Sentiment Analysis of Roman Urdu on E-Commerce Reviews Using Machine Learning

Article *in* Computer Modeling in Engineering and Sciences · February 2022

DOI: 10.32604/cmes.2022.019535

CITATIONS
5

READS
832

8 authors, including:

Bilal Chandio
University of Balochistan
3 PUBLICATIONS **16** CITATIONS
SEE PROFILE

Asadullah Shaikh
Najran University
115 PUBLICATIONS **817** CITATIONS
SEE PROFILE

Maheen Bakhtyar
Asian Institute of Technology
41 PUBLICATIONS **479** CITATIONS
SEE PROFILE

Junaid Baber
University of Balochistan
52 PUBLICATIONS **575** CITATIONS
SEE PROFILE

Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2022.019535 Published Online: 13 January 2022

ARTICLE

Sentiment Analysis of Roman Urdu on E-Commerce Reviews Using Machine Learning

Bilal Chandio[1], Asadullah Shaikh[2], Maheen Bakhtyar[1], Mesfer Alrizq[2], Junaid Baber[1], Adel Sulaiman[2,*], Adel Rajab[2] and Waheed Noor[3]

[1]Department of Computer Science and Information Technology, University of Balochistan, Quetta, 87300, Pakistan

[2]College of Computer Science and Information Systems, Najran University, Najran, 61441, Saudi Arabia

[3]Department of Information Technology, University of Balochistan, Quetta, 87300, Pakistan

[*]Corresponding Author: Adel Sulaiman. Email: alsuliman2007@gmail.com

Received: 28 September 2021 Accepted: 14 December 2021

ABSTRACT

Sentiment analysis task has widely been studied for various languages such as English and French. However, Roman Urdu sentiment analysis yet requires more attention from peer-researchers due to the lack of Off-the-Shelf Natural Language Processing (NLP) solutions. The primary objective of this study is to investigate the diverse machine learning methods for the sentiment analysis of Roman Urdu data which is very informal in nature and needs to be lexically normalized. To mitigate this challenge, we propose a fine-tuned Support Vector Machine (SVM) powered by Roman Urdu Stemmer. In our proposed scheme, the *corpus* data is initially cleaned to remove the anomalies from the text. After initial pre-processing, each user review is being stemmed. The input text is transformed into a feature vector using the bag-of-word model. Subsequently, the SVM is used to classify and detect user sentiment. Our proposed scheme is based on a dictionary based Roman Urdu stemmer. The creation of the Roman Urdu stemmer is aimed at standardizing the text so as to minimize the level of complexity. The efficacy of our proposed model is also empirically evaluated with diverse experimental configurations, so as to fine-tune the hyper-parameters and achieve superior performance. Moreover, a series of experiments are conducted on diverse machine learning and deep learning models to compare the performance with our proposed model. We also introduced the largest dataset on Roman Urdu, i.e., Roman Urdu e-commerce dataset (RUECD), which contains 26K+ user reviews annotated by the group of experts. The RUECD is challenging and the largest dataset available of Roman Urdu. The experiments show that the newly generated dataset is quite challenging and requires more attention from the peer researchers for Roman Urdu sentiment analysis.

KEYWORDS

Sentiment analysis; Roman Urdu; machine learning; SVM

1 Introduction

Sentiment analysis (SA) is the method of classifying and assessing the general opinion and sentiment of people against products, services, current affairs and other entities [1,2]. Sentiment

analysis is theoretically defined as the computational study of opinions, attitudes, sentiments, views, emotions and traits. Its a general term for activities such as identification of sentiment subjects and classification of sentiment polarities [3]. Sentiment analysis and opinion mining these two terms are being used alternatively [4]. Sentiment analysis is a branch of text classification which deals with classifying the user sentiments to either positive, negative or neutral. The sentiment analysis involves two common tasks that are subjectivity and polarity detection [5,6]. Subjectivity refers to predicting that a text is subjective or not and polarity refers to predicting that a subjective text is either positive or negative [6]. The document level sentiment analysis refers to considering the whole document to check the polarity whereas, at the sentence level, we have to analyze each sentence in a document [1,2] and on word level as the name implies the polarity of the word is being extracted.

Classification from a machine learning perspective is defined as approximating a mapping function (f) from input variables (X) to discrete output variables (y) [7]. The modus-operandi for classification is supervised learning because it entails a labeled dataset. Although some of the approaches which have been proposed were unsupervised or semi-supervised, most of the research work has focused on supervised learning [7,8]. The binary classification is mostly done in sentiment analysis. However, the multi-class SA problem has not been addressed properly. In the machine learning approach, pre-assigned labeled data is furnished to the classification algorithm in order to learn the patterns.

Rudimentarily, the sentiment analysis can be done using two different approaches, lexicon based and supervised machine learning based [9]. A labeled or annotated data is requisite for a supervised machine learning approach. Moreover, the supervised approaches are domain-dependent and re-training is often required upon arrival of new data. Whereas, the lexicon-based approaches do not entail the training data, rather the lexicons of words having certain weights coupled with their sentiment class are used to extract the overall sentiment of the text. The lexicon-based methods are found to be effective on conventional text but ineffective for Twitter's data because they may contain malformed and informal expressions. To overcome this issue of irregularity, more sophisticated machine learning based models are needed [6].

Sentiment analysis has become an important aspect of the analytics that companies must use to assess their market position[1], i.e., how other people think about such products and services also affect our own buying decisions [10]. Moreover, corporations and businesses are still keen on comprehending public opinion of their products and services, and this is becoming increasingly important to them as this information will aid them tremendously in enhancing the quality of their goods and services and eventually generating more profits. Therefore, it not only is applicable to companies and organizations, but also seeks applications in fields such as e-commerce, financial sphere, politics, current affairs, health and medicinal sciences and also history [1,11].

Recently, businesses are bombarded with massive amounts of data-more than 2.5 quintillion bytes of data are generated every day[2]. More than 80% data is unstructured which needs to be sorted first. Human agents will find it nearly impossible to manually examine this data accurately and effectively as a result of this. Sentiment analysis has motivated researchers and analysts to derive opinions of people on different goods, programs, activities and other entities. This has been

[1] https://towardsdatascience.com/sentiment-analysis-comparing-3-common-approaches-naive-bayes-lstm-and-vader-ab561f834f89

[2] https://towardsdatascience.com/quick-introduction-to-sentiment-analysis-74bd3dfb536c

made possible due to an astronomical rise in the volume of text data being made accessible on the Internet, not just in English but also in several regional languages across the world as well, along with the recent developments in the area of machine learning and deep learning.

This study is aimed at exploiting one of the resource starved language of South Asia, i.e., Roman Urdu. Roman Urdu uses the Latin script in contrast to regular Urdu, which uses the Nastaliq script. In order to process Roman Urdu text for any NLP task it would be appropriate to first understand the nature of this text and challenges faced in processing such text data.

1.1 Challenges in Processing Roman Urdu Text

Urdu is an Indo-aryan language being mostly spoken in sub-continent [12]. Roman Urdu is basically an amalgamation of English/Latin alphabets and the Urdu language. In other words, English alphabets are used to convey a message in Urdu. Users of the internet utilise Roman Urdu for socialising or seeking for information and are happy with the outcomes[3]. Prominent Urdu news websites has devised their own way of dealing Roman Urdu. Roman Urdu has provided a great advantage for those who do not understand the Arabic script or traditional Nastaliq script. Several social media plate-forms are working as laboratories for evolving this new Urdu script. The idea of romanizing the Urdu had been proposed several times.

In sub-continent people mostly share their views and sentiments in Roman Urdu which is very much irregular in nature and there is no such grammar or spelling criteria exists. Their exists a number of variants for a single word of Roman Urdu such as 'ahtejaj', 'ehtijaj', 'aehtajaaj' which means protest. Such irregular arrangements makes Roman Urdu difficult to process. Different researchers has proposed a rule based and machine learning based lexical normalization approaches to standardized this text [13]. An instance of Roman Urdu user review with different variants has been illustrated in Table 1 whereby it can be assessed that a single Roman Urdu sentence may be expressed in three different spellings. Everyone has his/her own spelling for the pronunciation of a word in Roman Urdu which makes it very much dynamic and irregular.

Table 1: An instance of Roman Urdu language sentence variants

Var	Roman Urdu review	English equivalent
1	Daraz k store pr mobile ki qeematen bohat munasib or achi hain	Mobile prices on daraz store are very reasonable and good
2	darz key storee per mubile ke kematen bht manaseb our ache hai	-do-
3	daraaz ka stoorre par mbile k kimatyn buht mnasib or acha hay	-do-

Another issue with Roman Urdu is that people mix English and Urdu words and then it would be a problem of detecting language. People share their ideas, suggestions, comments and opinions about a particular product, service, political entity and current affairs. There are so many user-generated opinions available on the web and from all those opinions, it is difficult to judge the number of positive and negative opinions [5].

[3] The role of Roman Urdu in multilingual information retrieval: A regional study, The Journal of Academic Librarianship

The Internet has been broadly user-centric. People are busy using various outlets to express their thoughts. Likewise, internet shopping has also become very popular and one of the most convenient forms of shopping [14]. Online sales of goods avoid the real hectic process of visiting the shops in person. Customers get the goods at their doorstep and safe payment is made easy online. Whenever individuals plan to purchase a product, they choose to read specific product reviews written by other individuals who have already used the same product. Online reviews offer a way to verify the popularity and effectiveness of products before purchase [12].

Over the past few years, online shopping in Pakistan has drastically increased[4]. The consumers are increasing day-by-day so the challenges to online shopping stores too. Decision making has always been crucial to business firms especially when it comes to e-commerce. The sentiment analysis of reviews is a key decision-making tool for e-commerce development [15]. An e-commerce website usually receives thousands of product reviews daily. The consumers of online shopping stores such as daraz.pk mostly express their reviews and feedback either in English or Roman Urdu. As discussed above, the nature of Roman Urdu is very much colloquial and the traditional sentiment analysis approaches would not be that much fruitful in this scenario. The foremost step in sentiment analysis is indeed pre-processing of the text *corpus* in order to clean and streamline all kinds of formal and informal expressions. The main challenge in the case of Roman Urdu pre-processing is its stemming. For resource rich languages such as English, French and German there exists a number of stemmers but for Roman Urdu, there is no single stemmer exists. We have proposed a human-coded dictionary based stemming for Roman Urdu. Stemming plays a significant role in pre-processing of any text *corpus*. The stemming is aimed at reducing the complexity and variance of a text which consequently triggers higher accuracy while classifying any document. The quality of stemming algorithms is typically measured in two different ways: (i) how accurately they map the variant forms of a word to the same stem; or (ii) how much improvement they bring to information retrieval systems.

This study deals primarily with sentiment analysis approaches focused on reviews in Roman Urdu language on Pakistani goods. Reviews are mainly positive, negative or neutral, and they evaluate and analyze the opinion of the customer, allowing retailers to enhance and boost the availability and efficiency of their goods, thereby positively impacting online shopping.

The major challenge while dealing with text classification problems is text representation. Text presentation is a mapping mechanism that, by a certain form, maps a word in a text space to a numerical vector space [16]. Much of the research in this area is shifted to deep learning, however, the traditional vector space model or bag-of-words model if powered by suitable pre-processing tools may yield better results. Moreover, by fine-tuning the hyper-parameters of traditional machine learning approaches such as Support Vector Machine (SVM), the desired results may be achieved. It is worth mentioning here that in a multi-class text classification environment it would be quite fruitful to break this into multiple binary classification problems. We too opted for this strategy to efficiently deal with this problem and will be discussed in forthcoming sections.

There are four main contributions in this research study: (1) We have publicly introduced the Roman Urdu dataset which contains 26 K+ instances obtained by the DarazPK portal and annotated by field experts, (2) We created an online system that can be used as a demo and also

[4] https://www.dawn.com/news/1606875

an API[5] that can easily be used as the baseline model for future research, (3) We also shared our all scripts, stemming dictionary, list of stop-words, and the trained model, and (4) We investigated different machine learning and deep learning schemes for the Sentiment Analysis (SA) of Roman Urdu reviews.

In our experimental pool, we have considered a set of machine learning and deep learning approaches to investigate the performance for Sentiment analysis of Roman Urdu. The machine learning approaches includes KNN [17], Naïve Bayes [18], Adaboost [19], SVM [20], Random Forest [21], Decision Tree, Logistic Regression [22]. Whereas, for deep learning we have considered RCNN [23], LSTM [24], CNN, RNN, and GRU [13]. Our proposed model employ the vector space model or bag-of-words model with TFIDF (term frequency-inverse document frequency) weighting scheme for feature selection. Our experimental results reveal that support vector machine (SVM) can better cater the sentiment analysis problem, in contrast to other machine learning and deep learning approaches. Provided that essential text pre-processing is carried-out and the model is fine-tuned accordingly. We have used the one-vs-rest (OVR) classification strategy to slice the multi-class problem into a number of binary classification problems. In order to evaluate the results, we have used precision, recall, F1-score and area under the curve (AUC) which is computed via outputs of the confusion matrix.

The rest of the paper is structured as follows. Section 2 presents related work. Section 3 focuses on the methodology and *corpus* generation and annotation process while Section 4 explores the experiments and results. Finally, Section 5 provides the conclusions and future directions.

2 Related Work

In natural language processing (NLP), Sentiment Analysis is an important research area that is still captivating the attention of researchers [1,25]. Various machine learning and deep learning model were proposed to solve this puzzle, however, not every model fits the needs of every kind of text data. The classification task involved in sentiment analysis can be categorized into four different domains such as subjectivity classification, word sentiment classification, document sentiment classification, and opinion extraction [26]. It was further extended that application of sentiment analysis is not only in assessing product reviews but can also be found on stock prediction, movie review, news articles, or political debates. In terms of the domains of the datasets, *corpus* types and scale, as well as multilingual context, the study finds several sentiment analysis issues [4,27]. A model built on the domain of one dataset such as politics may not yield good results for other domains such as product reviews [6,25]. The sentiment can be examined at different levels, namely document, sentence and aspect level [28,29]. The whole review is regarded as a basic information unit at the document level and is then classified into positive, negative or neutral opinions. In the same way, sentences are considered to be short documents. For the document as a whole, the sentiment at the document level is determined, which clarifies the sentiment associated with the target.

In the early days, the SA problem was dealt with lexicon bases approaches. A lexicon-based classifier SentiStrength was introduced earlier which was a supervised classifier and later it was upgraded to SentiStrength 2. The SentiStrength 2 an unsupervised cum supervised classifier that assigns separate scores (for both positive and negative) from 1–5 to a given text. This score

[5] https://sa-svm.mlcs.xyz/

actually decides the sentiment strength of a given text. SentiStrength was found to be slightly better than machine learning algorithms and logistic regression [6]. A lexicon-based approach using a contextual representation of words, called SentiCircles [9] which is aimed at extracting the semantics of words from their co-occurrence patterns and update their sentiment orientations accordingly was introduced. Contextual semantic approaches determine semantics from the co-occurrence patterns of words, also known as statistical semantics and have often been used for sentiment analysis. They claimed better accuracy for their SentiCircle lexicon based approach as compared to the traditional lexicon approaches, i.e., SentiStrength for lexicon labeling methods. Many researchers are of the view that the biggest challenge while classifying the text is the availability of annotated data which can be dealt with lexicon based approach for domain specific problems [25].

An aspect-based technique that relies on lexicons and rule-based technique to classify the user's opinion against government mobile apps was utilized [29]. Their approach is based on manually coded lexicons with a hybrid method to overcome the issue of domain knowledge and challenges to aspect based analysis. One of the major challenges in the aspect extraction technique is that some aspects are not explicitly mentioned in a review. A sentence with an explicit aspect contains a term that indicates the aspect category, while an implicit aspect would not be expressed by any specific word or term [29].

A comparative study of semantic text classification and traditional text classification was carried out [30]. Certain pros and cons of both approaches were identified. The semantic classification which is a knowledge-based approach to extract the polarity of a text is found effective than the machine learning approach. But on the other hand, the efficacy of semantic classification is limited to a specific domain of language for which a sentiment dictionary is being created such as SentiWord for the English language. A limited number of lexical databases restricts this approach to the specific domain of knowledge. Moreover, for both of the techniques, there exists a certain processing cost for knowledge-based semantic approach a huge knowledge data processing is needed, whereas, for the *corpus*-based system there is a cost for pre-processing. Such computational complexity would augment the running time of the classification algorithm. On the other hand, deep learning yields better results but computationally expensive and entails a huge amount of data [30].

In a comparative study (using BBC News and Digg dataset) it was concluded that lexicon-based classification outperformed the supervised machine learning approach especially detecting the subjectivity and objectivity of a text while in the case of detecting the polarity it was slightly better [31]. Moreover, Naive Bayes showed better accuracy as compared to maximum entropy. It was empirically found that the Support Vector Machine algorithm outperformed the Naive Bayes and Maximum Entropy classifiers, using a variety of features, such as term frequency feature weights, uni-grams and bi-grams, part-of-speech, etc. [32]. Our experiments too reveal the same but with a different configuration. The domain for experiments was movie reviews and for this they used IMDB movie reviews dataset. Before applying the machine learning algorithms features were extracted using the bag of features framework. Naive Bayes a text classification algorithm that is based on a Bayesian network, assigns a class to given text by calculating the posterior probability. The posterior probability limits Naive Bayes to perform optimally only on highly dependent features [18]. Some researchers have also proposed SVM with the chi-square method for feature selection and ultimately classifying the reviews. For this purpose two benchmark datasets, Pang *Corpus* and Taboada *Corpus* were used [33] and it was concluded that the uni-gram model outperforms the n-gram model, whereas, TFIDF plays a crucial role [34].

Noor et al. [12] has conducted a research on Roman Urdu sentiment analysis and proposed SVM. In their proposed configuration an accuracy of 60% was achieved. The possible cause of low accuracy is the absence of proper text standardization which is restricting the accuracy at a certain level. In our study, we have tried to overcome this problem by introducing the Roman Urdu stemmer. A research study on Roman Urdu News classification was conducted using Naive Bayes, Logistic Regression (LR), LSTM and CNN [35]. The study was aimed at classifying the news into five categories (health, business, technology, sports, international). They have utilized the phonetic algorithm for the lexical normalization and noise control in the *corpus*. It was identified that Naive Bayes outperforms the rest of classifiers. In an earlier research study, Roman Urdu sentiment classification experimented on Naive Bayes, Decision tree and KNN algorithm using ready-made tool WEKA which apparently favored Naive Bayes [17]. However, the size of their dataset mentioned was too small and not public.

Few researchers attempted the sentiment analysis problem using deep learning methods based on LSTM and RNN. A deep learning model RCNN was proposed and experimented on newly generated Roman Urdu dataset RUSA-19 [23]. There are 10,021 Roman Urdu sentences in the RUSA-19 *Corpus* that belong to five genres: (i) drama, movies, and talk shows; (ii) food and recipes, (iii) politics; (iv) apps, blogs, forums, and gadgets; and (v) sports gathered from numerous websites on social media. The creation of a new Roman Urdu dataset was the main contribution of their work. A deep learning based comparative study was conducted on Roman Urdu for SA task [24]. In this regard different machine learning classifiers among with proposed deep learning model LSTM were empirically evaluated. It was determined the deep learning model overshadows the machine learning models. In a recent research study a hybrid approach based on BiGRU and word embeddings were introduced [20] for the SA of Roman Urdu and reported good accuracy. However, the size of their dataset is too small and contains only 3241 sentiment reviews and not yet publicly available.

A study was conducted on a very small Roman Urdu dataset containing 150 positive and 150 negative reviews. The traditional bag-of-word model was used with three different machine learning algorithms, i.e., Naive Bayes, DT and KNN [36]. Their experimental results show that Naive Bayes has overshadowed the DT and KNN. Rafique et al. [37] has also employed different machine learning classifiers including SVM, Naive Bayes and Logistic Regression. In their empirical study they revealed that SVM outperform the rest of the classifiers. Habiba et. al. [38] conducted a study to compute the National Happiness Index (NHI) on Roman Urdu which is based on lexicon based and machine learning approach. To measure the NHI score, the suggested solution classifies a Roman Urdu sentiment that comprises three sub-opinions. To obtain discourse content, the algorithm divides the sentiment into multiple sub-opinions. Then, based on the derived discourse knowledge, it assigns positive or negative polarity to each sub-opinion. Finally, using the polarities of the sub opinions, it determines NHI. Sharf et al. [39] has identified that lexical normalization as one of most significant task in natural language processing. In order to normalize the Roman Urdu text they employed the phonetic based algorithms Soundex and NYSIIS. In their recent work [40] came up with another significant progress by creating a *corpus* of 15000 reviews acquired from various sources such as Twitter, Reddit, Urdu Poetry, etc. Their primary object was to conduct a discourse based sentiment analysis which uses the discourse information to extract the text features.

A study on lexical normalization of Roman Hindi and Urdu text, machine learning based approaches were employed [13]. Their research study proposed a novel technique called Transliteration based Encoding for Roman Hindi/Urdu text Normalization (TERUN). TERUN uses the

linguistic aspects of Roman Hindi/Urdu to transform lexically variant words to their canonical forms. It was identified that their proposed unsupervised phonetic scheme TERUN can better standardize the text in contrast to traditional phonetic algorithms such as Soundex, metaphone and NYSIIS. However, still Roman Urdu is found challenging while normalizing the text, as the nature of this text is quite variable.

In an attempt to deal with bilingual text based on Roman Urdu and English regarding people reviews on General Elections, 89,000 tweets were gathered [41]. They employed a lexicon centered approach by initially classifying the topic and language and later use the SentiStrength and Wordnet for generating the lexicons and subsequently the polarity of each tweet is calculated on the bases of this lexical dictionary. In a recent study the role of discourse information for Urdu SA is also studied [42]. It was identified that using discourse information bag-of-word (BOW) features can be improved which can later be classified using machine learning schemes. Do et al. [43] has introduced a RUBERT text representation, a bilingual model which is trained from scratch on BERT English model and monolingual model. RUBERT was aimed at mitigating two challenges that is Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). However, it can be perceived that although both English and Roman Urdu use the same Latin alphabets but still they are semantically different. Hence transfer learning in this regard may not yield the desired results.

In a recent study an attention based bidirectional CNN and RNN model, ABCDM was proposed for sentiment analysis [44]. Experiments were conducted on five review and three Twitter datasets and achieved good results for both long and short reviews. ABCDM exploits publicly available pre-trained GloVe word embedding vectors as the initial weights of the embedding layer. On top of the embedding layer, two bidirectional LSTM and GRU networks are used to extract both past and future contexts as semantic representations of the input text. In a research study to support the decision making using sentiment analysis for Chinese text reviews, Chen et al. [15] proposed TRG-DAtt. The proposed model was based on Target Relational Graph and Double Attention Network. The study reveals that the target words in reviews must be considered to accurately extract the emotional information. The sentiment analysis task is basically a classification problem which relies on extracted feature. Different feature engineering schemes such as Latent Dirichlet and Chi-Square is traditionally used to extract the key features which founds not reliable for short text. In a research study on unsupervised derivation of keyword summary for short length text a ranking based feature engineering method named Frequent Closed Wordsets Ranking (FCWRank) [45] was proposed. FCWRank mines all common closed wordsets from a cluster of short texts, then picks the most important wordset using a significance model that considers both the similarity of closed wordsets and the relationship between the closed wordset and the short text document. Subsequently, the keywords in wordset are sorted to semantically empower the model. A hybrid approach of two deep learning architectures namely Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) was proposed for sentiment analysis of reviews from different domains [26]. Their proposed Co-LSTM model was aimed at assessing sentiment analysis on social big data. Roman Urdu toxic comment classification was studied [46] to alienate the toxic and non-toxic comments. They produced a Roman Urdu toxic comments dataset, RUT (Roman Urdu Toxic) that contains over 72 thousand comments collected from popular social media platforms and manually labelled with high inter-annotator agreement. They experimented with different machine learning and deep learning methods using word embedding and finally proposed an ensemble approach. Sentiment information based network model(SINM) was proposed for Chinese text sentiment analysis [47]. The proposed model employees transformer

encoding and LSTM and Chinese emotional dictionary. SINM is aimed at initially learning the sentiment knowledge and later use auxiliary resources.

Contemplating on previous studies reveals that there is still room for improvement in this research area. It can be determined that not many NLP resources are available for resource-poor languages such as Roman Urdu. Especially, the lack of pre-processing tools such as stemming and lemmatization has compelled us to revisit the previous approaches of text processing. Most of the previous work on lexical normalization has not enabled this text to properly normalize the text and nor such standardization tools are made public. Our study is aimed at identifying the flaws in current approaches and propose an improved approach towards ameliorating the overall performance.

3 Methodology

This section is intended to deliberate the proposed methodology with a certain set of machine learning and deep learning experiments and stemming of Roman Urdu. In our pool of experiments, we have evaluated SVM, KNN, Naive Bayes, Adaboost, Random Forest (RF), Decision Tree (DT), Logistic Regression (LR) algorithm, RCNN [23], LSTM [24], CNN, RNN, and GRU. We propose a support vector machine for the sentiment analysis (SA) of Roman Urdu text. The abstract flow of our proposed approach is illustrated in Fig. 1. The framework uses vector space model, bag-of-word model for feature extraction, along with TFIDF weighting scheme, however, before feeding the data a much necessary stemming process is carried out which is based on human annotated lexical dictionary. The lexical dictionary is build to normalize the text data with variable spellings. Prior to this, no one has performed the stemming on Roman Urdu. We have introduced a dictionary based stemming method in order to cluster similar words in one group. This is intended to improve the feature selection process. The sentiment classification process starts with initially cleaning the data for unnecessary symbols and punctuation and later on a rule-based lexical normalization approach is applied and then data is sent for stemming, the process is known as pre-processing. Subsequently, the text features are extracted based on term-frequency and inverse document frequency (TFIDF) and placed in a vector space represented as $\mathbb{R}^n$. Besides so much critic, the traditional bag-of-word model still found good in extracting the key features, however, pre-processing plays a vital role by reducing the anomalies in text data. Soon after feature selection, this bag of features is now ready to be fed in the classification algorithm.

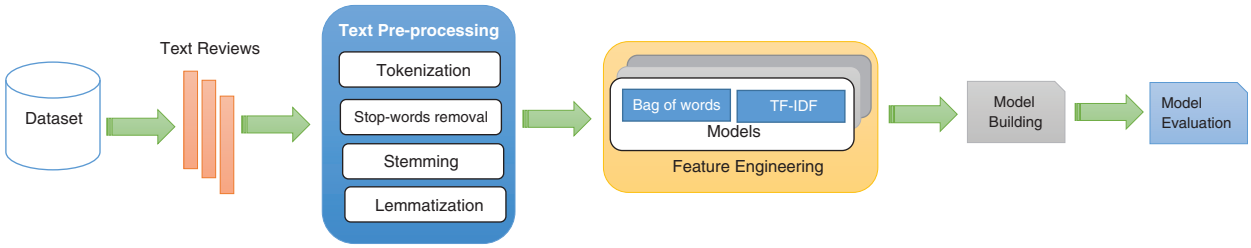


Figure 1: Abstract flow diagram of proposed framework

3.1 Lexical Normalization & Stemming

In order to normalize the text, we have employed the rule-based lexical normalization scheme also referred as hashing. The informal and variable nature of Roman Urdu text has posed a serious challenge while creating the lexical normalization rules. However, we have framed few

rules as mentioned in Table 2 to cater this problem, the rules are basically aimed at reducing the known suffix and infix from Roman Urdu words. The $ sign represents the ending string or suffix, ^ sign depicts the starting string, and + sign shows the multiples of a particular alphabet. The words such as "khamian" ("errors"), "buraian" ("evils"), "naikian" ("virtues"), "kahanian" ("stories") will be reduced to "khami","burai","naiki" and "kahani", respectively. It can be observed that if 'i' is followed by "an" than suffix "an" will be removed. Similarly, words like "insaniat" ("humanity"), "lisaniat" ("linguistics"), and "laqaooniat" ("lawlessness") will be replaced by "insan", "lisan" and "laqanon". Moreover, repeating alphabets will also be reduced to single alphabet such as "qanooon" ("law"), "boohatt" ("much") will be normalized to "qanon" and "bohat". Once the normalization rules are applied the normalized text is further standardize using human annotated lexical dictionary which is discussed below.

Table 2: Lexical normalization rules

#	*String*	*Replacement*	#	*String*	*Replacement*
1.	"ian" $	" "	11.	"ay"	"e"
2.	"niat" $	"n"	12.	"ie" $	"y"
3.	"iy+"	"i"	13.	"ain" $	"ein"
4.	"ia"	"i"	14.	"a+"	"a"
5.	"ten" $	"t"	15.	"j+"	"j"
6.	"tin" $	"t"	16.	"d+"	"d"
7.	"ar"	"r"	17.	"u"	"o"
8.	"ay" $	"e"	18.	"s+"	"s"
9.	"ey" $	"e"	19.	"ee+"	"e"
10.	"ih"	"eh"	20.	"ês"	"is"

Stemming is an inevitable step while pre-processing the text in NLP. The primary goal of stemming is to reduce the number of vocabularies by grouping similar words to a single stem word. The stemming process basically restricts a word to its stem or root word. In other words to lexically normalize the text stemming is employed. Their exists a number of stemmer for English and other resource-rich languages such as porter-stemmer [48] and snowball stemmer [49]. However, for Roman Urdu, there is not a single stemmer that exists. It is worth mentioning here that Roman Urdu stemming is much more challenging as compared to other languages. Because for a single word there exists a number of spelling variants which makes it challenging. The traditional prefix, infix and suffix method also known as rule-based method is not reliable in this case because of the extremely informal nature of the text. We propose a dictionary-based stemming function for Roman Urdu. For the purpose three graduates who have good knowledge of Roman Urdu were assigned to group the vocabularies extracted from the RUECD dataset. The vocabularies were grouped on the bases of intention to write the same word with different spellings. Ultimately, a stem word with the least length is assigned to overcome infliction in this informal text. An instance of lexically normalized words are recorded in Table 3. It can be observed that words are phonetically same but can be written with different spelling. Our future work is destined to automatically normalizing the text using an algorithmic scheme however, for now we have utilized the human annotated standardization scheme.

Table 3: Lookup table for lexically normalizing the Roman Urdu text

Words	Stem word	English word
aetbar, atbaar, etbar, aitbar	atbar	Trust
aziyat, aziyaten, aziat, azzeyat	azyt	Distress
aehtajaj, ahtejaji, ehtejaj	ahtjaj	Protest
behtar, behtareen, behtarin	bhtr	Better
qanooni, kanoon, qanoondan	qanon	Law

Our stemming machine is based on a mapping function f: $K \rightarrow S$ maps a word K to its corresponding stem word S. Here, K is finite set keys or terms against which we map a possible stem word from S. If the function fails to map the word to its stem the original word is returned. Additionally, the index for each word is separately managed using hashing function as illustrated in Fig. 2 so as to quickly retrieve the stem word. Using this mapping technique we perform the stemming on the whole *corpus* to reduce the anomalies.

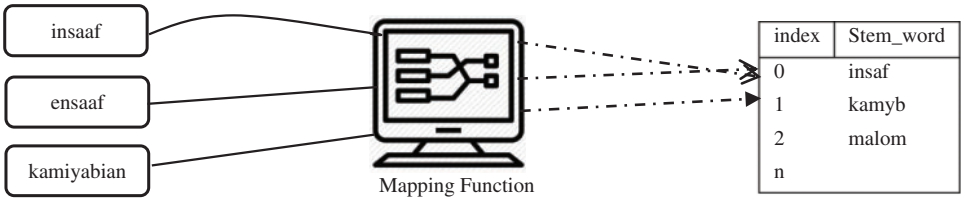


Figure 2: Roman Urdu stemmer mapping scheme

3.2 Feature Selection

As discussed above the term frequency and inverse document frequency (TFIDF) weighting scheme is applied for selecting the key features. Among a set of vocabularies $V = \{x_1, x_2, \ldots, x_n\}$ each vocabulary is being assigned a weight $W = \{w_1, w_2, \ldots, w_k\}$ for each separate document in our text *corpus*. The term frequency (tf) has the major drawback that it assigns a higher weight to the more frequently occurring terms. Consequently, the key terms are neglected which turns into poor feature selection. To overcome this problem we take the product of term-frequency (tf) and inverse-document frequency (idf). Equations in display format are separated from the paragraphs of the text.

3.3 TF-IDF Model

Term frequency-inverse document frequency is a statistical model for computing the weights of terms. The purpose of estimating the term weights is to determine that how important a term is to a document and for the whole *corpus*. In other words, key terms are identified which gives a better reflection of a certain class of document.

Term-frequency is a measure that determines how frequently a word occurs in a document. The term-frequency, t, is the number of times a term appears in our document. Every document is of different length it may possible that in long documents the term may appear many times. Therefore, we normalize the term by dividing it by the total length of document N. See Eq. (1) where N represents the total number of terms in a document and N_t represents, a number of times term t appears in a document.

$$\text{Term-Frequency (t)} = \frac{N_t}{N} \tag{1}$$

Inverse-document frequency is a measure that identifies how important a term to a document. Because term-frequency assigns almost equal importance to each term. There is a great possibility that mostly occurring terms such as stop words "wo, wahan, main, ye", etc. may gain higher weights. To weigh-down the mostly occurring terms and scaling up rare terms (which reflects the real contribution), we need to take the Inverse Document Frequency (IDF) as following:

$$\text{IDF (t)} = \log\left(\frac{N_d}{N_t}\right) \tag{2}$$

Here, N_d is the total number of document instances and N_t is a number of documents in which the term t appears. Eventually, we compute the product of tf and idf to amalgamate both frequency and importance factors. Soon after computing the desired weights, we place them in a vector space $\mathbf{R}^n$ which is a combination of rows and columns forming a feature matrix.

3.4 Classification Schemes

We have experimented our *corpus* on different classification approaches namely Support vector machine (SVM), K-Nearest Neighbor (KNN), Naive Bayes,Adaboost algorithm, Random Forest, Decision Tree (DT), Logistic Regression (LR), RCNN [23], LSTM [24], CNN, RNN, and GRU [13] to investigate either which classification schemes best fits the needs of Roman Urdu text. Our proposed classification scheme is based on a support vector machine algorithm (SVM) which is aimed at classifying the sentiments for multi-class text reviews. Support vector machine (SVM), a machine learning approach based on statistical learning theory. Provided a set of annotated user reviews for positive, negative, and neutral sentiment class, the SVM built a model to assign sentiment class to newly unseen input reviews. SVM model is an illustration of the text reviews as points in space, mapped so that the instances of the separate class are alienated by a vivid gap that is as wide as possible.

The SVM is initially being given the data for classification purpose. The data to be classified is represented as a single point in space, with each point represented by a feature vector x where, $x \in R^D$. Here, R^D is a vector space with D number of dimensions. The unique points are further mapped to a feature space x, $\phi(x) \in R^M$. Each input feature is mapped to feature space, this transformed basis vector $\phi(x)$ can be defined as:

$$\phi\,(x) : R^D \mapsto R^M \tag{3}$$

After mapping the input space to a feature space now we need to draw a separator line to classify the data points. This main line separator is also referred as decision boundary which divides the data points to its respective class as depicted in Fig. 3. Mathematically, this decision boundary is called a hyperplane. The hyperplane concept can better be understood with straight line which is represented as $y = mx + c$ where m is slope and c is intercept. Assuming that we are trying to fit a straight or linear line to classify the data points. The hyperplane equation now be represented as,

$$\mathbf{H} = \left\{\mathbf{x}|\mathbf{w}^T\mathbf{x} + b = 0\right\} \tag{4}$$

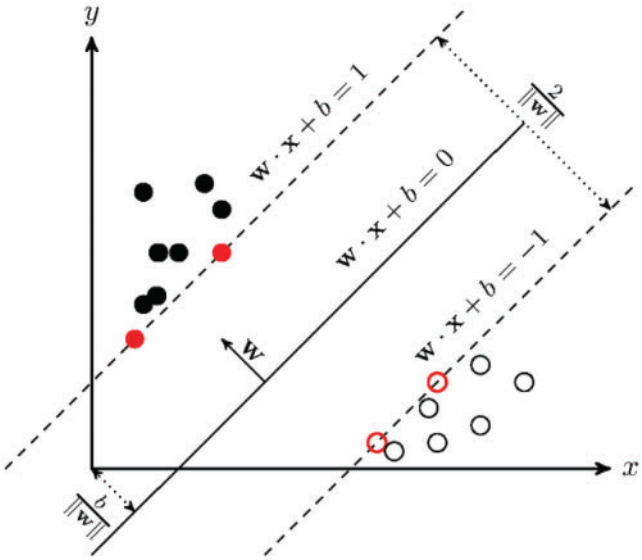


Figure 3: An illustration of SVM for linearly separable data

Here, b is an intercept and a bias term of the hyperplane equation. In the next phase, we have to find the line that segregates the data points in the optimal way by calculating the distance of points from a hyperplane. By finding the optimal hyperplane the miss-classification error can better be handled. Let the margin γ be defined as the distance from the hyperplane to the closest point across both classes. The closest points are known as the support vectors which play a vital role in assessing the marginal distance. Hyperplane can be placed in different directions, however, it is important to determine which hyperplane best alienates the data points. If γ is maximized than hyperplane ought to be somewhere in the middle of the two different classes. More specifically, the γ ought to be in a distance closest to the points in both the classes. In other case we just move the hyperplane to the points further away by increasing the γ In order to decide the optimal hyperplane, we compute the maximal marginal hyperplane (MMH) as a constrained optimization problem. The goal is to maximize the margin under the constraints that all features must lie on the correct side of the hyperplane. A much simpler formulation for the optimal solution would be the maximization of the minimization problem which is subsequently becomes the minimization as depicted in Eq. (5). The goal is to maximize the minimum distance.

$$\min_{\mathbf{w},b} \mathbf{w}^T\mathbf{w}, \qquad \text{such that} \qquad \forall i\, y_i \left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1 \tag{5}$$

where y_i is either $+1$ or -1, representing the two classes which may be either positive or negative to which x_i belongs. We need to identify the maximum marginal hyperplane which divides the point $y_i = +\ 1$ and $y_i = -1$. Here, w^T is the transpose of the normal vector to the hyperplane. Hence, the sentiment analysis task is formulated to determine which side of the hyperplane does the test instance falls into. In other words, if $(w^T x_i + b) \geq +\ 1$ then x_i falls into the positive class and if $(w^T x_i + b) \leq -1$ then x_i falls into the negative class.

The primary focus of SVM is to find the linear hyperplane to segregate the feature of user reviews into three classes that are positive, negative and neutral in case of ternary classification. In binary classification, the hyperplane alienates the mapped instances into two categories, i.e.,

positive and negative. Classifying the features decides the destiny of the user review class which may be one among the target labels.

An illustration of linearly separable data is illustrated in Fig. 3 whereby, black dots may depict positive class and white dots negative or neutral class. As we have opted OVR classification strategy, therefore, if black dots are positive than rest may treated as a single class. Ultimately, treating the multi-class problem as binary class problem. A linearly separable data ought to satisfy the two requirements as shown in Eq. (6).

$$\begin{aligned} \left(\mathbf{w}^T\mathbf{x}_i + b\right) &\geq +1 \text{ for } y_i = +1 \\ \left(\mathbf{w}^T\mathbf{x}_i + b\right) &\leq -1 \text{ for } y_i = -1 \end{aligned} \tag{6}$$

4 Experiments and Results

In this section, we discuss experiment design, dataset preparation, evaluation protocols, and results.

4.1 Corpus Generation and Statistics

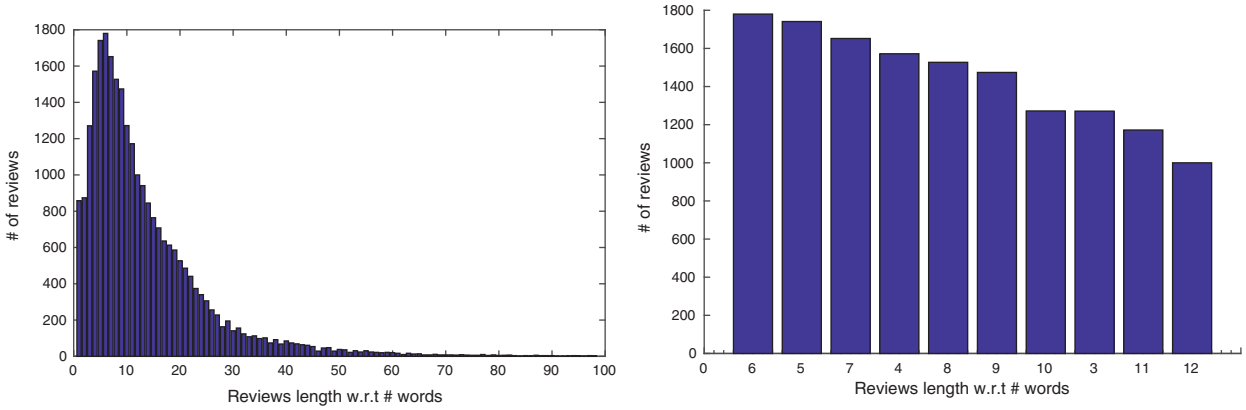
In our experimental settings, we introduce Roman Urdu e-commerce dataset (RUECD) to perform the sentiment analysis task. The Sentiment analysis task is performed for both binary and ternary classification. In binary classification positive and negative are the two classes whereas, for ternary classification we have three categories, i.e., negative (0), positive (1), and neutral (2). The RUECD dataset includes 26,824 user reviews having 9833 positive, 8252 negative and 8739 neutral reviews as illustrated in Table 4. The dataset is generated by the reviews of users on DarazPK. Different query terms are used and products are searched. Once the Roman Urdu based comments are extracted, then those are annotated by three experts. The experts labeled the comments based on annotation guidelines furnished to them. The whole annotation process and guidelines will be discussed in coming section. The dataset is very challenging and carefully created, it is an extension of our previous work [12]. It is also the largest dataset used for Roman Urdu. The dataset contains the shortest reviews of length 1 and the longest review of length 296 words. The average length of the reviews is 74 with standard deviation of 54. The median is also very close to the mean which is 66. The example of shortest views of length one and two are shown in Table 5, 40% of the reviews of length two are basically the tagging of friends or the users. The distribution of reviews length is show in Fig. 4. It can be seen that most of the reviews length is between 1–12 words, and very few long reviews exist. The long reviews are mostly easy to classify whereas short reviews are very tricky and contains many sarcastic expressions which hurts the accuracy of the model.

Table 4: RUECD Dataset description, there are total 26,824 reviews divided into two sets, i.e., training set and test set

Sentiment class	Training set	Test set	$\sum$
Negative (0)	6601	1651	8252
Positive (1)	7866	1967	9833
Neutral (2)	6991	1748	8739
$\sum$	21,458	5366	26,824

Table 5: Example of short reviews of length one (L1) and two (L2). The English of all Roman Urdu words are given in brackets

	Negative	Positive	Neutral
L1	juthi (lier)	mubarak (congratulations)	mubarak (congratulations)
	lanat (curse)	beshak (of course)	beshak (of course)
	bakwas (rubbish)	behtreen (best)	konse (which one)
	dukhon (sad)	ala (a slang)	lol
	fazooool (waste)	zabardast (great)	zaroor (definitely)
L2	chuta zahen (small mind)	no galazaat (no filthiness)	ji haazur (yes sir)
	sharam karo (shame on you)	shreef bcha (nobel child)	acha jee (Ok)
	fitay muh (a slang)	sahi kaha (well said)	bulkul bulkul (absolutely absolutely)
	dfa hoja (get out)	behtreen hogaya (That's great)	sachi yar (true buddy)
	toba kro (repent)	boot khuub (brilliant)	check kar (check it)
	jalti raho (stay jealous)	wah zbrdst (That's great)	kasay hoya (how went)



(a) (b)

Figure 4: The distribution of reviews length, (a) shows the overall distribution of length from 1 to 100, and (b) shows the top 10 most occurring reviews length

4.2 Annotation Process & Guidelines

This section is intended to deliberate the full annotation process that we opted to manually annotate the user reviews. The process involves the preparation of rules for annotation, manual annotation and subsequently estimating the inter-annotator agreement. Three annotators who are the native Urdu speakers and having excellent knowledge of Urdu and more specifically Roman Urdu were designated to annotate the dataset. A set of guidelines are prepared for annotators to be followed during the course of annotation.

4.2.1 Guidelines for Positive Class

A user review is counted as positive if all aspects in given text review are positive [50]. A user-review is combination of one or more sentences depicting sentiment against any product or services. If the user-review is expressing both positive and neutral sentiment, then positive class will be preferred [23]. All reviews that directly express sentiments of joy, thankfulness, optimism, enthusiasm, and hope will be marked as positive. Positive words like "acha" ("good"),

"khobsoorat" ("beautiful"), "behtareen" ("best"), "shandar" ("wonderful"), and "saaf" ("clean") should be used instead of negative ones like "Na", "Nahi" and "mat" as these terms flip the polarity [51]. In case there is agreement of approval then it will be marked as positive [52]. Sentences with illocutionary speech act like praise, congratulations are marked as positive [52].

4.2.2 Guidelines for Negative Class

If a sentence's aspect phrase exhibits a negative emotion, or there is sense of direct disagreement, then the sentence is categorised as negative [23]. A text review is marked as negative if the proportion of negative sentiment is higher than other sentiment classes. Sentences which includes banning, bidding, penalizing, or contains negative terms such as "bura" (bad), "bakwass" (rubbish), "zulam"(cruelness), "ganda" (dirty), and "mayosi"(dejection), without being modified by any other term like "Na", "Nahi", and "mat"; are considered negative [51,52]. All those reviews that express the feelings of sadness, anger, and violence are also marked as negative. If a statement has a negative word with a positive adjective, it is termed a negative sentence since the negative word negates the positive adjective [23].

4.2.3 Guidelines for Neutral Class

Sentence based on facts, or any thought is communicated, are considered neutral. Sentences with a low level of certainty and liability are classified as neutral sentences, such as "ho sakta hai (May be)", and "shayad (Perhaps)". A neutral sentence is one that has both positive and negative sentiments in terms of features and entities [23,50–52].

4.3 Corpus Annotation Process

The RUECD (Roman Urdu e-commerce dataset) dataset was manually annotated by human annotators. In order to prepare a benchmark dataset three annotators (α, β and γ) were designated to separately annotate the *corpus* data. The whole *corpus* were annotated by graduates who were native speakers of Urdu and also familiar with Roman Urdu. In order to evaluate annotation accuracy, we sampled a 300 reviews and assigned two annotators (α and β) to mark its polarity. The annotators were asked to follow the prepared annotation guidelines. Consequently, they introduced one of three categories: positive (1), negative (0) and neutral (2). The two annotators' conflicting pairings have been addressed, and the annotation guidelines have been modified as described above. Subsequently, the updated guidelines were use by two annotators for the whole *corpus*, in case of any disagreement, the third annotator (γ) was assigned to mark the polarity. We have achieved an Inter-Annotator Agreement (IAA) of 72.45% and Cohen Kappa score of 0.58 for the complete *corpus*. The accomplished moderate score depicts that prepared annotation guidelines were well understood and followed by the annotators.

The dataset is split into training and testing data with 80% and 20%, respectively. The training set is further divided into 80% and 20% subsets, where 80% set is used for training and 20% set is used for validation. The user reviews are represented using a vector space model with a TFIDF weighting scheme as discussed earlier. By computing the TFIDF against each document in the *corpus* we extract the features and embed them in a vector space. Moreover, the labels are converted into one-hot encoding to represent either of the three classes. Subsequently, this numerical text representation is fed into classification models. We have experimented with RCNN, Naive Bayes, KNN, Adaboost and support vector machine (SVM) for the sentiment classification task.

4.4 Evaluation Metric

In order to evaluate our model, we have used precision and recall as an evolution metric. The F1-score, overall accuracy and area under the ROC curve (AUC) are calculated to assess the performance of our classification model. This evaluation is based on the output of the confusion matrix as depicted in Figs. 5 and 6 which are True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). The confusion matrix is normalized for the better comprehension.

$$P = \frac{TP}{(TP + FP)} \tag{7}$$

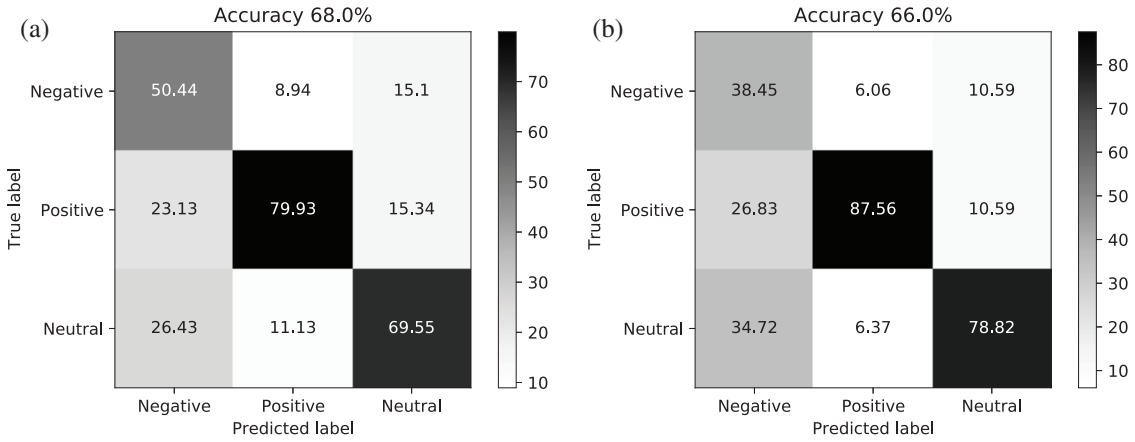


Figure 5: Confusion matrix output for multi-class sentiment classification on SVM of linear and sigmoid kernel. (a) Output of linear kernel (b) Output of sigmoid kernel

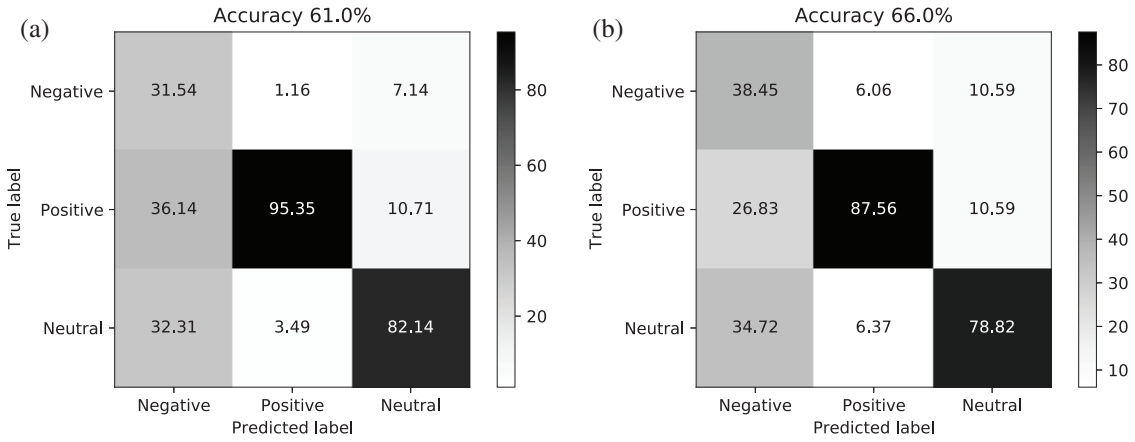


Figure 6: Confusion matrix output for multi-class sentiment classification on SVM of poly and RBF kernel. (a) Output of poly kernel (b) Output of RBF kernel

Precision (P) as shown in Eq. (7) is a measure that calculates the ratio of correctly labeled data by identifying how many document instances our model labeled belonged to the actual class label. The true positives (TP) depicts that prediction is +ve class and documents instance is actually +ve, whereas, the false positive (FP) is a false alarm which predicts the instance as +ve but belongs to −ve or neutral class.

$$R = \frac{TP}{(TP + FN)} \tag{8}$$

The recall (R) metric is used to determine the ratio of correctly +ve labeled by our model to all instances which were +ve. The recall is aimed at answering that among all the instances which are +ve, how many of those we correctly predict? Here in Eq. (8), the ratio of false-negative (FN) shows that how many instances were +ve but predicted as –ve which is the worst scenario. An earlier study on the relationship between precision and recall [53] has revealed that the higher the recall the better the model in terms of performance.

$$A = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{9}$$

Accuracy (A) is the proportion of the subjects correctly classified to the entire subject pool. This measure actually determines the overall performance of the model by identifying how many document instances were correctly labeled among all. In Eq. (9) the ratio of true negative (TN) shows that the instances which are predicted as −ve by our model are actually −ve.

$$\Omega = \frac{1}{2}\left(\frac{TP}{(TP + FN)} + \frac{TN}{(TN + FP)}\right) \tag{10}$$

The area under the ROC curve (Ω) exhibits a number of desirable properties over accuracy [54]. We have also considered AUC see Eq. (10) as an additional measure for assessing the performance of our proposed model.

$$F = 2.\frac{(P.R)}{(P + R)} \tag{11}$$

In order to seek the balance between precision and recall the F-measure is used. The F-measure (F) as shown in Eq. (11) is aimed at calculating the harmonic mean or average of the precision (P) and recall (R). A large number of True Negatives contribute to accuracy, which we do not focus on much in most business situations, whereas False Negative and False Positive have business costs (tangible and intangible), so F1-score might be a better measure to use if we need to find a balance between Precision and Recall.

4.5 Sentiment Classification

In our empirical evolution, we have used different machine learning (ML) and deep learning classification schemes. In order to compare the experimental results of the different deep learning model we have reproduced the previously proposed approaches so as to evaluate the power of both deep learning and machine learning models. Our proposed support vector machine (SVM) based on vector space model and powered by stemming has shown superior performance over the rest of machine learning and deep learning models. The experimental configuration for SVM is based on four different kernels that are linear, polynomial, sigmoid and RBF kernel to assess the

performance. In case of linear and sigmoid kernel we have tuned the value of C (a regularization parameter) to 0.86 to place a maximum marginal length hyperplane to alienate the features. Whereas, for polynomial kernel we have to set a larger value of C, i.e., 1000 which allows the low marginal length hyperplane. Similarly, the RBF kernel also entails a larger value of C which means we are allowing only few outliers. The experimental results of different kernels on SVM as shown in Figs. 5 and 6. It can be seen that the linear kernel outperforms all kernels. The linear kernel has shown superior performance with 68% accuracy which evident in Fig. 5a. On the other hand, sigmoid and RBF kernel as depicted in Figs. 5b and 6b too shown the better performance with 66% and 67% accuracy, respectively. However, the polynomial kernel as illustrated in Fig. 6a has achieved 61%–64% accuracy beside avoiding the outliers with greater value of C. Moreover, we have used One-vs-Rest (OvR) classification strategy with SVM, which is aimed at dealing the multi-class classification as multiple binary classification problems. The experimental results of different classification approaches for Roman Urdu sentiment analysis are illustrated in Table 6. The results depicts that our proposed SVM outperforms all baseline models, however, some deep learning and machine learning models has also shown good performance. It is worth mentioning here that Logistic regression has also shown a performance at par with our proposed scheme. On the other hand, among deep learning models LSTM [24] has shown better performance. The comparison includes different evaluation metrics, i.e., precision, recall, F1-score and overall accuracy of the model.

Table 6: Comparison of proposed framework with the baseline related models

Models	Precision	Recall	F1-score	Accuracy
KNN [17]	**0.70**	0.37	0.48	0.47
Naive Bayes [18]	**0.64**	0.59	0.61	0.62
Adaboost [19]	0.61	0.58	0.59	0.61
Random forest [21]	0.63	0.61	0.62	0.59
Decision tree	0.59	0.56	0.58	0.58
Logistic regression [22]	0.67	0.67	0.67	0.67
SVM [20]	0.59	0.58	0.58	0.61
RCNN [23]	**0.64**	0.62	0.63	0.63
LSTM [24]	**0.65**	**0.64**	**0.65**	**0.66**
CNN	0.63	0.62	0.62	0.63
RNN	0.63	0.63	0.63	0.63
GRU [13]	0.62	0.61	0.62	0.62
Proposed SVM	**0.68**	**0.71**	**0.69**	**0.68**

The precision and recall curves as illustrated in Fig. 7. It can be seen that the negative class has initially gained the precision but later incurred a decline with 0.69 average precision, the neutral class has shown good performance and scored an average precision of 0.67. Whereas, the positive class has outperformed the rest of the two classes and has gained an average precision of 0.80, the area under the precision-recall curve of these thrice classes has been observed as 0.73 which is quite satisfactory.

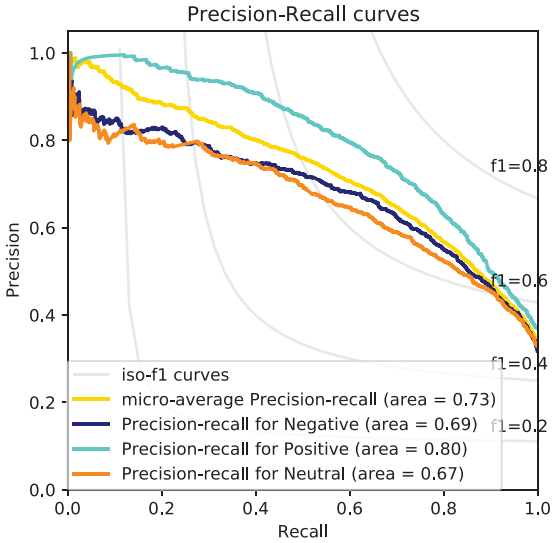


Figure 7: Precision and recall curves

The area under the curve (AUC) of ROC is another significant measure to determine the performance of our proposed model as illustrated in Fig. 8. The higher the AUC, the better the performance of the model in classifying the user reviews to either of three classes. It can be observed that AUC which is 0.83 is almost close to 1 which depicts that our classification model has performed better. Whereas, the individually calculated AUC of three classes reveals that the positive class has outperformed the negative and neutral class with an AUC of 0.84.

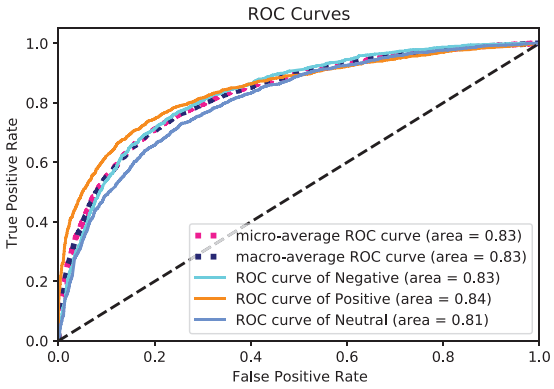


Figure 8: Area under the curve (AUC) of ROC

4.6 Results and Discussion

The experimental results as illustrated in Table 6 clearly show that our proposed SVM approach with stemming outperforms the rest of classification schemes with an accuracy of 68% which is clearly beating both the deep learning models and machine learning models. In our experimental settings we have used different kernels with SVM. However, linear and poly kernel has shown superior performance with smaller value of C. The C parameters guides the SVM optimizer that how much misclassification could be avoided. The larger value of C chooses the

smaller-margin hyperplane which usually faces miss-classification errors. Conversely, a very small value of C will prompt the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points. The experimental trial on Roman Urdu text revealed that a smaller value of C i.e. between 0 to 1 can better guide the SVM optimizer to correctly classify the sentiment. On the other hand, setting a larger value C such as 10000 cause the optimizer to increase the misclassification. Setting a more tiny value such as $C = 0$ will prompt the optimizer to not consider the error in classification.

The class-wise accuracy of our proposed model as recorded in Table 7 clearly shows that positive class outperformed in all departments, i.e., precision, recall, F1-score, and accuracy. However, the neutral and negative class has little suffered and observed a decline in precision and recall. The SVM recipe proposed by Noor et al. [12] also recorded in Table 6 has achieved an accuracy of 60% which is quite low as compared to our fine-tuned methodology. It can be observed that our proposed model has outperformed the deep learning model RCNN. The RCNN [23] model has achieved an accuracy of 63%. Examining the classification report as illustrated in Fig. 7 of our proposed model it is evident that positive class has shown better performance in both measures of precision and recall. Whereas, rest of the two classes has achieved a slightly less score in terms of precision and recall. The KNN [17] has shown good performance in terms of precision that is 70%, however, the recall is very low which has gravely affected the accuracy and scored 47%. Adaboost [19] algorithm which amalgamates the power of different classification algorithms has shown good performance over KNN with 61% accuracy. On the other hand, Naive Bayes [18] has shown improvement with an accuracy of 62% but unable to beat the SVM. The Adaboost algorithm has shown almost similar performance to Naive Bayes in terms of accuracy. The Random Forest [21] approach has achieved a performance almost at par with Naive Bayes but the overall accuracy is below 60% which is not welcoming. We have also experimented Decision Tree (DT) which has also shown an average level of performance with 59% precision, 57% recall and 0.58 F1-score. It was also observed that DT may only yield good results if max tree depth is tuned to almost 100 but still in our case it has achieved a maximum 58% accuracy. The Logistic Regression (LR) which is famous for regression and classification problem has shown a performance at par with our proposed SVM with 67% accuracy.

Table 7: Class-wise precision, recall, F1-score and accuracy on RUECD dataset using SVM with stemming

Sentiment class	Precision	Recall	F1-score	Accuracy
Negative (0)	0.66	0.66	0.66	
Positive (1)	0.73	0.71	0.72	
Neutral (2)	0.64	0.66	0.65	
Accuracy				0.68

On the other hand, deep learning models has also shown a better performance. We have experimented with LSTM which is famous for handling sequential data such as text or sound. In our experimental pool of deep learning models which includes LSTM [24], CNN, Classic RNN, RCNN [23] and GRU [24], the LSTM has outperformed the rest of deep learning models and certain machine learning models too with 66% accuracy. The rest of the models: RCNN, CNN, RNN and GRU has shown almost similar performance with 63% to 62% accuracy. However, RCNN has little gained in terms of precision. It's worth noting that word embedding is used

as the initial layer in all deep learning models. Deep neural networks which are often praised in contrast to machine learning models have an issue that it suffers from over-fitting owing to too many parameters. Whereas, machine learning models has only few parameters to tune which consequently prevent the model from over-fitting problem at some extent.

The empirical evaluation of both deep learning and machine learning models depicts that our proposed model has out-preformed all the baseline models. Moreover, the performance of our classification model based on the vector space model and SVM can be accessed via precision and recall curve and AUC. By fine-tuning the hyper-parameters and using different kernel functions we assessed the performance of our proposed model. The empirical evaluation shows that stemming has a significant effect on accuracy. By performing the stemming via our lexical normalization dictionary we have successfully reduced the anomalies from our text data which consequently improved the overall performance of the model. Furthermore, the systematic annotation process of the *corpus* has also played a vital role in solving the Roman Urdu sentiment analysis problem. The strategy of one-vs-rest (OvR) classification has proved fruitful in ameliorating the accuracy. Opting this strategy allows us to slice down the multi-class problem into the number of binary class problems which is inherently supported by SVM. The dataset is quite challenging that all famous models have accuracy below 70% which motivates the researchers to engage themselves to improve the accuracy. Roman Urdu is challenging as one word can be described by several possible spellings, and depending on the context, the meaning of the same word can be very different.

5 Conclusions and Future Work

Recent advancements in e-commerce across the world have changed business models. People prefer online shopping these days rather than going to malls physically. All online vendors give value to the comments received by the customers, and customers read all the comments before buying any product. Sentiment analysis plays a vital role in decision making for the vendors and also for the customers in e-commerce. There are several e-commerce portals in Pakistan in which DarazPK is very famous. Many users write their feedback in Roman Urdu or Urdu fonts. In this paper, we extracted Roman Urdu comments against several products and created our database. We annotated the comments based on expert opinions, as explained in the experimental section, and trained several famous machine learning and deep learning classifiers on this dataset. Experiments show that Roman Urdu sentiment analysis is a very challenging problem. Famous models such as LSTM, RNN, CNN, RCNN and GRU have received low accuracy. We also extended our previous SVM model by including the lexically normalized words of Roman Urdu and achieved an accuracy of 0.68. The results clearly depict that our proposed model clearly outperformed the baseline models. Moreover, a comparative study reveals that SVM, Logistic regression and LSTM has shown superior performance on the rest of the models. The dataset and scripts are kept online to easily re-implemented, extended, and used for research and commercial purposes. In our future work, we will extend this model for multilingual comments on Pakistan e-commerce portals. The languages used for comments on portals are either English, Urdu, or Roman Urdu.

Funding Statement: The authors would like to thank the Deputy for Study and Innovation, Ministry of Education, Kingdom of Saudi Arabia, for funding this research through a Grant (NU/IFC/INT/01/008) from the Najran University Institutional Funding Committee.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Nankani, H., Dutta, H., Shrivastava, H., Krishna, P. R., Mahata, D. et al. (2020). Multilingual sentiment analysis. In: *Deep learning-based approaches for sentiment analysis,* pp. 193–236. Berlin: Springer.
2. Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies, 5(1),* 1–167. DOI 10.2200/S00416ED1V01Y201204HLT016.
3. Feng, A., Gao, Z., Song, X., Ke, K., Xu, T. et al. (2020). Modeling multi-targets sentiment classification via graph convolutional networks and auxiliary relation. *Computers, Materials & Continua, 64(2),* 909–923. DOI 10.32604/cmc.2020.09913.
4. Medhat, W., Hassan, A., Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal, 5(4),* 1093–1113. DOI 10.1016/j.asej.2014.04.011.
5. Khushboo, T. N., Vekariya, S. K., Mishra, S. (2012). Mining of sentence level opinion using supervised term weighted approach of nave bayesian algorithm. *International Journal of Computer Technology and Applications, 3(3*), 987–991.
6. Thelwall, M., Buckley, K., Paltoglou, G. (2012). Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology, 63(1),* 163–173. DOI 10.1002/asi.21662.
7. van Engelen, J. E., Hoos, H. H. (2020). A survey on semi-supervised learning. *Machine Learning, 109(2),* 373–440. DOI 10.1007/s10994-019-05855-6.
8. Lin, C., He, Y. (2009). Joint sentiment/topic model for sentiment analysis. *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, Hong Kong, China.
9. Saif, H., He, Y., Fernandez, M., Alani, H. (2016). Contextual semantics for sentiment analysis of Twitter. *Information Processing & Management, 52(1),* 5–19. DOI 10.1016/j.ipm.2015.01.005.
10. Bakshi, R. K., Kaur, N., Kaur, R., Kaur, G. (2016). Opinion mining and sentiment analysis. *3rd International Conference on Computing for Sustainable Global Development*, vol. 2, pp. 452–455. New Delhi, India: IEEE.
11. Qutab, I., Malik, K. I., Arooj, H. (2020). Sentiment analysis for Roman Urdu text over social media, a comparative study. *CoRR*, *9(5),* 217–224.
12. Noor, F., Bakhtyar, M., Baber, J. (2019). Sentiment analysis in e-commerce using SVM on Roman Urdu text. *International Conference for Emerging Technologies in Computing*. Berlin: Springer.
13. Mehmood, K., Essam, D., Shafi, K., Malik, M. K. (2020). An unsupervised lexical normalization for Roman Hindi and Urdu sentiment analysis. *Information Processing & Management, 57(6),* 102368. DOI 10.1016/j.ipm.2020.102368.
14. Yang, L., Li, Y., Wang, J., Sherratt, R. S. (2020). Sentiment analysis for e-commerce product reviews in Chinese based on sentiment lexicon and deep learning. *IEEE Access, 8,* 23522–23530. DOI 10.1109/ACCESS.2020.2969854.
15. Chen, F., Xia, J., Gao, H., Xu, H., Wei, W. (2021). TRG-DAtt: The target relational graph and double attention network based sentiment analysis and prediction for supporting decision making. *ACM Transactions on Management Information Systems, 13(1),* 1–25. DOI 10.1145/3462442.
16. Wu, H., Liu, Y., Wang, J. (2020). Review of text classification methods on deep learning. *Computers, Materials & Continua, 63(3),* 1309–1321. DOI 10.32604/cmc.2020.010172.
17. Bilal, M., Israr, H., Shahid, M., Khan, A. (2016). Sentiment classification of Roman-Urdu opinions using Nave Bayesian, decision tree and KNN classification techniques. *Journal of King Saud University-Computer and Information Sciences, 28(3),* 330–344. DOI 10.1016/j.jksuci.2015.11.003.
18. Domingos, P., Pazzani, M. (1997). On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning, 29,* 103–130. DOI 10.1023/A:1007413511361.
19. Cao, Y., Miao, Q. G., Liu, J., Gao, L. (2013). Advance and prospects of adaboost algorithm. *Acta Automatica Sinica, 39(6),* 745–758. DOI 10.1016/S1874-1029(13)60052-X.
20. Mehmood, F., Ghani, M. U., Ibrahim, M. A., Shahzadi, R., Mahmood, W. et al. (2020). A precisely xtreme-multi channel hybrid approach for Roman Urdu sentiment analysis. *IEEE Access, 8,* 192740–192759. DOI 10.1109/ACCESS.2020.3030885.
21. Fang, X., Zhan, J. (2015). Sentiment analysis using product review data. *Journal of Big Data, 2(1),* 1–14. DOI 10.1186/s40537-015-0015-2.

22. Shah, K., Patel, H., Sanghvi, D., Shah, M. (2020). A comparative analysis of logistic regression, random forest and KNN models for the text classification. *Augmented Human Research, 5(1),* 1–16. DOI 10.1007/s41133-020-00032-0.
23. Mahmood, Z., Safder, I., Nawab, R. M. A., Bukhari, F., Nawaz, R. et al. (2020). Deep sentiments in Roman Urdu text using recurrent convolutional neural network model. *Information Processing & Management, 57(4),* 102233. DOI 10.1016/j.ipm.2020.102233.
24. Ghulam, H., Zeng, F., Li, W., Xiao, Y. (2019). Deep learning-based sentiment analysis for Roman Urdu text. *Procedia Computer Science, 147(8),* 131–135. DOI 10.1016/j.procs.2019.01.202.
25. Ikoro, V., Sharmina, M., Malik, K., Batista-Navarro, R. (2018). Analyzing sentiments expressed on Twitter by UK energy company consumers. *Fifth International Conference on Social Networks Analysis, Management and Security*, Valencia, Spain.
26. Behera, R. K., Jena, M., Rath, S. K., Misra, S. (2021). Co-LSTM: Convolutional lstm model for sentiment analysis in social big data. *Information Processing & Management, 58(1),* 102435. DOI 10.1016/j.ipm.2020.102435.
27. Zainuddin, N., Selamat, A. (2014). Sentiment analysis using support vector machine. *2014 International Conference on Computer, Communications, and Control Technology*, Langkawi, Malaysia, IEEE.
28. Moraes, R., Valiati, F., Gavião Neto, W. P. (2013). Document-level sentiment classification: An empirical comparison between SVM and ANN. *Expert Systems with Applications, 40(2),* 621–633. DOI 10.1016/j.eswa.2012.07.059.
29. Alqaryouti, O., Siyam, N., Monem, A. A., Shaalan, K. (2020). Aspect-based sentiment analysis using smart government review data. *Applied Computing and Informatics, (ahead-of-print),* 1143. DOI 10.1016/j.aci.2019.11.003.
30. Altnel, B., Ganiz, M. C. (2018). Semantic text classification: A survey of past and recent advances. *Information Processing & Management, 54(6),* 1129–1153. DOI 10.1016/j.ipm.2018.08.001.
31. Paltoglou, G., Gobron, S., Skowron, M., Thelwall, M., Thalmann, D. (2010). Sentiment analysis of informal textual communication in cyberspace. *Proceedings of Engage 2010*, pp. 13–25. Springer LNCS State-of-the-Art Survey, Vienna, Austria.
32. Pang, B., Lee, L., Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. *International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 79–86. Prague, Czech Republic: Association for Computational Linguistics.
33. Patil, G., Galande, V., Kekan, V., Dange, K. (2014). Sentiment analysis using support vector machine. *International Journal of Innovative Research in Computer and Communication Engineering, 2(1),* 2607–2612.
34. Mehmood, K., Essam, D., Shafi, K., Malik, M. K. (2019). Discriminative feature spamming technique for Roman Urdu sentiment analysis. *IEEE Access, 7,* 47991–48002. DOI 10.1109/ACCESS.2019.2908420.
35. Naqvi, R. A., Khan, M. A., Malik, N., Saqib, S., Alyas, T. et al. (2020). Roman Urdu news headline classification empowered with machine learning. *Computers, Materials & Continua, 65(2),* 1221–1236. DOI 10.32604/cmc.2020.011686.
36. Bilal, M., Israr, H., Shahid, M., Khan, A. (2016). Sentiment classification of Roman-Urdu opinions using Nave Bayesian, decision tree and KNN classification techniques. *Journal of King Saud University-Computer and Information Sciences, 28(3),* 330–344. DOI 10.1016/j.jksuci.2015.11.003.
37. Rafique, A., Malik, M. K., Nawaz, Z., Bukhari, F., Jalbani, A. H. et al. (2019). Sentiment analysis for Roman Urdu. *Mehran University Research Journal of Engineering & Technology, 38(2),* 463–470. DOI 10.22581/muet1982.1902.20.
38. Habiba, R., Awais, D. M., Shoaib, D. M. (2020). A technique to calculate national happiness index by analyzing Roman Urdu messages posted on social media. *ACM Transactions on Asian and Low-Resource Language Information Processing, 19(6),* 1–16. DOI 10.1145/3400712.
39. Sharf, Z., Rahman, S. U. (2017). Lexical normalization of Roman Urdu text. *International Journal of Computer Science and Network Security, 17(12),* 213–221.
40. Sharf Zareen, R. S. U. (2018). Performing natural language processing on Roman Urdu datasets. *International Journal of Computer Science and Network Security, 18(1),* 141–148.

41. Javed, I., Afzal, H., Majeed, A., Khan, B. (2014). Towards creation of linguistic resources for bilingual sentiment analysis of Twitter data. *International Conference on Applications of Natural Language to Data Bases/Information Systems*, pp. 232–236. Montpellier, France: Springer, Cham.
42. Awais, D. M., Shoaib, D. M. (2019). Role of discourse information in urdu sentiment classification: A rule-based method and machine-learning technique. *ACM Transactions on Asian and Low-Resource Language Information Processing, 18(4),* 1–37. DOI 10.1145/3300050.
43. Do, Q., Gaspers, J. (2019). Cross-lingual transfer learning with data selection for large-scale spoken language understanding. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 1455–1460. Hong Kong, China.
44. Basiri, M. E., Nemati, S., Abdar, M., Cambria, E., Acharya, U. R. (2021). ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis. *Future Generation Computer Systems, 115(3),* 279–294. DOI 10.1016/j.future.2020.08.005.
45. Cao, B., Wu, J., Wang, S., Gao, H., Fan, J. et al. (2021). Unsupervised derivation of keyword summary for short texts. *ACM Transactions on Internet Technology, 21(2),* 1–23. DOI 10.1145/3397162.
46. Saeed, H. H., Ashraf, M. H., Kamiran, F., Karim, A., Calders, T. (2021). Roman Urdu toxic comment classification. *Language Resources and Evaluation, 1–26(4),* 971–996. DOI 10.1007/s10579-021-09530-y.
47. Li, G., Zheng, Q., Zhang, L., Guo, S., Niu, L. (2020). Sentiment infomation based model for chinese text sentiment analysis. *2020 IEEE 3rd International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*, pp. 366–371. Shenyang, China: IEEE.
48. Willett, P. (2006). The porter stemming algorithm: Then and now. *Program, 40(3),* 219–223. DOI 10.1108/00330330610681295.
49. Porter, M. F. (2001). Snowball: A language for stemming algorithms. http://snowball.tartarus.org/texts/introduction.html.
50. Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S. et al. (2016). SemEval-2016 task 5: Aspect based sentiment analysis. *Proceedings of the 10th International Workshop on Semantic Evaluation*. San Diego, California, Association for Computational Linguistics.
51. Mehmood, K., Essam, D., Shafi, K., Malik, M. K. (2019). Sentiment analysis for a resource poor language–Roman Urdu. *ACM Transactions on Asian and Low-Resource Language Information Processing, 19(1)*, 1–15. DOI 10.1145/3329709.
52. Ayyaz, S., Qamar, U., Nawaz, R. (2018). HCF-CRS: A hybrid content based fuzzy conformal recommender system for providing recommendations with confidence. *PLoS One, 13(10),* e0204849. DOI 10.1371/journal.pone.0204849.
53. Buckland, M., Gey, F. (1994). The relationship between recall and precision. *Journal of the American Society for Information Science, 45(1),* 12–19. DOI 10.1002/(ISSN)1097-4571.
54. Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition, 30(7),* 1145–1159. DOI 10.1016/S0031-3203(96)00142-2.