scientific reports

OPEN

Roman urdu hate speech detection using hybrid machine learning models and hyperparameter optimization

Waqar Ashiq[1], Samra Kanwal[2], Adnan Rafique[3], Muhammad Waqas[4], Tahir Khurshaid[5]✉, Elizabeth Caro Montero[6,7,8], Alicia Bustamante Alonso[6,9,10] & Imran Ashraf[11]✉

With the rapid increase of users over social media, cyberbullying, and hate speech problems have arisen over the past years. Automatic hate speech detection (HSD) from text is an emerging research problem in natural language processing (NLP). Researchers developed various approaches to solve the automatic hate speech detection problem using different corpora in various languages, however, research on the Urdu language is rather scarce. This study aims to address the HSD task on Twitter using Roman Urdu text. The contribution of this research is the development of a hybrid model for Roman Urdu HSD, which has not been previously explored. The novel hybrid model integrates deep learning (DL) and transformer models for automatic feature extraction, combined with machine learning algorithms (MLAs) for classification. To further enhance model performance, we employ several hyperparameter optimization (HPO) techniques, including Grid Search (GS), Randomized Search (RS), and Bayesian Optimization with Gaussian Processes (BOGP). Evaluation is carried out on two publicly available benchmarks Roman Urdu corpora comprising HS-RU-20 corpus and RUHSOLD hate speech corpus. Results demonstrate that the Multilingual BERT (MBERT) feature learner, paired with a Support Vector Machine (SVM) classifier and optimized using RS, achieves state-of-the-art performance. On the HS-RU-20 corpus, this model attained an accuracy of 0.93 and an F1 score of 0.95 for the Neutral-Hostile classification task, and an accuracy of 0.89 with an F1 score of 0.88 for the Hate Speech-Offensive task. On the RUHSOLD corpus, the same model achieved an accuracy of 0.95 and an F1 score of 0.94 for the Coarse-grained task, alongside an accuracy of 0.87 and an F1 score of 0.84 for the Fine-grained task. These results demonstrate the effectiveness of our hybrid approach for Roman Urdu hate speech detection.

Keywords Hate speech detection, Deep learning, Model optimization, Urdu text classification

During the past decade, the usage of social media platforms like Twitter, Instagram, Facebook, etc. became very popular. People post or comment on social media posts and express their views. The difference in opinion on social media platforms can change into a debate and harsh words are exchanged sometimes. Negative, disrespectful, and abusive comments violating human dignity are all categorized as hate speech and an increasing trend has been observed recently. Due to the ease of accessibility and widespread use of the Internet, computers, tablets, and cell phones, cyberbullying may occur at any time and anywhere, making it a severe issue.

Social media platforms and websites serve as a hub of communication for individuals all around the world. People who are geographically, religiously, racially, and culturally separated from one another (such as the

[1]Department of Software Engineering, University of Management and Technology, Lahore 54590, Pakistan. [2]Department of Computer Science, University of Management and Technology, Lahore 54590, Pakistan. [3]School of Information and Communications Technology, University of Tasmania, Launceston 7250, Australia. [4]Department of Mathematics, University of Education, Vehari 61100, Pakistan. [5]Department of Electrical Engineering, Yeungnam University, Gyeongsan 38541, Republic of Korea. [6]Universidad Europea del Atlantico., Isabel Torres 21, Santander 39011, Spain. [7]Universidad Internacional Iberoamericana Arecibo, Puerto Rico 00613, USA. [8]Universidade Internacional do Cuanza, Cuito, Bie, Angola. [9]Universidad Internacional Iberoamericana, Campeche 24560, Mexico. [10]Universidad de La Romana, La Romana, Dominican Republic. [11]Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, Republic of Korea. ✉email: tahir@ynu.ac.kr; imranashraf@ynu.ac.kr

separation of the Indian Sub-continent into India and Pakistan) frequently engage in verbal violence against one another[1]. It is defined as "abusive communication targeting specific group traits such as culture, religion, or gender"[2]. When writing reviews or comments on online products, videos, or articles, users typically prefer and feel more comfortable using their native language than English[3]. Abusive language in posts or comments should not be visible to other users since it encourages cyberbullying. As a result, it is critical to develop an autonomous system for detecting, stopping, or prohibiting harmful language or hate speech before it is published online. According to[4] Twitter is one of the most popular social networking sites, with 330 million active users and 200 billion tweets per year. Although the large majority of these users express their emotions or share their views, others misuse their freedom by tweeting controversial content. Hate speech is a sort of content that has caused controversy.

With over 231.3 million speakers worldwide, Urdu is the 10th most spoken language[5]. It is the official language of Pakistan, a country of 220 million people[6]. Urdu is spoken in many countries other than Pakistan, including India, the United Kingdom, the United States, Canada, and the Middle East[7]. However, there are various difficulties in creating digital content in Urdu. For instance, the Urdu alphabets differ significantly from the English alphabet, and the English keyboard cannot be used to type Urdu. Furthermore, the Urdu language includes 36 alphabets, but English keyboards only support 26 alphabets. Because of the significant discrepancy in the number of alphabets, mapping the Urdu alphabet to an English keyboard is problematic. Because of these difficulties, most Urdu speakers communicate in a distinct script that employs English alphabets to write Urdu phrases, technically known as Roman Urdu[8].

In previous research, hate speech detection (HSD) has been solved as a text classification task under supervised learning. A range of approaches to detect hate speech from the text are developed for various languages such as English[9,10], Spanish[11–13], Bengali[14–16], Arabic[17,18], Indonesian[19–21], Urdu[22]and Roman Urdu[1,8,23]. The developed approaches are broadly categorized into machine learning using manually extracted features, deep learning (DL), and transfer learning. The main focus of this study is on Roman Urdu text. Roman Urdu HSD task is solved mainly by using hand-crafted feature engineering methods such as count vectorizer, N-Gram vectorizer, word-level term frequency, character-level features, word embeddings[8], and word and character N-grams[1]. Traditional hand-crafted feature engineering is an eminently tedious, time-taking process and requires human ability, while automatic feature engineering extracts and learns features directly from raw text. Although promising results are reported by these methods, HSD is still an emerging research area and requires investigation of novel methods for Roman Urdu to enhance performance for Roman Urdu text classification.

HSD in Roman Urdu is uniquely challenging due to its non-standardized nature, where speakers use English alphabets to write Urdu words, resulting in inconsistencies in spelling and grammar. Previous methods predominantly relied on manual feature engineering, which is labor-intensive and less effective for handling the linguistic intricacies of Roman Urdu. This study introduces a novel hybrid approach that combines deep learning models and transformer-based architectures with traditional machine learning classifiers. The term "hybrid" refers to the integration of two distinct approaches. Specifically, convolutional neural networks (CNN), long short-term memory networks (LSTM), bidirectional LSTM (BiLSTM), gated recurrent units (GRU), and the multilingual BERT (MBERT) transformer are employed for automatic feature extraction, while machine learning (ML) classifiers such as support vector machines (SVM) are used for classification. The hybrid approach leverages the deep learning models' ability to capture complex patterns from the Roman Urdu text, while the machine learning classifiers optimize classification performance through structured data. In the hybrid approach, the deep learning models allow for automatic feature extraction from raw Roman Urdu text, capturing deep contextual and semantic information that traditional machine learning techniques would not be able to capture on their own. This justifies the use of deep learning and transformer for feature extraction, as it reduces the need for manual feature engineering and can better handle the complexities of Roman Urdu, which lacks standardized spelling and grammar. Additionally, ML models are effective for classification due to their interpretability, robustness, and ease of optimization using hyperparameter tuning.

Furthermore, to optimize the performance of ML models, we applied different hyperparameter optimization approaches comprising grid search (GS), randomized search (RS), and Bayesian optimization with Gaussian process (BOGP). This integration of automatic feature extraction and hyperparameter optimization techniques enhances the robustness of our approach to detecting hate speech in Roman Urdu, addressing the specific challenges posed by its linguistic properties. The novelty of our approach lies in the integration of deep learning models and transformer-based models (MBERT) for feature extraction, which reduces the need for manual feature engineering, combined with traditional machine learning models for classification and enhances the performance through hyperparameter tuning, which, to our knowledge, has not been explored in this domain. The proposed model is evaluated on two publicly available benchmarks: Roman Urdu corpora HS-RU-20 corpus[8]and RUHSOLD corpus[23]. The first dataset contains three classes including neutral, hostile, and hate speech offensive while the second dataset has coarse-grained levels (i.e. abusive, normal) and fine-grained levels (i.e. abusive, normal, religious hate, sexism, and profane).

A considerable range of experiments is performed to ratify the usefulness of the proposed hybrid model. On both corpora, complete experimentation is divided into two distinct phases. In the first phase, experiments are conducted using the proposed hybrid models with the default parameter settings of machine learning algorithms. In the second phase, we performed experiments with hyperparameter optimization methods to optimize the prediction performance. In the end, we compared the results using standard evaluation measures to check the effect of optimization and compared the results of the proposed methods with the baseline of both corpora. By addressing the limitations of traditional feature engineering and applying an innovative hybrid model, this study significantly advances hate speech detection in Roman Urdu. The key research questions guiding this study are as follows:

- RQ1: How can we effectively leverage deep learning and transformer models for automatic feature extraction in Roman Urdu hate speech detection, where traditional manual feature engineering may fall short?
- RQ2: How does the combination of automatic feature extraction with machine learning classifiers improve classification performance for hate speech detection in Roman Urdu text?
- RQ3: What is the impact of hyperparameter optimization techniques (GS, RS, BOGP) on enhancing the classification performance for both binary and multiclass hate speech detection tasks?The primary contributions of this study are given below:
-
- Developed a hybrid model that combines deep learning models and transformers as automatic feature learners with different MLAs for classification. This approach leverages the strengths of both deep learning and traditional machine learning techniques to improve hate speech detection in Roman Urdu text.
- Applied various HPO techniques, including GS, RS, and BOGP to optimize the performance of the MLAs in the hybrid model. HPO helps to fine-tune the models and improve their prediction performance which enhances the effectiveness of hate speech detection.
- Proposed hybrid model and HPO methods are evaluated on two publicly available benchmark hate speech corpora: HS-RU-20 corpus and RUHSOLD corpus. The experimental results demonstrate the effectiveness of the proposed approach and obtained state-of-the-art results using MBERT-SVM-RS combination on both corpora.The rest of the article is organized as follows. Section "Literature Review" demonstrates the detailed literature review of existing resources and approaches for hate speech detection tasks. The proposed hybrid model for both binary and multiclass hate speech detection tasks is presented in Sect. "Proposed Hybrid Model for Roman Urdu Hate Speech Detection". Experimental results are presented in Sect. "Dataset" . Finally, the conclusion of the study and some future directions are illustrated in Sect. "Data Preprocessing".

Literature review

In literature, multiple resources and approaches for hate speech detection are developed for different languages such as English[9,10,24], Spanish[11–13], Bengali[14–16], Arabic[17,18], Indonesian[19–21], Italian[25], Urdu[22]and Roman Urdu[1,8,23]. Mainly the hate speech detection problem is solved as a text classification under supervised learning. A comprehensive overview of hate speech detection corpora and approaches is given below.

Waseem et al[9]. developed the corpus of 16k labeled tweets. To evaluate the corpus they applied the character n-gram feature extraction approach with a logistic regression (LR) classifier for classification. Results indicate that the highest F1 score of 0.74 using 10-fold cross-validation is obtained. In another study, Malmasi et al[24]. used n-gram and skip-gram feature extraction approaches on the social media hate speech corpus comprising 14509 annotated instances[26]. The study obtained the highest accuracy of 78% using the SVM classifier.

Considerable efforts have been made to solve the hate speech detection problem using transfer learning methods in English. Rizoiu et al[27]. developed a transfer learning approach using BiLSTM on two hate speech corpora: corpus from[28]that discriminates between racist, sexist, and harmless comments and corpus from[26] which distinguished between hateful, offensive, and harmless comments. Experimental evaluation showed the highest micro F1 score of 78% and 72%, respectively on both corpora.

Later, Mehmood et al[29]. proposed passion-Net an attention-based deep learning evaluation model for the identification of hate speech in the Roman Urdu language. By integrating attention mechanisms, thus their model demonstrated an 8.7% increase in the F1 score and an 18.6% increase in recall for tasks involving fine-grained classification. This highlights the possibility of using attention mechanisms as a tool for improving models through the attention mechanism. Linguistic variables, which is especially effective in the case of complicated text classification problems including hate speech detection. Nasir et al[30]. discussed the work of developing an LR-based model for Roman Urdu yielding 81% accuracy in neutral to hostile classification and 87% in the identifying of offensive-hate speech. This result shows how well the classical ML-based models perform on the hate setting of tasks related to speech detection, in Roman Urdu as well. Maqbool et al[31]. worked towards implementing data augmentation techniques so as to reduce the vices of limited data in the identification of hate speech. By adding new variables to the RUHSOLD dataset as well as by using their model they have employed m-BERT for their model and got their model success rate of 91.3% accuracy. This work illustrates the performance of using pre-trained transformer models together with data augmentation techniques which is very essential especially when optimization is needed in environments characterized by limited availability of resources.

Some notable efforts have been made to address the Roman Urdu hate speech detection problem. Khan et al[8]. prepared an RU-HS-20 corpus comprising 5000 instances by collecting Roman Urdu tweets for hate speech detection. The authors divided the corpus into neutral-hostile and hate speech-offensive classes comprising 5000 and 3570 instances, respectively. To evaluate the corpus, the authors applied various feature extraction methods with different machine learning classifiers including Naive Bayes (NB), random forest (RF), LR, and SVM. LR classifier with count vectorizer achieves the best results with a 0.84 accuracy score. LR classifier also produced the highest F1 score of 0.90 and 0.75 for neutral-hostile and hate speech-offensive tasks, respectively.

Rizwan et al[23]. developed a benchmark hate speech and offensive language detection corpus that contains 10012 Roman Urdu tweets. The corpus is divided into two levels by authors including coarse-grained level and fine-grained level. The coarse-grained level consists of two classes normal and abusive. The fine-grained level comprises five classes including abusive, religious hate, sexism, and profane comprising 2402, 782, 839, and 640 instances, respectively. For text classification, several deep learning models are employed including LSTM, GBDT, FastText with CNN, Bi-LSTM with attention, etc. BERT with CNN-gram obtained the best results with 0.90 each for accuracy and F1 score at the coarse-grained level task and the highest accuracy and F1 score of 0.82 and 0.75, respectively, at the fine-grained level task.

Malik et al[32]. concentrated on identifying hate speeches and targeting the communities in the Nastaliq Urdu language. Their work used transformer-based models including Urdu-RoBERTa and Urdu-DistilBERT;

the results were 86.58% of hate speech identification accuracy and 84.17% accuracy in finding out target communities. The work provides another piece of evidence of the ability of transformer-based models to fine-tune the low-resource language understanding tasks. Cyberbullying detection was suggested by Atif et al[33]. based on a Roman Urdu model using GRU-based tweets. It also obtained a 0.97% accuracy and an F1 score of 0.97% of the test results illustrating the ability of GRU to capture sequential dependencies in textual data. Also, the authors carried out a user behavior analysis where the profiles were categorized as normal, suspicious, and abusive. Ullah et al[34]. proposed a DistilBERT-based system for hate speech detection in multilingual contexts but especially for low-resource languages such as Roman Urdu. Their model gave an F1-score of 0.6369, as they defined it as a measure of the model's performance on a code-mixed Telugu-English dataset. This work proves how light models' transformers work when applied to hate speech detection tasks, especially where computing power is limited. Jahangir et al[35]. used and compared the results of both the Support Vector Machines (SVM) and Logistic Regression in detecting cyberbullying in Roman Urdu posts on social media. Their model was found to be 92% accurate. An accuracy of 86% is caregiver; all these make the use of the ML approach especially for classification tasks in multilingual environments to be effective if enhanced with better features.

For sentiment analysis of the Urdu text, Khan et al[36]., employed basic and advanced MLand DL algorithms of which the best reached the F1 score 82.05% with Logistic Regression. This work demonstrates that the fusion of standard machine learning techniques with deep learning techniques to perform text classification in low-resource languages is efficient. Al Maruf et al[37]. surveyed the recent advances in Bengali hate speech detection and analyzed that the cross-lingual transfer learning technique and contextual information are crucial. These findings translate directly to Roman Urdu hate speech detection where, as mentioned earlier, there is a lack of annotated datasets and cross-lingual approaches could prove helpful in surmounting this problem. Later, Khan et al[38]. applied this hybrid method to abusive language detection in the Urdu language considering traditional ML models and deep learning models with attention. In the case of the DUAL dataset of more than 12,000 tweets, the proposed RF attained an F1 measure of 0.96%, while the best result was obtained by Bi-LSTM, with attention and Word2Vec embeddings. Thus, future work in this area should focus on improving the attention mechanisms to facilitate better detection with low-resource languages such as Urdu with an accuracy of 0.95%.

Razi and Ejaz[39]proposed detection of cyberbullying in the multilingual mixed text of Urdu, Roman Urdu, and English using MBERT and MuRIL models fine-tuned for the detection purpose. They got an F1 score of 0.92. While work has been done in multilingual hate speech detection, we show in this study, through an ablation study with MuRIL, how transfer learning is very effective, scoring 92%. To the best of our knowledge, Khan et al[40]. proposed a multi-class sentiment analysis model utilizing MBERT from the Urdu text, which previously reported an F1 score of 81.49%. Their research also showed that using BERT to improve text classification performance can have a high impact, especially on posts in resource-poor languages including the Urdu language as a result of having less labeled data for hate speech identification. They applied and experimented with spare ML, including the DL algorithms of RF with the binary relevance strategy, and attested that the F1 score discovered was 56.1%. This work underlines the important fact that Emotion classification in Low-resource languages is not an isolated task; it goes hand in hand with hate speech detection.

In the research done by Amjad et al[41]., the authors researched the effects of data augmentation in specific towards the voice recognition system. Although in their study, they mainly investigate Pakistani racial speaker recognition, some of the discovered concepts, such as data augmentation of limited datasets with techniques including pitch shifting and time stretching, could be potentially beneficial for optimizing hate speech detection models for Roman Urdu, knowing the fact that collection of large datasets labeled for this purpose often becomes an issue. Bade et.al[42] proposed a model using Random forest with TF-IDF to classify hate speech in code-mixed Telugu language considering social media posts. The authors' model had a macro F1 of 0.492, demonstrating that the language processing task is not a straightforward one as a result of code-mixed languages which is prevalent in the detected Roman Urdu hate speech.

Gandhi et al[43]. recently delivered a versatile review of the recently employed approaches for hate speech detection and remarked on the increasing application of Explainable AI in this problem. The review also found out that some of the transformer-based models for example BERT, and its variants have surpassed traditional ML models in the detection of hate speech not to mention the fact that these models have established new benchmarks, particularly in Roman Urdu which is a low resource language. In the context of the present paper, it is worth describing the multilingual framework for hate speech enhanced by, DL and and transformer models in 13 languages to detect hate speech proposed by Hashmi et al[44].. In developing the programmed study schedule, they obtained an F1 score of 0.95 for Roman Urdu which is higher than the previous studies. This research thus highlights the necessity of involving language-specific knowledge together with cross-lingual transfer for better hate speech classification across the multilingual setting.

In a recent study, Khan et al[45]. designed a model whereby they combined CNN with LSTM for sentiment analysis of Roman Urdu and English script. The same high accuracy was reached by the model, and the best results were obtained by developing SVM classifiers along with Word2Vec, a continuous bag of words (CBoW) which produced 0.904 of accuracy on the RUSA dataset. It emphasizes the use of the traditional classifiers of the ML family combined with deep learning for Roman Urdu text classification.

Akhter et al[1]. proposed a Roman Urdu dataset and an Urdu offensive dataset by gathering the comments on Youtube videos. Roman Urdu dataset comprised 147000 instances labeled with two classes offensive and non-offensive. To evaluate the corpus, the authors used word and character n-grams for feature extraction and applied many machine-learning classifiers to classify the comments. LogitBoost achieved the best F1 score of 99.2% using character tri-gram features.

Ali et al[22]. developed an Urdu hate lexicon and used it to create an annotated dataset of 10526 Urdu tweets. Moreover, as a baseline study, the authors deployed multiple machine-learning approaches for hate speech identification. They also employ transfer learning to deal with pre-trained FastText Urdu word embeddings and multi-lingual BERT embeddings. Finally, the authors show that using different variations of BERT, BERT, xlm-roberta, and distil-BERT achieved the highest F1 scores of 0.68, 0.68, and 0.69, respectively, on multi-class classification problems.

Table 1 presents the summary of various resources and methods developed for hate speech detection in different languages. However, we noted that no one explored the hybrid models comprising deep learning and machine learning models. Additionally, research on hate detection in the Roman Urdu language is rather scarce. To fill this gap, this research proposes a hybrid approach in which deep learning models are employed as automatic feature learners and machine learning algorithms for classification to detect hate speech from Roman Urdu text. Furthermore, hyperparameter optimization methods are explored to optimize the performance of classifiers.

Proposed hybrid model for roman urdu hate speech detection

In this study, we propose a hybrid model where "hybrid" refers to the integration of two distinct approaches: DL models CNN, LSTM, BiLSTM, GRU, and transformer model MBERT are utilized as automatic feature learners, and ML models are used for classification. The hybrid nature lies in leveraging the strengths of DL and transformer models in feature learning and the effectiveness of ML models in classification. This combination offers more robust performance for Roman Urdu HSD compared to using either approach in isolation. Previously, no such hybrid approach has been implemented to address the problem of HSD for Roman Urdu text.

Traditional hand-crafted feature extraction methods such as term frequency-inverse document frequency (TF-IDF) and a bag of words (BoW) are effectively utilized in many classification tasks and produce promising results. However, several drawbacks are associated with these methods such as manual extraction of these features, and time consumption. Similarly, these methods do not consider the semantic relationship between terms. To overcome these limitations, deep learning models and transformer model MBERT are used, which process the data automatically to extract important features and train the models effectively.

Year	Article	Language	Corpus	Best Method	Best Results
2024	Nasir et al[30].	Roman Urdu	10,526 Tweets	Logistic Regression	Accuracy = 87% for offensive-hate task, 81% for neutral-hostile task
2024	Maqbool et al[31].	Roman Urdu	RUHSOLD++ Augmented (10,012 Tweets)	m-BERT + Augmentation	Accuracy = 91.3%, F1 = 0.91 with 50% augmented data
2024	Malik et al[32].	Nastaliq Urdu	HSTC Corpus	Urdu-DistilBERT	Accuracy = 86.58% for hate speech detection, 84.17% for target community
2024	Atif et al[33].	Roman Urdu	Cyberbullying Dataset	GRU	Accuracy = 97%, F1 = 97% for cyberbullying detection
2024	Khan et al[38].	Urdu	DUAL Dataset (12,000 Tweets)	Random Forest, Bi-LSTM with Attention	F1 = 0.96 (RF), Accuracy = 0.95 (Bi-LSTM with Attention)
2024	Ullah et al[34].	Roman Urdu and Telugu	Code-mixed Social Media	DistilBERT	F1 = 0.6369
2024	Jahangir et al[35].	Roman Urdu	Roman Urdu Social Media Posts	SVM, Logistic Regression	Accuracy = 92.86%
2024	Razi and Ejaz[39]	Urdu, Roman Urdu, and English	Mixed-language Social Media	m-BERT, MuRIL	F1 = 0.92 (MuRIL)
2024	Khan et al[40].	Urdu	Multilingual Sentiment Corpus	m-BERT	F1 = 81.49%
2024	Hashmi et al[44].	Roman Urdu	Multilingual Corpus (13 Languages)	ML, DL, and Transformers	F1 = 0.95 for Roman Urdu
2024	Gandhi et al[43].	Roman Urdu	Comprehensive Review	Explainable AI and Transformers	BERT-based models outperform traditional ML in Roman Urdu hate speech detection
2024	Bade et al[42].	Telugu	Code-mixed Telugu Social Media	Random Forest, TF-IDF	Macro F1 = 0.492
2024	Al Maruf et al[37].	Bengali	Bengali Hate Speech Corpus	Cross-lingual Transfer Learning	Insights transferable to Roman Urdu
2023	Mehmood et al[29].	Roman Urdu	RUHSOLD	Passion-Net with Attention	F1 score improvement by 8.7%, recall improvement by 18.6%
2022	Khan et al[45].	Roman Urdu and English	RUSA Dataset	CNN-LSTM, SVM + Word2Vec CBOW	Accuracy = 0.904 for Roman Urdu sentiment classification
2022	Ashraf et al[46].	Urdu	Multi-label Emotion Classification Corpus	Random Forest, Binary Relevance	F1 = 56.1%
2022	Amjad et al[41].	Roman Urdu	Augmented Speaker Recognition Dataset	Data Augmentation	Insights transferable to hate speech detection
2022	Ali et al[22].	Urdu	10,526 Tweets	Distil-Bert	F1 = 0.69
2021	Khan et al[36].	Urdu	Sentiment Corpus	Logistic Regression	F1 = 82.05%
2020	Rizwan et al[23].	Roman Urdu	RUHSOLD (10,012 Tweets)	BERT+CNN-gram	Accuracy = 0.90, F1 score = 0.90 for Coarse-grained task Accuracy = 0.82, F1 score = 0.75 for Fine-grained task
2020	Akhter et al[1].	Roman Urdu and Urdu	Urdu Offensive (1,47,000 Comments)	Char tri-grams, LogitBoost	F1 = 99.2% for Roman Urdu task
2019	Rizoiu et al[27].	English	16K Tweets and 22,304 Tweets	BiLSTM	F1 = 73.89%
2017	Malmasi et al[24].	English	14,509 Tweets	Char 4-grams	Accuracy = 78.0%
2016	Waseem et al[9].	English	16K Tweets	Char n-grams	F1 = 73.89%

Table 1. Summary of previous works on hate speech detection.

Figure 1 shows the workflow of the adopted methodology. We employed four deep learning models including CNN, LSTM, BiLSTM, and GRU as automatic feature learners. Due to their inherent benefits in NLP applications, deep learning models are chosen for feature learning in our hate speech detection task. Deep learning has the potential to improve the detection of subtle patterns in hate speech since it can automatically learn pertinent features from raw text data, eliminating the need for extracting manual features. In addition, the analysis of textual hierarchical and sequential structures, such as words, phrases, and sentences, are frequently used in the process of detecting hate speech. Deep learning models are an appropriate option for such tasks because they are capable of capturing long-term dependencies and hierarchies, allowing for a deeper understanding of the context and meaning of the language used in hate speech. After that extracted features are passed to the machine learning algorithms such as SVM, RF, LR, and NB for hate speech categorization. To optimize the performance of machine learning algorithms we employed three hyperparameter optimization methods including GS, RS, and BOGP. These hyperparameter optimization approaches were chosen for their complementing strengths. Grid Search is more efficient for smaller parameter spaces, but Random Search is more efficient for bigger search spaces. When evaluating models is expensive or when dealing with noisy data, Bayesian Optimization comes in handy. We can efficiently explore the hyperparameter space and determine the ideal parameters that lead to enhanced performance of our hate speech detection models by combining these methods.

Dataset

For experiments, this study used two benchmark Roman Urdu hate speech corpora including HS-RU-20 corpus[8]and RUHSOLD corpus[23]. Both corpora are separated into two different subtasks, as presented in Table 2. HS-RU-20 corpus comprises 5000 labeled instances with unbalanced distribution over two distinct classes neutral and hostile containing 1430 and 3570 instances respectively in the neutral-hostile task. The 3570 instances of the hostile class are further divided into two classes hate speech and offensive comprising 833 and 2737 instances, respectively in the hate speech-offensive task. Similarly, the RUHSOLD corpus consists of 10012 unbalanced annotated instances with two subtasks coarse-grained and fine-grained tasks. In the coarse-grained task, instances are distributed over two different classes normal and abusive/offensive comprising 5349 and 4663 samples, respectively. The instances of the abusive/offensive class are further categorized into four different classes abusive, religious hate, sexism, and profane comprising 2402, 782, 839, and 640 instances, respectively in the fine-grained task. The normal class containing 5349 instances is also included in the fine-grained task.

Data preprocessing

The data of both corpora HS-RU-20 corpus and RUHSOLD corpus is a collection of raw tweets that contain multiple issues such as punctuation marks, extra spaces, hyperlinks, etc. So initially we clean the data using different preprocessing steps before the feature engineering process. Different data preprocessing steps are applied to both corpora. We removed all the punctuation marks (i.e. ? &,#,!) in data preprocessing. We used regular expressions (RE) and removed all uniform resource locators (URLs), extra spaces, and numeric values from the text because they do not provide any useful information for model training. All stopwords such as 'a', 'the', 'is' are also removed from the text using regular expression. All the data is also converted to lowercase during preprocessing.

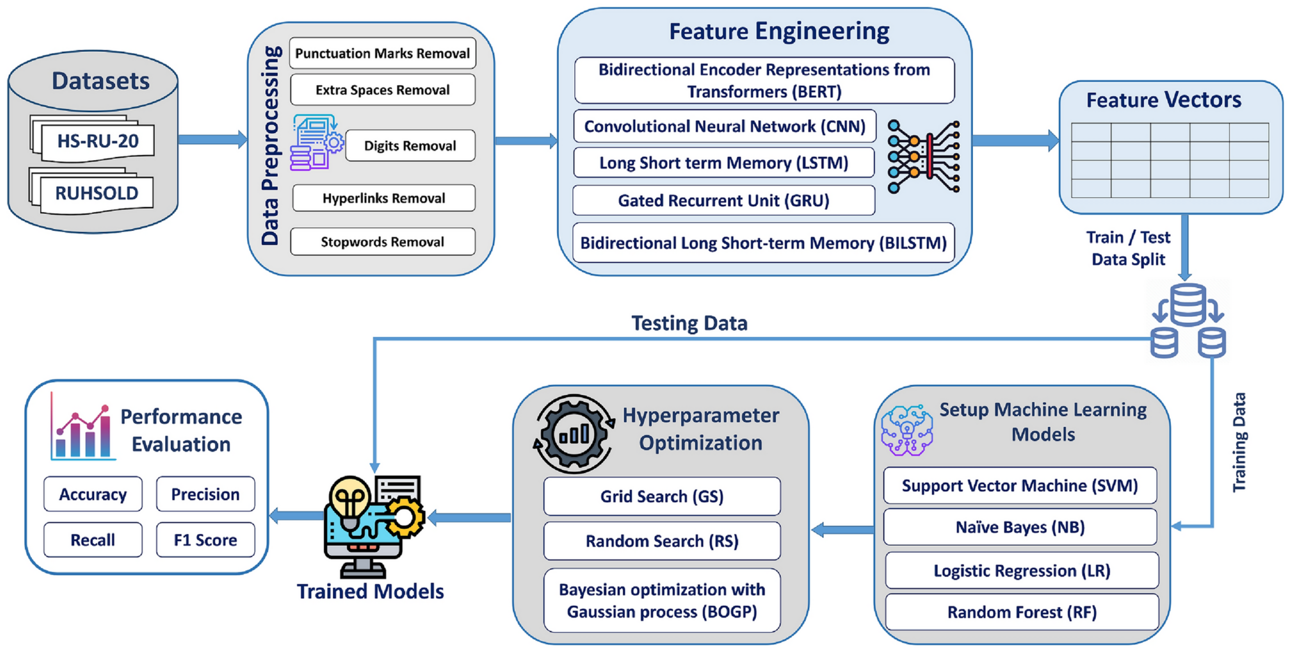
Datasets
HS-RU-20
RUHSOLD
Data Preprocessing
Punctuation Marks Removal
Extra Spaces Removal
Digits Removal
Hyperlinks Removal
Stopwords Removal
Feature Engineering
Bidirectional Encoder Representations from Transformers (BERT)
Convolutional Neural Network (CNN)
Long Short term Memory (LSTM)
Gated Recurrent Unit (GRU)
Bidirectional Long Short-term Memory (BILSTM)
Feature Vectors
Train / Test Data Split
Testing Data
Training Data
Setup Machine Learning Models
Support Vector Machine (SVM)
Naïve Bayes (NB)
Logistic Regression (LR)
Random Forest (RF)
Hyperparameter Optimization
Grid Search (GS)
Random Search (RS)
Bayesian optimization with Gaussian process (BOGP)
Trained Models
Performance Evaluation
Accuracy
Precision
Recall
F1 Score

Fig. 1. Workflow diagram of the adopted methodology.

	Neutral-Hostile Task		Hate Speech-Offensive Task	
HS-RU-20 Corpus	Classes	No. of Instances	Classes	No. of Instances
	Neutral	1430	Hate Speech	833
	Hostile	3570	Offensive	2737
	Total	5000	Total	3570
RUHSOLD Corpus	Coarse-grained Task		Fine-grained Task	
	Classes	No. of Instances	Classes	No. of Instances
	Normal	5349	Normal	5349
	Abusive/Offensive	4663	Abusive/Offensive	2402
			Religious Hate	782
			Sexism	839
			Profane/Untargeted	640
	Total	10,012	Total	10,012

Table 2. Details of HS-RU-20 and RUHSOLD corpora for hate speech detection.

Deep learning models

In this study, deep learning models are employed as automatic feature learners. Deep learning models process information analogous to the human brain[47]. Deep learning approaches achieved notable results in the different NLP tasks. Two primary steps are involved in deep learning models to perform a particular classification task. First, converting the raw text into embedding vectors, and second, building a model to perform classification. To address the task of hate speech detection in Roman Urdu, we utilized CNN, LSTM, BiLSTM, and GRU as automatic feature learners. These deep learning models were chosen primarily because of their abilities to capture local features through convolutional processes, represent temporal associations in sequential data, and process information in both directions to capture both past and future contexts. Our proposed hybrid approach, by employing these deep learning models, is better able to extract and represent complex linguistic patterns, which enhances the performance of hate speech detection in Roman Urdu. The detail of each deep learning model is discussed in the subsections below:

CNN is a popular feed-forward approach that processes information only in the forward direction[48]. In recent years, CNN has achieved promising results in various classification-related tasks. It attracts the broad attention of researchers because of its good feature learning and classification capability[49]. CNN architecture comprises three primary layers: convolutional layer, pooling layer, and fully connected layer. The input is provided in the form of embeddings to the convolutional layer. The convolutional layer extracts the output of neurons by applying a convolutional operation to the input through filters. Each convolutional layer is followed by a pooling layer which reduces computational complexity, performs down-sampling along with dimensionality, and decreases the number of parameters without loss of important data. Finally, the fully connected layer generates class scores or feature maps, and then a layer with softmax activation is applied for classification.

LSTM[50]is a specific type of recurrent neural network (RNN) model. RNN architecture differs from CNN because it is capable of keeping previous contextual information in memory and applying it to current information for learning sequences. RNN model learns only a limited range of context, this problem is often referred to as the vanishing gradient problem. To address the limitations of RNN, the LSTM model is proposed, which is capable of learning long-term dependencies[51]. LSTM architecture comprises a cell state (to store memory) and three gates, i.e. input gate, forget gate, and output gate. The input gate decides which information to pass into the memory block at the current time step. The forget gate decides which information is to keep and reset from the memory cell, and the output gate manages the flow of output in the next hidden states.

BiLSTM[52] adopts the concept of the LSTM model with some enhancements in learning strategy. LSTM model learns the temporal sequences and long-pattern dependencies perfectly but fails to understand the complete sentence structure. The LSTM model considered only preceding information during learning, whereas the bidirectional LSTM learns sequences from both directions. The bidirectional LSTM model comprises two separate hidden layers connected to a single output. The working of the first hidden layer is similar to the LSTM model, whereas the second layer, learns the sequence in a backward direction. BiLSTM combined the outputs of both layers, which increases the contextual information to produce better results.

GRU[53] is also a well-known variation of the RNN model and achieves notable performance in sequence learning problems. GRU model is similar to the LSTM and is designed to address the problem of vanishing and exploiting gradient. Complexity and computing speeds are the major limitations of LSTM and bidirectional LSTM models. GRU overcomes these limitations by reducing the gates to two i.e. update gate and reset gate. The update gate is responsible for deciding how much preceding information is retained and to be passed to the next time steps. The forget gate decides how often previous information to be forgotten.

MBERT is a pre-trained transformer model designed to handle multiple languages by leveraging a shared vocabulary and a common architecture across languages. It is trained on the concatenation of texts from 104 languages without explicit cross-lingual alignment, making it highly effective for tasks like multilingual text classification, translation, and sentiment analysis. MBERT encodes text from different languages into a shared embedding space, allowing it to generalize across languages for tasks such as hate speech detection, even in resource-limited settings. This makes it ideal for multilingual NLP applications without requiring language-specific fine-tuning.

Machine learning algorithms

For classification, we freeze the last layer, i.e. a softmax layer of deep learning models, and use the ML algorithms. In this study, we used four machine learning algorithms including RF, SVM, NB, and LR to detect hate speech in Roman Urdu.

Hyperparameter optimization methods

Hyperparameters are machine learning model parameters that are set before the training process. In contrast to model parameters, which are learned from training data, hyperparameters are set by the model developer and stay constant during the training process. These hyperparameters affect the learning algorithm's behavior and can have a substantial influence on the model's performance and generalization capabilities. In general, building an efficient machine learning model is a complicated and time-consuming procedure that entails selecting a suitable algorithm and improving the system model by optimizing its hyperparameters. The importance of tuning these hyperparameters originates from their large effect on the machine learning model's performance. Tuning these parameters can have a significant influence on the model's capacity to generalize and distinguish between different types of hate speech. For instance, In the RF algorithm, the desired count of estimators and the highest depth of trees for the analysis, influence the model's complexity and overfitting potential. The kernel and regularization parameter (C) used in SVM can have a significant influence on the decision boundary and classification accuracy.

We selected to tune these hyperparameters particularly because they are essential components that affect the behavior and adaptability of machine learning algorithms in hate speech detection. We can find the optimal settings that optimize the performance of our models on the Roman Urdu hate speech data by fine-tuning these parameters. To optimize the performance of each machine learning algorithm, we employed three hyperparameter optimization methods including GS, RS, and BOGP. Although there are a number of additional hyperparameter optimization techniques, we chose these three because of their reliable performance in various circumstances. Every approach presents a different trade-off between effectiveness and performance, which qualifies them as suitable options for our hate speech detection task. These techniques are also frequently utilized in the literature and have succeeded in a number of ML applications. Muslim et al[54]. used the GS method with an SVM classifier for product review sentiment classification and improved the accuracy. Omotehinwa et al[55]. optimized the performance of RF and XGBoost using the GS method for spam email classification. Alzanin et al[56]. used a random search method to improve the performance of different machine learning algorithms in Arabic text classification. Valarmathi et al[57]. applied various HOP methods including RS, and GS for heart disease prediction and obtained outstanding results. We ensure a rigorous and thorough search for the optimum hyperparameter configurations, resulting in more accurate and higher performance in our hate speech detection models, by employing a broad range of hyperparameter optimization strategies.

GS[58] is a brute-force method that requires comprehensive searching for a predefined set of parametric values from configuration space. It is an exhaustive search that assesses all possible combinations of hyperparameters by performing a Cartesian product of a user-defined set of parametric values. GS can be efficiently executed and parallelized. However, a major drawback associated with GS is its high-dimensional configuration space for searching hyperparameter values. The count of evaluations rises exponentially as the count of hyperparameters increases. This method performs well with a small configuration space of hyperparameters.

RS[59] method was developed to address the issues associated with GS. The working process of RS is very similar to GS, but it chooses the pre-defined hyperparameter values randomly from configuration space within upper and lower bounds. This method is efficient and has the ability to search from large configurations or search spaces. The major benefit of RS is that it can be efficiently parallelized because every evaluation is independent. RS randomly chooses the fixed count of parametric values combination from search space which leads to enhancing the performance by minimizing the time complexity. The major disadvantage of GS and RS is that these are time-consuming methods because each evaluation is separate from prior evaluations.

BO[60]is the most widely used hyperparameter optimization method which works in an iterative manner. Unlike earlier discussed methods, BO uses the previously achieved results to find future evaluation points. BO comprises two essential elements, a surrogate method, and an acquisition method. The main aim of the surrogate method is to fit each experimental point into the objective method. In BO, the Gaussian process[61] is a traditional surrogate method used to perform the objective function modeling.

In terms of contribution to the body of knowledge, the discussion around hyperparameter optimization highlights the uniqueness and importance of our study. While previous research has focused on hate speech recognition, little attention has been paid to the unique issues associated with the Roman Urdu language. We emphasize the need to customize hate speech detection models to the specific linguistic characteristics of Roman Urdu by thoroughly exploring and tuning hyperparameters using GS, RS, and BOGP. Our comprehensive evaluation of hyperparameter configurations is critical in improving the accuracy and generalization capabilities of hate speech detection in Roman Urdu. Furthermore, comparing numerous hyperparameter optimization approaches sheds light on the efficiency and usefulness of various optimization strategies, incorporating to improved understanding of optimizing machine learning algorithms for categorizing hate speech. The thorough investigation and novel utilization of hyperparameter optimization in the context of Roman Urdu hate speech detection distinguishes our work as a valuable and distinct contribution to the current body of knowledge in NLP.

Model evaluation

To evaluate the performance of proposed hybrid models on Roman Urdu corpora, we employed four evaluation measures accuracy, precision, recall, and F1 score. Several studies have used these evaluation metrics to evaluate the performance of models for Roman Urdu HSD[8,23,27,31,33]. The following formulas are used for the evaluation metrics

$$Accuracy(Acc) = \frac{\text{Count of Correctly Predicted Examples}}{\text{Total Count of Examples}} \tag{1}$$

$$Precision(P) = \frac{TP}{TP + FP} \tag{2}$$

$$Recall(R) = \frac{TP}{TP + FN} \tag{3}$$

$$F_1 score = \frac{2 \times P \times R}{P + R} \tag{4}$$

where true positive (TP) symbolizes the proportion of correctly categorized positive examples; false positive (FP) indicates the number of negative examples incorrectly categorized as positive examples; true negative (TN) denotes the proportion of correctly categorized negative examples, and false negative (FN) denotes the number of positive examples incorrectly categorized as negative examples.

Experimental results

Word embeddings

The occurrence of words and characters in a corpus represents the crude features. Each word is recognized as a separate feature and the system seeks to initiate learning vector representations in a particular context. There are various methods to represent these features, but the most common methods are one-hot vector representation, and word embeddings (dense vector representation). We used the word embeddings method that is most extensively used in text classification tasks[62]. Word embeddings provide a similar representation vector of each word, which reduces the size of vocabulary without losing important information. In this study, we used global vector (GloVe)[63] word embedding for vector representation which is based on an unsupervised global learning model. It creates data of the co-occurred words in matrix form and for each word, context words are searched in corpora within a specific window size. This procedure creates linear substructures of vectors that are referred to as word embedding. To perform experiments we used 100-dimensional pre-trained GloVe word embedding comprising 27 billion tokens and 1.2 million unique words. Other types of word embeddings, such as FastText, Word2Vec, and BERT-based embeddings, are also available. Each embedding approach has distinct advantages that might be advantageous based on the goal and corpus characteristics. However, we selected Glove since it is the most commonly utilized in many NLP tasks. The pre-trained GloVe embeddings were originally trained on English text. However, Roman Urdu shares significant lexical overlap with English due to the use of the Roman alphabet, and many Roman Urdu terms are either directly borrowed from English or transliterated. This overlap allows English-trained embeddings like GloVe to capture relevant features from Roman Urdu text.

For the transformer model, MBERT converts text into embeddings through a process of tokenization and contextualized encoding. First, it tokenizes the input text into subword units using WordPiece, which handles diverse language scripts. The tokens are then fed into the transformer layers of MBERT, where each token is represented as a dense vector (embedding) based on its context within the sentence. This contextualized representation captures the meaning of the word in relation to the surrounding words, generating embeddings that are useful for downstream tasks like multilingual hate speech detection.

Hyperparameter tuning

Hyperparameter tuning is a fundamental process of determining the combination of optimal values during model training. These optimal sets of parameters help to improve the model's performance. In this study, we randomly tested different parameter values during initializing, compiling, and training the model, as presented in Table 3. Regarding the number of hidden layers, we tested 1,2, and 3 hidden layers for all models and identified that the 3-layer CNN model and 1-layer LSTM, BiLSTM, and GRU models are more effective. Hidden layers are critical elements of deep learning models since they are responsible for learning and demonstrating complicated patterns in data. In the CNN model, we used a 1_D convolutional layer with a filter size of 2 to extract the features. We tried different hidden units such as 64, 128, and 256, and observed that 64 units are more suitable during training in all models. We applied different activation functions such as ReLu and RMSprob and observed that ReLu produced better results. Non-linearity is introduced into neural networks through activation functions, allowing them to learn and approximate complicated correlations in data. Dropout layers avoid overfitting by eliminating a subset of neurons at random during learning. We experimented with dropout rates of 0.1, 0.2, and 0.3. A dropout rate of 0.2 was determined to be the most effective in limiting overfitting while not lowering the model's ability to learn significant patterns. Dropout rates of 0.1 resulted in less regularization, which might lead to overfitting, whereas rates of 0.3 were too high and limited the model's capacity to learn effectively from the data. All models are trained with 10 epochs with a batch size of 64.

For MBERT, we utilized the bert-base-multilingual-cased version of Multilingual BERT (mBERT). Key parameters include setting the maximum token length to 128, which ensures that all input text is either truncated or padded to a fixed length for consistency during processing. The embeddings were extracted from the last

Model	Parameters	Parameter values tested	Best selected parameter
CNN	Hidden layers	1, 2, 3	3
	Hidden units	64, 128, 256	64
	Activation function	ReLu, tanh, softmax	ReLu
	Dropout	0.1, 0.2, 0.3	0.2
LSTM	Hidden layers	1, 2, 3	1
	Hidden units	64, 128, 256	128
	Activation function	ReLu, tanh, softmax	ReLu
	Dropout	0.1, 0.2, 0.3	0.2
BiLSTM	Hidden layers	1, 2, 3	1
	Hidden units	64, 128, 256	128
	Activation function	ReLu, tanh, softmax	ReLu
	Dropout	0.1, 0.2, 0.3	0.2
GRU	Hidden layers	1, 2, 3	1
	Hidden units	64, 128, 256	128
	Activation function	ReLu, tanh, softmax	ReLu
	Dropout	0.1, 0.2, 0.3	0.2

Table 3. Parameter settings of deep learning models.

hidden layer of MBERT, and we averaged these contextualized representations to obtain a dense vector for each text. Truncation and padding were enabled to efficiently manage variable-length input sequences, ensuring that MBERT could process multilingual data for tasks like HSD.

Ablation study

In this section, we attempt to conduct a more sensitive ablation study of the choice of hyperparameter and model components for hybrid models. This will help in isolating the effect of individual hyperparameters on the final accuracy and F1 score by tuning one parameter at a time while the other parameters remain constant. We applied different numbers of hidden layers 1, 2, and 3 for all models. It turned out that in the case of the CNN model, the application of three hidden layers allowed us to improve performance by up to 2% according to the F1 score in comparison with a model with just one layer. In the case of LSTM, BiLSTM, and GRU, on the other hand, the best performance was provided by a model architecture with one layer, since overfitting occurred when additional layers were added, especially in the case of smaller datasets like HS-RU-20.

Dropout is essential to prevent overfitting. We chose the dropout rates 0.1, 0.2, and 0.3. Indeed, the ablation study showed that a dropout rate of 0.2 consistently yielded the best balance between regularization and model learning capacity. When setting the dropout rate to higher values (0.3), it resulted in 5% worse F1 scores, as this already too-strong regularization hindered the model from learning important patterns in the data. While a dropout of 0.1 resulted in overfitting, specifically of the RUHSOLD dataset, it turned out to be too low to apply effective regularization.

In the experiments, we also used 64, 128, and 256 hidden units across all the models. Similar to most models, the best value is 128 hidden units for both LSTM and GRU. However, their increase to 256 did not bring significant improvements but introduced more complexity, which manifested in longer training time without substantial performance gain.

We compare two optimizers: Adam and RMSprop. For all models, the Adam optimizer behaves consistently better than RMSprop by 3% in terms of F1 score on average. We believe this must be due to its adaptive learning rate, thus enabling it to converge faster and perform better when using the LSTM-based model due to its faster convergence along with better generalization.

The configuration space of hyperparameters for machine learning algorithms is presented in Table 4. The hyperparameter optimization methods GS, RS, and BOGP select the best parameter values from the configuration space during model training. The effect of the best combination of parameter values corresponding to each machine learning algorithm's performance is discussed in the results section.

Evaluation methodology

The problem of hate speech detection from Roman Urdu is treated as a text classification task under a supervised learning methodology. In the HS-RU-20 corpus, the hate speech detection task is processed as a binary classification problem in both subtasks (i.e. Neutral-Hostile and Hate Speech-Offensive tasks). In the RUHSOLD corpus, the coarse-grained task is classified into two classes (Normal and Abusive) and treated as a binary classification problem, whereas the fine grained-task processed as a multi-class classification problem and classified into five classes (Abusive, Normal, Religious Hate, Sexism, Profane).

We performed all the experiments in Python 3.9.13 with multiple libraries including Scikit-learn and Keras backend TensorFlow. For performance assessment, we used the train-test split approach with a ratio of 80% of data used for training and 20% for testing. We used 10% of training data for validation in deep learning models.

Machine learning algorithm	Parameters	Parameter values
Random forest	n_estimators	sp_randint(10,100)
	max_features	sp_randint(1,64)
	max_depth	sp_randint(5,50)
	min_samples_leaf	sp_randint(1,11)
	min_samples_split	sp_randint(2,11)
	criterion	[’gini’,’entropy’]
Logistic regression	penalty	[’l1’,’l2’]
	C	np.logspace(-3,3,7)
	solver	[’newton-cg’, ’lbfgs’, ’liblinear’]
Support vector machine	’C’	[1,10, 100]
	kernel	[’linear’,’poly’,’rbf’,’sigmoid’]
Naive Bayes	’alpha’	[0.01, 0.1, 0.5, 1.0, 10.0]

Table 4. Hyperparameters for machine learning algorithms.

	Neutral-Hostile Task					Hate Speech-Offensive Task			
Features	Model	Accuracy	Precision	Recall	F1 score	Accuracy	Precision	Recall	F1 score
CNN	SVM	0.85	0.85	0.86	0.85	0.84	0.81	0.83	0.80
	RF	0.83	0.82	0.84	0.83	0.83	0.81	0.82	0.81
	LR	0.85	0.84	0.85	0.85	0.81	0.80	0.82	0.81
	NB	0.82	0.83	0.83	0.82	0.80	0.78	0.79	0.80
LSTM	SVM	**0.88**	**0.83**	**0.85**	**0.90**	**0.86**	**0.83**	**0.84**	**0.82**
	RF	0.86	0.80	0.83	0.89	0.84	0.81	0.83	0.81
	LR	0.87	0.80	0.84	0.87	0.83	0.79	0.81	0.81
	NB	0.85	0.81	0.84	0.86	0.83	0.80	0.80	0.80
BiLSTM	SVM	0.87	0.86	0.85	0.85	0.85	0.81	0.81	0.81
	RF	0.85	0.84	0.86	0.84	0.83	0.81	0.82	0.79
	LR	0.81	0.82	0.84	0.83	0.83	0.80	0.80	0.80
	NB	0.84	0.80	0.83	0.82	0.81	0.80	0.81	0.81
GRU	SVM	0.82	0.80	0.82	0.81	0.81	0.77	0.79	0.79
	RF	0.80	0.79	0.79	0.79	0.77	0.78	0.78	0.78
	LR	0.81	0.81	0.80	0.80	0.79	0.80	0.77	0.80
	NB	0.79	0.76	0.81	0.79	0.78	0.78	0.79	0.78

Table 5. Results obtained using hybrid models with default parameters on HS-RU-20 corpus. Bold values shows the best scores for each evaluation metric.

Results and analysis

The results obtained from the hybrid models using various hyperparameter optimization methods (HPOMs) on the HS-RU-20 and RUSHOLD corpus reveal insightful trends across multiple ML algorithms. First, the achieved results using deep learning and transformer models as feature learners with machine learning algorithms as classifiers using default parameters are discussed. Later, the performance of hyperparameter optimization methods is examined on each corpus.

Results obtained using HS-RU-20 corpus

The primary focus of the analysis using the HS-RU-20 corpus is on two main tasks: Neutral-Hostile Task and Hate Speech-Offensive Task. These two tasks are evaluated across different feature extraction models such as MBERT, CNN, LSTM, BiLSTM, and GRU, paired with various MLAs such as SVM, RF, LR, and NB.

Table 5 presents the results of hybrid models with default parameters of machine learning algorithms on the HS-RU-20 hate speech corpus. Regarding the neutral-hostile task, the overall results indicate that the LSTM feature learner with the SVM algorithm obtained the highest performance with 0.88 accuracy and 0.90 F1 scores. The GRU feature learner with NB gives the lowest performance and has a 0.79 score each for accuracy and an F1 score. Regarding the hate speech-offensive task, the feature learner LSTM model and SVM algorithm produced the highest results with 0.86 accuracy and 0.82 F1 scores.

Figure 2 presents the results of hybrid models with default parameters of machine learning algorithms on the HS-RU-20 hate speech corpus. Regarding the neutral-hostile task, the overall results indicate that the deep learning model LSTM feature learner with the SVM algorithm obtained the highest performance with 0.88 accuracy and 0.90 F1 scores. However, when using the transformer model MBERT feature learner combined

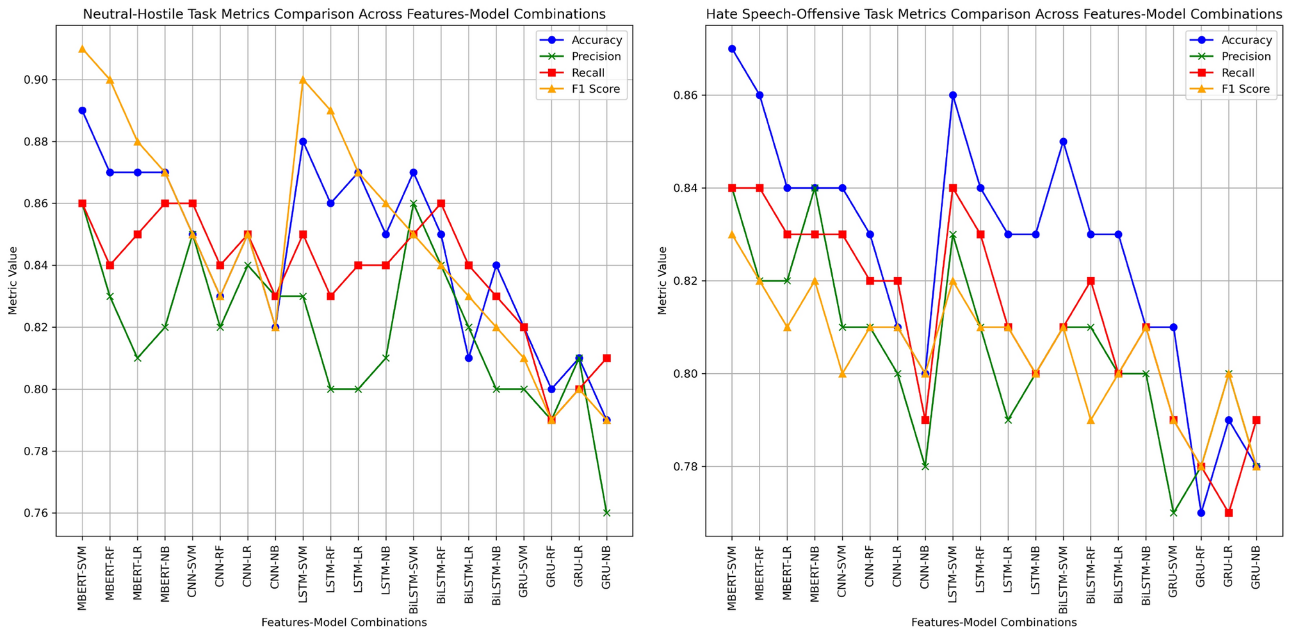

Fig. 2. Results obtained using hybrid models with default parameters on HS-RU-20 corpus.

with SVM, the performance reaches 0.89 accuracy and an F1 score of 0.91, indicating that MBERT slightly outperforms LSTM in this task. The GRU feature learner with NB gives the lowest performance and has a 0.79 score each for accuracy and an F1 score. For the Hate Speech-Offensive task, the MBERT-SVM hybrid model achieves the highest performance, with an accuracy of 0.87 and an F1 score of 0.83, closely followed by the LSTM-SVM combination with an accuracy of 0.86 and an F1 score of 0.82.

From the results, a significant finding is that the SVM algorithm consistently achieves better results than other machine learning algorithms for both sub-tasks, regardless of the feature extraction model. In particular, MBERT-SVM and LSTM-SVM are highly effective in categorizing neutral-hostile and hate speech-offensive content in Roman Urdu text. MBERT's strength in feature extraction is likely due to its ability to leverage pre-trained multilingual embeddings that capture deeper semantics across multiple languages, including Roman Urdu. This makes MBERT an ideal candidate for hate speech detection in resource-scarce languages. Also, it indicates that SVM is especially effective in categorizing neutral-hostile and distinguishing between hate speech and offensive content in Roman Urdu text. The SVM is better than other classifiers at finding hate speech because it can work well with both small and large datasets. In hate speech detection, text data usually has many different features. The formulation of SVM enables it to find the best line that separates different categories of hate speech in a very effective way.

Additionally, the deep learning-based LSTM feature learner performs significantly better than other feature learners like CNN, BiLSTM, and GRU. The LSTM feature learner is highly adapted to understanding the context of Roman Urdu hate speech since it has the capability to capture long-term relationships in sequences. The importance of utilizing these feature learners for identifying hate speech is highlighted by the development of deep learning methodologies, in particular LSTM.

Overall, findings highlight that LSTM-based feature learners and SVM classifiers can potentially be combined to provide the most significant performance for hate speech detection in Roman Urdu. One of the reasons for their effectiveness in this endeavor is largely due to the LSTM model's ability to learn complicated patterns and the SVM algorithm's proficiency with high-dimensional data. These results highlight the usefulness of the hybrid approach that has been put out and demonstrate its capacity to handle problems with detecting hate speech in Roman Urdu on social media platforms.

Results also show that against every automatic feature learner, the SVM machine learning algorithm produced the best results for both sub-tasks. The overall performance of the models indicates that LSTM-based features when used with SVM obtain the best performance.

The results of the proposed hybrid models, which utilize optimization methods for both the Neutral-Hostile Task and the Hate Speech-Offensive Task on the HS-RU-20 corpus, are presented in Table 6. Figures 3 and 4, respectively, illustrate the performance of models for both datasets. For the Neutral-Hostile Task, the best results were obtained using MBERT as the feature learner and SVM as the classifier, optimized with RS, achieving an accuracy of 0.93 and an F1 score of 0.95, outperforming all other models. Similarly, for the Hate Speech-Offensive Task, MBERT-SVM with RS optimization also obtained state-of-the-art results with an accuracy of 0.89 and an F1 score of 0.88.

Compared to other optimization methods, RS consistently outperformed GS and BOGP, particularly when applied to the MBERT-SVM hybrid model. RS's ability to efficiently explore the hyperparameter space by randomly sampling from predefined ranges allowed it to uncover optimal hyperparameter combinations that

			Neutral-Hostile Task				Hate Speech-Offensive Task			
Features	Model	HPO	Accuracy	Precision	Recall	F1 score	Accuracy	Precision	Recall	F1 score
CNN	SVM	GS	0.85	0.85	0.87	0.86	0.84	0.82	0.83	0.80
		RS	0.87	0.86	0.87	0.89	0.86	0.82	0.84	0.81
		BOGP	0.85	0.85	0.85	0.87	0.85	0.83	0.84	0.81
	RF	GS	0.83	0.84	0.84	0.84	0.83	0.81	0.84	0.81
		RS	0.84	0.82	0.82	0.85	0.84	0.82	0.83	0.83
		BOGP	0.84	0.83	0.83	0.84	0.83	0.82	0.85	0.81
	LR	GS	0.85	0.85	0.87	0.86	0.81	0.81	0.82	0.83
		RS	0.86	0.86	0.86	0.87	0.83	0.81	0.80	0.85
		BOGP	0.85	0.85	0.86	0.85	0.82	0.81	0.83	0.84
	NB	GS	0.82	0.83	0.84	0.83	0.80	0.80	0.81	0.81
		RS	0.83	0.84	0.85	0.84	0.81	0.79	0.81	0.82
		BOGP	0.82	0.83	0.83	0.83	0.81	0.78	0.80	0.80
LSTM	SVM	GS	0.88	0.85	0.86	0.91	0.86	0.84	0.84	0.86
		RS	**0.90**	**0.85**	**0.87**	**0.93**	**0.87**	**0.84**	**0.86**	**0.86**
		BOGP	0.89	0.83	0.86	0.91	0.87	0.83	0.85	0.85
	RF	GS	0.86	0.81	0.83	0.88	0.85	0.83	0.85	0.84
		RS	0.87	0.81	0.81	0.89	0.86	0.82	0.84	0.85
		BOGP	0.86	0.82	0.84	0.88	0.86	0.81	0.85	0.84
	LR	GS	0.87	0.81	0.84	0.88	0.84	0.80	0.83	0.81
		RS	0.88	0.82	0.86	0.89	0.85	0.80	0.82	0.83
		BOGP	0.87	0.82	0.83	0.87	0.84	0.81	0.82	0.82
	NB	GS	0.85	0.82	0.84	0.86	0.83	0.81	0.82	0.81
		RS	0.86	0.81	0.83	0.87	0.83	0.82	0.83	0.82
		BOGP	0.85	0.82	0.83	0.86	0.83	0.81	0.82	0.81
BiLSTM	SVM	GS	0.87	0.86	0.87	0.85	0.86	0.80	0.82	0.82
		RS	0.87	0.85	0.88	0.86	0.87	0.83	0.85	0.83
		BOGP	0.87	0.85	0.86	0.85	0.86	0.82	0.82	0.82
	RF	GS	0.85	0.86	0.86	0.84	0.83	0.83	0.85	0.81
		RS	0.86	0.85	0.87	0.85	0.84	0.81	0.84	0.82
		BOGP	0.85	0.83	0.85	0.85	0.84	0.83	0.83	0.81
	LR	GS	0.82	0.83	0.83	0.83	0.83	0.81	0.81	0.82
		RS	0.83	0.84	0.85	0.85	0.83	0.80	0.80	0.83
		BOGP	0.83	0.84	0.84	0.83	0.83	0.82	0.82	0.82
	NB	GS	0.84	0.81	0.82	0.82	0.81	0.83	0.81	0.81
		RS	0.84	0.83	0.84	0.83	0.82	0.82	0.84	0.82
		BOGP	0.85	0.82	0.83	0.82	0.81	0.81	0.85	0.81
GRU	SVM	GS	0.82	0.82	0.85	0.81	0.81	0.78	0.79	0.79
		RS	0.82	0.84	0.82	0.82	0.82	0.79	0.81	0.80
		BOGP	0.82	0.81	0.83	0.81	0.81	0.78	0.82	0.79
	RF	GS	0.80	0.79	0.80	0.79	0.77	0.79	0.78	0.78
		RS	0.81	0.79	0.81	0.80	0.77	0.79	0.80	0.79
		BOGP	0.80	0.79	0.79	0.79	0.78	0.78	0.79	0.78
	LR	GS	0.81	0.82	0.80	0.80	0.79	0.82	0.79	0.80
		RS	0.82	0.82	0.83	0.81	0.80	0.81	0.82	0.81
		BOGP	0.82	0.80	0.81	0.80	0.79	0.82	0.78	0.80
	NB	GS	0.79	0.77	0.79	0.79	0.78	0.80	0.82	0.78
		RS	0.80	0.78	0.79	0.80	0.79	0.78	0.80	0.79
		BOGP	0.79	0.77	0.78	0.79	0.78	0.79	0.80	0.78
CV (Baseline)[8]	LR	-	0.84	0.84	0.97	0.90	0.84	0.69	0.82	0.75

Table 6. Results obtained using hybrid models with parameter optimization on HS-RU-20 corpus. Bold values shows the best scores for each evaluation metric.

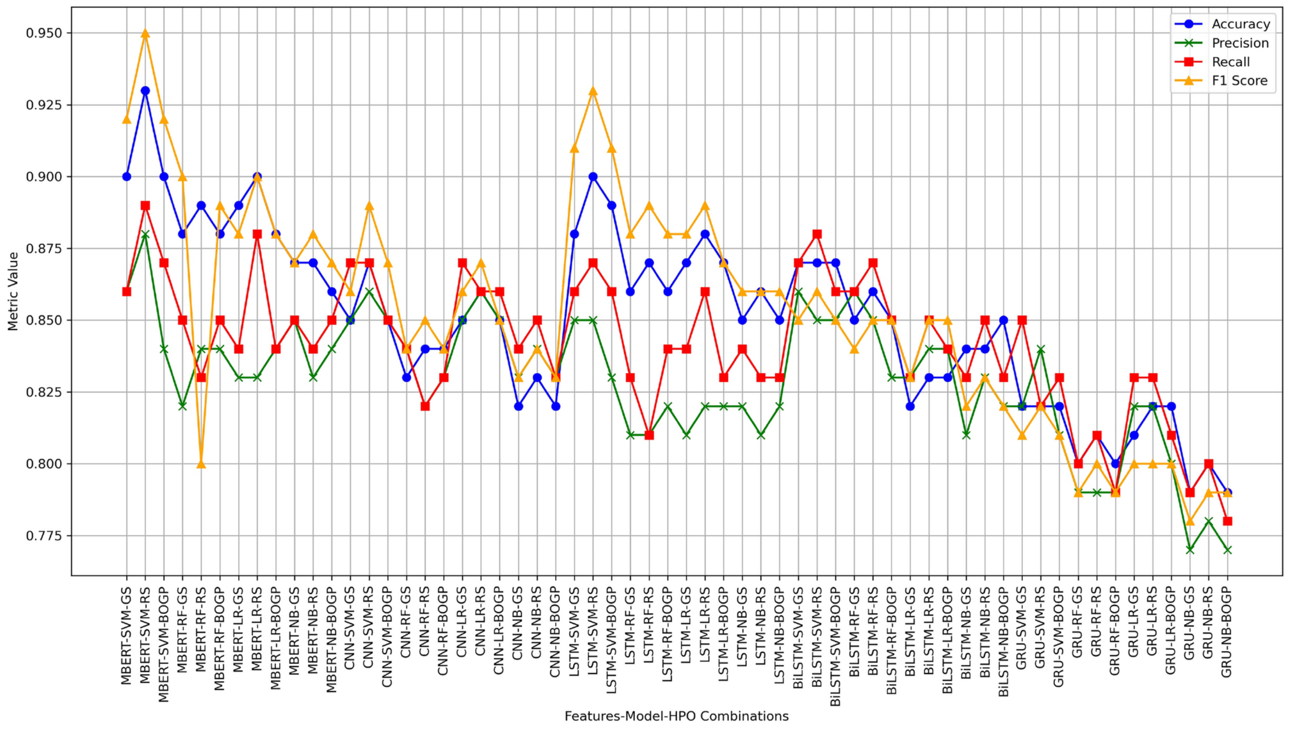

Fig. 3. Performance of hybrid models on Neutral-Hostile Task with HPO (HS-RU-20 corpus).

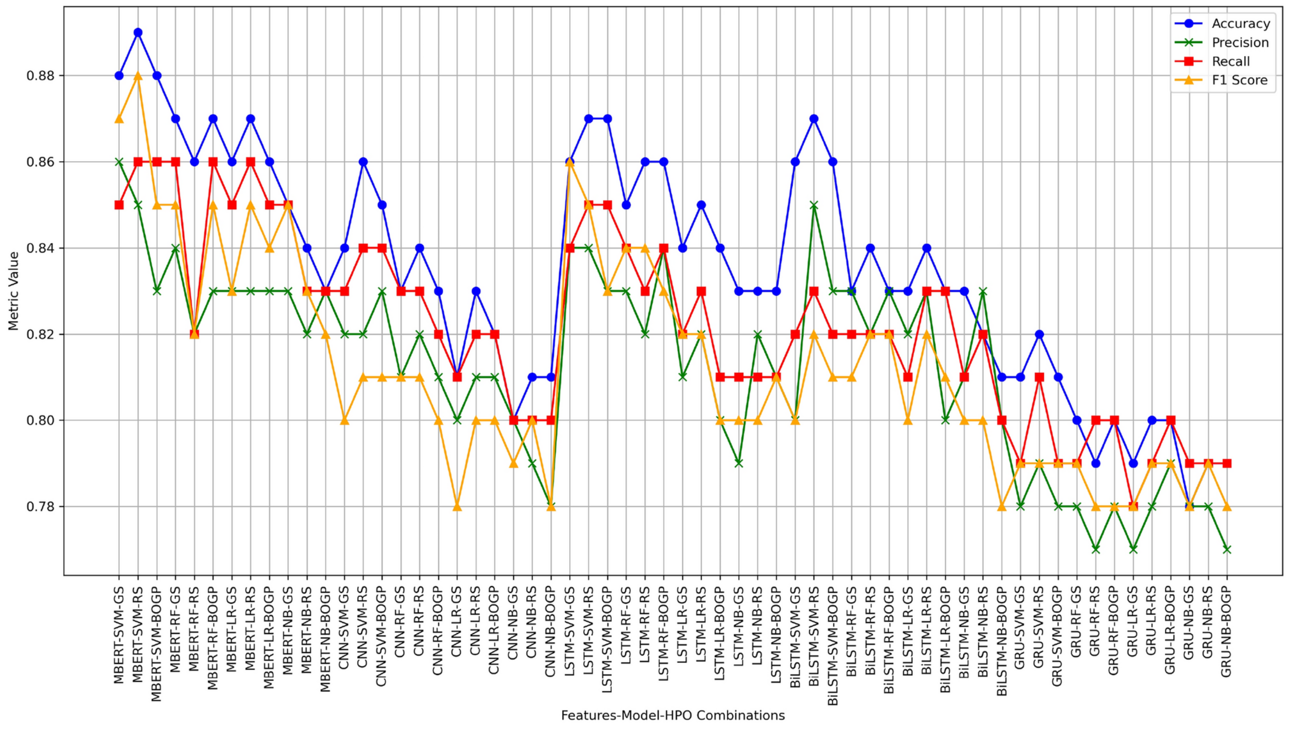

Fig. 4. Performance of hybrid models on Hate Speech-Offensive Task with HPO (HS-RU-20 Corpus).

led to better model performance. This suggests that RS is more suitable for complex feature learners like MBERT and LSTM.

For the neutral-hostile task, the obtained performance of models shows that the LSTM model produces the best results when used with the SVM classifier. Using the RS hyperparameter optimization, its performance is increased and it achieves the highest results with a 0.90 accuracy and a 0.93 F1 score. Similarly, for the hate speech-offensive task, the proposed approach shows the best results with LSTM and SVM and obtained state-of-the-art results with a 0.87 accuracy score and a 0.87 F1 score using RS optimization.

The overall results indicate that our experiments demonstrate the value of using various optimization techniques to improve the performance of machine learning algorithms. Notably, the performance of the SVM combined with the LSTM feature learner is greatly improved by the RS optimization method, demonstrating its remarkable significance. A more thorough investigation of the hyperparameter space is possible due to RS's capacity to randomly sample hyperparameters from predetermined ranges. Compared to the standard grid search optimization method, this random sampling method may uncover better hyperparameter combinations that result in enhanced model performance.

Regarding the performance of optimization methods in our study, the RS method consistently outperforms GS and BOGP, despite the fact that both of them show improvements over default parameter values. Since RS can efficiently explore the hyperparameter space and produce competitive results with fewer iterations than GS and BOGP, the RS technique is a viable alternative for hyperparameter optimization in the detection of hate speech.

The proposed hybrid model, along with hyperparameter optimization (HPO) methods, indicates excellent performance compared to previous literature results on the RU-HS-20 corpus[8]. The results obtained from our proposed model with HPO outperformed the baseline results and obtained state-of-the-art results for both sub-tasks. For the Neutral-Hostile Task, the baseline results achieved 0.84 accuracy and an F1 score of 0.90 using a count vectorizer with an LR classifier. In contrast, our hybrid model with MBERT-SVM and RS optimization achieved 0.93 accuracy and an F1 score of 0.95, clearly demonstrating the benefit of using advanced feature learners and optimization techniques.

For the Hate Speech-Offensive Task, the baseline reported 0.84 accuracy and an F1 score of 0.75, whereas our MBERT-SVM hybrid model achieved 0.89 accuracy and an F1 score of 0.88 after RS optimization. These results underscore the effectiveness of deep learning-based feature extraction over traditional methods such as the count vectorizer. The results reveal that MBERT as a feature learner, when combined with SVM and optimized with RS, consistently outperforms other models. The MBERT model's ability to capture rich multilingual embeddings plays a crucial role in enhancing feature extraction for Roman Urdu, which lacks standardized grammar and spelling. This highlights the advantage of using pre-trained transformer models for hate speech detection in resource-scarce languages.

Additionally, the use of HPOMs improved the performance of the machine learning classifiers. By choosing the optimal hyperparameters from the configuration space, the RS approach significantly increased overall performance. Notably, state-of-the-art results were obtained employing the MBERT feature learner with SVM classifier and RS optimization. Similarly, our proposed HPOMs surpassed the baseline results for the hate speech offense task. We found that HPOMs showed higher abilities to enhance the performance of machine learning classifiers in both sub-tasks of the HS-RU-20 corpus.

These results highlight the significance of the proposed hybrid approach for detecting hate speech. The classic count vectorizer, which depends on word frequency information, cannot match the capabilities of the MBERT feature learner to automatically learn the proper features from the text input. Furthermore, a hybrid approach, combining deep learning with traditional machine learning methods, improves the discrimination of hate speech by providing a more comprehensive representation of the data.

Additionally, the optimization techniques, particularly the randomized search (RS) method, play a significant role in optimizing the machine learning algorithms. The RS method's capacity to effectively investigate various hyperparameter combinations enables the identification of optimal settings, improving model performance.

In conclusion, this research introduces an innovative and efficient method for detecting hate speech in Roman Urdu text, surpassing other approaches in the literature. On the HS-RU-20 corpus, state-of-the-art results are obtained by combining MBERT feature learner, SVM classifier, and hyperparameter optimization through RS.

Results obtained using RUSHOLD corpus

In this work, we expanded the use of the proposed hybrid model and HPO methods to the RUHSOLD hate speech corpus, which is divided into two distinct tasks: coarse-grained (comprising two classes) and fine-grained (containing five classes). The main reason for using two different datasets for experiments is just the validation of the proposed methods. The results presented in Table 7 indicate the significance of the proposed approach in hate speech detection on the RUHSOLD corpus.

The results presented in Fig. 5 indicate the significance of our approach in hate speech detection on the RUHSOLD corpus. For the Coarse-grained Task, the results reveal that the transformer model MBERT feature learner combined with the SVM algorithm produced the highest performance, achieving an accuracy of 0.91 and an F1 score of 0.89. This performance is higher than the top-performing DL-based CNN-SVM combination, which achieved an accuracy of 0.90 and an F1 score of 0.89. This indicates that MBERT's capacity to capture rich contextual information from Roman Urdu text allows for more accurate coarse-grained classification compared to CNN.

For the Fine-grained Task, MBERT-SVM also outperformed all other models, achieving an accuracy of 0.82 and an F1 score of 0.79. This is again higher than the CNN-SVM combination, which reached an accuracy of 0.82 and F1 score of 0.79. MBERT's deep semantic understanding of Roman Urdu proves highly effective in distinguishing between more nuanced categories of hate speech in this fine-grained task. In contrast, the GRU feature learner paired with NB produced the lowest performance for both tasks, with an accuracy of 0.83 and an F1 score of 0.83 for the Coarse-grained Task, and an accuracy of 0.81 with an F1 score of 0.78 for the Fine-grained Task. This demonstrates the limitations of GRU in feature extraction for Roman Urdu compared to other models like CNN and MBERT.

		Coarse-grained Task				Fine-grained Task			
FEM	MLA	Accuracy	Precision	Recall	F1 score	Accuracy	Precision	Recall	F1 score
CNN	SVM	**0.90**	**0.85**	**0.87**	**0.89**	**0.82**	**0.80**	**0.83**	**0.79**
	RF	0.85	0.82	0.81	0.84	0.80	0.80	0.81	0.77
	LR	0.87	0.80	0.83	0.86	0.81	0.81	0.83	0.78
	NB	0.82	0.82	0.84	0.81	0.79	0.80	0.81	0.76
LSTM	SVM	0.88	0.82	0.84	0.87	0.80	0.80	0.81	0.78
	RF	0.83	0.85	0.84	0.82	0.78	0.80	0.82	0.78
	LR	0.84	0.84	0.85	0.83	0.79	0.81	0.82	0.77
	NB	0.81	0.81	0.83	0.80	0.80	0.81	0.87	0.75
BiLSTM	SVM	0.87	0.82	0.84	0.85	0.81	0.78	0.78	0.77
	RF	0.81	0.80	0.80	0.81	0.78	0.79	0.80	0.78
	LR	0.82	081	0.83	0.80	0.79	0.81	0.82	0.74
	NB	0.80	0.80	0.80	0.81	0.77	0.78	0.80	0.76
GRU	SVM	0.84	0.82	0.83	0.86	0.79	0.80	0.81	0.76
	RF	0.80	0.82	0.84	0.81	0.79	0.78	0.80	0.74
	LR	0.81	0.81	0.83	0.80	0.77	0.79	0.81	0.76
	NB	0.80	0.82	0.79	0.82	0.76	0.78	0.79	0.75

Table 7. Results obtained using hybrid models with default parameters on RUHSOLD corpus. Bold values shows the best scores for each evaluation metric.

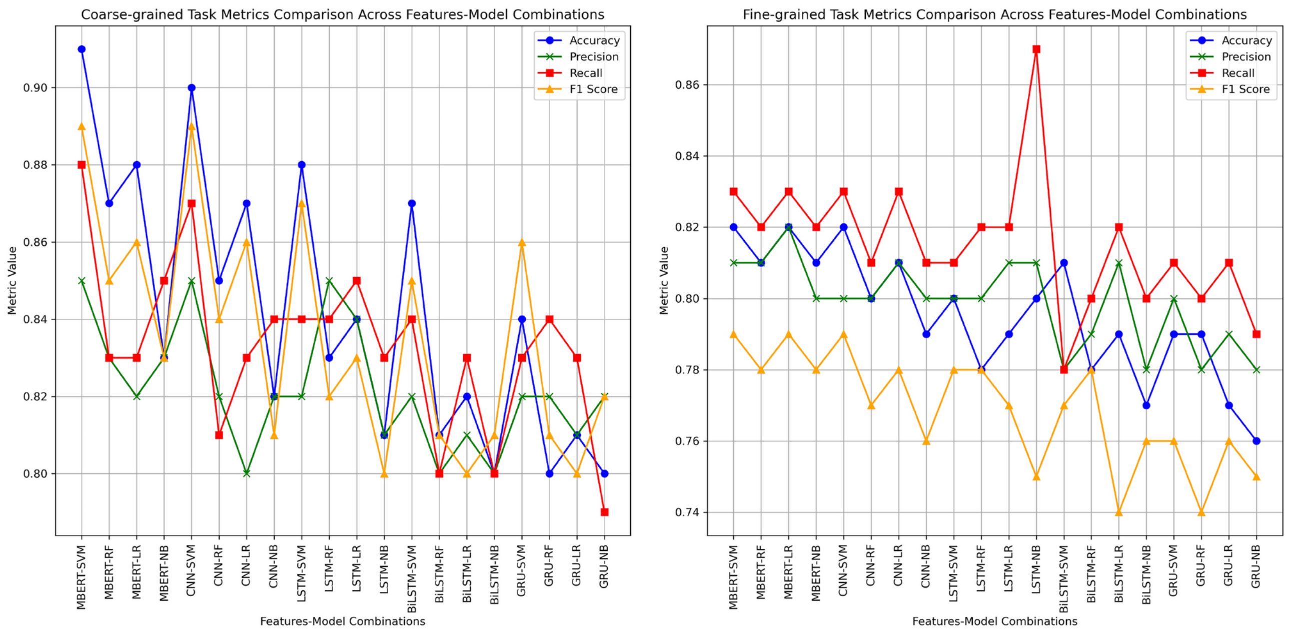

Fig. 5. Results obtained using hybrid models with default parameters on RUHSOLD corpus.

Experimental results on the RUHSOLD corpus clearly show that the transformer model MBERT feature learner, when combined with SVM, delivers the best performance for both Coarse-grained and Fine-grained tasks. MBERT's ability to capture multilingual representations proves particularly advantageous in Roman Urdu hate speech detection, surpassing even deep learning models, which had previously performed well in this domain.

A notable observation is that while deep learning CNN-SVM combinations perform well, the transformer-based MBERT-SVM combination further enhances classification accuracy due to MBERT's advanced feature extraction capabilities. MBERT captures deeper semantic and contextual information from Roman Urdu text, which is crucial for distinguishing between subtle variations in hate speech categories, especially in the fine-grained task.

Table 8 results of proposed Hybrid models with HPOMs on RUHSOLD corpus. Each ML algorithm's performance was further enhanced by the use of three HPO techniques. The results of proposed models with HPOMs on RUHSOLD corpus to optimize the performance of machine learning algorithms for Coarse-grained

			Coarse-grained Task				Fine-grained Task			
FEM	MLA	HPOMs	Accuracy	Precision	Recall	F1 score	Accuracy	Precision	Recall	F1 score
CNN	SVM	GS	0.91	0.87	0.88	0.90	0.83	0.80	0.85	0.80
		RS	**0.93**	**0.87**	**0.89**	**0.92**	**0.84**	**0.82**	**0.83**	**0.81**
		BOGP	0.90	0.89	0.90	0.91	0.82	0.81	0.85	0.80
	RF	GS	0.86	0.83	0.84	0.85	0.80	0.80	0.81	0.77
		RS	0.87	0.82	0.82	0.86	0.81	0.82	0.85	0.78
		BOGP	0.85	0.83	0.83	0.85	0.80	0.81	0.84	0.78
	LR	GS	0.87	0.80	0.83	0.87	0.82	0.83	0.84	0.79
		RS	0.88	0.83	0.85	0.89	0.83	0.81	0.82	0.79
		BOGP	0.87	0.82	0.84	0.86	0.81	0.79	0.80	0.80
	NB	GS	0.83	0.85	0.87	0.82	0.80	0.81	0.83	0.77
		RS	0.84	0.85	0.86	0.83	0.79	0.80	0.83	0.78
		BOGP	0.82	0.83	0.83	0.81	0.80	0.81	0.82	0.78
LSTM	SVM	GS	0.88	0.83	0.85	0.87	0.81	0.81	0.81	0.78
		RS	0.89	0.84	0.85	0.88	0.81	0.82	0.83	0.80
		BOGP	0.89	0.85	0.86	0.87	0.82	0.84	0.84	0.79
	RF	GS	0.84	0.85	0.84	0.82	0.78	0.80	0.82	0.78
		RS	0.85	0.81	0.85	0.84	0.79	0.82	0.83	0.79
		BOGP	0.85	0.82	0.86	0.83	0.78	0.81	0.81	0.78
	LR	GS	0.85	0.84	0.85	0.84	0.79	0.81	0.82	0.77
		RS	0.86	0.82	0.83	0.85	0.79	0.81	0.83	0.78
		BOGP	0.84	0.82	0.86	0.84	0.80	0.82	0.81	0.77
	NB	GS	0.82	0.81	0.83	0.80	0.80	0.82	0.86	0.75
		RS	0.83	0.80	0.81	0.82	0.81	0.83	0.85	0.76
		BOGP	0.82	0.81	0.82	0.81	0.80	0.81	0.87	0.76
BiLSTM	SVM	GS	0.87	0.82	0.84	0.86	0.81	0.79	0.80	0.78
		RS	0.89	0.81	0.83	0.87	0.82	0.80	0.81	0.79
		BOGP	0.88	0.84	0.84	0.86	0.82	0.79	0.80	0.78
	RF	GS	0.82	0.81	0.80	0.81	0.79	0.80	0.80	0.78
		RS	0.84	0.83	0.85	0.83	0.80	0.80	0.81	0.80
		BOGP	0.82	0.82	0.82	0.82	0.80	0.81	0.81	0.79
	LR	GS	0.83	0.81	0.83	0.80	0.79	0.82	0.82	0.74
		RS	0.85	0.83	0.84	0.83	0.80	0.80	0.83	0.76
		BOGP	0.85	0.81	0.83	0.81	0.79	0.82	0.84	0.75
	NB	GS	0.81	0.81	0.82	0.81	0.77	0.80	0.80	0.76
		RS	0.83	0.82	0.83	0.83	0.79	0.81	0.81	0.78
		BOGP	0.81	0.84	0.85	0.82	0.78	0.81	0.83	0.77
GRU	SVM	GS	0.84	0.83	0.85	0.86	0.79	0.80	0.81	0.76
		RS	0.86	0.82	0.84	0.87	0.80	0.81	0.82	0.77
		BOGP	0.84	0.81	0.84	0.87	0.80	0.82	0.84	0.76
	RF	GS	0.81	0.83	0.84	0.81	0.79	0.79	0.81	0.74
		RS	0.82	0.84	0.87	0.83	0.80	0.79	0.82	0.75
		BOGP	0.81	0.83	0.85	0.82	0.79	0.80	0.81	0.74
	LR	GS	0.81	0.83	0.84	0.80	0.77	0.79	0.81	0.76
		RS	0.83	0.80	0.85	0.82	0.79	0.80	0.82	0.77
		BOGP	0.82	0.81	0.82	0.81	0.78	0.79	0.80	0.76
	NB	GS	0.80	0.83	0.83	0.82	0.77	0.79	0.81	0.75
		RS	0.82	0.84	0.85	0.83	0.78	0.80	0.81	0.77
		BOGP	0.81	0.82	0.83	0.82	0.77	0.79	0.80	0.76
BERT + CNN-gram	-	-	0.90	0.90	0.90	0.90	0.82	0.75	0.74	0.75
(Baseline)[23]										

Table 8. Results obtained using hybrid models with HPOMs on RUSHOLD corpus. Bold values shows the best scores for each evaluation metric.

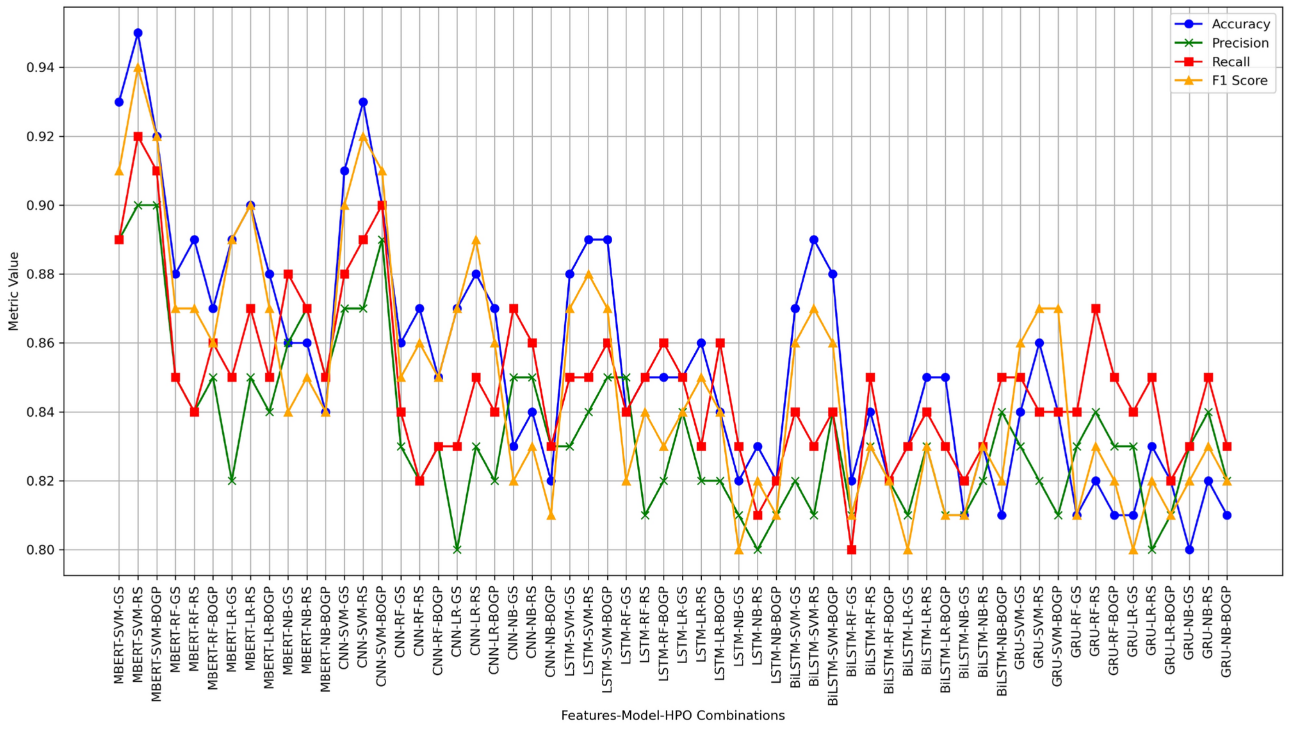

Fig. 6. Performance of hybrid models on Coarse-grained Task with HPO (RUSHOLD Corpus).

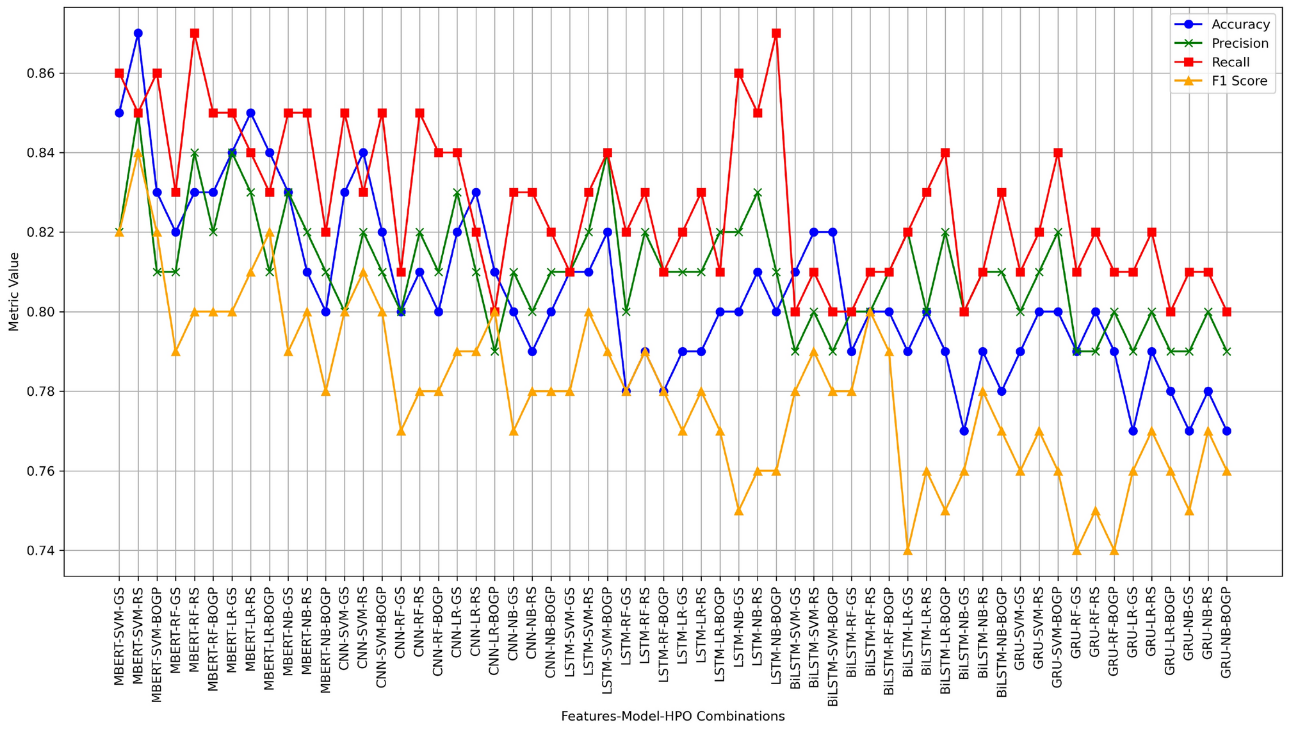

Fig. 7. Performance of hybrid models on Fine-grained Task with HPO (RUSHOLD Corpus).

task and Fine-grained task are presented in Figs. 6 and 7 respectively. The results indicate that the performance of each machine learning is improved using different HPOMs. For the Coarse-grained Task, the MBERT-SVM combination using RS optimization achieved state-of-the-art results with an accuracy of 0.95 and an F1 score of 0.94, significantly surpassing other combinations. For the Fine-grained Task, MBERT-SVM with RS optimization also delivered the highest performance, with an accuracy of 0.87 and an F1 score of 0.84. These results validate the effectiveness of the hybrid approach and demonstrate how hyperparameter optimization can enhance the performance of machine learning models in hate speech detection.

Regarding DL model combinations, for the Coarse-grained task, the CNN model with SVM classifier achieved the best results using the RS hyperparameter optimization method, obtaining an accuracy of 0.93 and an F1 score of 0.92 with optimized parameters ('kernel': 'linear', 'C': 10). Similarly, for the Fine-grained task, the CNN feature learner with SVM attained outstanding results using RS optimization, achieving an accuracy of 0.84 and an F1 score of 0.81 with the same optimized parameters.

The results from the RUHSOLD corpus clearly indicate that MBERT combined with SVM and optimized using RS hyperparameter tuning delivers the best performance for both tasks. The superior performance of MBERT is attributed to its ability to capture rich, context-aware embeddings that are well-suited for Roman Urdu hate speech detection. Meanwhile, the RS optimization technique proved to be the most effective method for improving model performance by identifying optimal hyperparameter configurations that significantly enhance classification accuracy and F1 scores. By surpassing baseline results from previous studies[23], the proposed hybrid models demonstrate the advantage of integrating deep learning feature learners with machine learning classifiers.

As a result of using both the HS-RU-20 and RUHSOLD corpora, the current research demonstrates the efficacy of the proposed models and hyperparameter optimization techniques in Urdu HSD for Roman Urdu text. The MBERT feature learner, combined with the SVM classifier and RS hyperparameter optimization, leads to state-of-the-art results on both corpora, outperforming preceding baseline techniques, presented in Fig. 8. The findings clearly indicate that the selection of the feature learner significantly impacts the model's

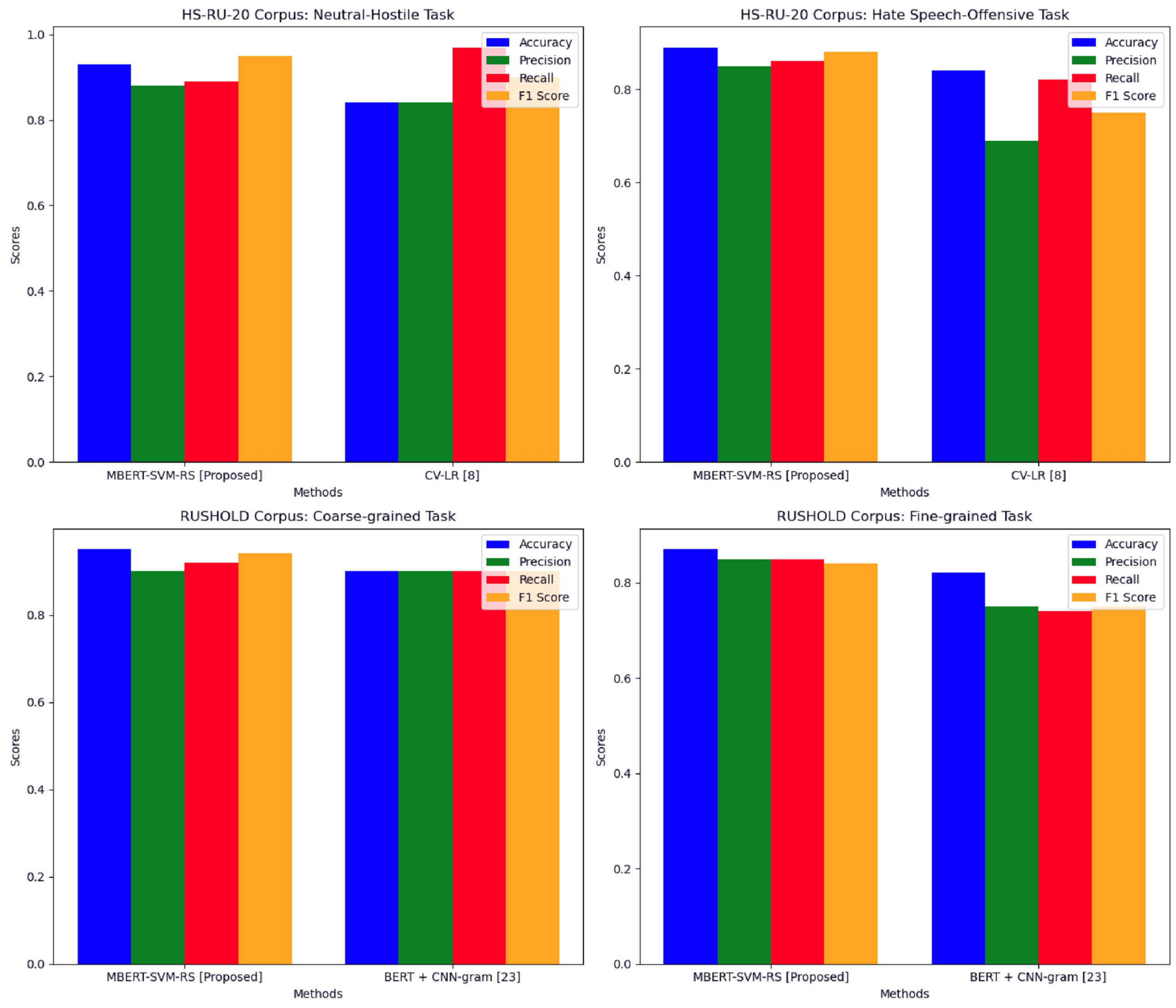

Fig. 8. Performance comparison of proposed and existing methods on both HS-RU-20 and RUSHOLD Corpus Corpus).

performance. However, despite the variability in feature learner performance, the SVM classifier consistently produced excellent results across both datasets, highlighting its robustness in text classification tasks.

The MBERT as a feature learner further enhanced model performance. On both datasets, the MBERT-SVM combination outperformed other models, with RS-optimized MBERT achieving a 0.95 accuracy and 0.94 F1 score on the RUHSOLD Coarse-grained Task, and a 0.87 accuracy and 0.84 F1 score on the Fine-grained Task. This demonstrates MBERT's strength in generating contextually rich embeddings for Roman Urdu text, which significantly improves classification outcomes.

The benefit of using word embeddings, specifically GloVe embeddings and MBERT embeddings, is evident through the improved performance of the models. GloVe embeddings, used with deep learning models, provided rich, dense vector representations of words, enabling the models to capture semantic relationships and improve classification accuracy. This was noticeable in the CNN-SVM, and LSTM-SVM combination, where the use of GloVe embeddings contributed to the performance of both datasets. On the other hand, MBERT embeddings offered a more contextualized understanding of Roman Urdu text, leading to state-of-the-art results when combined with SVM. MBERT-SVM, with its ability to capture deep, multilingual semantic features, achieved the highest performance, particularly on the RUHSOLD dataset, with an accuracy of 0.95 on the Coarse-grained Task and 0.87 on the Fine-grained Task, demonstrating the value of using advanced embeddings in hate speech detection.

Based on these results, it is evident that HPO methods, particularly RS, have greatly enhanced the performance of machine learning algorithms .RS consistently delivered the best outcomes for sub-tasks in both datasets, outperforming GS and BOGP. The effectiveness of RS lies in its ability to efficiently explore the hyperparameter space by testing randomly sampled values, which often leads to optimal configurations that might be missed by methods that exhaustively search the entire space, such as GS.

Our findings highlight the importance of selecting appropriate feature learners and applying hyperparameter optimization to improve model performance. The MBERT-SVM combination with RS optimization sets a new benchmark for hate speech detection in Roman Urdu. These results contribute to advancing research in hate speech detection for low-resource languages and offer a solid foundation for future work in this area.

Conclusion and future work

This study presents a novel hybrid model utilizing multilingual BERT (MBERT) transformer and deep learning models CNN, LSTM, BiLSTM, and GRU as an automatic feature extractor combined with machine learning classifiers for Roman Urdu hate speech detection. The MBERT-SVM combination, enhanced through random search (RS) hyperparameter optimization, named MBERT-SVM-RS, achieves state-of-the-art results on both benchmark corpora, HS-RU-20 and RUHSOLD. The model consistently outperformed other approaches, achieving an accuracy of 0.95 and an F1 score of 0.94 on the RUHSOLD Coarse-grained Task, and an accuracy of 0.87 and an F1 score of 0.84 on the Fine-grained Task. The superior performance of MBERT can be attributed to its ability to generate rich, contextual embeddings that capture the complexities of Roman Urdu text, making it highly effective for hate speech detection. The RS optimization method further enhances performance by efficiently searching the hyperparameter space, leading to optimal configurations that improve model accuracy. These results demonstrate that the proposed MBERT-SVM-RS hybrid model provides substantial improvements over traditional feature extraction methods. In the future, we plan to explore additional embeddings, such as BERT-based variants, FastText, and Word2Vec, to further enhance the model's robustness and effectiveness in hate speech detection.

Availability of data and materials

"The datasets used and/or analysed during the current study available from the corresponding author on reasonable request."

Received: 14 June 2024; Accepted: 6 November 2024
Published online: 19 November 2024

References

1. Akhter, M. P., Jiangbin, Z., Naqvi, I. R., Abdelmajeed, M. & Sadiq, M. T. Automatic detection of offensive language for urdu and roman urdu. *IEEE Access* **8**, 91213–91226 (2020).
2. Djuric, N., Zhou, J., Morris, R., Grbovic, M., Radosavljevic, V., & Bhamidipati, N. Hate speech detection with comment embeddings. In: Proceedings of the 24th International Conference on World Wide Web, pp. 29–30 (2015)
3. Noor, F., Bakhtyar, M., & Baber, J. Sentiment analysis in e-commerce using svm on roman urdu text. In: International Conference for Emerging Technologies in Computing, pp. 213–222 (2019). Springer
4. Statista: Number of monthly active Twitter users worldwide from 1st quarter 2010 to 1st quarter 2019. https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/ (2022)
5. Ethnologue: What are the top 200 most spoken languages? https://www.ethnologue.com/guides/ethnologue200 (2022)
6. Worldometer: Pakistan population. https://www.worldometers.info/world-population/pakistan-population/ (2022)
7. Daud, A., Khan, W. & Che, D. Urdu language processing: a survey. *Artificial Intelligence Review* **47**(3), 279–311 (2017).
8. Khan, M. M., Shahzad, K. & Malik, M. K. Hate speech detection in roman urdu. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* **20**(1), 1–19 (2021).
9. 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, pp. 88–93 (2016)
10. De Gibert, O., Perez, N., García-Pablos, A., & Cuadros, M. Hate speech dataset from a white supremacy forum. arXiv preprint arXiv:1809.04444 (2018)
11. García-Díaz, J. A., Jiménez-Zafra, S. M., García-Cumbreras, M. A., & Valencia-García, R. Evaluating feature combination strategies for hate-speech detection in spanish using linguistic features and transformers. Complex & Intelligent Systems, 1–22 (2022)

12. Plaza-del-Arco, F. M., Molina-González, M. D., Urena-López, L. A. & Martín-Valdivia, M. T. Comparing pre-trained language models for spanish hate speech detection. *Expert Systems with Applications* **166**, 114120 (2021).
13. Almatarneh, S., Gamallo, P., Pena, F. J. R., & Alexeev, A. Supervised classifiers to identify hate speech on english and spanish tweets. In: International Conference on Asian Digital Libraries, pp. 23–30 (2019). Springer
14. Romim, N., Ahmed, M., Talukder, H., & Islam, S. Hate speech detection in the bengali language: A dataset and its baseline evaluation. In: Proceedings of International Joint Conference on Advances in Computational Intelligence, pp. 457–468 (2021). Springer
15. Karim, M. R., Dey, S. K., Islam, T., Sarker, S., Menon, M. H., Hossain, K., Hossain, M. A., & Decker, S. Deephateexplainer: Explainable hate speech detection in under-resourced bengali language. In: 2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA), pp. 1–10 (2021). IEEE
16. Ishmam, A. M., & Sharmin, S. Hateful speech detection in public facebook pages for the bengali language. In: 2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 555–560 (2019). IEEE
17. Aldjanabi, W., Dahou, A., Al-qaness, M. A., Elaziz, M. A., Helmi, A. M., & Damaševičius, R. Arabic offensive and hate speech detection using a cross-corpora multi-task learning model. In: Informatics, vol. 8, p. 69 (2021). MDPI
18. Duwairi, R., Hayajneh, A. & Quwaider, M. A deep learning framework for automatic detection of hate speech embedded in arabic tweets. *Arabian Journal for Science and Engineering* **46**(4), 4001–4014 (2021).
19. Fauzi, M. A. & Yuniarti, A. Ensemble method for indonesian twitter hate speech detection. *Indonesian Journal of Electrical Engineering and Computer Science* **11**(1), 294–299 (2018).
20. Putri, S. D. A., Ibrohim, M. O., & Budi, I. Abusive language and hate speech detection for indonesian-local language in social media text. In: International Conference on Computing and Information Technology, pp. 88–98 (2021). Springer
21. Alfina, I., Mulia, R., Fanany, M. I., & Ekanata, Y. Hate speech detection in the indonesian language: A dataset and preliminary study. In: 2017 International Conference on Advanced Computer Science and Information Systems (ICACSIS), pp. 233–238 (2017). IEEE
22. Ali, R., Farooq, U., Arshad, U., Shahzad, W. & Beg, M. O. Hate speech detection on twitter using transfer learning. *Computer Speech & Language* **74**, 101365 (2022).
23. Rizwan, H., Shakeel, M. H., & Karim, A. Hate-speech and offensive language detection in roman urdu. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 2512–2522 (2020)
24. Malmasi, S., & Zampieri, M. Detecting hate speech in social media. arXiv preprint arXiv:1712.06427 (2017)
25. Santucci, V., Spina, S., Milani, A., Biondi, G., & Di Bari, G. Detecting hate speech for italian language in social media. In: EVALITA 2018, Co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), vol. 2263 (2018)
26. Davidson, T., Warmsley, D., Macy, M., & Weber, I. Automated hate speech detection and the problem of offensive language. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 11, pp. 512–515 (2017)
27. Rizoiu, M. -A., Wang, T., Ferraro, G., & Suominen, H. Transfer learning for hate speech detection in social media. arXiv preprint arXiv:1906.03829 (2019)
28. Waseem, Z. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In: Proceedings of the First Workshop on NLP and Computational Social Science, pp. 138–142 (2016)
29. Mehmood, F. et al. Passion-net: a robust precise and explainable predictor for hate speech detection in roman urdu text. *Neural Computing and Applications* **36**(6), 3077–3100 (2024).
30. Nasir, S., Seerat, A., & Wasim, M. Hate speech detection in roman urdu using machine learning techniques. In: 2024 5th International Conference on Advancements in Computational Sciences (ICACS), pp. 1–7 (2024). IEEE
31. Maqbool, F., Spahiu, B., & Maurino, A., et al. Impact of data augmentation on hate speech detection in roman urdu (2024)
32. Malik, M. S. I., Nawaz, A., & Jamjoom, M. M. Hate speech and target community detection in nastaliq urdu using transfer learning techniques. IEEE Access (2024)
33. Atif, A., Zafar, A., Wasim, M., Waheed, T., Ali, A., Ali, H., & Shah, Z. Cyberbullying detection and abuser profile identification on social media for roman urdu. IEEE Access (2024)
34. Ullah, F., Zamir, M., Arif, M., Ahmad, M., Felipe-Riveron, E., & Gelbukh, A. Fida@ dravidianlangtech 2024: A novel approach to hate speech detection using distilbert-base-multilingual-cased. In: Proceedings of the Fourth Workshop on Speech, Vision, and Language Technologies for Dravidian Languages, pp. 85–90 (2024)
35. Jahangir, M.T., Ahmad, M., & Rehman, H. Efficient intelligent system for cyberbullying detection in english and roman urdu social media posts. Journal of Computing & Biomedical Informatics (2024)
36. Khan, L., Amjad, A., Ashraf, N., Chang, H.-T. & Gelbukh, A. Urdu sentiment analysis with deep learning methods. *IEEE access* **9**, 97803–97812 (2021).
37. Al Maruf, A. et al. Hate speech detection in the bengali language: a comprehensive survey. *Journal of Big Data* **11**(1), 97 (2024).
38. Khan, A., Ahmed, A., Jan, S., Bilal, M., & Zuhairi, M. F. Abusive language detection in urdu text: Leveraging deep learning and attention mechanism. IEEE Access (2024)
39. Razi, F., & Ejaz, N. Multilingual detection of cyberbullying in mixed urdu, roman urdu, and english social media conversations. IEEE Access (2024)
40. Khan, L., Amjad, A., Ashraf, N. & Chang, H.-T. Multi-class sentiment analysis of urdu text using multilingual bert. *Scientific Reports* **12**(1), 5436 (2022).
41. Amjad, A., Khan, L. & Chang, H.-T. Data augmentation and deep neural networks for the classification of pakistani racial speakers recognition. *PeerJ Computer Science* **8**, 1053 (2022).
42. Bade, G., Kolesnikova, O., Sidorov, G., & Oropeza, J. Social media hate and offensive speech detection using machine learning method. In: Proceedings of the Fourth Workshop on Speech, Vision, and Language Technologies for Dravidian Languages, pp. 240–244 (2024)
43. Gandhi, A., Ahir, P., Adhvaryu, K., Shah, P., Lohiya, R., Cambria, E., Poria, S., & Hussain, A. Hate speech detection: A comprehensive review of recent works. Expert Systems, 13562 (2024)
44. Hashmi, E., Yayilgan, S.Y., Hameed, I.A., Yamin, M.M., Ullah, M., & Abomhara, M. Enhancing multilingual hate speech detection: From language-specific insights to cross-linguistic integration. IEEE Access (2024)
45. Khan, L., Amjad, A., Afaq, K. M. & Chang, H.-T. Deep sentiment analysis using cnn-lstm architecture of english and roman urdu text shared in social media. *Applied Sciences* **12**(5), 2694 (2022).
46. Ashraf, N. et al. Multi-label emotion classification of urdu tweets. *PeerJ Computer Science* **8**, 896 (2022).
47. Zhang, L., Wang, S. & Liu, B. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **8**(4), 1253 (2018).
48. Young, T., Hazarika, D., Poria, S., & Cambria, E. Recent trends in deep learning based natural language processing. ieee Computational intelligenCe magazine 13(3), 55–75 (2018)
49. Song, P., Geng, C., & Li, Z. Research on text classification based on convolutional neural network. In: 2019 International Conference on Computer Network, Electronic and Automation (ICCNEA), pp. 229–232 (2019). IEEE
50. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural computation* **9**(8), 1735–1780 (1997).
51. Luan, Y., & Lin, S. Research on text classification based on cnn and lstm. In: 2019 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), pp. 352–355 (2019). IEEE
52. Isnain, A. R., Sihabuddin, A. & Suyanto, Y. Bidirectional long short term memory method and word2vec extraction approach for hate speech detection. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)* **14**(2), 169–178 (2020).

53. Dey, R., & Salem, F. M. Gate-variants of gated recurrent unit (gru) neural networks. In: 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1597–1600 (2017). IEEE
54. Muslim, M. A. Support vector machine (svm) optimization using grid search and unigram to improve e-commerce review accuracy. *Journal of Soft Computing Exploration* **1**(1), 8–15 (2020).
55. Omotehinwa, T. O. & Oyewola, D. O. Hyperparameter optimization of ensemble models for spam email detection. *Applied Sciences* **13**(3), 1971 (2023).
56. Alzanin, S. M., Azmi, A. M. & Aboalsamh, H. A. Short text classification for arabic social media tweets. *Journal of King Saud University-Computer and Information Sciences* **34**(9), 6595–6604 (2022).
57. Valarmathi, R. & Sheela, T. Heart disease prediction using hyper parameter optimization (hpo) tuning. *Biomedical Signal Processing and Control* **70**, 103033 (2021).
58. Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. Algorithms for hyper-parameter optimization. Advances in neural information processing systems **24** (2011)
59. Bergstra, J., & Bengio, Y. Random search for hyper-parameter optimization. Journal of machine learning research **13**(2) (2012)
60. Eggensperger, K., Feurer, M., Hutter, F., Bergstra, J., Snoek, J., Hoos, H., & Leyton-Brown, K. Towards an empirical foundation for assessing bayesian optimization of hyperparameters. In: NIPS Workshop on Bayesian Optimization in Theory and Practice, vol. 10 (2013)
61. Rasmussen, C. E. Gaussian processes in machine learning. In: Summer School on Machine Learning, pp. 63–71 (2003). Springer
62. Abid, F., Alam, M., Yasir, M. & Li, C. Sentiment analysis through recurrent variants latterly on convolutional neural network of twitter. *Future Generation Computer Systems* **95**, 292–308 (2019).
63. Pennington, J., Socher, R., & Manning, C. D. Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)

Author contributions
W.A. conceptualization, data curation and writing - original manuscript. S.K. conceptualization, formal analysis and writing - original manuscript. A.R. methodology, formal analysis, data curation and writing - original manuscript. M.W. software, methodology, and project administration. T.K. visualization, software, and investigation. E.C.M. investigation, funding acquisition and project administration. A.B.A. validation, resources and visualization. I.A. supervision, validation and writing - review & edit. All authors reviewed the manuscript and approved it.

Funding
This research is funded by the European University of Atlantic.

Declarations

Conflict of Interest
“The authors declare no conflict of interests.”

Ethics approval and Consent to participate
“Not applicable.”

Consent for publication
“Not applicable.”

Code availability
“Not applicable.”

Additional information
Correspondence and requests for materials should be addressed to T.K. or I.A.

Reprints and permissions information is available at www.nature.com/reprints.