Comparative Analysis of Machine Learning and Deep Learning Models in Social Media Sentiment Analysis

Diaa Salama AbdElminaam[1], Jana Ahmed[2], Shahd Emad[3],
Haya Walid[4], Habiba Darwish [5], Laila Amgad [6]
Faculty of Computer Science
Misr International University, Cairo, Egypt
diaa.salama[1], jana2208912[2],
shahd2202743[3], haya2202798[4], habiba2206738[5], laila2201298 [6]{@miuegypt.edu.eg}

***Abstract*—Social media sites are a rich source of user-generated content, offering real-time insights into public sentiment. Understanding user sentiment in both Arabic and English is essential for applications ranging from marketing to policy analysis. Accurate sentiment analysis of Arabic and English posts is vital for understanding linguistic, cultural, and contextual differences in user expression. This study applies traditional machine learning algorithms—specifically, CatBoost, Ridge Classifier, Logistic Regression, Naive Bayes, XGBoost, and LightGBM—as well as deep learning models including BiLSTM, DistilBERT, AraBERT, CNN, and RNN, to classify sentiment in Arabic and English texts independently. Each language model was trained on preprocessed, labeled datasets and later tested on Reddit posts retrieved through the Python Reddit API Wrapper (PRAW) to assess performance in a real-world setting. Model performance was assessed using standard evaluation metrics, including Accuracy, Precision, Recall, and F1-score. For English sentiment classification, BiLSTM achieved the highest F1-score and Accuracy of 81% among deep learning models. For Arabic data, BiLSTM again outperformed other deep learning models with an F1-score and Accuracy of 82%. Among traditional machine learning techniques, LightGBM achieved the best performance for English texts with an F1-score and Accuracy of 63%, and for Arabic texts, Logistic Regression was the top performer, yielding an F1-score Accuracy of 80%. Results indicate that deep learning methods yield improved classification performance over traditional machine learning techniques, particularly when tailored to language-specific characteristics. The findings emphasize the importance of language-aware model selection in social media analysis.**

Keywords: Sentiment Analysis, Arabic Language Processing, Text Classification, Machine Learning, Deep Learning, BiLSTM, DistilBERT, AraBERT, Support Vector Machines, CatBoost, Ridge Classifier, LightGBM, Logistic Regression, Naive Bayes, Convolutional Neural Networks, Recurrent Neural Networks, Natural Language Processing (NLP)

I. INTRODUCTION

Social Media has become a powerful platform where people express their opinions, emotions, and attitudes on a wide range of topics, influencing public perception and shaping societal trends. Analyzing sentiment from these vast streams of text—especially in multiple languages like Arabic and English—provides valuable insights for businesses, policymakers, and researchers seeking to understand public mood and consumer behavior. The rapid growth of Arabic and English content created both and opportunity and a challenge for automated sentiment analysis.

Sentiment analysis helps decode the underlying emotions behind posts, reviews, and comments, enabling more informed decisions in marketing, product development, and social engagement strategies. By leveraging natural language processing techniques to analyze both Arabic and English social media data, this paper aims to capture and interpret the nuanced sentiments expressed by diverse user groups, thereby contributing to improved communication, targeted outreach, and enhanced understanding of public opinion in a globally connected world.

Because of the exponential growth in multilingual social media content and the inherent complexity of sentiment expression across different languages, it is crucial to develop effective automated sentiment analysis systems. Employing machine intelligence presents a viable strategy for sentiment classification within linguistic context. Machine learning algorithms are capable of processing extensive datasets of social media posts, thereby revealing intricate patterns and language-based relationships that are fundamental for precise sentiment prediction. In this study, we introduce a data-driven approach that utilizes both traditional machine learning methods, including CatBoost, Ridge Classifier, Logistic Regression, Naive Bayes, XGBoost, and LightGBM, as well as advanced deep learning techniques to be specifically adapted to the subtleties of Arabic and English sentiment analysis. Our approach involves training separate models on language-specific social media datasets to identify emotional expressions and cultural nuances withing each respective language. Through systematic comparison of these methodologies, we aim to determine the most effective approaches for sentiment classification in both Arabic and English contexts. We employ deep learning architectures including DistilBERT, AraBERT, BiLSTM, RNN, and

CNN alongside traditional machine learning algorithms, all of which are carefully trained and evaluated against the respective datasets to assess their predictive capabilities for sentiment analysis. Preprocessing methods including text normalization, tokenization, and stemming are carefully applied to refine data quality in preparation for model training.

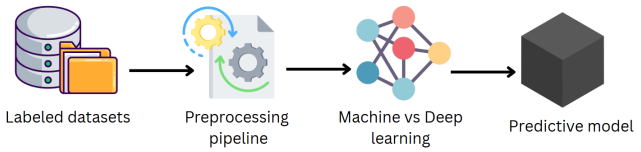

Fig. 1. Sentiment Analysis Workflow

II. Related Work

The field of sentiment analysis has seen substantial advancements, particularly with the proliferation of social media data. Numerous studies have explored various methodologies, ranging from traditional machine learning to sophisticated deep learning architectures, across different languages. This research builds upon this foundation by specifically investigating sentiment analysis for Arabic and English social media content independently, utilizing both machine learning and deep learning approaches.

The authors investigate sentiment analysis in English social media data [1]. They explore various text representation types, including TF-IDF and word embeddings, and compare the performance of Naïve Bayes, Support Vector Machines (SVM), and Artificial Neural Networks (ANN) on Twitter and IMDB movie review datasets. The paper's objective is to determine the most effective model for English sentiment classification. The research found that Artificial Neural Networks consistently achieved the best accuracy, highlighting the potential of neural approaches for this task.

Deep learning approaches for sentiment analysis on electronic social media were investigated by Derbentsev et al. (2022) [2]. The authors developed three deep neural network models: a Convolutional Neural Network (CNN), a CNN-Long Short-Term Memory (LSTM) hybrid, and a Bi-Directional LSTM-CNN (BiLSTM-CNN) hybrid. Using GloVe and Word2Vec embeddings, the models were evaluated on the IMDB Movie Reviews and Twitter Sentiment 140 datasets, benchmarking against Logistic Regression. This research found that the CNN model achieved the best accuracy for the IMDB dataset (90.1%) while the BiLSTM-CNN model showed the highest accuracy for the sentiment 140 dataset (82.1%), demonstrating performance comparable to state-of-the-art models.

This study proposes a sentiment analysis framework to review Arabic text, investigating two textual representations: TF-IDF and word embeddings via Word2Vec [3];. They suggested and evaluated various methods for categorizing sentiments in Arabic text based on a dependable dataset, including Long Short-Term Memory (LSMT), hybrid LSTM-CNN, Convolutional Neural Network (CNN), Logistic Regression, Random Forest, Decision Tree, Support Vector Machines (SVM), and Multinomial Naive Bayes (MNB). The findings indicatd that these methods enhanced Accuracy, Precision, Recall, and F1-score, with LR and SVM classifiers achieving the highest Accuracy at 87%, while deep learning models like LSTM (86.41%) and CNN-LSTM (86.10%) also showed strong performance.

Deep learning networks were utilized by Ali, Abd El Hamid, and Youssif (2019) to develop a classification sentiment analysis system for movie review datasets [4]. This research introduced, comparative results of different deep learning networks, with Multilayer Perceptron (MLP) serving as a baseline. They developed and applied Long Short-Term Memory (LSTM), Convultional Neural Network (CNN), and a hybrid LSTM-CNN model on the IMDB dataset, which comprises 50,000 movie reviews. Data preparation included pre-processing with Word2Vec for word embedding. The findings indicated that the hybrid CNN-LSTM model surpassed the performance of MLP, individual CNN, and LSTM networks, achieving an accuracy of 89.2%. Moreover, these deep learning models exhibited superior performance when contrasted with conventional methods such as SVM, Naive Bayes, and RNTN, as observed in other studies utilizing English datasets.

Deep learning approaches for Arabic sentiment analysis are presented by Mohamed and Kora (2020) [5]. Acknowledging the limitations in Arabic content sentiment analysis due to morphological complexities, dialectal varieties, and corpus scarcity, their contribution is twofold: first, they introduce a corpus of forty thousand labeled Arabic tweets spanning several topics; second, they present three deep learning models, namely CNN, LSTM, and RCNN, for Arabic sentiment analysis. Validating the models' performance with the help of word embedding on their proposed corpus, the experimental results indicate that LSTM, with an average of 81.31%, outperforms CNN and RCNN. Additionally, applying data augmentation to the corpus significantly increases LSTM accuracy by 8.3%.

A deep learning-based model to predict the polarity of opinions and sentiments in Arabic text is proposed by Alharbi, Kalkatawi, and Taileb (2021) [6]. The study addresses the growing importance of classifying subjective text from social networks and blogs. Two types of recurrent neural networks (RNNs) are leveraged to learn higher-level representations, and to mitigate data dependency and increase model effectiveness, three distinct classification algorithms are utilized to produce

the final output. Experimental results demonstrate that their model achieved high accuracy, ranging between 81.1‘% and 94.32% across selected datasets. Furthermore, the proposed model significantly reduced the relative classification error rate by up to 26% compared to existing state-of-the-art models.

III. Proposed Methodology

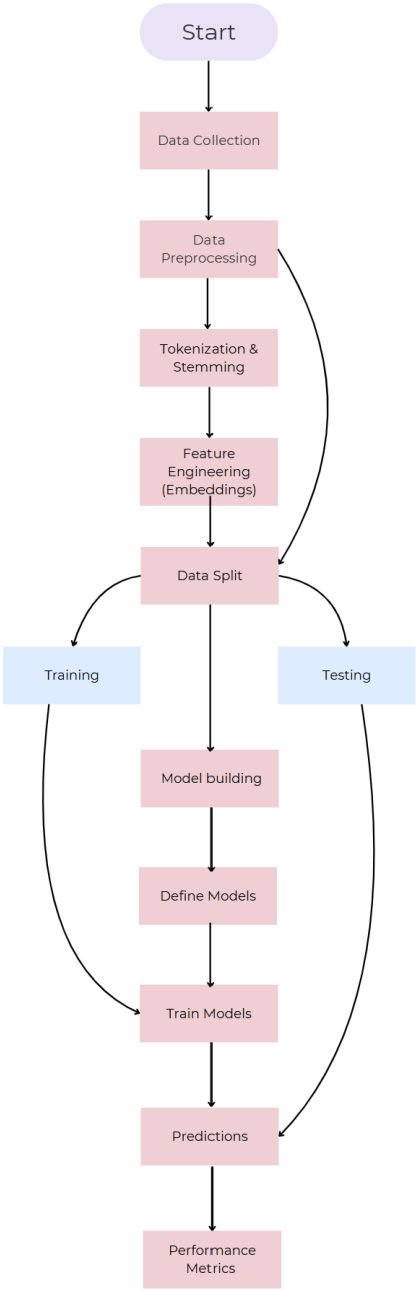

Fig. 2. NLP Methodology

A. *English Dataset*

The Sentiment140 dataset serves as our primary English-language corpus for sentiment analysis. This benchmark dataset contains 1.6 million tweets collected via Twitter's API between 2009 and 2011, providing a robust foundation for NLP applications. The raw data contains six columns, but we focus exclusively on the tweet text and sentiment labels for our analysis.

TABLE I
FEATURES OF ENGLISH DATASET

Features	Type	Description
ID	Integer	Unique tweet identifier
Date	Object	Timestamp of the tweet
Flag	Object	Query flag
User	Object	Twitter username
Tweet	Object	Raw tweet content
Target	Integer	Label (0=negative, 4=positive)

1) **Data Loading and Initial Preparation:**
As a preliminary step in our sentiment analysis pipeline, we begun by loading a large English-language dataset. The dataset was imported from a CSV file using the appropriate character encoding, `ISO-8859-1`. This choice was necessary to prevent decoding issues, such as the commonly encountered `UnicodeDecodeError` when attempting to read the file using the default UTF=8 encoding. The error typically arises due to the presence of special characters or byte sequences incompatible with UTF-8, which is a frequent occurrence in social media text. By explicitly specifying the correct encoding, we ensured that all character, accented letters and special symbols - were correctly parsed, preserving the integrity of the data.

Upon successful import, we assigned descriptive column names to enhance readability and interpretability: `target`, `ids`, `date`, `flag`, `user` and `tweet`. A preliminary inspection of the dataset confirmed that all 1.6 million entries were intact, with no missing values or duplicated rows across any of the columns.

Since the primary focus of our analysis is on classifying the sentiment expressed in each tweet, we retained only the relevant columns: `tweet` and `target`. The remaining metadata - such as user ID, timestamps, and flag indicators - were excluded from further analysis as they do not contribute directly to the sentiment classification task.

2) **Text Cleaning and Normalization:**
Following data loading, a comprehensive text preprocessing step was performed to clean the tweet content and prepare it for sentiment analysis. The raw tweets contained a variety of noisy elements such as user mentions, hyperlinks, hashtags, and excessive whitespace, which are generally irrelevant for sentiment detection and can introduce noise into the modeling process. To address this, we developed a custom

cleaning function that systematically removed URLs, mentions and hashtags using regular expressions. This ensured that only the core textual content of each tweet was retained

Subsequently, the cleaned text was normalized to enforce consistency across the dataset. The normalization process included converting all characters to lowercase, removing non-alphanumeric characters while preserving key punctuation marks (such as periods, commas, exclamation points, and question marks), and reducing repeated punctuation to a single instance (eg: converting !!! to !). Additionally, redundant whitespace was removed to streamline the text structure. To further enhance the textual quality, we applied the *ftfy* (fixes text for you) library, which automatically corrects encoding issues and ensures that the tweets are represented in a clean, readable format. These normalization steps were applied to the cleaned tweets, and the resulting output was stored in a new column titled `normalized_tweet`. This preserved the original cleaned tweets while enabling downstream tasks to work on the standardized text, thereby reducing noise and improving the overall reliability and performance of sentiment classification models.

3) **Duplicate Handling and Conflict Resolution:**
Next step was to investigate duplicate entries in the dataset, particular focusing on tweets that were repeated with inconsistent labels. Initially, using only the normalized tweet content (`normalized_tweet`) for duplication detection revealed 64973 duplicates among over 1.5 million tweets. However, when both tweet content and target labels were considered, the number of full-record duplicated dropped to 47,969 indicating that several identical tweets by their normalized content and aggregated their associated labels. This revealed 3,658 unique tweets with multiple conflicting labels (same tweet labeled differently), sugessting significant labelling noise

To better illustrate these conflicts, a sample of the detected conflciting tweets and their associated labels is dipslayed. Each row displays a unique normalized tweet followed by the set of target labels it received

TABLE II
EXAMPLE OF CONFLICTING TWEETS

Normalized Tweet	Negative	Positive
ah have fun! take lots of picture of carnival stuff like cotton candy and horsies! jealous	1	1
another rainy day	11	1
at work again	7	1
being bored	17	2
but then again, it's the chinese 'a's, which is tough! oh well, i did my best.	1	1
but why?	7	8

To address these inconsistencies, we implemented a conflict resolution strategy based on label frequency and tweet length. Very short tweets (fewer than 2 words) were dropped due to lack on context. For longer tweet with clear majority label agreement, we retained the most frequent label. In ambiguous cases - such as when label counts were tied or different by ony one - the tweet gets discarded to preserve data quality. This resolution process was applied across all conflicting tweets.

After conflict resolution, the number of tweets with label inconsistencies dropped to zero, confirming that all remaining tweets had consistent labeling. Temporarily, the number of repeated tweets (regardless of labels) rose to 63,199 due to reintegration of resolved records. These were later duplicated, leaving the dataset free of both label conflicts and duplicate entries

4) **Tweet Filtering:**
To ensure data quality and eliminate noise from the corpus, we conducted a multi-faceted outlier detection process targeting atypical tweets that may impair downstream modeling. The following strategies were applied

a) Short Tweet Detection: Tweets containing fewer than two tokens were flagged as potential outliers due to insufficient semantic content. A total of 884 tweets were identified in this category. Example include fragments such as "what", "true", and blank entries. These were removed from the dataset to maintain linguistic relevance and avoid padding sparse inputs during vectorization

b) Long Tweet Check: Conversely, excessively verbose tweets may indicate spam or abnormal use cases. Tweets, exceeding 100 tokens were flagged for review. However, no such instances were found.

c) Garbled Text Detection: Checked for tweets containing non-ASCII characters, which often arise from encoding issues (eg: â€™ instead of '). Also implemented a targeted character

replacement function to normalize common problematic sequences to replace ½ with 0.5 and remove noise characters)

d) Repeated Characters: To detect non-informative or exaggerated expressions (e.g, "soooo happpyyyyy"), we flagged tweets containing repeated characters appearing more than four times. this resulted in 26,954 tweets exhibiting such patterns. These were normalized by compressing elongated sequences to at most two consecutive characters(from "cooooool" to "cool"), reducing lexcial redundancy without discarding user sentiment expression

e) Numeric-Only Tweets: Finally, scanned for tweets composed entirely of numbers (possibly spaced), which may indicate noise or mislabeling. A regex pattern was used to capture such cases. After batch-wise evaluation for scalability, no tweet containing only numbers were found

5) **Class Distribution Analysis:**
The dataset exhibits a nearly balanced class distribution, with approximately 51% of the tweets labeled as negative and 49% labeled as positive. This slight imbalance is minimal and does not pose a significant threat to model performance or evaluation metrics. The balanced nature of the dataset ensures that the classification models are less likely to be biased toward a particular class.

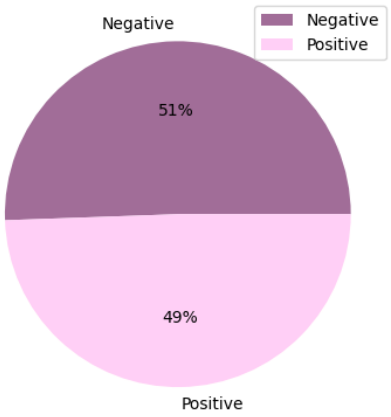

Fig. 3. Class distribution

B. *Arabic Dataset*

330K Arabic Sentiment Analysis Reviews Dataset

The second dataset consists of 330,000 Arabic product reviews, each review is annotated with a binary sentiment label (0 for negative and 1 for positive). This dataset provides a robust foundation for training and evaluating Machine Learning and Deep Learning models for NLP tasks. The large volume of the data, balanced labeling and linguistic diversity support the development of models capable of generalizing across various dialects.

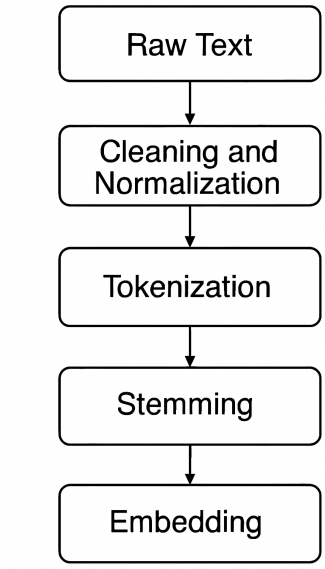

Fig. 4. Preprocessing Steps

A fundamental aspect of data science is data preparation, which involves preparing datasets for use and evaluation by machine learning and deep learning models. This essential phase includes several steps intended to improve the quality of the data and make it suitable for analytical tasks.

*1) **Preprocessing and Normalization Steps**:* We implemented a series of modular cleaning functions to ensure that the input text is standardized and stripped of noise. These are cleaning steps that significantly improved the consistency and the quality of the raw data.

1) **Stopwords Removal:**
Arabic stopwords were filtered using NLTK, excluding critical negation words to preserve sentiment context.

2) **Noise Removal:**
URLs, hashtags, mentions, numbers, and Latin characters were removed to retain only meaningful Arabic content.

3) **Punctuation Normalization:**
All punctuation was removed except for essential Arabic sentence makers (,, ?, !, .) to preserve the meaning.

4) **Character Normalization:**
Unified variant forms of letters and reduced character elongations

5) **Diacritics and Tatweel Removal:** Stripped diacritical marks and tatweel to reduce orthographic noise

6) **Emoji Removal:** Eliminated emojis using the emoji library for clean textual analysis

7) **Gibberish Filtering:** Removed non-Arabic characters using Unicode-based regular expressions

C. *Text Representation Pipeline: From Tokens to Vectors*

- **Arabic:**
  1) **Tokenization:**
  We utilized Farasa Arabic NLP toolkit to split complex Arabic words into sub-words which is a crucial step for morphologically rich languages.

  2) **Stemming:**
  Farasa's stemmer was applied on the tokenized text to reduce words to their root forms while preserving semantics and enhancing the model's generalization capabilities

  3) **Embedding:**
  To convert Arabic text into numerical representations suitable for machine learning and deep learning models, we embedded the stemmed tokens using FastText word vectors (cc.ar.300.vec). The result is a feature matrix, ready for input into models. This strategy captures semantic similarity and sub-word information, which is especially useful for handling Arabic morphology and unseen word forms.

- **English:**
  1) **Tokenization:**
  Tokenization involves two granularity levels: unigram and trigrams. For unigram tokenization, raw text is split into individual words using NLTK's `word_tokenize`, followed by stopword removal (excluding negations) and stemming. This isolates meaningful.

  For trigram tokenization, sequences of three consecutive stemmed tokens are generated (e.g., "the movie was bad" $\rightarrow$ [("movi", "wa", "bad")]). Trigrams capture contextual phrases that signal sentiment (e.g., "do not recommend"), addressing limitations of bag-of-words models by preserving local word order.

  2) **Stemming:**
  The project utilizes the porter stemmer from NLTK to reduce inflectional forms of the words to their root forms (e.g: "running" $\rightarrow$"run"). This step ensures uniformity in lexical representation while preserving sentiment-bearing negations excluded from the stopword list. The stemmer's rule-based approach strikes a balance between computational efficiency and linguistic consistency, critical for sentiment analysis tasks where word stems often carry polarity.

  3) **Embedding:**
  The model employs pre-trained Glove embeddings (100-dimensional vectors from glove.6b.100d) to initialize the embeddings layer, leveraging semantic relationships captured from large-scale corpora. The vocabulary is built from trigram sequences, converted to integer indices via keras tokenizer. The embedding layer maps these indices to fixed-dimensional vectors, with the glove weights frozen during training to preserve pre-trained semantic features. Out-of-vocabulary trigrams are zero-padded, ensuring consistent input lengths for the downstream bidirectional LSTM, CNN and RNN architectures. This approach combines the efficiency of transfer learning with contextual sensitivity of neural networks for sentiment classification.

D. *Used Machine Learning Models*

The models that were used in both datasets CatBoost, Logistic Regression, Naive Bayes and XGBoost. In addition, for the Arabic dataset SVM was added to the Arabic's models. On the other hand, in the English dataset LightGBM and Ridge Classifier were added to the english models. For all models in Arabic, dense FastText embeddings were used as input features, while for English, trigram token features were vectorized using CountVectorizer then data were splitted then passed to the models then finally tuned. The results of all models showed efficiency and solid performances.

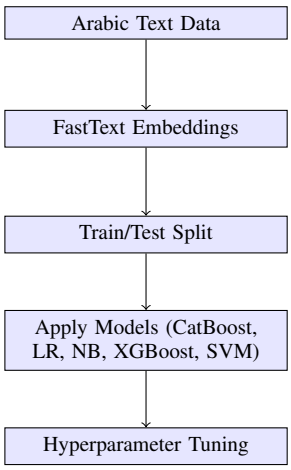

Fig. 5. Arabic Dataset Machine Learning Pipeline

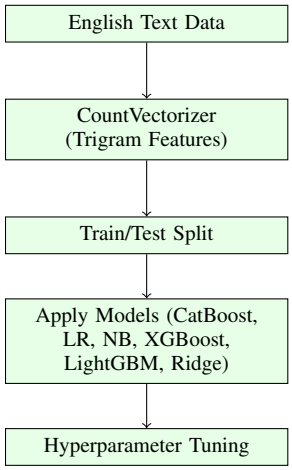

Fig. 6. English Dataset Machine Learning Pipeline

1) **CatBoost**

The CatBoost model is a gradient boosting model that was chosen for Arabic and English text analysis because it has a strong ability to handle non-numeric and sparse data without heavy preprocessing. It is a smart way for handling categorical variables and resistance to overfitting makes it a good choice for the language sentiment.

a) **CatBoost for Arabic dataset:**

TABLE III
CATBOOST CLASSIFICATION RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.797	0.784	0.781
AUC-ROC (macro OvR)	—	—	0.863

Class	Precision	Recall	F1-Score	Support
0	0.77	0.79	0.78	32,629
1	0.79	0.77	0.78	33,371
Macro Avg	0.78	0.78	0.78	66,000
Weighted Avg	0.78	0.78	0.78	66,000

b) **CatBoost for English dataset:**

TABLE IV
CATBOOST CLASSIFICATION RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.6256	0.6179	0.6164

Class	Precision	Recall	F1-Score	Support
-1	0.61	0.67	0.64	141567
1	0.62	0.56	0.59	138451
Macro Avg	0.62	0.62	0.62	280018
Weighted Avg	0.62	0.62	0.62	280018

2) **Logistic Regression**

Logistic regression is a fundamental linear model that finds common in both binary and multiclass classification tasks, making it a simple fit for sentiment analysis. Its interpretability and computational efficiency makes it suitable for text-based tasks where input features are often high-dimensional. Regardless its simplicity, logistic regression is surprisingly efficient, especially when used in combination with feature engineering techniques like CountVectorizer or TF-IDF. The most important thing in this model is its simplicity, robustness to overfitting in low-dimensional settings, and ability to show probabilistic outputs, which are useful in real-world sentiment analysis projects.

a) **Logistic Regression for Arabic dataset:**

TABLE V
LOGISTIC REGRESSION RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.798	0.798	0.796
AUC-ROC (macro OvR)	—	—	0.87

Class	Precision	Recall	F1-Score	Support
0	0.79	0.80	0.80	32629
1	0.81	0.79	0.80	33371
Macro Avg	0.80	0.80	0.80	66,000
Weighted Avg	0.80	0.80	0.80	66,000

b) **Logistic Regression for English dataset:**

TABLE VI
LOGISTIC REGRESSION RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.5984	0.5986	0.5975

Class	Precision	Recall	F1-Score	Support
-1	0.60	0.60	0.60	141567
1	0.59	0.60	0.59	138451
Macro Avg	0.60	0.60	0.60	280018
Weighted Avg	0.60	0.60	0.60	280018

3) **Naive Bayes**

Naive Bayes is a classifier that is nearest to the Bayes theorem, assuming that all the features are independent. In spite of its simplicity, it has proven to be extremely effective in text classification tasks, including sentiment analysis, which benefits from the sparse and discrete representation of textual data. Thus, this model particularly excels in high-dimensional input spaces, like those resulting from a vectorized representation of text—commonly used in our NLP pipeline. In addition, it also performs surprisingly well in smaller datasets and offers really fast training times, making it an ideal candidate for those situations where you require something highly scalable and also a bit iterative.

a) **Naive Bayes for Arabic dataset:**

b) **Naive Bayes for English dataset:**

4) **XGBoost**

Extreme Gradient Boosting (XGBoost) is a highly efficient and scalable implementation of gradient boosting algorithms, and it is especially effective in tasks of sentiment analysis. It also builds on a solid foundation of gradient boosting with regularization and pruning strategies to reduce overfitting while maintaining high performance. Although the core algorithm is powerful

TABLE VII
NAIVE BAYES RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.643	0.645	0.642

Class	Precision	Recall	F1-Score	Support
0	0.61	0.77	0.68	32629
1	0.70	0.52	0.60	33371
Macro Avg	0.65	0.64	0.64	660000
Weighted Avg	0.65	0.64	0.64	660000

TABLE VIII
NAIVE BAYES RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.5817	0.5832	0.5811

Class	Precision	Recall	F1-Score	Support
-1	0.58	0.60	0.59	141567
1	0.58	0.57	0.57	138451
Macro Avg	0.58	0.58	0.58	280018
Weighted Avg	0.58	0.58	0.58	280018

in its own right, XGBoost also incorporates a number of useful techniques for handling missing values, optimizing computational resources, and achieving top-flight accuracy, especially in ensemble-based settings. All of these strengths make XGBoost a go-to algorithm for many data scientists, and its frequent appearance in winning competition solutions is a testament to its capabilities.

a) **XGBoost for Arabic dataset:**

TABLE IX
XGBOOST RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.8343	0.7917	0.7901

Class	Precision	Recall	F1-Score	Support
0	0.78	0.80	0.79	32629
1	0.80	0.78	0.79	33371
Macro Avg	0.79	0.79	0.79	66000
Weighted Avg	0.79	0.79	0.79	66000

b) **XGBoost for English dataset:**

TABLE X
XGBOOST RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.6705	0.6782	0.6279

Class	Precision	Recall	F1-Score	Support
-1	0.62	0.67	0.65	141567
1	0.63	0.59	0.61	138451
Macro Avg	0.63	0.63	0.63	280018
Weighted Avg	0.63	0.63	0.63	280018

5) **Ridge Classifier**

Ridge Classifier is a linear classification model that applies L2 regularization to prevent overfitting, particularly when dealing with multicollinearity and high-dimensional feature sets. In sentiment analysis, Ridge Classifier performs well when input features are highly sparse. Its point of strength lies in balancing bias and variance through regularization, leading to better generalization on unseen data. The model also offers a closed-form solution, which ensures fast training and makes it suitable for baseline and production-ready systems in NLP.

a) **Ridge Classifier for English dataset:**

TABLE XI
RIDGE CLASSIFIER RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.5983	0.5983	0.5974

Class	Precision	Recall	F1-Score	Support
-1	0.60	0.60	0.60	141567
1	0.59	0.60	0.60	138451
Macro Avg	0.60	0.60	0.60	280018
Weighted Avg	0.60	0.60	0.60	280018

6) **SVM**

Support Vector Machines are potent margin-based classifiers that are especially good in high-dimensional spaces. SVMs are well-suited for sentiment analysis tasks because they can and do construct optimal decision boundaries in complex feature spaces. They are particularly valuable when the number of features exceeds the number of observations, which is not unusual in the world of NLP. One of the key strengths of SVMs is their robustness to overfitting. Overfitting, you may recall, is a bad phenomenon that occurs when a model learns the training data too well and fails to make accurate predictions on new, unseen data. SVMs don't have this problem to anywhere near the extent that, say, logistic regression has. SVMs have this strength partly because they use a kind of regularization that helps models perform better on the competition held at Kaggle, which is our main concern here.

a) **SVM for Arabic dataset:**

TABLE XII
SVM RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.7989	0.7992	0.7964

Class	Precision	Recall	F1-Score	Support
0	0.79	0.81	0.80	32629
1	0.81	0.79	0.80	33371
Macro Avg	0.80	0.80	0.80	66000
Weighted Avg	0.80	0.80	0.80	66000

7) **LightGBM**

The Light Gradient Boosting Machine (LightGBM) is a boosting algorithm designed for high efficiency and

speed. It stems from the gradient boosting framework, with optimizations like tree growth by leaves and decision splitting by histograms, which together reduce training time significantly, with no apparent increase in error rate. Sentiment analysis is a common task in natural language processing. For such tasks, LightGBM (in gradient boosting mode) is particularly effective for two reasons: it supports large feature spaces (up to hundreds of thousands of features), and it has a very efficient implementation that works well in a distributed setting. However, since the main task is text classification, one important aspect in evaluating the performance of such algorithms is their speed and scalability, as well as the quality of the obtained results.

a) **LightGBM for English dataset:**

TABLE XIII
LIGHTGBM RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.6826	0.6325	0.6303

Class	Precision	Recall	F1-Score	Support
-1	0.63	0.67	0.65	141567
1	0.64	0.59	0.61	138451
Macro Avg	0.63	0.63	0.63	280018
Weighted Avg	0.63	0.63	0.63	280018

E. Used Deep Learning Models

Common principles between the English and Arabic datasets guided the model selection, training, and evaluation, including the systematic partitioning of data into training, validation, and test sets, the use of optimization algorithms for model training, and the assessment of performance using established metrics such as accuracy, precision, recall, and F1-score. Our approach incorporated both traditional deep learning architectures, such as recurrent and convolutional neural networks, and advanced Transformer-based models to leverage their respective strengths in natural language understanding.

*1) **Traditional Deep Learning Architectures**:* For both English and Arabic, several traditional deep learning models were constructed with a shared foundational structure. Each model typically began with an initial word embedding layer, converting tokenized input sequences into dense vector representations. Following this, specialized processing layers (which were either recurrent or convolutional) were introduced, accompanied by dropout layers for regularization, before a terminal dense output layer applied sigmoid activation function for binary classification. For compilation, the models utilized the Adam optimizer and a binary crossentropy loss function, with accuracy designated as the primary performance metric. Training involved iterating for up to 10 epochs with a batch size of 32. To enhance generalization and prevent overfitting, ModelCheckpoint was employed to save the best performing model based on validation accuracy, and EarlyStopping with a patience of 3 on validation loss was utilized to halt training when performance ceased to improve.

***Bidirectional Long Short-Term Memory (BiLSTM) Model**:* The Bidirectional LSTM model was chosen for its efficacy in capturing long-range dependencies and contextual information by processing sequential data in both forward and backward directions. This bidirectionality is particularly advantageous for sentiment analysis, where the sentiment expressed can be heavily influenced by the interplay of words across a sentence, including those preceding and succeeding a given term. For instance, understanding negation or complex sentiment expressions often requires appreciating context from both temporal directions.

For the English dataset, this architecture comprised a Bidirectional(LSTM(units)) layer, where units (32 or 64) and dropout rate (0.3 or 0.5) were hyperparameters tuned during experimentation. The input was derived from tokenized and padded sequences, with the initial Embedding layer initialized using pre-trained GloVe embeddings.

TABLE XIV
ENGLISH BILSTM CLASSIFIER RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.8233	0.8133	0.81

Class	Precision	Recall	F1-Score	Support
0	0.81	0.82	0.81	69829
1	0.82	0.80	0.81	69158
Macro Avg	0.81	0.81	0.81	280018
Weighted Avg	0.81	0.81	0.81	280018

For the Arabic dataset, a two-layer LSTM architecture was implemented to capture the temporal dependencies within Arabic text. This model featured 128 hidden units per layer, providing substantial capacity to learn complex sequential patterns. A dropout rate of 0.5 was applied between layers to mitigate overfitting. The model's final output was passed through a fully connected layer to predict sentiment classes.

TABLE XV
ARABIC BILSTM CLASSIFIER RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.8278	0.8165	0.82

Class	Precision	Recall	F1-Score	Support
0	0.81	0.83	0.82	32629
1	0.83	0.81	0.82	33371
Macro Avg	0.82	0.82	0.82	66000
Weighted Avg	0.82	0.82	0.82	66000

***Simple Recurrent Neural Network (RNN) Model**:* The Simple Recurrent Neural Network model served as a foundational baseline for sequential processing. This architecture highlights the inherent capability of recurrent layers to recognize "temporal patterns," referring to the sequential relationships and ordering of words that contribute

to meaning and sentiment within a text. While less adept at long-term dependency learning than LSTMs, it provides a crucial comparative perspective on the benefits of more complex recurrent architectures.

For the English dataset, the Simple RNN model incorporated a SimpleRNN(units) layer, with units (32 or 64) and dropout rate (0.3 or 0.5) as configurable hyperparameters. The input followed the same tokenization, padding, and GloVe embedding strategy as the BiLSTM.

TABLE XVI
ENGLISH RNN CLASSIFIER RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.6836	0.7055	0.71

Class	Precision	Recall	F1-Score	Support
0	0.72	0.68	0.70	69829
1	0.69	0.73	0.71	69158
Macro Avg	0.71	0.71	0.71	138987
Weighted Avg	0.71	0.71	0.71	138987

For the Arabic dataset, a custom RNN model was implemented using PyTorch, featuring a two-layer architecture with a hidden dim of 128. This model processes sequences to capture temporal dependencies within the Arabic text. It was optimized using torch.optim.Adam and trained with nn.CrossEntropyLoss(). Training involved iterating over a predefined number of epochs, with performance monitored on training, validation, and test loaders, evaluating metrics such as Loss, Accuracy, Mean Squared Error (MSE), and a composite Performance Score.

TABLE XVII
ARABIC RNN CLASSIFIER RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.8073	0.8073	0.8033

Class	Precision	Recall	F1-Score	Support
0	0.80	0.81	0.80	32629
1	0.81	0.80	0.80	33371
Macro Avg	0.80	0.80	0.80	66000
Weighted Avg	0.80	0.80	0.80	66000

Convolutional Neural Network (CNN) Model: Convolutional Neural Networks, traditionally prominent in image processing, have demonstrated significant utility in Natural Language Processing (NLP) for their ability to extract local and position-invariant features from text. In this context, a 1D convolution layer effectively identifies n-gram-like patterns (e.g., phrases or specific word combinations) indicative of sentiment, regardless of their exact position within a sentence.

The non-linear ReLU (Rectified Linear Unit) activation function was applied after the convolutional operations. Defined as

$$\text{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

ReLU introduces essential non-linearity, enabling the model to learn complex, non-linear relationships within the data, which is critical for representing intricate linguistic patterns. Its computational efficiency also contributes to faster training.

For the English dataset, the CNN model utilized a Conv1D layer with 128 filters and a kernel size of 5, followed by a GlobalMaxPooling1D layer to capture the most salient features across the sequence. The input was based on GloVe embeddings.

TABLE XVIII
ENGLISH CNN CLASSIFIER RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.7846	0.7910	0.79

Class	Precision	Recall	F1-Score	Support
0	0.79	0.79	0.79	69829
1	0.79	0.79	0.79	69829
Macro Avg	0.79	0.79	0.79	138987
Weighted Avg	0.79	0.79	0.79	138987

For the Arabic dataset, the Convolutional Neural Network employed a single 1D convolution layer with 100 filters and a kernel size of 3, designed to detect finer local patterns. This layer was followed by a ReLU activation function and an adaptive max-pooling layer to efficiently summarize the most relevant features. The pooled features were subsequently passed through a dropout layer and a fully connected layer for final classification.

TABLE XIX
ARABIC CNN CLASSIFIER RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.5987	0.5963	0.5975

Class	Precision	Recall	F1-Score	Support
0	0.59	0.61	0.60	32629
1	0.60	0.59	0.60	33371
Macro Avg	0.60	0.60	0.60	66000
Weighted Avg	0.60	0.60	0.60	66000

*2) **Transformer-based Models:*** Transformer models represent a paradigm shift in NLP. By utilizing self-attention mechanisms that dynamically assess the significance of various words within a sequence, these models are able to capture long-range dependencies and complex contextual relationships with greater efficacy than conventional recurrent models. These models are typically pre-trained on massive text corpora and then fine-tunes on smaller, task-specific datasets.

***English Sentiment Analysis (DistilBERT)**:* For English sentiment analysis, a DistilBERT model was utilized, specifically the *TFDistilBertForSequenceClassification* architecure pre-trained on the d*istilbert-base-uncased* checkpoint. The *DistilBertTokenizerFast* from the same pre-trained checkpoint was employed for tokenization, handling padding to a max-length of 128 and truncation. Hyperparameter optimization was performed using *Optuna* to minimize validation loss. The tuned hyperparameters included batch size, learning rate, number of training epochs, and weight decay.

TABLE XX
DISTILBERT RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.8519	0.8368	0.8024

Class	Precision	Recall	F1-Score	Support
0	0.8378	0.7557	0.7947	105488
1	0.7726	0.8502	0.8096	102992
Macro Avg	0.8052	0.8030	0.8021	208480
Weighted Avg	0.8056	0.8024	0.8020	208480

***Arabic Sentiment Analysis (AraBERT)**:* For Arabic sentiment analysis, the AraBERT model(aubmindlab/bert-uncased-arabertv02) was utilized. AraBERT is a BERT-based model pre-trained extensively on a large corpus of Arabic text, making it highly specialized and effective for understanding the nuances of the Arabic language, including its rich morphology and specific linguistic structures. The model was fine-tuned for binary sentiment classification using an AutoTokenizer for handling text tokenization and input formatting. Hyparameter optimization was performed using Optuna, which systematically explored optimal values for learning rate, weight decay, and batch size.

TABLE XXI
ARABERT RESULTS AFTER TUNING

Metric	Training	Validation	Testing
Accuracy	0.8519	0.8368	0.8024

Class	Precision	Recall	F1-Score	Support
0	0.8506	0.8813	0.8657	17895
1	0.7425	0.6886	0.7145	8894
Macro Avg	0.7965	0.7849	0.7901	26789
Weighted Avg	0.8147	0.8173	0.8155	26789

F. Performance Metrics

For evaluating the performance and dependability of the sentiment analysis models, a set of classification-specific metrics was employed. A comprehensive assessment of the models' effectiveness in correctly classifying positive and negative sentiments is provided by these metrics.

***Accuracy**:* Accuracy measures the proportion of total predictions that were correct. It is calculated by dividing the number of correctly identified instances by the total count of instances.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{1}$$

***Precision**:* Precision quantifies the percentage of true positive identifications among all positive predictions. This measure is especially critical in situations where the cost associated with a false positive is substantial.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}} \tag{2}$$

***Recall (Sensitivity)**:* Recall quantifies the percentage of all actual positive instances that were successfully detected. It is crucial when the cost of a false negative is high.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}} \tag{3}$$

***F1-Score**:* The F1-Score is the harmonic mean of the Precision and Recall. It provides a single metric that balances both precision and recall, being particularly useful for imbalanced datasets.

$$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

IV. RESULTS AND ANALYSIS

COMPARISON OF MACHINE LEARNING MODELS ON THE ARABIC DATASET

Model Name	Precision	Recall	F1-Score	Accuracy
CatBoost	0.78	0.78	0.78	0.78
SVM	0.78	0.82	0.80	0.79
Logistic Regression	0.79	0.80	0.80	0.80
Naive Bayes	0.64	0.65	0.64	0.64
XGBoost	0.79	0.79	0.79	0.79

COMPARISON OF MACHINE LEARNING MODELS ON THE ENGLISH DATASET

Model Name	Precision	Recall	F1-Score	Accuracy
CatBoost	0.61	0.62	0.62	0.62
Ridge Classifier	0.59	0.60	0.60	0.60
Logistic Regression	0.59	0.60	0.60	0.60
Naive Bayes	0.58	0.58	0.58	0.58
XGBoost	0.67	0.63	0.63	0.63
LightGBM	0.68	0.63	0.63	0.63

The analysis of model performance with respect to Arabic and English datasets shows that models perform relatively better on the Arabic dataset compared to the English dataset. For the Arabic dataset, Logistic Regression, XGBoost, and

CatBoost gave the most stable performance with respect to all evaluation metrics, achieving F1-scores in the range of 0.78 to 0.80. Naive Bayes, however, performed the worst out of all models for both datasets with very low scores in precision and F1-score, particularly in comparison to the other models.

Performance across models in the English dataset was comparatively lower. LightGBM and XGBoost were the best with an F1-score and accuracy of approximatively 0.63, while other models such as Logistic Regression and Ridge Classifier performed at around 0.60. This indicates that less complex ensemble techniques are preferable for the English dataset in these models.

These findings suggest that the usefulness of a model depends pliantly on the language features of the dataset. Furthermore, they demonstrate that ensemble models like XGBoost, CatBoost, and LightGBM perform well across the two datasets. With more English language focused feature enhancement, the corrective activities needed to improve performance on the dataset could help close the gap.

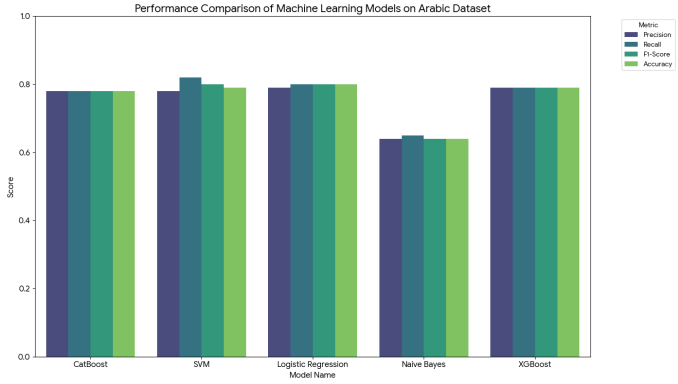

Fig. 7. Comparison of Machine Learning Models on the Arabic Dataset

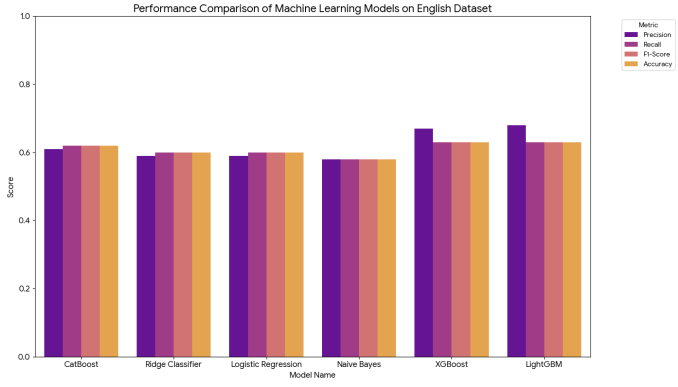

Fig. 8. Comparison of Machine Learning Models on the English Dataset

COMPARISON OF DEEP LEARNING MODELS ON THE ARABIC DATASET

Model Name	Precision	Recall	F1-Score	Accuracy
BiLSTM	0.82	0.82	0.82	0.82
RNN	0.80	0.80	0.80	0.8033
CNN	0.6	0.6	0.6	0.5975
AraBERT	0.7965	0.7849	0.7901	0.8173

COMPARISON OF DEEP LEARNING MODELS ON THE ENGLISH DATASET

Model Name	Precision	Recall	F1-Score	Accuracy
BiLSTM	0.81	0.81	0.81	0.81
RNN	0.71	0.71	0.71	0.71
CNN	0.79	0.79	0.79	0.79
DistilBERT	0.8056	0.8024	0.8020	0.8024

The analysis of deep learning model performance with respect to Arabic and English datasets shows strong results, particularly for language-specific models.

For the Arabic dataset, BiLSTM demonstrated the strongest performance, achieving an F1-score of 0.82 and an accuracy of 0.82. AraBERT and RNN also performed commendably, with accuracies around 0.80. The CNN model, however, showed significantly lower accuracy of 0.5975.

Performance across models in the English dataset was also strong, with BiLSTM leading with an F1-score of 0.81 and an accuracy of 0.81. DistilBERT followed closely with an F1-score of 0.8020 and an accuracy of 0.8024. The CNN model achieved an F1-score of 0.79 and accuracy of 0.79, while the RNN model had comparatively lower scores, with an F1-score and accuracy of 0.71.

These findings suggest that advanced deep learning architectures, particularly those designed to capture sequential dependencies and language nuances like BiLSTM and transformer-based models (DistilBERT, AraBERT), are highly effective for sentiment analysis across both languages. They generally offer improved classification performance over traditional machine learning techniques, emphasizing the critical role of refined model selection that is customized to language-specific traits for effective social media analysis.

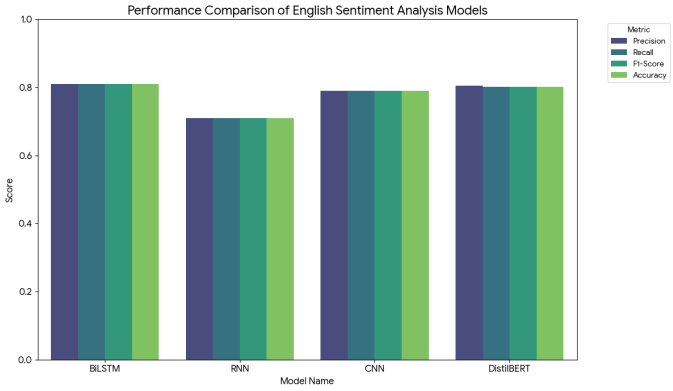

Fig. 9. Comparison of Deep Learning Models on the English Dataset

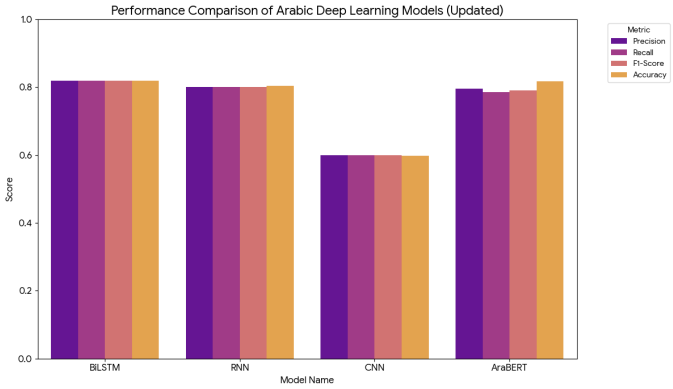

Fig. 10. Comparison of Deep Learning Models on the Arabic Dataset

V. PRAW API Testing Results

A. Arabic Sentiment Prediction

The trained BiLSTM model performed sentiment classification on the processed embeddings. The model's output probabilities were converted to binary sentiment labels using an argmax operation:

- **Label 0**: Negative sentiment
- **Label 1**: Positive sentiment

TABLE XXII
Sample Arabic Sentiment Analysis Results (BiLSTM)

Comment (Arabic)	**Predicted Label**
يعني واللة بجد انا جربت بيع الصور بيع الشعارات التسويق بالعمولة الطباعة عند الطلب بيع المقالات بس مكسبتش ولا جنية لحد دلوك ضيعت وقت علي الفاضي حد جرب الطرق دي ونفعت معاه وجربها إزاي	Negative
حبيبي الغالي وفقك الله وشكرا جدا علي كلماتك الجميله ونصيحتك وسام لي قبل نقدك بل بالعكس اخذ نقدك واضعه ببرواز من ذهب يشبعك شكرا من القلب و ب اذن الله اكون احسن ابشر	Positive
كيف؟ مجرد تحفيز أم تنويم مغناطيسي؟؟	Negative
اتمنى انهم يمسكونك ويعيدون تأهيلك استح وخاف من ربك على الاقل	Negative
عندك حق شكرا لتعليقك	Positive
تبدو الكثير من الروايات مفتعلة أو أن أفكارها تجارية	Negative

B. English Sentiment Prediction

TABLE XXIII
English Sentiment Analysis Results (BERT)

English Comment	**Predicted Label**
i mean it looks great. good on my bro..	Positive
this doesn't seem church related at all.	Negative
sounds stupid, but uno. i think its on steam, but dont know for certain as i dont have a pc	Negative
you worked with what you had back then	Positive
is there a way to fix a lb button on generic xbox controller that wont respond unless mashed down in a certain way?	Neutral

TABLE XXIV
English Sentiment Analysis Results (BiLSTM)

English Comment	**Predicted Label**
Windows Was The Problem All Along	Negative
I never thought Id see one in the wild... .	Negative
the first game that emotionally devastated me (FF9)	Positive
Take your time, you got this	Positive
Parents use culture to justify toxic behavior Culture is supposed to represent part of your identity.	Negative

VI. Conclusion

This research showcased the effectiveness of diverse machine learning and deep learning models for sentiment analysis across Arabic and English datasets. The evaluation considered key metrics such as Precision, Recall, F1-Score, and Accuracy.

For the English dataset, deep learning models like DistilBERT and BiLSTM showed strong performance, achieving metrics in the range of 80-81%. In contrast, traditional machine learning models such as XGBoost and LightGBM performed comparably well within their category for the English dataset

with metrics around 63-68%.

On the Arabic dataset, deep learning models, specifically BiLSTM and AraBERT, also exhibited high effectiveness, with performance metrics ranging from 80-82%. Among the machine learning models applied to the Arabic dataset, Logistic Regression, SVM, and XGBoost were among the top performers, generally achieving scores around 79-80%.

Overall, deep learning architectures consistently achieved higher performance metrics compared to traditional machine learning models across both English and Arabic sentiment analysis tasks. This highlights their capability in capturing complex patterns within textual data, making them highly suitable for advanced natural language processing applications like sentiment analysis. The continued advancement of these algorithms is crucial for developing more accurate and nuanced sentiment prediction systems in various linguistic contexts.

REFERENCES

[1] M. S. Basarslan and F. Kayaalp, "Sentiment analysis with machine learning methods on social media," *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, vol. 9, no. 3, pp. 5–15, sep 2020. [Online]. Available: https://revistas.usal.es/cinco/index.php/2255-2863/article/view/ADCAIJ202093515

[2] V. D. Derbentsev, V. S. Bezkorovainyi, A. V. Matviychuk, O. M. Pomazun, A. V. Hrabariev, and A. M. Hostryk, "Sentiment analysis of electronic social media based on deep learning," in *Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART)*. SciTePress, 2022, pp. 119–128. [Online]. Available: https://www.scitepress.org/Papers/2022/119323/119323.pdf

[3] G. S. Hussein and A. Riad, "Arabic sentiment analysis using deep learning and machine learning approaches," *Journal of Computing and Communication*, vol. 3, no. 2, pp. 10–22, july 2024. [Online]. Available: https://journals.ekb.eg/article_380113.html

[4] N. M. Ali, M. M. Abd El Hamid, and A. Youssif, "Sentiment analysis for movies reviews dataset using deep learning models," *International Journal of Data Mining & Knowledge Management Process (IJDKP)*, vol. 9, no. 2/3, may 2019. [Online]. Available: https://www.researchgate.net/publication/333607586_SENTIMENT_ANALYSIS_FOR_MOVIES_REVIEWS_DATASET_USING_DEEP_LEARNING_MODELS

[5] A. Mohammed and R. Kora, "Deep learning approaches for arabic sentiment analysis," *Neural Computing and Applications*, vol. 32, pp. 10 583–10 594, 2020. [Online]. Available: https://www.researchgate.net/publication/335969728_Deep_learning_approaches_for_Arabic_sentiment_analysis

[6] A. Alharbi, M. Kalkatawi, and M. Taileb, "Arabic sentiment analysis using deep learning and ensemble methods," *Arabian Journal for Science and Engineering*, vol. 46, pp. 8913–8923, may 2021. [Online]. Available: https://link.springer.com/article/10.1007/s13369-021-05475-0