SN Computer Science (2025) 6:205
https://doi.org/10.1007/s42979-025-03741-8

ORIGINAL RESEARCH

Multilingual Hate Speech Detection: Innovations in Optimized Deep Learning for English and Arabic Hate Speech Detection

Hassan AL-Sukhani[1] · Qusay Bsoul[2] · Abdelrahman H. Elhawary[3] · Ziad M. Nasr[3] · Ahmed E. Mansour[4] · Radwan M. Batyha[5] · Basma S. Alqadi[6] · Jehad Saad Alqurni[7] · Hayat Alfagham[8] · Magda M. Madbouly[9,10]

Received: 8 September 2024 / Accepted: 5 December 2024

Abstract

This paper presents the development of a multilingual hate speech detection model that effectively processes and classifies content in both Arabic and English. The study leverages both traditional machine learning models, such as K-Nearest Neighbors (KNN), Naive Bayes, and Support Vector Machines (SVM), as well as advanced deep learning models, specifically Bi-directional Long Short-Term Memory (Bi-LSTM) networks. A key challenge addressed is the classification of mixed-language content, which is common on social media platforms in the MENA region. To enhance detection performance, preprocessing techniques were applied to the text data, and the Synthetic Minority Over-sampling Technique (SMOTE) was used to balance the dataset. The results show that the Bi-LSTM model outperformed traditional machine learning approaches, particularly in identifying hate speech across multiple languages. The proposed model demonstrates superior accuracy and robustness in handling mixed-language input, providing a more effective solution for real-world hate speech detection tasks.

Keywords Hate speech detection · Natural language processing (NLP) · Mixed Arabic and English training · Machine learning · Deep learning · Class imbalance · Sentiment analysis · Multilingual NLP

✉ Ahmed E. Mansour
dotahmed@aol.com

Hassan AL-Sukhani
h.sukhni@jadara.edu.jo

Qusay Bsoul
q.bsoul@aau.edu.jo

Abdelrahman H. Elhawary
A.Elhawary@gmail.com

Ziad M. Nasr
z.Nasr@gmail.com

Radwan M. Batyha
r_batiha@asu.edu.jo

Basma S. Alqadi
bsalqadi@imamu.edu.sa

Jehad Saad Alqurni
jalqurni@iau.edu.sa

Hayat Alfagham
hmalfagham@iau.edu.sa

Magda M. Madbouly
mmadbouly@alexu.edu.eg

[1] Cybersecurity Department, Faculty of Science and Information Technology, Jadara University, Irbid, Jordan

[2] Cybersecurity Department, College of Computer Sciences and Informatics, Amman Arab University, Amman, Jordan

[3] Faculty of Engineering, Ain Shams University, Cairo, Egypt

[4] Faculty of Computer Science, Misr International University, Cairo, Egypt

[5] Faculty of Information Technology, Applied Science Private University, Amman 11931, Jordan

[6] Computer Science Department, College of Computer and Information Sciences, Al Imam Mohammad Ibn Saud Islamic University, Riyadh, Saudi Arabia

[7] Department of Educational Technologies, College of Education, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam, Saudi Arabia

[8] Department of MIS, College of Applied Studies and Community Service, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam, Saudi Arabia

[9] Institute of Graduate Studies & Research, Alexandria University, Alexandria, Egypt

[10] Faculty of Computer and Data Science, Alexandria University, Alexandria, Egypt

Published online: 20 February 2025

Introduction

Hate speech on social media platforms has become a growing concern due to its harmful effects on individuals and communities [1]. Platforms like Twitter and Facebook are increasingly used to spread abusive language and harmful ideologies, necessitating the development of automated systems to detect and mitigate such content. While significant progress has been made in detecting hate speech in monolingual environments, the problem becomes more complex when the content spans multiple languages, particularly in regions like the Middle East and North Africa (MENA), where users frequently mix Arabic and English in a single post [2].

Hate speech detection is particularly significant in multilingual contexts such as the MENA region, where social media content often includes a mix of Arabic and English, known as code-switching. The coexistence of languages within a single text introduces linguistic challenges that traditional monolingual models cannot address effectively. This linguistic blending, commonly referred to as code-switching, presents unique challenges for hate speech detection models. Recent studies indicate that over 25% of social media content in the MENA region is written in a mix of Arabic and English. This prevalence highlights the urgent need for detection models capable of effectively processing multilingual inputs. Traditional machine learning models, typically trained on monolingual datasets, struggle with these scenarios due to their reliance on language-specific features and lack of contextual understanding across languages. As a result, these models often misclassify mixed-language posts, either overgeneralizing or failing to detect hate speech.

Existing models tend to focus on either Arabic or English and struggle with language-specific nuances, leading to inaccuracies in detecting hate speech in mixed-language environments. The gap in the existing literature lies in the limited research on multilingual or cross-lingual hate speech detection. While various models have been developed to address hate speech in either Arabic or English, few studies have tackled the challenge of mixed-language content, which is increasingly common on social media. This paper seeks to fill this gap by proposing a model that combines traditional machine learning techniques with deep learning approaches, specifically tailored for detecting hate speech in mixed Arabic-English contexts. By incorporating preprocessing techniques and a deep learning architecture that captures dependencies across languages, this model addresses these limitations and enhances detection accuracy for code-switched text. By addressing the challenges of language imbalance and code-switching, this study contributes to the advancement of multilingual hate speech detection.

As a summary our contributions are as follows:

- We introduce a novel approach to detecting hate speech in mixed Arabic and English social media content, leveraging a combination of machine learning and deep learning techniques.
- The research presents a balanced multilingual dataset that incorporates both Arabic and English, addressing the imbalance of available monolingual datasets.
- We propose a Bi-directional Long Short-Term Memory (Bi-LSTM) deep learning model, which outperforms traditional machine learning models in handling code-switched text.
- The study uses advanced preprocessing techniques and the Synthetic Minority Over-sampling Technique (SMOTE) to address data imbalance, improving the model's generalization capability across languages.

The rest of the paper is organized as follows: section two presents the collected related work. Section three explores the key challenges that faced us while trying to train this model. Section four introduces the proposed models. The experimental results are shown in section five. Section six shows a detailed comparison between the different proposed models. Section seven concludes the paper and provides some future works.

Related Work

Hate speech detection has been an active area of research, with various machine learning and deep learning techniques being employed to improve accuracy and robustness. Traditional machine learning models [3] the authors provides a comprehensive overvie such as Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Naive Bayes (NB), Decision Trees (DT), and Random Forest (RF) have demonstrated early success in detecting hate speech by leveraging features like word embeddings and term frequency-inverse document frequency (TF-IDF). More recently, deep learning models have gained prominence, including Bi-directional Long Short-Term Memory (Bi-LSTM) and Convolutional Neural Networks (CNN) [4], which excel at capturing contextual and sequential data from text. Additionally, transformer-based models like BERT have further advanced the field by offering enhanced language understanding. Ensemble models, combining multiple classifiers, have been explored to enhance detection performance, while data balancing techniques like SMOTE have been applied to address class imbalance issues, particularly in hate speech datasets where offensive content is often underrepresented [5]. These diverse approaches collectively form the foundation of modern hate speech detection systems.

Traditional Machine Learning Models

Support Vector Machines (SVM)

Badjatiya et al. [6] explores deep learning techniques for hate speech detection on Twitter using word embeddings and various neural network models. The paper utilizes CNN and LSTMs, achieving notable performance improvements by employing word embeddings for detecting hate speech on Twitter. Del Vigna et al. [7] focuses on detecting hate speech in Italian Facebook comments using machine learning approaches. The paper introduces the first manually annotated Italian hate speech corpus and compares SVM and LSTM classifiers for detecting hate speech on Facebook.

K-Nearest Neighbors (KNN)

Faris et al. [8] focuses on detecting hate speech in Arabic using word embedding and deep learning techniques such as CNN and LSTM. The paper creates a dataset for Arabic hate speech detection and combines CNN with LSTM, achieving high performance.

Naive Bayes (NB)

Nobata et al. [9] presents a machine learning-based approach for detecting abusive language in online content. It develops a supervised classification method with NLP features that outperforms deep learning models in this domain.

Decision Trees (DT)

Omar et al. [10] compares traditional machine learning and deep learning algorithms for hate speech detection in Arabic on OSNs. The study introduces an Arabic hate speech dataset and finds that RNN outperformed other classifiers with 98.7% accuracy.

Random Forest (RF)

Omar et al. [10] also evaluates Random Forest as one of the models in the comparison of machine learning and deep learning algorithms for Arabic hate speech detection. Nobata et al. [9] evaluates Random Forest along with other models for detecting abusive language in online content, showing its effectiveness in this domain.

Deep Learning Models

Bi-directional Long Short-Term Memory (Bi-LSTM)

Badjatiya et al. [6] presents a deep learning approach for Twitter hate speech detection using word embeddings. The paper compares CNN and Bi-LSTM models for hate speech detection, with Bi-LSTM demonstrating high performance. Omar et al. [10] evaluates Bi-LSTM along with traditional ML and DL algorithms, finding that Bi-LSTM performs well in comparison to traditional machine learning algorithms for detecting hate speech in Arabic.

Convolutional Neural Networks (CNN)

Badjatiya et al. [6] focuses on applying deep learning techniques for detecting hate speech on Twitter, using CNNs for word embeddings and achieving high performance. Alshalan et al. [11] explores hate speech detection in the Saudi Twittersphere using deep learning approaches, with CNN models applied for Arabic hate speech detection and showing strong results.

BERT-Based Approaches

Alshalan et al. [11] applies BERT-based approaches for hate speech detection in Saudi Twitter, demonstrating the model's effectiveness in this domain. MacAvaney et al. [12] addresses challenges in hate speech detection and proposes BERT-based models for better interpretability, contributing to the improvement of detection systems.

Ensemble Models

Zimmerman et al. [13] explores the use of ensemble deep learning models to improve hate speech detection. The paper demonstrates the effectiveness of combining multiple deep learning models to enhance performance. Zimmerman et al. [14] focuses on improving hate speech detection through ensemble learning techniques. The study shows that combining various deep learning models results in improved performance.

Data Balancing Techniques

SMOTE (Synthetic Minority Over-Sampling Technique)

Omar et al. [10] applies SMOTE to balance an Arabic dataset for hate speech detection, improving model performance by addressing class imbalance in the dataset.

While SMOTE is effective for class balancing, it has limitations, especially with languages like Arabic that have complex linguistic structures and variations. Generating synthetic samples for Arabic text can introduce inaccuracies, as SMOTE does not account for the contextual and grammatical subtleties required in multilingual datasets. To mitigate these issues, we explored additional options,

including ADASYN, which prioritizes more challenging cases, potentially enhancing model robustness in the presence of nuanced Arabic content.

Our work builds upon these existing approaches by focusing on the specific challenge of detecting hate speech in mixed Arabic-English content, a context that has been largely overlooked in the literature. Traditional models either focus on monolingual text or struggle with the dynamic nature of multilingual content due to their reliance on language-specific features [15]. Our proposed Bi-LSTM model diverges from these approaches by capturing the sequential dependencies between mixed-language text, making it more suitable for handling code-switching, which is prevalent in social media posts from the MENA region. Furthermore, our use of SMOTE to balance the dataset addresses a key issue that many previous studies have neglected-the imbalance in language representation, particularly between Arabic and English. By ensuring a more balanced dataset, our model is able to generalize more effectively, leading to improved detection performance across both languages. This innovation positions our work as a significant contribution to the field of multilingual hate speech detection, particularly for real-world applications where the ability to process mixed-language content is critical. Table 1 provides a comprehensive overview of the main models used in hate speech detection, highlighting the pros and cons of each. The table illustrates how different models, including traditional machine learning approaches, deep learning techniques, and ensemble methods, perform in monolingual and multilingual contexts, emphasizing the strengths and limitations of each approach in handling complex linguistic data, such as mixed-language content.

Challenges

Informal Nature of Posts and Comments

Unlike professional newspaper articles, textbooks, research papers, and other formal writings. The informal nature of posts, tweets, blogs and comments present some unique challenges for standardizing these inputs to any machine learning or deep learning model [11].

This is simply due to the fact that informal text content tend to have more spelling mistakes, grammatical errors, and unnecessary punctuation marks.

All of these things had to be taken care of in the data pre-processing stage of the machine learning cycle to avoid incorrect word embeddings and to be able to produce a proper numerical vector representation of any text input to the proposed models.

Data Imbalance—Class Outputs

One of the biggest challenges encountered while working on this problem set is that the data was imbalanced towards hate speech content. Which if left as is would have biased the model towards always predicting hate speech regardless of the input. Rendering any Machine learning or deep learning model ineffective for the task. This was compensated for using SMOTE as discussed in the "Methodology" section.

Table 1 Comparison of hate speech detection models

Model/approach	Pros	Cons
Traditional machine learning (e.g., SVM, KNN, Naive Bayes)	Simple and fast to train Good performance on monolingual datasets	Struggles with mixed-language content Limited context understanding Requires extensive feature engineering
Random forest and decision trees	Can handle non-linear data Robust and interpretable High accuracy in some monolingual tasks	Computationally expensive for large datasets Less effective in multilingual or cross-lingual settings
Bi-directional long short-term memory (Bi-LSTM)	Captures sequential context Effective for mixed-language content Handles long dependencies between words	Computationally expensive Requires large datasets for optimal performance
Convolutional neural networks (CNN)	Performs well with text classification High accuracy in detecting monolingual hate speech	Struggles with context and sequential dependencies Limited in capturing cross-lingual relationships
BERT-based approaches	State-of-the-art performance in NLP tasks Pre-trained on diverse datasets Effective in handling cross-lingual data	Requires significant computational resources Fine-tuning for specific tasks is challenging
Ensemble models	Combines strengths of multiple models Reduces variance and improves accuracy	Computationally expensive Complex to implement and interpret
SMOTE (synthetic minority over-sampling technique)	Balances imbalanced datasets Improves performance in minority class detection	Can generate noisy data Less effective when classes are highly overlapping

Data Imbalance—Arabic to English Inputs

There was also imbalance in the ratio of Arabic words to English words in the dataset. There is a large variety of English datasets of various sizes available for use since most of the world uses English professionally or informally. However, there were less Arabic resources and datasets available we performed SMOTE after mixing together the Arabic and English dataset in hopes that it would fix this imbalance as a subsequent of evening out the class labels.

Linguistic Challenges in Mixed-Language Content

Working with mixed-language content like Arabic and English introduces specific linguistic challenges, including code-switching and dialectal variations. Arabic dialects vary significantly across regions (e.g., Egyptian, Gulf, Levantine), and these differences impact both text preprocessing and model training. To address these complexities, our preprocessing pipeline includes customized tokenization and stemming techniques tailored to each dialect, enhancing the model's ability to handle language-specific nuances. By incorporating these tailored techniques, we aim to improve the model's capacity to accurately process and represent mixed-language content and dialectal variations, ultimately enhancing detection accuracy in multilingual contexts.

Methodology

The proposed machine learning life cycle is broken down into these steps as shown in the Fig. 1:

Algorithm 1 Hate Speech Detection Using Deep Learning

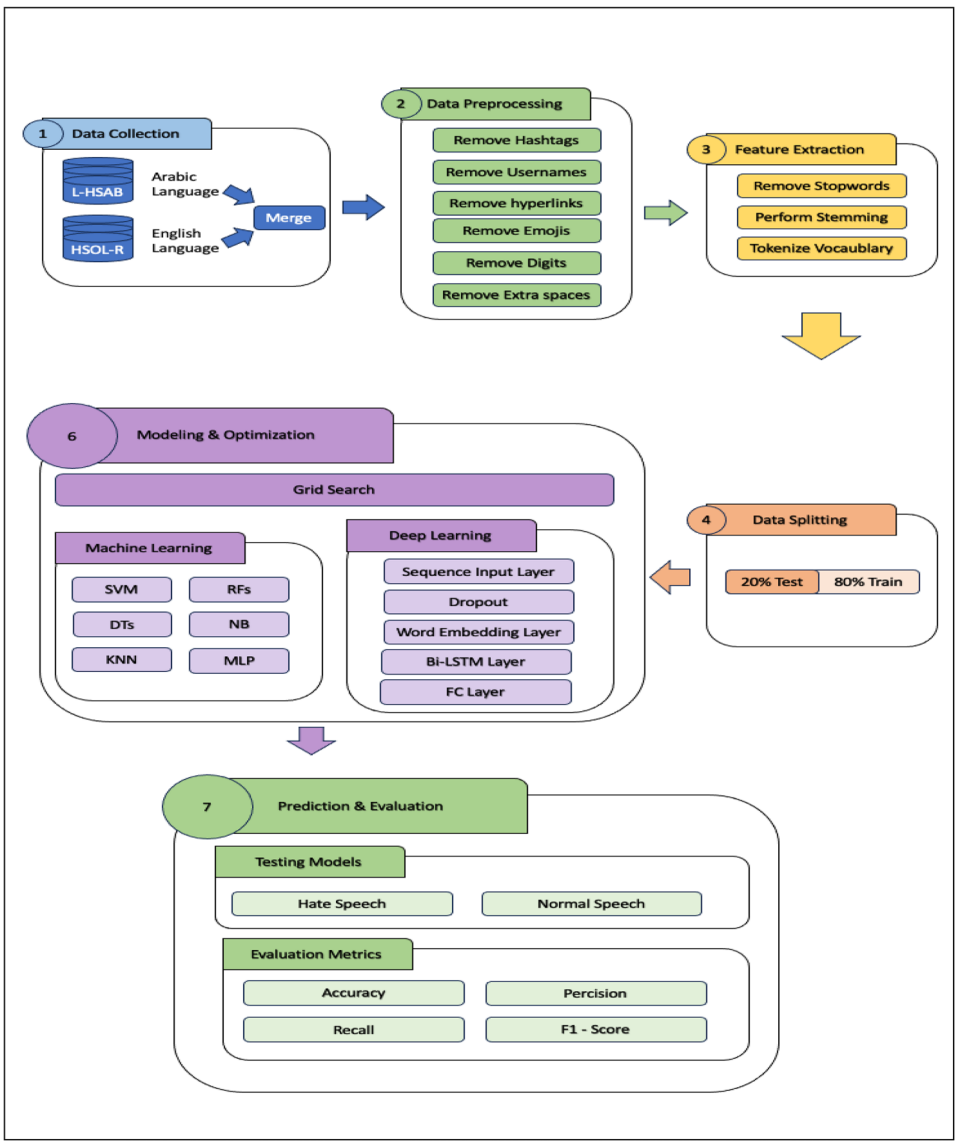


Fig. 1 The proposed lifecycle of hate speech detection

Input: Arabic and English text datasets
Output: Hate speech classification (Hate or Non-Hate)
procedure HATESPEECHDETECTION
    Step 1: Data Collection
    Collect datasets in Arabic and English.
    Use datasets such as L-HSAB for Arabic and a relevant English dataset.
    Apply SMOTE to balance the dataset if needed.
    Step 2: Data Preprocessing
    Remove hashtags, usernames, emojis, digits, and extra spaces.
    Apply stemming to convert words to their base forms.
    Tokenize the text to convert words into numerical vectors.
    Step 3: Feature Extraction
    For traditional machine learning models, use TF-IDF for feature extraction.
    For deep learning models, use word embeddings to represent the text data.
    Step 4: Data Splitting
    Split the data into training and testing sets using a suitable ratio.
    Step 5: Model Training
    Machine Learning Models:
    - Train models such as K-Nearest Neighbors (KNN), Naive Bayes (NB), Decision Trees (DT), Random Forest (RF), Support Vector Machines (SVMs), and Multilayer Perceptron (MLP).
    - Perform hyperparameter tuning using grid search or other techniques.
    Deep Learning Model:
    - Use a Bi-directional Long-Short Term Memory (Bi-LSTM) model.
    - Fine-tune the Bi-LSTM model's hyperparameters, such as the number of LSTM units and dropout rate.
    Step 6: Prediction and Evaluation
    Evaluate model performance using metrics like Accuracy, Precision, Recall, and F1-score.
    Compare the performance of all models on the test dataset.
    Consider the trade-offs between precision, recall, and F1-score when selecting the best model.
end procedure

- Data Collection
- Data Preprocessing
- Feature Extraction
- Data Splitting
- Modeling and Hyperparameter Tuning
- Prediction and Evaluation Metrics.

The steps performed for each life cycle item are outlined within this section.

Data Collection

Our study drew upon the insights offered by two distinct datasets, each tailored to address the intricacies of hate speech detection in different linguistic domains: Arabic and English. The first dataset, known as The Levantine Hate Speech and Abusive (L-HSAB), was meticulously curated to capture the nuances of hate speech in Arabic. Comprising a comprehensive collection of 5,846 records, L-HSAB provides a valuable resource for training and evaluating models in the context of the Arabic language.

The second dataset, sourced from the mrmorj/hate-speech-and-offensive-language repository, is an English-language dataset boasting a substantial 24,783 records. However, a notable challenge surfaced during our analysis-the dataset exhibited an imbalanced distribution between the interest classes. To address this imbalance, we employed the Synthetic Minority Over-sampling Technique (SMOTE), a well-established technique in machine learning, to augment the minority class instances. This strategic oversampling ensured a more balanced representation of hate and

non-hate speech instances within the English dataset, fostering improved model performance and generalization.

Recognizing online discourse's inherent diversity, we merged these two datasets. This amalgamation aimed at creating a more robust and comprehensive training corpus, incorporating the distinctive linguistic characteristics of both Arabic and English. The unified dataset thus encompassed 30,629 records, combining the rich linguistic nuances present in L-HSAB with the augmented representation of the English dataset after SMOTE. This approach not only broadened the scope of our study but also facilitated the development of models capable of effectively discerning hate speech in a multilingual context.

The decision to merge datasets was grounded in the belief that a diverse training set would empower our models to navigate the intricacies of hate speech detection across languages. This approach not only underscores the universality of the problem but also aligns with the real-world challenge of addressing hate speech in the global, multilingual online landscape. The subsequent sections detail the modeling techniques and hyperparameter tuning applied to leverage this comprehensive dataset for hate speech detection in both Arabic and English.

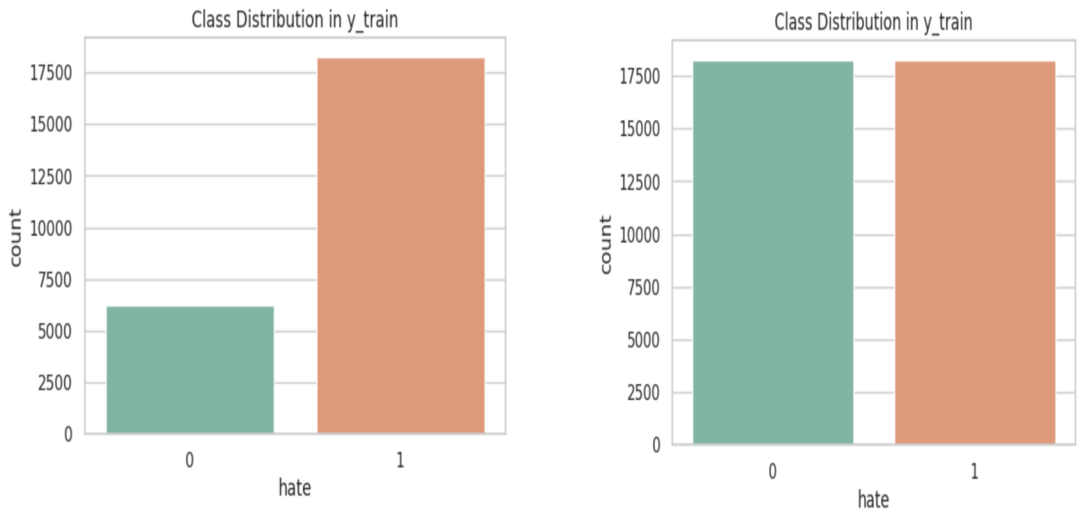
Data Quality and Language Imbalance

Arabic Dataset Quality and Origin

Our Arabic dataset, L-HSAB, was curated to reflect authentic and diverse forms of hate speech within the Arabic-speaking community, comprising both formal and informal language. We verified dataset integrity by sampling and manual annotations, ensuring relevance to common expressions of hate speech in MENA-based social media. The English dataset was sourced from the mrmorj repository, which is widely used in hate speech detection research.

Impact of Imbalance on Model Performance

The imbalance between Arabic and English datasets presented a challenge, as models trained on a higher volume of English data may demonstrate a bias towards English-language patterns. This was addressed through SMOTE applied across both language subsets, improving the models' ability to generalize. However, we acknowledge that further refinement, such as targeted over-sampling of unique Arabic linguistic patterns, may benefit future iterations. Arabic Dataset Quality and Origin: Our Arabic dataset, L-HSAB, was curated to reflect authentic and diverse forms of hate speech within the Arabic-speaking community, comprising both formal and informal language. We verified dataset integrity by sampling and manual annotations, ensuring relevance to common expressions of hate speech in MENA-based social



(a) Frequency of each Class Before applying SMOTE

(b) Frequency of each Class After applying SMOTE

Fig. 2 Comparison of class frequency before and after applying SMOTE

media. The English dataset was sourced from the mrmorj repository, which is widely used in hate speech detection research.

Data Preprocessing and Feature Extraction

Since the input data comes mainly from Twitter, it includes hashtags, usernames, and emojis that will confuse the models. One of the first steps is to remove these hashtags, usernames, and emojis from the input data, as well as digits and extra spaces before or after a word. This is done to ensure that what is left is a set of valid Arabic or English words that can be made sense of later.

Grammar is beneficial to us to gain context and understanding of an action's time and actors. However, for the basic need of sentiment analysis [16], the fact that the same word can have multiple forms that all imply the same meaning complicates things. The words "stopping", "stopped", "stop" all mean the same thing, and on their own are all normal (not hateful) words.

We applied stemming to the words to push them back to their "tense-free" forms. Such that "stopping" and "stopped" become "stop". This simplifies the tokenization process (turning the words to numbers), and maintains the context about the word that the model needs to have.

Then, the words are turned into tokens based on the dataset's vocabulary. for instance, the word trash becomes the number 14. This is necessary as no single machine learning or deep learning algorithm works on strings as input data, and they always expect numbers.

We then had the problem that the data was imbalanced, as shown in Fig. 2a. We then used the Synthetic Minority Oversampling Technique (SMOTE) to try and compensate for the lack of normal text artificially before training the model. This caused the class frequency to almost be split equally between normal and hate after applying SMOTE, as shown in Fig. 2b.

Finally data was split into training and testing sets. The training sets will be used to train the machine learning models, while the testing set will validate and measure the different models' performance.

Model Selection and Justification

In this study, we selected three machine learning models-K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and Bi-directional Long Short-Term Memory (Bi-LSTM)-to evaluate their performance in hate speech detection for mixed Arabic-English content. Each model was chosen for its strengths in handling different aspects of the problem:

- **K-Nearest Neighbors (KNN)**: KNN was selected for its simplicity and effectiveness in detecting local patterns within the dataset. KNN is an instance-based learning algorithm that is well-suited for smaller, structured datasets, making it useful for identifying patterns in linguistic clusters. However, KNN can be sensitive to hyperparameter choices, particularly the number of neighbors, and may struggle with high-dimensional, complex data like mixed-language text.
- **Support Vector Machines (SVM)**: SVM was chosen for its ability to handle high-dimensional spaces and create decision boundaries that are effective in separating classes. It is particularly effective when the classes are not linearly separable, which is often the case in text classification tasks. SVM offers a balanced trade-off between precision and recall, which is important when dealing with the imbalanced nature of hate speech detection. However, while SVM showed competitive performance, it did not capture the sequential dependencies between words in mixed-language content as effectively as Bi-LSTM.
- **Bi-directional Long Short-Term Memory (Bi-LSTM)**: Bi-LSTM was prioritized due to its ability to capture long-range dependencies and contextual relationships within the text. This is especially important for hate speech detection in mixed-language contexts, where code-switching between Arabic and English can occur within the same sentence. The sequential nature of language in these contexts requires models like Bi-LSTM, which can consider both past and future context when making predictions. Bi-LSTM demonstrated superior performance in handling the complexities of multilingual content, including code-switching, and outperformed traditional machine learning models in terms of both accuracy and robustness.

Modeling and Hyperparameter Tuning

To optimize the hate speech detection models, an extensive exploration of models and hyperparameters was conducted. The process involved grid search, systematically tuning hyperparameters to identify optimal configurations. This approach thoroughly examines the model's performance across various parameter values, facilitating the selection of the most effective settings.

K Nearest Neighbour (KNN)

Description K Nearest Neighbors (KNN) stands as a prominent instance-based learning algorithm, renowned for its simplicity and effectiveness in classification tasks. The fundamental principle underlying KNN is to discern the class of

a given data point by examining the majority class among its k nearest neighbors within the feature space. In essence, this algorithm operates on the assumption that similar instances tend to share common characteristics, and thus, the classification of a data point can be informed by the prevalent class within its proximity.

The operational concept of KNN involves the measurement of distance between data points in the feature space. Commonly utilized distance metrics include Euclidean distance, Manhattan distance, or other suitable measures depending on the nature of the data. When a classification decision is required for a particular data point, KNN identifies its k closest neighbors based on the chosen distance metric. Subsequently, the algorithm assigns the class label to the data point by considering the majority class among these nearest neighbors. This proximity-driven approach allows KNN to adapt to the local structure of the data, making it particularly effective in scenarios where instances of similar classes tend to cluster together.

Mathematical Formulation

$$\hat{y} = \text{argmax}_{y_i}\left(\sum_{j=1}^{k} I(y_i = y_j)\right) \tag{1}$$

Hyperparameters Number of neighbors (k)

Naive Bayes (NB)

Description Naive Bayes, a probabilistic model rooted in Bayes' theorem, stands out for its simplicity and efficacy, especially in text classification tasks. This algorithm relies on the feature independence assumption, simplifying the complex task of modeling joint probability distributions by considering each feature as conditionally independent, given the class label. Despite its seemingly naive assumption, Naive Bayes often produces compelling results, particularly in scenarios where this approximation aligns well with the underlying data structure.

In text classification, Naive Bayes excels due to its adept handling of high-dimensional feature spaces inherent in textual data. Its efficiency in modeling the probability of a document belonging to a specific class is underscored by the algorithm's straightforward yet powerful mathematical formulation, applying Bayes' theorem. This formulation expresses the posterior probability of a class given an observation in terms of the prior probability and likelihood. Naive Bayes, with its pragmatic independence assumption, thus offers a versatile and enduring solution for text classification, providing efficiency and interpretability in various applications.

Mathematical Formulation

$$P(y|x_1, x_2, ..., x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1, x_2, ..., x_n)} \tag{2}$$

Decision Trees (DT)

Description: Decision Trees recursively split the dataset based on feature conditions, creating a tree structure with decision rules.

Mathematical Formulation: The decision tree algorithm involves recursively partitioning the data based on feature conditions to maximize information gain or minimize impurity.

Hyperparameters: Maximum depth, minimum samples split

Random Forest (RF)

Description: Random Forest is an ensemble of decision trees. It improves robustness and accuracy by aggregating predictions from multiple trees.

Mathematical Formulation:

$$\hat{y}_{RF} = \frac{1}{T}\sum_{t=1}^{T} \hat{y}_t \tag{3}$$

where $\hat{y}_{RF}$ is the random forest prediction, T is the number of trees, and $\hat{y}_t$ is the prediction of the t^{th} tree.

Hyperparameters: Number of trees, maximum depth of trees

Support Vector Machines (SVMs)

Description: SVMs find the hyperplane that maximizes the margin between classes in the feature space, thus separating data points into different classes.

Mathematical Formulation:

$$\text{minimize}\left(\frac{1}{2}||w||^2 + C\sum_{i=1}^{N} \xi_i\right) \tag{4}$$

subject to $y_i(w \cdot x_i - b) \geq 1 - \xi_i$ for all i,

where w is the weight vector, b is the bias term, ξ_i are slack variables, and C is the regularization parameter.

Hyperparameters: Regularization parameter (C), kernel type

Multilayer Perceptron (MLP)

Description: MLP is a neural network model with multiple layers of interconnected nodes, capable of capturing complex non-linear relationships within the data.

Mathematical Formulation:

$$a_j^{(l)} = \sigma\left(\sum_{i=1}^{n^{(l-1)}} w_{ij}^{(l)} a_i^{(l-1)} + b_j^{(l)}\right) \tag{5}$$

where $a_j^{(l)}$ is the activation of the jth neuron in layer l, $w_{ij}^{(l)}$ is the weight from neuron i in layer $l-1$ to neuron j in layer l, $b_j^{(l)}$ is the bias of neuron j in layer l, and σ is the activation function.

Hyperparameters: Number of layers, number of neurons per layer, learning rate

Bi-directional Long-Short Term Memory (Bi-LSTM)

Description: Bi-LSTM is a deep learning model designed for sequential data, considering both past and future context when making predictions.

Mathematical Formulation:

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \tag{6}$$

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}) \tag{7}$$

$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}) \tag{8}$$

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \tag{9}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{10}$$

$$h_t = o_t \odot \tanh(c_t) \tag{11}$$

where x_t is the input at time t, h_t is the hidden state at time t, i_t,f_t, g_t, o_t are the input, forget, cell, and output gates, $\odot$ denotes element-wise multiplication, W and b are weight and bias parameters, and σ is the sigmoid activation function.

Hyperparameters: Number of LSTM units, dropout rate These models, each with its distinctive characteristics and mathematical foundations, were subjected to hyperparameter tuning through grid search, enabling the identification of optimal configurations for hate speech detection in both Arabic and English text.

Methodology Justification

Choice of Models

We selected a range of models-K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and Bi-directional Long Short-Term Memory (Bi-LSTM)-each catering to different facets of the multilingual hate speech detection task. KNN was chosen due to its simplicity and effectiveness in capturing local class patterns, particularly beneficial for smaller, similar linguistic clusters within the dataset. SVM was utilized for its capability to separate classes with a clear hyperplane, which is effective when dealing with high-dimensional data. Bi-LSTM was prioritized due to its strength in sequential data, which aligns with the linguistic dependencies and context-specific features necessary for detecting nuanced hate speech in mixed-language contexts, especially where Arabic and English co-exist in code-switching scenarios [17].

Rationale for SMOTE over Other Techniques

We opted for the Synthetic Minority Over-sampling Technique (SMOTE) to address data imbalance because it enhances the minority class by generating synthetic samples rather than duplicating data points, as seen in techniques like Random Oversampling. SMOTE is particularly effective in NLP tasks where generating similar context-aware sentences is crucial for model generalization. We considered alternatives such as ADASYN but selected SMOTE as it maintains linguistic consistency without introducing significant noise, which could degrade model performance.

Prediction and Evaluation Metrics

In evaluating the performance of our hate speech detection models, we rely on several key metrics, each offering unique insights into the models' effectiveness in

Table 2 Bi-LSTM model performance metrics

	Precision	Recall	F1-Score	Support
0	0.83	0.89	0.86	4554
1	0.89	0.81	0.85	4554
Accuracy			0.85	9108
Macro Avg	0.86	0.85	0.85	9108
Weighted Avg	0.86	0.85	0.85	9108

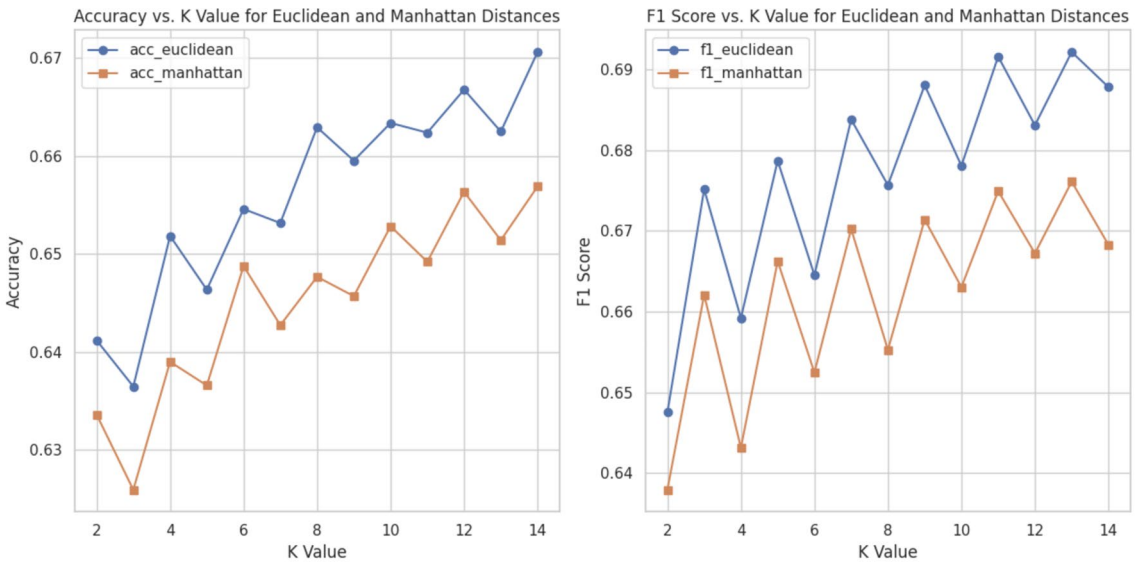


Fig. 3 Accuracy and F1-Score—KNN

Table 3 KNN model performance metrics (Euclidean)

Accuracy	F1	Recall	Precision
0.641	0.648	0.659	0.636

Accuracy	F1	Recall	Precision
0.591	0.55	0.59	0.64

Fig. 4 Naive Bayes model performance metrics

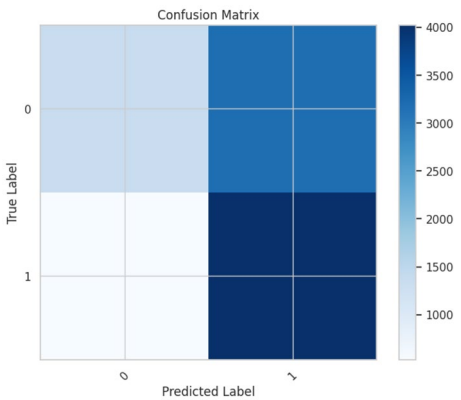


Fig. 5 Confusion matrix—Naive Bayes

distinguishing between hate speech and non-hate speech instances. These metrics are:

- **Accuracy:** The ratio of correctly predicted instances to the total number of instances. It provides an overall measure of model correctness and is calculated as:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Instances}} \quad (12)$$

- **Precision:** The proportion of true positive predictions among all instances predicted as positive. Precision focuses on the accuracy of positive predictions and is calculated as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (13)$$

- **Recall:** The proportion of true positive predictions among all actual positive instances. Recall gauges the

Best Hyperparameters:

Criterion:	gini
Max Depth:	10
Max Features:	None
Min Samples Leaf:	4
Min Samples Split:	2

Accuracy	F1	Recall	Precision
0.819	0.82	0.82	0.82

Fig. 6 Decision trees model performance metrics

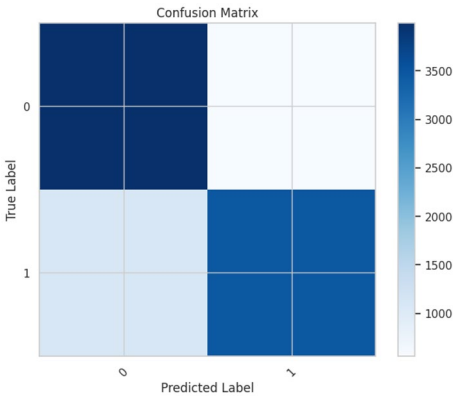


Fig. 7 Confusion matrix—decision trees

Best Hyperparameters:

Criterion:	entropy
Max Depth:	None
Max Features:	sqrt
Min Samples Split:	2

Accuracy	F1	Recall	Precision
0.810	0.81	0.81	0.81

Fig. 8 Random forest model performance metrics

model's ability to identify all relevant instances and is calculated as:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{14}$$

- **F1-Score:** The harmonic mean of precision and recall, providing a balanced measure of a model's performance. It is calculated as:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{15}$$

- **Support:** The number of actual occurrences of each class in the specified dataset. It gives context to the precision and recall values.
- **Macro Average:** The average of precision, recall, and F1-score calculated independently for each class. It treats all classes equally, regardless of class imbalance.
- **Micro Average:** The aggregate metric calculated by summing up individual true positives, false positives, and false negatives across all classes. It provides an overall measure that considers class imbalances.

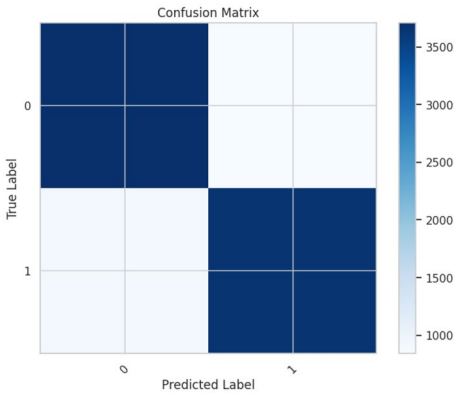


Fig. 9 Confusion matrix—random forest

Best Hyperparameters:

C: 1 Gamma: auto Kernel: rbf

Accuracy	F1	Recall	Precision
0.654	0.64	0.65	0.69

Fig. 10 SVM model performance metrics

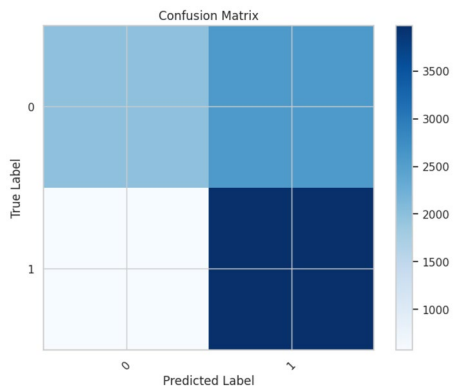


Fig. 11 Confusion matrix—SVM

These metrics collectively offer a comprehensive evaluation of our models' performance in hate speech detection, considering both overall accuracy and the nuanced aspects of precision, recall, and F1-score for each class.

Results

Bi-directional Long-Short Term Memory (Bi-LSTM)

See Table 2.

K Nearest Neighbors (KNN)

See Fig. 3.

Results for n_neighbors = 2, distance metric = Euclidean:

See Table 3.

Naive Bayes

See Figs. 4 and 5.

Decision Trees (DTs)

Best Hyperparameters:

Criterion:	gini
Max depth:	10
Max features:	None
Min samples leaf:	4
Min samples split:	2

See Figs. 6 and 7.

Random Forest

Best Hyperparameters:

Criterion:	Entropy
Max depth:	None
Max features:	sqrt
Min samples split:	2

See Figs. 8 and 9.

Best Hyperparameters:

Activation: logistic
Gamma: 0.0001
Kernel: 100
Learning Rate: constant
Max Iterations: 50

Accuracy	F1	Recall	Precision
0.617	0.62	0.62	0.62

Fig. 12 MLP model performance metrics

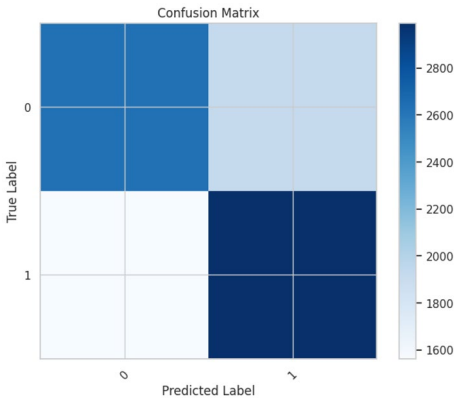


Fig. 13 Confusion matrix—multilayer perceptron

Support Vector Machines (SVMs)

Best Hyperparameters:

C:	1	Gamma: auto	Kernel: rbf

See Figs. 10 and 11

Multilayer Perceptron (MLP)

Best Hyperparameters:

Activation:	Logistic
Gamma:	0.0001
Kernel:	100
Learning rate:	Constant
Max iterations:	50

See Figs. 12 and 13.

Table 4 Overall Model Performance Comparison

Model	Accuracy	Precision	Recall	F1-Score
Bi-LSTM	0.85	0.86	0.85	0.85
KNN	0.641	0.636	0.659	0.648
Naive Bayes	0.591	0.64	0.59	0.55
Decision trees	0.819	0.82	0.82	0.82
Random forest	0.810	0.81	0.81	0.81
SVMs	0.654	0.69	0.65	0.64

Discussion: The Bi-LSTM model demonstrated robust performance with high precision, recall, and F1-score for both classes. KNN's performance varied with different hyperparameter configurations, showing the sensitivity of the model to these choices. Naive Bayes exhibited lower accuracy and precision for class 0 but achieved higher recall and F1-score for class 1. Decision Trees and Random Forest models showed competitive performance, with Random Forest slightly outperforming Decision Trees in terms of accuracy. SVMs exhibited a balanced trade-off between precision and recall, achieving a reasonable F1-score.

Error Implications in Real-World Applications: **False Positives (Classifying Non-Hate Speech as Hate Speech):** False positives could lead to unnecessary censorship, potentially infringing on users' freedom of expression. This is particularly sensitive in multilingual contexts where regional slang or culturally specific language may be misinterpreted as offensive. A high rate of false positives could result in user frustration, mistrust in platform moderation policies, and the unintentional silencing of minority voices. For this reason, precision is especially crucial in hate speech detection to avoid undue censorship of benign content. **False Negatives (Failing to Detect Actual Hate Speech):** False negatives, or failing to flag actual hate speech, can lead to the unmoderated spread of harmful content, posing psychological harm and contributing to community division. On social media platforms, this could allow the proliferation of abusive or violent rhetoric, risking real-world harm and eroding user safety. Given the risks associated with undetected hate speech, a high recall rate is vital to ensure that harmful content is minimized effectively. Achieving a balance between precision and recall is critical, and the ideal balance may vary depending on the specific application. In high-risk environments (e.g., platforms serving vulnerable populations), prioritizing recall may be more appropriate, while platforms focusing on open discourse may opt to emphasize precision.

Comparison

In this section, we compare the performance of the different hate speech detection models based on the evaluation metrics discussed in section above. The models considered for comparison are Bi-directional Long-Short Term Memory (Bi-LSTM), K Nearest Neighbors (KNN), Naive Bayes, Decision Trees (DTs), Random Forest, and Support Vector Machines (SVMs).

Overall Performance

Table 4 provides an overview of the overall performance metrics for each model.

Discussion

From the results, we observe that the Bi-LSTM model achieved the highest overall accuracy, precision, recall, and F1-score. This indicates its effectiveness in capturing complex relationships within the data, especially in the context of hate speech detection.

Decision Trees and Random Forest also performed well, with high accuracy and balanced precision and recall. The ensemble nature of Random Forest provided a slight improvement over individual Decision Trees.

KNN showed sensitivity to hyperparameter configurations, leading to variable performance. Naive Bayes exhibited lower accuracy but higher recall for class 1, indicating its capability to capture instances of hate speech, though at the cost of more false positives.

SVMs demonstrated a balanced trade-off between precision and recall, achieving a reasonable F1-score.

Consideration of Use Case

The choice of the most suitable model depends on the specific requirements of the hate speech detection application. If high accuracy and balanced precision and recall are crucial, the Bi-LSTM model stands out. Decision Trees and Random Forest offer a good balance between accuracy and interpretability. Naive Bayes may be considered if minimizing false negatives is a priority.

It's essential to consider the trade-offs between different metrics based on the consequences of false positives and false negatives in the application domain.

Conclusion and Future Work

Conclusion

In conclusion, this study demonstrates the effectiveness of combining traditional and deep learning models for multilingual hate speech detection, especially in contexts involving code-switching between Arabic and English. By addressing the challenges of language imbalance and dataset quality, our approach represents a significant advancement in real-world hate speech detection applications.

Future Work

There are several areas for future research that could enhance the effectiveness and applicability of multilingual hate speech detection models:

1. Improving Dataset Quality and Diversity: One limitation in the current study is the imbalance and limited size of Arabic hate speech datasets compared to English. Future

work could focus on expanding the Arabic dataset by collecting data from additional sources, such as Facebook and Instagram, and incorporating a broader range of dialects, including North African, Gulf, and Egyptian dialects, to better represent the diversity within Arabic-speaking communities. Additionally, a balanced dataset across multiple languages would allow models to generalize better in mixed-language environments.

2. Addressing Sarcasm and Implicit Hate Speech: Sarcasm and implicit forms of hate speech are challenging for models to detect, especially in a multilingual context. In future work, advanced labeling processes that capture nuanced language, such as sarcasm, irony, and implicit hate, could be implemented. Data annotation could involve linguists or cultural experts to ensure that subtle linguistic markers of sarcasm and implicit bias are effectively labeled. Furthermore, training models with labeled sarcastic instances could improve the accuracy of detecting covert hate speech.

3. Exploring Alternative Model Architectures: While Bi-LSTM showed strong performance in this study, alternative architectures may further improve accuracy. Future research could explore the application of Transformer-based models, such as BERT and multilingual BERT (mBERT), which are well-suited for NLP tasks and may better capture the complexity of code-switched language. Transformer models could potentially improve contextual understanding and adapt well to mixed-language data due to their attention mechanisms. Additionally, ensemble learning approaches combining various architectures (e.g., CNN-BiLSTM and Transformer ensembles) could be examined to determine if performance gains can be achieved by leveraging the strengths of multiple architectures.

4. Dynamic Data Augmentation Strategies: Future studies could incorporate dynamic data augmentation techniques beyond SMOTE to enhance data diversity. Techniques such as back-translation (translating text to another language and back again) or language-specific augmentations could improve model robustness, particularly when dealing with limited datasets for minority languages. By addressing these future research directions, the field of multilingual hate speech detection could move closer to real-world applications that are robust, culturally sensitive, and adaptable to diverse language inputs. We hope this study lays the groundwork for further advancements in this area.

Author Contributions All authors contributed equally to the research and manuscript preparation.

Funding Not applicable.

Data Availability The data that support the findings of this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest The authors have declared that there is no conflict of interest. Non-financial conflict of interest.

Informed Consent Not applicable.

Research Involving Human and/or Animals Not applicable.

References

1. Al-Hassan A, Al-Dossari H. Detection of hate speech in social networks: a survey on multilingual corpus. In: 6th International Conference on Computer Science and Information Technology, vol. 10; 2019;. p. 10–5121.
2. Khabirova E. Integration of English loanwords in Arab online communication; 2020. p. 1583–9. https://doi.org/10.15405/epsbs.2020.08.183.
3. Mullah NS, Zainon WMNW. Advances in ml algorithms for hate speech detection in sm: a review. J Adv Mach Learn. 2021;9:88367–72.
4. Tahosin F, Shill PC, Alam MGR. Multi-modal hate speech detection using machine learning. Preprint 2023. Published on ResearchGate.
5. Rana A, Jha S. Emotion based hate speech detection using multimodal learning. 2022. arXiv preprint arXiv:2202.06218.
6. Badjatiya P, Gupta S, Gupta M, Varma V. Deep learning for hate speech detection in tweets. In: WWW 2017 Companion; ACM. 2017. p. 29–30.
7. Del Vigna F, Cimino A, Dell'Orletta F, Petrocchi M, Tesconi M. Hate me, hate me not: hate speech detection on Facebook. In: Proceedings of the First Italian Conference on Cybersecurity (ITASEC17) 2017. CEUR Workshop Proceedings.
8. Faris H, Aljarah I, Habib M, Castillo PA. Hate speech detection using word embedding and deep learning in the Arabic language context. In: Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2020). SCITEPRESS - Science and Technology Publications, Lda; 2022. p. 453–60.
9. Nobata C, Tetreault J, Thomas A, Mehdad Y, Chang Y. Abusive language detection in online user content. In: Proceedings of the 25th International Conference on World Wide Web. ACM. International World Wide Web Conference Committee (IW3C2); 2016. https://doi.org/10.1145/2872427.2883062.
10. Omar A, Mahmoud TM, Abd-El-Hafeez T. Comparative performance of machine learning and deep learning algorithms for Arabic hate speech detection in osns. In: Proceedings of the International Conference on Artificial Intelligence and Computer Vision. Springer Nature; 2020. p. 247–57. https://doi.org/10.1007/978-3-030-44289-7_24.
11. Alshalan R, Al-Khalifa H. A deep learning approach for automatic hate speech detection in the Saudi Twittersphere. Appl Sci. 2020;10(23):8614. https://doi.org/10.3390/app10238614.
12. MacAvaney S, Yao H-R, Yang E, Russell K, Goharian N, Frieder O. Hate speech detection: challenges and solutions. PLoS ONE. 2019;14(8):0221152.
13. Zimmerman S, Fox C, Kruschwitz U. Improving hate speech detection with deep learning ensembles. J Mach Learn Res. 2021;22(14):1–18.
14. Zimmerman S, Kruschwitz U, Fox C. Improving hate speech detection with deep learning ensembles. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018); 2018.

15. Haapoja J, Laaksonen S-M, Lampinen A. Gaming algorithmic hate-speech detection: stakes, parties, and moves. Soc Media Soc. 2020;6(2):2056305120924778.
16. Aref A, Al Mahmoud RH, Taha K, Al-Sharif M. Hate speech detection of Arabic short text. In: Proceedings of the International Conference on Information Technology, Computer Science and Applications (ITCSE) and its associated workshops: Natural Language Computing and Applications (NLCA), Information and Computing Applications in Industry and Engineering (ICAIT), Computer Applications in Image and Multimedia Processing (CAIML), Image Processing and Pattern Recognition (ICDIPV), Cryptography and Information Security (CRYPIS), Wireless and Mobile Communication (WiMo); . Computer Science & Information Technology (CS & IT); 2020. p. 81–94. https://doi.org/10.5121/csit.2020.100507.
17. Waseem Z, Hovy D. Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter. In: Proceedings of the NAACL Student Research Workshop; 2016. p. 88–93.