Proceedings of the Pakistan Academy of Sciences: A Physical and Computational Sciences 62(4): 277-288 (2025)
Copyright © Pakistan Academy of Sciences
ISSN (Print): 2518-4245; ISSN (Online): 2518-4253
http://doi.org/10.53560/PPASA(62-4)845

Pakistan Academy of Sciences

Research Article

Improving Roman Urdu Topic Classification through Custom Stemming and an SGD-Optimized Machine Learning Pipeline

Muhammad Aqeel[1], Irfan Qutab[2], Khawar Iqbal[3*], Habiba Fiaz[4], and Hira Arooj[5]

[1]School of Software, Northwestern Polytechnical University, Xi'an, China
[2]Department of Engineering, University of Modena and Reggio Emilia, Modena, Italy
[3]Riphah School of Computing and Innovation, Riphah International University, Lahore, Pakistan
[4]School of Mathematics and Statistics, Northwestern Polytechnical University, Xi'an, China
[5]Department of Mathematics and Statistics, The University of Lahore, Sargodha, Pakistan

Abstract: All over social media and internet platforms, Roman Urdu content is extremely casual, inconsistent, and linguistically diversified, which makes it hard to interpret through conventional Natural Language Processing (NLP) techniques. This paper proposes a strong topic-classification framework for Roman Urdu, integrating Stochastic Gradient Descent (SGD)-optimized machine learning, dictionary-assisted stemming, and custom lexical normalization in order to overcome those challenges. The method consists of structured preprocessing, reduction of repeated letters, rule-based normalization, extraction of TF-IDF features, and the evaluation of a few classifiers including Logistic Regression (LR), Support Vector Machine (SVM), Naïve Bayes (NB), Decision Tree (DT), K-Nearest Neighbors (KNN), along with the proposed model of SGD. The proposed classifier outperformed all the baseline models with an accuracy of 95%, according to the experimental results on the four-class dataset comprised of Politics, Sports, Education, and Religion. The results depict the importance of stemming and normalization to improve feature quality and reduce orthographic variability in low-resource languages. All things considered, this study provides a repeatable and efficient pipeline for Roman Urdu subject classification and thus lays a concrete foundation for further Roman Urdu NLP research.

Keywords: Roman Urdu Stemmer, TF-IDF, Stochastic Gradient Descent, Topic Classification, Machine Learning.

1. INTRODUCTION

Topic classification using Natural Language Processing (NLP) is a major application, where machines classify texts into predefined categories. Topic classification refers to classifying a document into predefined topics such as social media, news, or reviews. Efficient topic classification systems for multiple languages are becoming more important with the rapid increase in online contents, especially social media contents. Large Language Models [1] or deep learning models for specialized domains [2] are some of the recent advancements that were taken into consideration. Efficient topic classification systems for multiple languages are becoming increasingly important with the rapid growth in online contents, especially social media contents. In South Asia, Roman Urdu which is a form of Urdu written in Latin script is frequently practiced. Roman Urdu undergoes an informal language with limited resources, regardless of its increasing popularity, which leads to substantial challenges for automated text classification [3]. By formulating a high accuracy topic classification system particularly for Roman Urdu, integrating its lexical variation and morphological irregularities, this study aims to address these shortcomings. Roman Urdu is used in a significant portion of South Asian discussion forums because Urdu is one of the languages that are most frequently used in the world [4]. Roman Urdu's lack of standard orthographic structures and a more informal atmosphere of social

Received: December 2023; Revised: November 2025; Accepted: December 2025
* Corresponding Author: Khawar Iqbal <khawar.iqbal@riphah.edu.pk>

media have contributed to the growing number of non-standard spellings, which makes automated text categorization far more challenging [5, 6]. For that reason, it is vital to build such tools that can arrange and classify this massive amount of user-generated content for improved information access and interpretation.

Roman Urdu has received little attention in recent studies, which mainly focused on text classification for high resource languages like English. Techniques using deep learning for text classification have been previously investigated by Minaee *et al.* [7]. These techniques perform well in settings where resources are abundant, but they show limitations when applied to languages with limited resources such as Roman Urdu. In this regard, Gasparetto *et al.* [8] studied algorithms for text categorization and also demonstrated how hard it can be to apply these approaches to unstructured and informal texts such as Roman Urdu. While TF-IDF (Term Frequency-Inverse Document Frequency) is an established feature extraction method [9], it has yet to be studied extensively on Roman Urdu due to the presence of nonstandard spelling and irregular forms in the language that render such methods very difficult to apply. Similarly, Hussain *et al.* [10] carried out a detailed study on Roman Urdu sentiment detection but did not present any preprocessing mechanism, which is considered crucial in topic classification. Similarly, the study carried out by Arshad *et al.* [11] on the recognition of emotions in Roman Urdu text failed to consider the specific preprocessing requirements of the language.

Although, Pakray *et al.* [12] focused on low resource language processing, issues related to Roman Urdu were not sufficiently focused on, where its informal expressions and spelling irregularities make classification a highly challenging job. As far as stemming is concerned, although it has been well studied for languages like English, it does not suffice to handle Roman Urdu, and an efficient stemmer for Roman Urdu remains missing. Adimulam *et al.* [13] focused on transfer learning in languages with very minimal resources. However, the unique morphological constraints pertaining to Roman Urdu were not clearly explored in this work. Avetisyan and Broneske [14] made an effort to review low resource languages but did not provide any customized solution for Roman Urdu, which further gives weight to the importance of effective preprocessing. Similarly, Ògúnremí *et al.* [15], while discussing decolonizing NLP for low resource languages, did not explore those very unique complexities existing in Roman Urdu text.

While the studies of Sandu *et al.* [16] and Chen *et al.* [17] focused on text extraction techniques for social media, they did not cater specifically to Roman Urdu but rather focused their approach on strongly resourced languages. Ghafoor *et al.* [18] studied multilingual text processing, but again, their work did not cover methods that could cater to the rich lexical features of Roman Urdu. Even though TF-IDF is a widespread feature extraction technique, it needs further tuning to deal with informal writing patterns of Roman Urdu. Kumar *et al.* [19] assessed deep learning for hyperspectral image classification, failing to assess the challenge of text classification for low-resourced languages like Roman Urdu. Additionally, Faheem *et al.* [20] investigated part of speech tagging for Roman Urdu but did not expand their work to topic classification and Hussain *et al.* [10] addressed the challenges of emotion recognition in Roman Urdu; however, their work did not discuss topic categorization, which considers a broader perspective of Roman Urdu textual characteristics.

Roman Urdu text categorization has drawn more interest, especially in view of complications linked with the detection of sentiment and emotions. The work of Ilyas *et al.* [21] identified the recognition of emotions in code mixed Roman Urdu-English text, their research has avoided specific challenges that arise when dealing with pure Roman Urdu text, such as the irregular spelling and lack of standardization of the language.

In the same direction, Chandio *et al.* [22] have proposed an attention-driven Residual Unit–Bidirectional LSTM (RU-BiLSTM) framework for sentiment analysis targeting Roman Urdu, but they failed to take into account carefully the difficulty of the topic classification, opening a way to deal with a greater variety of textual structures. Nabeel *et al.* [23] used machine learning (ML) models to classify emotions in Roman Urdu posts but the struggles of classifying topics within this language context were not taken into account by them. Khan *et al.* [24] worked on the sentiment analysis for Roman Urdu from a multilingual point of view, they predominantly focused on emotion identification, leaving a gap in the establishment of broader topic classification systems. More generalized issue of topic categorization, which has not yet explored, was also ignored by Rana *et al.* [25], who contributed in the area of Roman Urdu language by offering an unsupervised method for analysis of sentiments on social media short text classification.

Tejaswini *et al.* [26] examined social media text interpretation using NLP methods and hybrid deep learning models for detecting depression, and the work of Lavanya and Sasikala [27] explored text classification in social healthcare settings using NLP and deep learning, both of these studies mainly relied on sentiment analysis and did not address the specific challenges of topic classification, which is the focus of our work. The need for improved approaches to Roman Urdu text processing becomes clear when considering that Akhter *et al.* [28] focused on identifying abusive language in both Urdu and Roman Urdu but did not extend the analysis to topic categorization. Similarly, Mehmood *et al.* [29] proposed a discriminant approach for feature spamming and played their role in the analysis of sentiment for Roman Urdu; however, their research work did not incorporate topic classification.

Mehmood *et al.* [30] used a hybrid approach for sentiment analysis of Roman Urdu through the Xtreme multi-channel technique. However, their work still had some shortcomings since it missed the aspect of topic classification. Saeed *et al.* [31] worked on the area of toxic comment classification for Urdu and Roman Urdu by developing the PURUTT corpus, which aimed at enhancing the detection of toxic comments. However, their work does not tackle the key issue of topic classification.

In conclusion, despite some progress made in sentiment analysis and toxic comment detection for Roman Urdu-Urdu, there is still a gap in the application of such techniques to topic classifications. Feature extraction techniques such as TF-IDF and n-gram techniques have gained considerable attention, however, issues such as non-standard spelling, colloquial language use, and small datasets still exist. Therefore, the proposed study strengthens the Stochastic Gradient Descent (SGD) by developing a more accurate topic classification technique and a Roman Urdu stemmer.

2. MATERIALS AND METHODS

Roman Urdu stemming and a vast amount of ML experiments form the basis of this study's methodology. Logistic Regression (LR) [9], Support Vector Machine (SVM) [30], SGD, K-Nearest Neighbors (KNN), Naïve Bayes (NB) and Decision Tree (DT) [32] were among the algorithms whose performances we assessed. The establishment of a method for Roman Urdu text topic classification using SGD is a major accomplishment of this study. Figure 1 is a conceptual illustration of our proposed methodology. Our method incorporates the use of the TF-IDF weighting scheme, but just before inserting the data into the model, a lexical dictionary is utilized to guide a critical stemming process. By contemplating the various spellings and variations in Roman Urdu, this dictionary contributes in standardizing the text. The main purpose of this step is to improve the feature selection process.

It starts with data cleaning, which deletes irrelevant symbols and punctuation marks from the text. Next, lexical normalization is conducted by using a rule-based approach, followed by stemming. Together, these form the preprocessing stage of the work, which is really important to

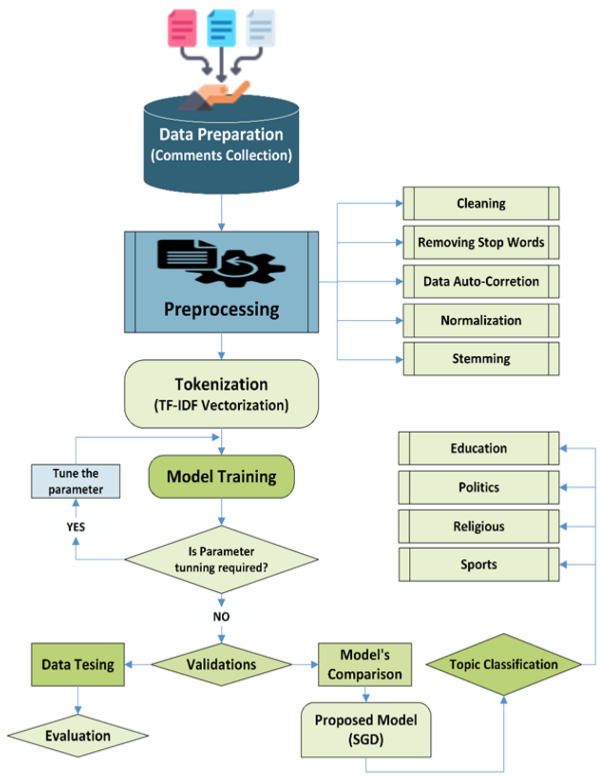


Fig. 1. Proposed model methodology workflow.

handle the irregularities present in Roman Urdu text. A TF-IDF vectorizer was then applied for feature extraction, while a number of ML models were subsequently used for the classification.

2.1. Dataset

The Roman Urdu dataset[1] used in this research has been collected from Kaggle, a well monitored platform acknowledged for its rich dataset repository and data science competitions. This dataset is a very valuable collection of text data, particularly in the Roman Urdu language, which covers a wide range of topics and sentiments. The corpus is collected from online forums and social blogs, hence offering a rich and reliable repository of real-world linguistic interactions and individual opinions. It provides a very useful insight into how people express their sentiments and opinions in Roman Urdu about diverse topics. The dataset consists of 4065 comments, hence, the data is labeled with categories like politics (1398), sports (1092), education (851), and religious (724). The politics and sports categories are most represented, followed by the education and religious comments, as captured in Figure 2, thereby reflecting an imbalanced yet diverse distribution in the corpus.

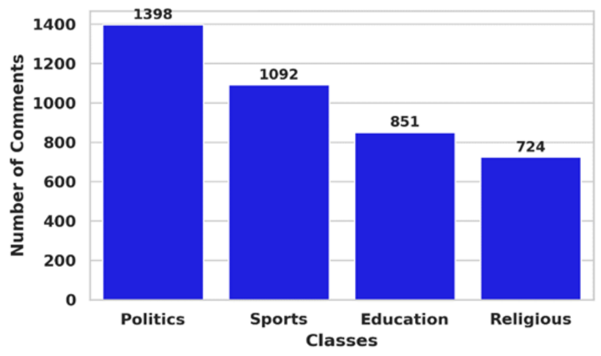


Fig. 2. Roman Urdu Dataset.

2.2. Preprocessing

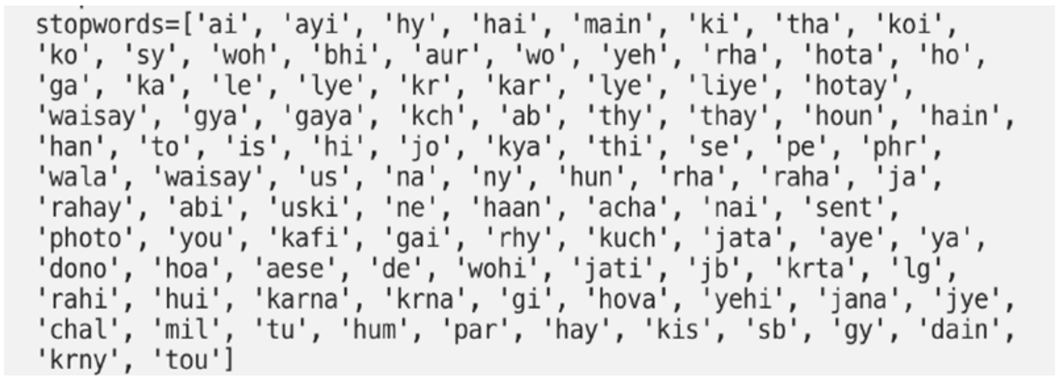
Preprocessing is important as it retains only the significant words and removes the rid of rest. Filler words like “punch lines,” “number characters” and “stop words” were deleted. The data preprocessing decreases computation time and size of the data. Doing that in NLTK library (Python), several operations are performed including removing the unnecessary words and characters, auto correcting and stemming.

2.2.1. Remove Stopping Words

Stop words are those common and repetitive words, which do not appear as useful information for the sentiment prediction. The idea of stop words was first introduced by Luhn [33]. In this paper, we perform a manual selection for these stop words. We will use a curated set of Urdu stop words to efficiently remove irrelevant words, reducing the data processing step. Figure 3 shows the stop words of Roman Urdu.

2.2.2. Data Auto Correction

For the unstructured Roman Urdu used in informal comments over the web, people usually use incorrect syntactical structures, hence the mining process is complicated. Hence, someone might stretch out characters of a word “bohttttttttt khubbbbbb” instead of the desired “boht khub” meaning “well done” in response to this our system attempts to resolve these ill formedness as by identifying the correct syntactic composition of words in order to facilitate better analysis [34].



```
stopwords=['ai', 'ayi', 'hy', 'hai', 'main', 'ki', 'tha', 'koi',
'ko', 'sy', 'woh', 'bhi', 'aur', 'wo', 'yeh', 'rha', 'hota', 'ho',
'ga', 'ka', 'le', 'lye', 'kr', 'kar', 'lye', 'liye', 'hotay',
'waisay', 'gya', 'gaya', 'kch', 'ab', 'thy', 'thay', 'houn', 'hain',
'han', 'to', 'is', 'hi', 'jo', 'kya', 'thi', 'se', 'pe', 'phr',
'wala', 'waisay', 'us', 'na', 'ny', 'hun', 'rha', 'raha', 'ja',
'rahay', 'abi', 'uski', 'ne', 'haan', 'acha', 'nai', 'sent',
'photo', 'you', 'kafi', 'gai', 'rhy', 'kuch', 'jata', 'aye', 'ya',
'dono', 'hoa', 'aese', 'de', 'wohi', 'jati', 'jb', 'krta', 'lg',
'rahi', 'hui', 'karna', 'krna', 'gi', 'hova', 'yehi', 'jana', 'jye',
'chal', 'mil', 'tu', 'hum', 'par', 'hay', 'kis', 'sb', 'gy', 'dain',
'krny', 'tou']
```

Fig. 3. Stop words in Roman Urdu.

2.2.3. Normalization and Stemming

A rule-based approach named hashing with the incorporation of lexical strategies for normalizing the Roman Urdu text is utilized by researchers of [35]. We have developed some guidelines to overcome this issue. These guidelines attempt to minimize the use of shared suffixes and infixes of the Roman Urdu words. In Table 1, an indication of the end of a string or suffix is shown by '$' sign, the start of any string by '^' sign, and repetition of any alphabet is '+'.

So, for example, words such as "khamian" (flaws), "achaaiyaan" (goodness), and "kitabain" (books) become "khami", "achai", and "kitab" respectively. One of the interesting things that can be noticed here is that the suffix "an" is removed when the letter "i" is observed before it. Also, expressions such as "taqreebaat" (ceremonies), "chakkay" (Sixes), and "haqooq" (rights) become "taqreb", "chakka", and "haq" respectively. Moreover, repeated letters are reduced to a single representation, as noticed in the normalizations of "qanooon" to "qanon" and "boohatt" to "bohat". Finally, after the application of these guidelines, the normalized text is then standardized using a human-annotated lexical dictionary.

The stemmer used in the data preprocessing step is intended to reduce words to their root form. Though there could be scenarios where the stem does not match with the root, this is still effective since related words tend to belong to the same stem despite the root not being proper itself. There are numerous stemmers for the English language or any other language that is gifted with rich linguistic resources. Examples of such stemmers include the Porter stemmer [36] and the Snowball stemmer [37]. The situation of stemming words for Roman Urdu is far more complex as compared to other languages.

Table 1 provides some examples of lexically normalized words. It is clear that the words in Table 2 have the same sound or pronunciation but with varying spellings. The stem word generation is dependent on a mapping function that is precisely given by f: N → S, where N denotes a finite set of words against which we strive to link plausible stem words that belong to set S. This function of mapping is set to establish the correct stem word S for the term N, boosting the efficiency of the stem word generation. If the mapping function is unsuccessful in identifying a stem word, then the root word is used. So, for ensuring effective search for the stem word, there is separate indexing of each word by means of a hashing function. Therefore, by using the map function, the entire document is exposed to the stemming process to remove any possibilities of inconsistencies or anomalies.

2.3. Model Training and Validation Phase

The data was divided into model's training and validation subsets as part of the dataset partitioning process [38, 39]. In particular, 70% of the dataset was reserved for model training, and the left over 30% was allocated for validation. Further insights into this division are provided in Table 3, revealing that 2845 comments were incorporated for model's training, and 1220 comments were employed for validation purposes.

2.4. Pipeline

A pipeline combines various estimation procedures into a single step, simplifying the ML process [38]. A pipeline involves the progressive implementation of a set of transformers (data modeling), followed

Table 1. Rules for Lexical Normalization.

Sr. No.	String	Replacement
1.	"ian" $	'i'
2.	"niat" $	"ni"
3.	"iy+"	'i'
4.	"ia"	'i'
5.	"ih"	"eh"
6.	"ay"	'e'
7.	"ie" $	'y'
8.	"ee+"	'e'
9.	"es"	'is'
10.	"ar"	'r'

Table 2. Stemming of Roman Urdu.

Roman Words	Stemming	English
siasat, syasat, sayasat	syast	Politics
parhaye, parhaee, parhai	prhai	Study
kitabain, kitaabain, ketabain	kitab	Books
taqreebaat, tareebat, taqrebaat	taqreeb	Ceremony
achaiyaan, achaian, achaiyan	achai	Goodness

Table 3. Training and Testing Sets Description.

Class	Training Set	Test Set	Total
Politics	994	404	1398
Sports	748	344	1092
Education	596	255	851
Religious	507	217	724
Total	2845	1220	4065

by an estimator at the end (ML model) [39]. The transformation stage includes the methods fit() and transform(), while the estimator includes fit() and predict(). Although an estimator always implements fit(), it may not necessarily implement predict(). Briefly, pipelines are designed with fit(), transform(), and predict() capabilities, allowing the entire pipeline to be fitted to the training data and then applied consistently to the test data without repeating each step manually. A pipeline is then built to convert words into vectors, extract features, and fit the model. In this work, function names such as fit(), transform(), and predict() are written with parentheses to indicate that they refer to callable methods (the () denotes that these are functions that can be executed with arguments), as commonly defined in machine learning libraries.

2.5. Feature Extraction

The step of feature selection involves the utilization of TF-IDF weighting scheme, a widely used method in text classification [32, 34]. This scheme assigns specific weights to individual vocabulary terms, belonging to the set $V = \{v_1, v_2 ... v_n\}$, for each document within the text corpus, in order to estimate their importance [7]. These weights, denoted as $W = \{w_1, w_2 ... w_k\}$, aim to reflect the significance of each vocabulary term. Nevertheless, the term frequency (TF) approach's shortcoming lies in its tendency to give higher weights to frequently appearing terms, which could lead to the neglect of crucial terms and subsequent subpar feature selection. Through the following characteristics, size of the feature can be evaluated.

2.6. TF-IDF Vectorizer

Term Frequency Inverse Document Frequency (TF-IDF) approach has broader utilization to transform text into a numerical illustration for prediction after training the ML models [8]. TF-IDF vectorizer takes into account a word's average prominence in a document [32]. When dealing with the most frequently used words, this is a great method. We can penalize them by using it. TF-IDF vectorizer applies a frequency-based weighting factor to the word counts. Table 4 displays the example of feature extraction using TF-IDF. Equation (1) shows the formulation of TF − IDF value in a particular document 'd' for a specific 't' th term:

$$TF - IDF(t,d) = TF(t,d) \times IDF(t) \tag{1}$$

The term frequency TF (t, d) is for 't' th term in document 'd'. While Inverse Document Frequency for 't' th term throughout the corpus is represented as IDF (t).

2.7. Classification Scheme

Our classification framework employs a diverse set of ML algorithms to classify topics in Roman Urdu text. These algorithms include Multinomial Logistic Regression (MLR), SVM, Naive Bayes, LR, Decision Tree, and our proposed approach based on SGD to explore the classification schemes that most suit the requirements of Roman Urdu text. The framework we have devised for topic classification is rooted in the utilization of the SGD algorithm [40]. This approach is used for the effective classification of topics in multi-class text reviews. The best algorithm emerged here is SGD, which showed the highest accuracy in categorizing Roman Urdu text. SGD is also an iterative optimization algorithm that plays a key role in the training of ML models [41].

It plays a very contributive role in text classification for Roman Urdu text in our research. The algorithm updates model parameters in an

Table 4. Feature Extraction by using TF-IDF.

Sr. No.	Words	TF-IDF
1.	talem	0.53109389
2.	games	0.57735026
3.	cricket	1.69314718
4.	hamesha	0.29207003
5.	reham	0.41802398
6.	khelta	0.70710678
7.	hifazat	0.26017797
8.	insan	0.24783099
9.	afsos	0.28194161
10.	tawajo	0.33762465

iterative manner, where it considers sometimes a single training example or a small batch every time. Inherent with this stochastic nature, it introduces randomness into the process, allowing the algorithm to avoid local minima and enabling quick convergence, especially in the case of large datasets.

This can be given, mathematically, by an update rule for SGD as:

$$\theta_{t+1} = \theta_t - \eta \nabla f(\theta_t; x_i; y_i) \tag{2}$$

Here θ_t represents the model's parameter vector at iteration t. $\nabla f(\theta_t; x_i; y_i)$ denotes the gradient of the loss function, f with respect to θ_t, evaluated on training example (x_i, y_i). While η, a hyperparameter, is the learning rate and decides the step size in the updates of the parameters. In this scenario, x_i shows input feature vector and y_i displays its respective target label for i-th data point used in the computation of the gradient of the loss function.

We implemented an SGD model based on a well-organized pipeline approach. This was composed of two significant parts: the TF-IDF vectorizer and the SGD classifier. The TF-IDF vectorizer played an important role in converting the text data into a numerical representation by assigning words with numeric values according to their weights in TF-IDF. These weights determine the importance of words within the text corpus. The processed data would then serve as an input to the SGD classifier, which utilizes the SGD optimization technique in training a linear classifier for binary classification problems. The "hinge" choice of loss function played an instrumental role in informing the optimization process, while the "l2" penalty contributed toward regularization. The parameter "max_iter" controlled the maximum number of iterations that should result from the optimization process. Through these components and by combining them in a pipeline configuration, we have successfully engineered a robust and flexible SGD model that can be applied to text classification tasks.

3. RESULTS AND DISCUSSION

The main results of our work demonstrate the efficiency of the proposed methodology for Roman Urdu topic classification. Our model, enhanced through the integration of SGD and a custom Roman Urdu stemmer, outperforms well-established models like LR, SVM, NB, DT, and kNN with regularity, which is also supported by prior works that state that quality preprocessing has a great effect on the classification result in low-resource languages [7, 10]. An achieved accuracy of 95 percent reflected the importance of efficient cleaning and TF-IDF transformation, such a relation is also supported through previous studies on Roman Urdu text processing [32]. A number of factors create this improvement. First of all, Roman Urdu-specific stemming rules and customized normalization reduce spelling inconsistencies and noise, thereby mitigating known limitations in previously reported Roman Urdu classification works [10, 42]. Second, TF-IDF is able to provide a sparse feature space that is efficiently handled by the linear SGD classifier, which further supports the previously found observations regarding the efficiency of linear models for short and informal text [7]. Overall, our results confirm that combining language aware preprocessing with an optimized linear classifier leads to more accurate topic categorization and offers strong potential for broader Roman Urdu text classification applications [32].

3.1. Evaluation Metrics

The efficiency of the classifier's is then assessed by using recall, F1-score and precision. Confusion Matrix of our proposed model is also displayed to illustrate the model's functionality.

3.1.1. Accuracy

From the perspective of examining classification models, accuracy is a fundamental metric. The magnitude of successful predictions of a model is an elementary description of its accuracy. Mathematically, we can formulate it as:

$$Accuracy = \frac{No.\ of\ correct\ predictions}{Total\ no.\ of\ predictions} \tag{3}$$

In the context of binary classification, accuracy is simplified in terms of negatives and positives as:

$$Accuracy = \frac{TN + TP}{TP + TN + FP + FN} \tag{4}$$

3.1.2. Precision and Recall

In the context of information extraction, precision and recall are most commonly applied. The record numbers that have been reclaimed are considered precision, whereas the total record numbers that have been recovered are termed as recall. Meanwhile Precision and recall are inversely related, this highlights the impact of having a reliable classification system to offer context for their variances.

Mathematical interpretation of both terms in classification task is given as:

$$Precision = \frac{True\ Positive}{False\ Positive\ +\ True\ Positive} \tag{5}$$

$$Recall\ = \frac{True\ Positive}{False\ Negative\ +\ True\ Negative} \tag{6}$$

3.1.3. F1-Score

F-measurement, F-score or F1 are similar calculation of the check. The percentage of correctly recognized positive outcomes is a common way to measure precision p, which are divided by percentage of all samples classified as positive, while recall r is the percentage of correctly identified positive results, which are divided by percentage of all examples categorized as positive.

$$F1 - Score = \frac{2\ \times\ Precison\ \times\ Recall}{Precison\ +\ Recall} \tag{7}$$

3.1.4. Confusion Matrix

Error matrix is another name for confusion matrix, in ML and classification. It is a table that clearly shows where a model makes mistakes. It helps illustrate model's effectiveness or efficiency by comparing its predictions with the original results. The main goal is to analyze the classifier's efficiency. By depicting both predicted and actual values, the confusion matrix offers a visual representation of disparities. This evaluation draws on insights from the confusion matrix, illustrated in Figure 4. Which encompasses metrics for topic classification. Correct predictions are positioned along the diagonal for visualization with the proper labelling of Politics, Sports, Education and Religious classes.

3.2. Topic Classification

In the context of the experimental study, various ML techniques of classification were used for the task. In order to ensure an unbiased comparison, replication of the earlier proposed solutions was carried out for measurement of the efficiency and validity of the ML models. Table 5 shows the experimental results of various solutions of classification with regard to Roman Urdu topic classification tasks. These experimental results clearly show that the proposed solution of SGD with enhancement of the stemmed solution outperformed all other solutions with its enhanced performance capability. In addition, various other solutions using ML also found effective solutions. It is pertinent to note that solutions by LR and by SVM found solutions equivalent to that of our proposed solution for better understanding with various metrics like recall, precision, F1, and accuracy.

Apparently, the class-wise accuracy of analysis models, as shown in Figure 4, clearly reveals that religious class shows better advancement in terms of each recall, F1-measure, precision, and total accuracy. At the same time, there was a slight drop in precision and recall for politics and support classes. Though Table 5 shows the efficiency of our models relative to other models. When comparing, there was a relative low accuracy of 61% by the SVM model developed by Mehmood *et al.* [30] relative to our fine-tuned models. Notably, even the proposed models by us showed better efficiency relative to the deep learning models Recurrent Convolutional Neural Network (RCNN) with an accuracy of 63%. Moreover, the KNN models [32] showed better efficiency relative to precision with a precision of (70%), though relative to recall, it is ineffective

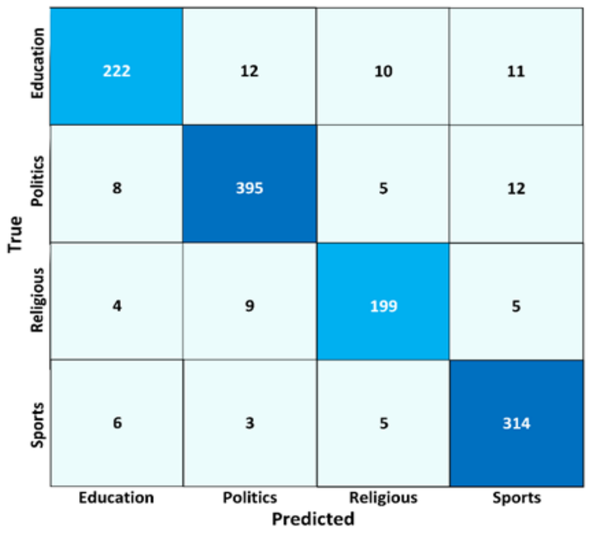


Fig. 4. Confusion Matrix of proposed model (SGD) for Topic Classification of Roman Urdu.

Table 5. Comparative evaluation metrics for proposed and existing models.

Model	Precision	Recall	F1-Score	Accuracy
LR	**0.94**	0.94	0.94	0.94
SVM	**0.94**	0.93	0.94	0.94
Naïve Bayes	**0.90**	0.84	0.86	0.86
Decision Tree	**0.83**	0.83	0.83	0.84
KNN	**0.87**	0.86	0.87	0.87
SVM [30]	0.59	0.58	0.58	0.61
KNN [32]	0.70	0.37	0.48	0.47
LSTM [42]	0.65	0.64	0.65	0.66
Random Forest [43]	0.63	0.61	0.62	0.59
RCNN [44]	0.64	0.62	0.63	0.63
Proposed SGD	**0.95**	0.94	0.94	**0.95**

with low recall that caused the lowest accuracy of 47%. At the same time, the Random Forest models' approach [42] showed relative efficiency relative to NB models, though it gained an accuracy of below 60%, which is unsatisfactory. Additionally, Naive Bayes showed relative efficiency with achieved accuracy of 62%, though it failed to achieve better efficiency relative to the SGD models [43]. At the same time, the efficiency of DT models showed moderate result with the precision of 59%, recall of 57%, and F1-measure of 0.58. Finally, LR and SVM models showed relative efficiency relative to ours with impressive accuracy of 94%. This shows that it is effective relative to regression models as well as classifications.

Figure 5 summarizes the detailed analysis of various models of ML for sentiment classification. This graph is more of a representation of the efficiency of the model in terms of Precision, Recall, F1 Score, and Accuracy of six models: LR, SVM, NB, DT, k-NN, and proposed model. This

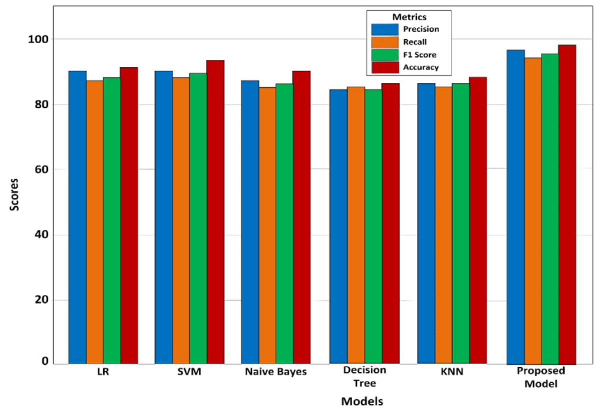


Fig. 5. Comparison of models' performances for Roman Urdu text classification.

graph aptly expresses the measures of the models using four bars for each of the models, representing each of the mentioned factors. It is worthy to note that the proposed model gets the maximum number of counts via these factors, highlighting the effectiveness of the proposed model for sentiment analysis.

4. CONCLUSIONS

In this work, we discussed topic classification for Roman Urdu text with several ML algorithms, including MLR, SVM, NB, Random Forest, DT, and our proposed SGD model supplemented with a Roman Urdu Stemmer. Our approach included extensive data preprocessing and feature extraction so that an optimal classification pipeline was achieved. Among all of the tried models, the SGD model performed best, achieving the maximum accuracy value of 95%. That means the proposed parameter optimization method in the SGD model showed better performance improvement in topic classification for Roman Urdu text. Though promising, we note some limitations of the current study, namely, the adoption of a single train/test split without any evaluation by other measures such as cross-validation that more completely showcases the generalization of the model. Furthermore, further works are needed to address the issues of class imbalance and the application of more advanced methods, such as cross-validation, which could make the results more robust. This study provides validation significant for Roman Urdu topic classification. This could be used in social media monitoring, content categorization, and public discourse studies. Future work will concentrate on refining the SGD model, expanding the dataset, and integrating additional linguistic features to enhance classification performance further.

5. ACKNOWLEDGMENTS

The authors acknowledge with gratitude the institutional support and resources that enabled the successful execution of this study.

6. CONFLICT OF INTEREST

The authors confirm that they have no conflicts of interest related to this publication. No financial or personal relationships affected how the study was designed, carried out, or interpreted.

7. REFERENCES

1. N. Pangakis and S. Wolken. Knowledge distillation in automated annotation: Supervised text classification with LLM-generated training labels. *Proceedings of the Sixth Workshop on Natural Language Processing and Computational Social Science (NLP+CSS 2024)*, *Mexico City, Mexico*, pp. 113-131 (2024). https://aclanthology.org/2024.nlpcss-1.9.pdf
2. Y. Xie, Z. Li, Y. Yin, Z. Wei, G. Xu, and Y. Luo. Advancing legal citation text classification: A Conv1D-based approach for multi-class classification. *Journal of Theory and Practice of Engineering Science* 4(2): 15-22 (2024).
3. K. Mehmood, D. Essam, K. Shafi, and M.K. Malik. An unsupervised lexical normalization for Roman Hindi and Urdu sentiment analysis. *Information Processing and Management* 57(6): 102368 (2020).
4. G.F. Simons and C.D. Fennig (Eds.). Ethnologue: Languages of the World (20th Edition). *SIL International, Dallas, USA* (2017).
5. G.I. Akabuike and I.C. Onuh. English spelling variations in social media among select students of English language in Nnamdi Azikiwe University. *Ansu Journal of Language and Literary Studies* 5(1): 52-64 (2025).
6. J. Tatemura. Virtual reviewers for collaborative exploration of movie reviews. *Proceedings of the 5th International Conference on Intelligent User Interfaces*, *New Orleans, LA, USA* pp. 272-275 (2000). https://doi.org/10.1145/325737.325870
7. S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao. Deep learning-based text classification: A comprehensive review. *ACM Computing Surveys* 54(3): 62 (2021).
8. A. Gasparetto, M. Marcuzzo, A. Zangari, and A. Albarelli. A survey on text classification algorithms: From text to predictions. *Information* 13(2): 83 (2022).
9. S. Daud, M. Ullah, A. Rehman, T. Saba, R. Damaševičius, and A. Sattar. Topic classification of online news articles using optimized machine learning models. *Computers* 12(1): 16 (2023).
10. N. Hussain, A. Qasim, G. Mehak, O. Kolesnikova, A. Gelbukh, and G. Sidorov. Hybrid machine learning and deep learning approaches for insult detection in Roman Urdu text. *AI* 6(2): 33 (2025).
11. M.U. Arshad, M.F. Bashir, A. Majeed, W. Shahzad, and M.O. Beg. Corpus for emotion detection on Roman Urdu. *22nd International Multitopic Conference (INMIC 2019), Islamabad, Pakistan* pp. 1-6 (2019). https://doi.org/10.1109/INMIC48123.2019.9022782
12. P. Pakray, A. Gelbukh, and S. Bandyopadhyay. Natural language processing applications for low-resource languages. *Natural Language Processing* 31(2): 183-197 (2025).
13. T. Adimulam, S. Chinta, and S.K. Pattanayak. Transfer learning in natural language processing: Overcoming low-resource challenges. *International Journal of Enhanced Research in Science Technology and Engineering* 11(2): 65-79 (2022).
14. H. Avetisyan and D. Broneske. Large language models and low-resource languages: An examination of Armenian NLP. *Findings of the Association for Computational Linguistics: IJCNLP-AACL 2023 (Findings)* pp. 199-210 (2023). https://aclanthology.org/2023.findings-ijcnlp.18.pdf
15. T. Ògúnrẹ̀mí, W.O. Nekoto, and S. Samuel. Decolonizing NLP for low-resource languages: Applying Abebe Birhane's relational ethics. *GRACE: Global Review of AI Community Ethics* 1(1): 1-13 (2023). https://ojs.stanford.edu/ojs/index.php/grace/article/view/2584/1546
16. A. Sandu, L.A. Cotfas, A. Stănescu, and C. Delcea. A bibliometric analysis of text mining: Exploring the use of natural language processing in social media research. *Applied Sciences* 14(8): 3144 (2024).
17. Q. Chen, W. Wang, K. Huang, and F. Coenen. Zero-shot text classification via knowledge graph embedding for social media data. *IEEE Internet of Things Journal* 9(12): 9205-9213 (2021).
18. A. Ghafoor, A.S. Imran, S.M. Daudpota, Z. Kastrati, R. Batra, and M.A. Wani. The impact of translating resource-rich datasets to low-resource languages through multi-lingual text processing. *IEEE Access*

9: 124478-124490 (2021).

19. V. Kumar, R.S. Singh, M. Rambabu, and Y. Dua. Deep learning for hyperspectral image classification: A survey. *Computer Science Review* 53: 100658 (2024).
20. A. Faheem, F. Ullah, U. Azam, M.S. Ayub, and A. Karim. Part of speech (POS) tagging in Roman Urdu: Datasets and models. *Language Resources and Evaluation* 59(4) pp. 4285-4312 (2025).
21. A. Ilyas, K. Shahzad, and M. Kamran Malik. Emotion detection in code-mixed Roman Urdu-English text. *ACM Transactions on Asian and Low-Resource Language Information Processing* 22(2): 48 (2023).
22. B.A. Chandio, A.S. Imran, M. Bakhtyar, S.M. Daudpota, and J. Baber. Attention-based RU-BiLSTM sentiment analysis model for Roman Urdu. *Applied Sciences* 12(7): 3641 (2022).
23. Z. Nabeel, M. Mehmood, A. Baqir, and A. Amjad. Classifying emotions in Roman Urdu posts using machine learning. *Mohammad Ali Jinnah University International Conference on Computing (MAJICC), (15th-17th July 2021), Karachi, Pakistan* pp. 1-7 (2021). https://doi.org/10.1109/MAJICC53071.2021.9526273
24. I.U. Khan, A. Khan, W. Khan, M.M. Su'ud, M.M. Alam, F. Subhan, and M.Z. Asghar. A review of Urdu sentiment analysis with multilingual perspective: A case of Urdu and Roman Urdu language. *Computers* 11(1): 3 (2021).
25. T.A. Rana, K. Shahzadi, T. Rana, A. Arshad, and M. Tubishat. An unsupervised approach for sentiment analysis on social media short text classification in Roman Urdu. *Transactions on Asian and Low-Resource Language Information Processing* 21(2): 28 (2021).
26. V. Tejaswini, K. Sathya Babu, and B. Sahoo. Depression detection from social media text analysis using natural language processing techniques and hybrid deep learning model. *ACM Transactions on Asian and Low-Resource Language Information Processing* 23(1): 4 (2024).
27. P.M. Lavanya and E. Sasikala. Deep learning techniques on text classification using Natural Language Processing (NLP) in social healthcare network: A comprehensive survey. *3rd International Conference on Signal Processing and Communication (ICPSC), (13th-14th may 2021), Coimbatore, India* pp. 603-609 (2021). https://doi.org/10.1109/ICSPC51351.2021.9451752
28. M.P. Akhter, Z. Jiangbin, I.R. Naqvi, M. Abdelmajeed, and M.T. Sadiq. Automatic detection of offensive language for Urdu and Roman Urdu. *IEEE Access* 8: 91213-91226 (2020).
29. K. Mehmood, D. Essam, K. Shafi, and M.K. Malik. Discriminative feature spamming technique for Roman Urdu sentiment analysis. *IEEE Access* 7: 47991-48002 (2019).
30. F. Mehmood, M.U. Ghani, M.A. Ibrahim, R. Shahzadi, W. Mahmood, and M.N. Asim. A precisely Xtreme-multi channel hybrid approach for Roman Urdu sentiment analysis. *IEEE Access* 8: 192740-192759 (2020).
31. H.H. Saeed, T. Khalil, and F. Kamiran. Urdu toxic comment classification with PURUTT corpus development. *IEEE Access* 13: 21635-21651 (2025).
32. M. Bilal, H. Israr, M. Shahid, and A. Khan. Sentiment classification of Roman-Urdu opinions using Naïve Bayesian, Decision Tree, and KNN classification techniques. *Journal of King Saud University-Computer and Information Sciences* 28(3): 330-344 (2016).
33. H.P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development* 2(2): 159-165 (1958).
34. S. Tariq, T.A. Rana, and F. Shahzadi. A comparative study of sentiment analysis in Urdu and Roman Urdu: The neglected realms. *CSI Transactions on ICT* 13(2): 193-211 (2025).
35. B. Chandio, A. Shaikh, M. Bakhtyar, M. Alrizq, J. Baber, A. Sulaiman, and W. Noor. Sentiment analysis of Roman Urdu on e-commerce reviews using machine learning. *CMES-Computer Modeling in Engineering and Sciences* 131(3): 1263-1287 (2022).
36. P. Willett. The Porter stemming algorithm: then and now. *Program* 40(3): 219-233 (2006).
37. M.F. Porter. Snowball: A language for stemming algorithms. Snowball Project, *Cambridge, UK* (2001). http://snowball.tartarus.org/texts/introduction.html
38. L. Wratten, A. Wilm, and J. Göke. Reproducible, scalable, and shareable analysis pipelines with bioinformatics workflow managers. *Nature Methods* 18(10): 1161-1168 (2021).
39. Y. Zhou, Y. Yu, and B. Ding. Towards mlops: A case study of ml pipeline platform. *International Conference on Artificial Intelligence and Computer Engineering (ICAICE), (23rd-25th October 2020), Beijing, China* pp. 494-500 (2020). https://doi.org/10.1109/ICAICE51518.2020.00102
40. J. Wang and G. Joshi. Cooperative SGD: A unified framework for the design and analysis of local update SGD algorithms. *Journal of Machine*

Learning Research 22(1): 9709-9758 (2021).

41. S.H. Haji and A.M. Abdulazeez. Comparison of optimization techniques based on gradient descent algorithm: A review. *PalArch's Journal of Archaeology of Egypt/Egyptology* 18(4): 2715-2743 (2021).
42. H. Ghulam, F. Zeng, W. Li, and Y. Xiao. Deep learning-based sentiment analysis for Roman Urdu text. *Procedia Computer Science* 147: 131-135 (2019).
43. L.C. Yu, J.L. Wu, P.C. Chang, and H.S. Chu. Using a contextual entropy model to expand emotion words and their intensity for the sentiment classification of stock market news. *Knowledge-Based Systems* 41: 89-97 (2013).
44. Z. Mahmood, I. Safder, R.M.A. Nawab, F. Bukhari, R. Nawaz, A.S. Alfakeeh, and S.U. Hassan. Deep sentiments in Roman Urdu text using recurrent convolutional neural network model. *Information Processing and Management* 57(4): 102233 (2020).