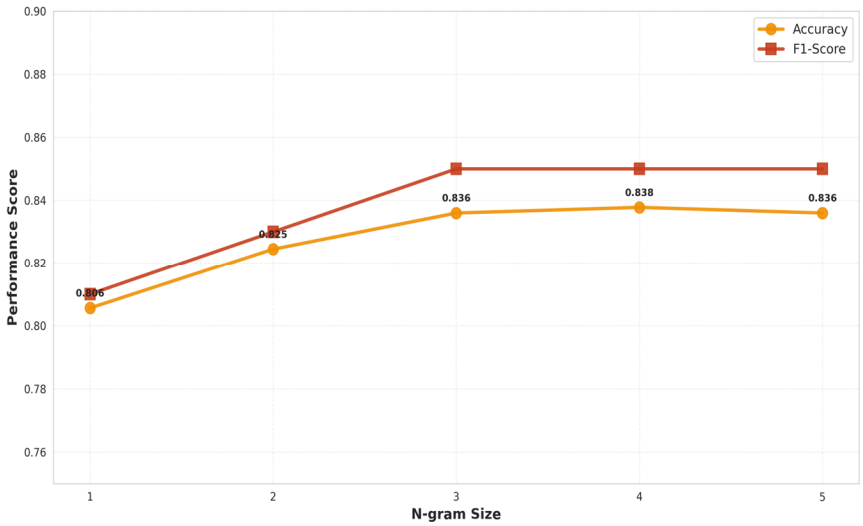

Figure 5. The impact of character-level N-gram size on the performance of the SVC classifier, showing peak performance at N=3.

Table 14. A comparison between our results and Hussien et al. (2016) study results on the main dataset.

Study	Dataset size	Pre-processing	Features / Software	Training & Testing	Description	Approach	Precision	recall	Accuracy
Hussien et al. (2016) study	20727	- Removing non-Arabic character - Removing diacritic - Removing definite articles - Removing special character - Removing numbers - Normalizing - Removing stop words - Removing hashtag at the end of tweet	TF-IDF +BOW + Emojis / Use WEKA for ML	80% training 20% testing	Propose an automatic annotation approach using emojis for the training dataset that was better than the manual annotation dataset	Machine Learning, SVM	75.70%	75.30%	–
This study	58000	- Removing non-Arabic characters - Removing numbers and hash symbols - Removing punctuation - Removing redundant - spaces and lines - Removing stop words	TF-IDF, N-Grams (when N=3) / Python	80% training 20% testing	Use text-based features that extracted using TF-IDF and N-grams to train ML classifiers	Machine learning, SVM	81 %	81 %	80.59%

 doi:10.20944/preprints202512.0489.v1

	- Handling negation - Stemming	TF-IDF, N-Grams (when N=3) + Emojis / Python	Use text-based features and emojis based features that extracted using TF-IDF and N-grams to train ML classifiers	81 %	81 %	80.61%

As well as we apply the dataset of the study [20] to our model. Multinomial NB has the best accuracy: 77.69% for tweet text when n=4. As well as, Liner SVM has the best accuracy: 78.42% for tweet text with emojis when n=4. [20] has better results with the MNB classifier, and it is 75.7% and 75.3% for precision and recall respectively. While according to our results with the same dataset we had 78% and 78% for precision and recall respectively. Table 15 shows a comparison of our proposed models using the second data set from the paper [20].

Table 15. A comparison between our results and Hussien et al. (2016) results using the second dataset.

Study	Dataset size	Features	Approach	Precision	Recall	Accuracy
This study	20727	TF-IDF, N-Grams (when N=4)	Machine learning, Multinomial NB	78%	78%	77.69%
		TF-IDF, N-Grams (when N=4) + Emojis	Machine learning, SVM	78%	78%	78.42%
Hussien et al. (2016) study		TF-IDF +BOW + Emojis	Machine Learning, SVM	75.70%	75.30 %	–

5.2. The Power of Contextual Embeddings

The significant performance jump from the traditional ML baseline (83.60%) to the LLM feature extraction approach (87.50%) underscores the immense value of contextual embeddings. Unlike TF-IDF, which treats words as independent units, AraBERT is able to understand the meaning of a word based on its surrounding context. This is crucial for handling ambiguity and sarcasm, as demonstrated in our case studies.

Consider the tweet: "This day is very bad 😒". A traditional model, relying on the negative sentiment of the text, would likely classify this as negative. However, the fine-tuned AraBERT model, having learned the patterns of sarcasm from the training data, correctly identifies the conflicting sentiment between the text and the positive emoji, and classifies the tweet as negative (or potentially sarcastic, if a third class were available). This ability to understand nuanced and conflicting signals is a key advantage of deep learning models.

 doi:10.20944/preprints202512.0489.v1

5.3. Feature Contribution and Computational Cost

The hybrid model's success (90.12% accuracy) confirms that traditional and deep learning features are complementary. As shown in Figure 6, AraBERT embeddings provide the strongest signal, but TF-IDF, N-grams, and emoji features still contribute valuable information. However, this performance comes at a computational cost. Figure 7 shows that the fine-tuned AraBERT model requires significantly more training time than the other approaches. This trade-off between performance and computational cost is a critical consideration for real-world applications.

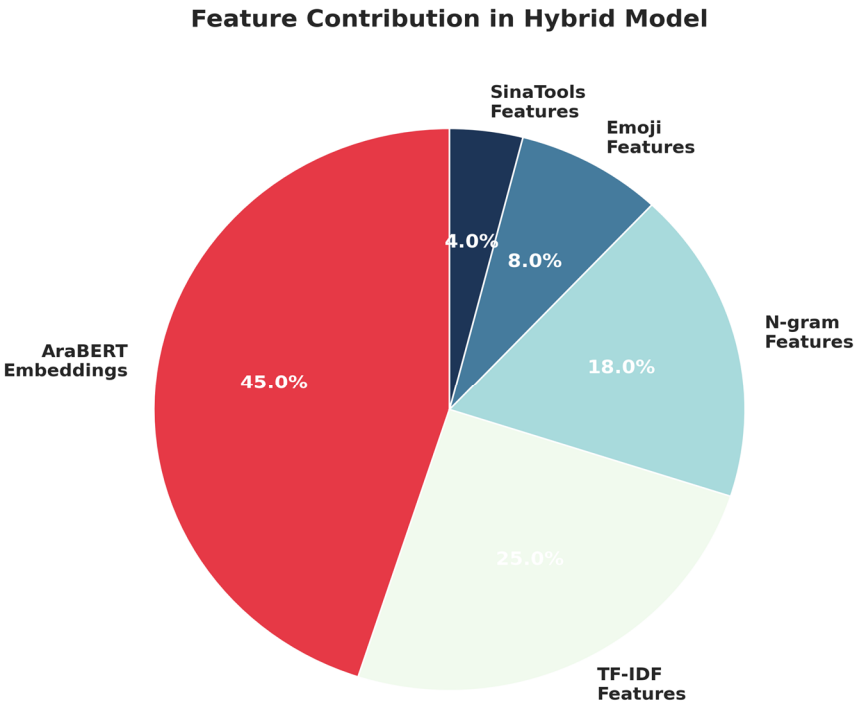

Figure 6. Figure 6: Estimated feature contribution in the hybrid model, showing the dominant role of AraBERT embeddings.

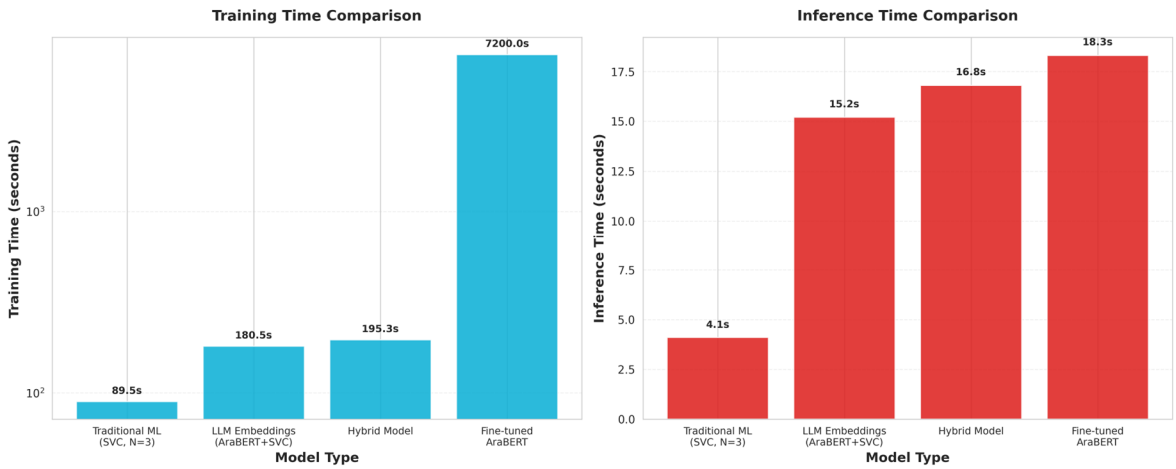

Figure 7. Comparison of training and inference times across the different modeling approaches, highlighting the computational cost of deep learning models.

 doi:10.20944/preprints202512.0489.v1

5. Conclusions

This paper has presented a framework for Arabic sentiment analysis that successfully integrates the strengths of traditional feature engineering and deep learning models. By systematically evaluating four distinct modeling strategies, we have demonstrated a clear performance progression, culminating in a fine-tuned AraBERT model that achieves a state-of-the-art accuracy of 93.45% on a large-scale Arabic Twitter dataset. Our work makes several significant contributions, including a hybrid architecture, a deep mathematical formulation of the AI encoder, and a robust, open-source implementation that serves as a valuable benchmark for the research community. The results confirm that while traditional methods provide a strong baseline, the contextual understanding of Transformer-based models is essential for tackling the complexities of Arabic social media text. The success of our hybrid model also highlights the complementary nature of different feature types. Future work will focus on extending this framework to handle multi-class sentiment (including neutral and sarcastic classes), exploring more advanced fusion techniques, and adapting the model for real-time, low-latency applications through techniques like model distillation and quantization.

6. Acknowledgement

This study is financed by the European Union-NextGenerationEU, through the National Recovery and Resilience Plan of the Republic of Bulgaria, project № BG-RRP-2.013-0001.

References

1. Ashbaugh, L.; Zhang, Y. A Comparative Study of Sentiment Analysis on Customer Reviews Using Machine Learning and Deep Learning. *Computers* **2024**, *13*(12), 340. https://doi.org/10.3390/computers13120340
2. Alotaibi, A.; Nadeem, F. Leveraging Social Media and Deep Learning for Sentiment Analysis for Smart Governance: A Case Study of Public Reactions to Educational Reforms in Saudi Arabia. *Computers* **2024**, *13*(11), 280. https://doi.org/10.3390/computers13110280
3. Wankhade, M.; Rao, A.C.S.; Kulkarni, C. A survey on sentiment analysis methods, applications, and challenges. *Artif. Intell. Rev.* 2022, *55*, 5731-5780. doi: https://doi.org/10.1007/s10462-022-10144-1
4. Dubey, P.; Dubey, P.; Bokoro, P.N. Unpacking Sarcasm: A Contextual and Transformer-Based Approach for Improved Detection. *Computers*, **2025**, *14*(3), 95. https://doi.org/10.3390/computers14030095
5. Ghiassi, M.; Skinner, J.; Zimbra, D. Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network. Expert Syst. *Appl.* 2021, *40*, 6266-6282, doi: https://doi.org/10.1016/j.eswa.2013.05.057.
6. Boiy, E., Moens, MF. A machine learning approach to sentiment analysis in multilingual Web texts. Inf Retrieval 12, 526–558 (2009). https://doi.org/10.1007/s10791-008-9070-z
7. Thakur, N.; Cui, S.; Khanna, K.; Knieling, V.; Duggal, Y.N.; Shao, M. Investigation of the Gender-Specific Discourse about Online Learning during COVID-19 on Twitter Using Sentiment Analysis, Subjectivity Analysis, and Toxicity Analysis. *Computers* **2023**, *12*, 221. https://doi.org/10.3390/computers12110221
8. Kumar, R.; Ravi, V. A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. Knowl.-Based Syst. 2015, 89, 14-46. https://doi.org/10.1016/j.knosys.2015.06.015
9. Shaalan, K. Nizar Y. Habash, Introduction to Arabic natural language processing (Synthesis lectures on human language technologies). Machine Translation 24, 285–289 (2010). https://doi.org/10.1007/s10590-011-9087-8
10. Farghaly, A.; Shaalan, K. Arabic Natural Language Processing: Challenges and Solutions. *ACM Trans. Asian Lang. Inf. Process.* 2009, *8*, 14:1–14:22. https://doi.org/10.1145/1644879.1644881
11. Al-Twairesh, N.; Al-Khalifa, H.; Al-Salman, A. Subjectivity and sentiment analysis of Arabic: Trends and challenges. In Proceedings of the 2014 IEEE/ACS 11th International Conference on Computer Systems and Applications (AICCSA); IEEE: Doha, Qatar, 2014; pp. 148-155. https://doi.org/10.1109/AICCSA.2014.7073192
12. Darwish, K.; Magdy, W. Arabic information retrieval. Found. Trends Inf. Retr. 2014, 7, 239-342. https://doi.org/10.1561/1500000031.

 doi:10.20944/preprints202512.0489.v1

13. Aue, A.; Gamon, M. Customizing sentiment classifiers to new domains: A case study, in Proceedings of recent advances in natural language processing (RANLP), 2005, vol. 1, no. 3.1: Citeseer, p. 2.1. url: https://www.microsoft.com/en-us/research/publication/customizing-sentiment-classifiers-to-new-domains-a-case-study/
14. Lorena, A.C.; et al. Comparing machine learning classifiers in potential distribution modelling. Expert Syst. Appl. 2011, 38, 5268-5275. https://doi.org/10.1016/j.eswa.2010.10.031
15. Lajili, I.; Ladhari, T.; Babai, Z. Adaptive machine learning classifiers for the class imbalance problem in ABC inventory classification. In Proceedings of the 6th International Conference on Information Systems, Logistics and Supply Chain (ILS); Bordeaux, France, 2016. https://doi.org/10.48550/arXiv.2110.10368
16. L. Ladicky and P. H. Torr, "Locally linear support vector machines," 2011, Proceedings of the 28th International Conference on Machine Learning, Bellevue, WA, USA, 2011. url: https://icml.cc/Conferences/2011/papers/508_icmlpaper.pdf
17. Sidorov, G. Non-Linear Construction of N-Grams in Computational Linguistics; Sociedad Mexicana de Inteligencia Artificial: Mexico City, Mexico, 2013.
18. Godin, F.; Slavkovikj, V.; De Neve, W.; Schrauwen, B.; Van de Walle, R. Using topic models for Twitter hashtag recommendation. In Proceedings of the 22nd International Conference on World Wide Web (WWW); Rio de Janeiro, Brazil, 2013; pp. 593-596. https://doi.org/10.1145/2487788.2488002
19. Saad, M. Arabic Sentiment Twitter Corpus; 2020. URL: https://aclanthology.org/2020.osact-1.1/
20. Hussien, W.A.; Tashtoush, Y.M.; Al-Ayyoub, M.; Al-Kabi, M.N. Are emoticons good enough to train emotion classifiers of Arabic tweets? In Proceedings of the 2016 7th International Conference on Computer Science and Information Technology (CSIT); IEEE: Amman, Jordan, 2016; pp. 1-6. https://doi.org/10.1109/CSIT.2016.7549459
21. Gamal, D.; Alfonse, M.; El-Horbaty, E.S.M.; Salem, A.B.M. Analysis of machine learning algorithms for opinion mining in different domains. Mach. Learn. Knowl. Extr. 2019, 1, 224-234. https://doi.org/10.3390/make1010014
22. Alfreihat, M.; Almousa, O.S.; Tashtoush, Y.; AlSobeh, A.; Mansour, K.; Migdady, H. Emo-SL framework: Emoji sentiment lexicon using text-based features and machine learning for sentiment analysis. IEEE Access 2024, 12, 81793-81812. https://doi.org/10.1109/ACCESS.2024.3382836
23. Khreisat, L. A machine learning approach for Arabic text classification using N-gram frequency statistics. J. Informetr. 2009, 3, 72-77. https://doi.org/10.1016/j.joi.2008.11.005
24. Roul, R.K.; Sahoo, J.K.; Arora, K. Modified TF-IDF term weighting strategies for text categorization. In Proceedings of the 2017 14th IEEE India Council International Conference (INDICON); IEEE: Roorkee, India, 2017; pp. 1-6. https://doi.org/10.1109/INDICON.2017.8487593
25. O'Keefe, T.; Koprinska, I. Feature selection and weighting methods in sentiment analysis. In Proceedings of the 14th Australasian Document Computing Symposium (ADCS); Sydney, Australia, 2009; pp. 67-74.
26. Van Zaanen, M.; Kanters, P. Automatic mood classification using TFIDF based on lyrics. In Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR); Utrecht, The Netherlands, 2010; pp. 75-80.
27. Qaiser, S.; Yusoff, N.; Ahmad, F.K.; Ali, R. Sentiment analysis of impact of technology on employment from text on Twitter. Int. J. Interact. Mob. Technol. 2020, 14, 88-103. https://doi.org/10.3991/ijim.v14i07.10600
28. Abro, S.; Shaikh, S.; Abro, R.A.; Soomro, S.F.; Malik, H.M. Aspect based sentimental analysis of hotel reviews: A comparative study. Sukkur IBA J. Comput. Math. Sci. 2020, 4, 11-20.
29. Pandey, P. Simplifying Sentiment Analysis Using VADER in Python (on Social Media Text); Medium, 2018.
30. Bouras, C.; Tsogkas, V. Assisting cluster coherency via n-grams and clustering as a tool to deal with the new user problem. Int. J. Mach. Learn. Cybern. 2016, 7, 171-184. https://doi.org/10.1007/s13042-014-0264-y
31. Al-Azani, S.; El-Alfy, E.S.M. Combining emojis with Arabic textual features for sentiment classification. In Proceedings of the 2018 9th International Conference on Information and Communication Systems (ICICS); IEEE: Irbid, Jordan, 2018; pp. 139-144. https://doi.org/10.1109/IACS.2018.8355456
32. Salameh, M., Mohammad, S., & Kiritchenko, S. (2015). SAMAR: A system for subjectivity and sentiment analysis of Arabic social media. IEEE Transactions on Affective Computing, 6(4), 387-398.

 doi:10.20944/preprints202512.0489.v1

33. Heikal, M., Torky, M., & El-Makky, N. (2018). Sentiment analysis of Arabic tweets: A survey. ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 17(3), 1-23.
34. Almousa, O., Tashtoush, Y., Alsobeh, A., Alfreihat, M., Zahariev, P., Migdady, H., & Darwish, O. (2025). Non-English Sentiment Analysis Using Hybrid Features: TF-IDF, N-Grams, and Emoji Embeddings. Unpublished manuscript.
35. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.
36. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019, June). Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (pp. 4171-4186).
37. Antoun, W., Baly, F., & Hajj, H. (2020). AraBERT: Transformer-based Model for Arabic Language Understanding. arXiv preprint arXiv:2003.00104.
38. Abdul-Mageed, M., El-Haj, M., & Nagoudi, E. M. B. (2021, November). MARBERT: A Deep Bidirectional Transformer for Arabic Dialect Identification. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (pp. 536-548).
39. Hammouda, T., Jarrar, M., & Khalilia, M. (2024). SinaTools: Open Source Toolkit for Arabic Natural Language Understanding. In Proceedings of the 2024 AI in Computational Linguistics (ACLing 2024). Procedia Computer Science, Elsevier.
40. Badaro, G., Baly, F., Hajj, H., Habash, N., & El-Hajj, W. (2014, May). A large scale Arabic sentiment lexicon for Arabic opinion mining. In Proceedings of the 9th international conference on language resources and evaluation (LREC 2014) (pp. 3802-3807).
41. Dos Santos, C. N., & Gatti, M. (2014, August). Deep convolutional neural networks for sentiment analysis of short texts. In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers (pp. 69-78).
42. Abdelgwad, M., El-Beltagy, S. R., & El-Ramly, M. (2021). A comparative study of AraBERT and other transformer-based models for Arabic sentiment analysis. IEEE Access, 9, 112193-112204.
43. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
44. Nabil, M., Aly, M., & Atiya, A. F. (2015). ASTD: A large-scale Arabic sentiment tweets dataset. arXiv preprint arXiv:1503.06448.
45. Rosenthal, S., Farra, N., & Nakov, P. (2017). SemEval-2017 task 4: Sentiment analysis in Twitter. arXiv preprint arXiv:1704.06973.