Received July 21, 2021, accepted August 6, 2021, date of publication August 12, 2021, date of current version August 23, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3104308

UTSA: Urdu Text Sentiment Analysis Using Deep Learning Methods

UZMA NAQVI, ABDUL MAJID, AND SYED ALI ABBAS
Department of Computer Sciences and Information Technology, The University of Azad Jammu & Kashmir, Muzaffarabad 13100, Pakistan
Corresponding author: Syed Ali Abbas (ali.abbas@ajku.edu.pk)

ABSTRACT The Internet has seen substantial growth of regional language data in recent years. It enables people to express their opinion by incapacitating the language barriers. Urdu is a language used by 170.2 million people for communication. Sentiment analysis is used to get insight of people opinion. In recent years, researchers' interest in Urdu sentiment analysis has grown. Application of deep learning methods for Urdu sentiment analysis has been least explored. There is a lot of ground to cover in terms of text processing in Urdu since it is a morphologically rich language. In this paper, we propose a framework for Urdu Text Sentiment Analysis (UTSA) by exploring deep learning techniques in combination with various word vector representations. The performance of deep learning methods such as Long Short-Term Memory (LSTM), attention-based Bidirectional LSTM (BiLSTM-ATT), Convolutional Neural Networks (CNN) and CNN-LSTM is evaluated for sentiment analysis. Stacked layers are applied in sequential model LSTM, BiLSTM-ATT, and C-LSTM. In CNN, various filters are used with single convolution layer. Role of pre-trained and unsupervised self-trained embedding models is investigated on sentiment classification task. The results obtained show that the BiLSTM-ATT outperformed other deep learning models by accomplishing 77.9% accuracy and 72.7% F1 score.

INDEX TERMS Sentiment analysis, deep learning models, Urdu language processing, sentence classification, word embeddings.

I. INTRODUCTION

Social media forums, blogs, comments, and reviews provide opinionated data about issues, products, and services. As in recent pandemic period, a sudden burst of internet usage has been reported [1]. According to Statista [2], there are 4.66 billion active internet users till October 2020. Increased usage of internet encouraged it to transform from monolingual to multilingual platform.

In the last decade, the presence of different language websites, including Urdu, has substantially increased. Urdu is an official language of Pakistan and India's schedule language used by millions of people worldwide for communicating. Most visited sites in Pakistan offer their content in Urdu [3]. Urdu presents some challenges for language processing, such as Urdu uses formal and informal verb forms, and each noun has an either masculine or feminine gender. Similarly, Urdu language has loan words from Persian, Arabic and Sanskrit languages. Urdu is written from right to left and boundary between words is not always distinguishable such as 'ادھرکیارکھاہے' (what lays there) is understandable, although it has no space between words.

Sentiment Analysis (SA) is a term used to extract subjective information by applying natural language processing techniques. It is used to classify text as positive(p), neutral(o) or negative(n) regarding a product, service, topic, event etc. SA helps researchers to explore trends which in turns help in strategic planning. SA is a classification process that can be performed at document level, sentence level and aspect level [4]. At Document-level, document is classified as negative or positive based on document text expression. If the document contains more positive sentences, it is classified as positive, classified as negative if more negative sentences than positive, neutral if the number of positive sentences is equal to number of negative sentences. It does not apply to documents that contain multiple entities to evaluate. At Sentence level classification, the first step is to identify whether the sentence is subjective or objective. An objective sentence gives some information, while a subjective sentence provides views and opinions. In the case of subjective, it determines whether it

The associate editor coordinating the review of this manuscript and approving it for publication was Zhan Bu.

is negative, positive, or neutral. Aspect level determines the sentiments of identified entities at very fine-grained level. It identifies both sentiments about aspects and associated targets [5].

There are several approaches for sentiment analysis: lexicon-based, Machine Learning (ML) and Deep Learning (DL). Lexicon-based methods make use of a words list and associated word sentiment. Two automated approaches are used to compile words list. One is dictionary-based approach that works with known orientation words and by searching synonyms and antonyms in publicly available word collections to add more words in the original list. This process works iteratively until no new word is found. Sentiment score is calculated by taking the summation of polarity of lexicons in the text if found in dictionary. Deduction is made in sentiment score in case of lexicon with negative polarity. The construction of a lexicon dictionary is possible manually (Wordnet, VADER), automatically or semi-automatically (SenticNet). Most frequently used lexicons are unigrams. However only using unigrams is not sufficient for SA such as a same word may has opposite orientation in different circumstances. Main disadvantage of this approach is unavailability of domain specific lexicons [6]. The solution of dictionary-based is corpus-based approach in which list of seed words (adjectives) is expanded by using the corpus of same domain documents. Words that appear together expected to have same polarity. On the other hand, ML methods depend upon supervised classification approaches to detect and frame sentiment as positive or negative. However, this approach requires labeled data to train classifiers. Several probabilistic, linear, decision tree, rule based, and unsupervised methods can be used for SA.

When domain understanding is deficient for feature learning, DL techniques outperform others when used for feature engineering [7]. DL provides excellent benefits for complex problems such as image classification, natural language processing, and speech recognition. DL for sentiments analysis is a preferred approach than conventional machine learning methods. DL methods such as recurrent neural networks (RNN) and convolutional neural networks (CNN) outperformed manual feature generation-based classifiers [8]. DL network performance continues to enhance as more data is available. DL techniques can be made to work perfectly well, even with small datasets, through fine-tuning and transfer learning [9]–[12]. The execution time is relatively higher for DL models as it need to be trained with large size data set, but once it is trained, it takes less time to test. RNN is used for sequential and time series data through the hidden units. The Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM) and Gated Recurrent Unit (GRU) are the popular variations in RNN-based representation learning methods. These forms of networks have shortcomings, longer sentences are difficult to handle as all the source information is compressed into fixed-size vector. Attention mechanism can be applied along with these neural networks to attain better performance [13]. CNN is recently got attention for text classification. CNN uses convolutional filters to recognize patterns in data. Processing time of CNN based models is ten times less than the RNN based models [14]. Textual data is converted to numerical form for DL classification models to learn data patterns. Selection of word representation can affect the classification. Probability of accurate classification can be increased if the encoding represents semantic relationship between words. Word embeddings are vector representations of words in the given data that capture the context of the underlying words in relation to other words in the sentence. It means same word that appears in different contexts will have different vector representation. Word embedding models can provide pre-trained word embedding and can also be finetuned to create embeddings for the target data. DL models using word embedding achieve better performance [15]. There are several word embedding models such as Word2Vec [16] and fastText [17].

Contents shared on social media (except reviews) are mostly very noisy. Tools such as POS taggers and parsers are required to rectify data for accurate classification. Therefore, a significant amount of preprocessing is required before any analysis. Adjectives are mostly used as sentiment units, but nouns are also good sentiment carriers and improve performance. The key to sentiment classification is the selection of a set of effective features. Some of the features are Bag of Words (BOW), Parts of Speech (POS), sentiment words, rules of opinion, sentiment shifters, and syntactic dependencies.

Very limited work has been done for Urdu text sentiment analysis [18], [19]. Techniques developed for other languages such as English could not be applied in Urdu SA. There are several challenges in Urdu language SA as it is a morphologically rich language. It still lacks language processing resources such as stop words lists, lemmatization and stemming tools as compared to other languages. In Urdu, to tokenize, spaces have to be inserted/removed between words as the boundary between words is not clearly marked, such as 'کتنااچھاکھاناہے'. Word order in Urdu sentence may be different, but the meaning would remain the same such as 'میٹھے آم ہیں' and 'آم میٹھے ہیں' have the same sense.

The purpose of current study is to search optimal solution for SA of Urdu data in digital format. In this paper, sentence level Urdu Text Sentiment Analysis (UTSA) by exploring deep learning techniques in combination with various word vector representations is proposed. Main contribution of research is as follows.

1) A labelled dataset consisting of 6000 sentences labelled with sentiments(p,n) is developed. No such public dataset is available as best of our knowledge. The dataset will be made available publicly.
2) Data set is preprocessed by removing stop words, foreign language characters, punctuations and alphanumeric characters for noise reduction and enhancing feature extraction. As very few preprocessing resources are available so supporting data lists are created.
3) We proposed the framework where various DL techniques such as of stacked LSTM, BiLSTM-ATT, CNN,

and C-LSTM are explored for sentiment classification. We investigated the classification performance based on pre-trained vector models trained on large corpora and self-trained models trained on our dataset.

4) The performance of our proposed framework (UTSA) on Urdu language dataset is analyzed. Furthermore, experiment results are provided to show the performance of DL methods for Urdu text sentiment classification, especially LSTMs robustness in handling Urdu text.

II. LITERATURE REVIEW

SA is the most researched topic for text classification in the last decade. In this section we discussed approaches in literature applied for classification of textual data written in Urdu language.

A. LEXICON-BASED APPROACH

SentiUnits [18] were created by identifying sentiment words/phrases in the sentence. That also included orthographic, phonological, syntactic, and morphological features along the word. Sentence polarity was computed by adding the polarity of the SentiUnits. In their later work [20], authors associated SentiUnits with their targets through shallow parsing chunking. Lexicon expanded to include nominal appraisal head words, and modifiers. They determined sentence polarity by identifying all SentiUnits for a specific target. They also presented SenitUnits in JSON format and two step classification model in [21]. Along with the adjectives, nouns, and negations, they also included verbs, intensifiers, and context dependent words for lexicon-based SA. Authors [22] found lexicon-based approach performed way better than supervised machine learning methods for sentiment classification. They utilized adjectives, verbs, and nouns as sentiment carrier words along with negation intensifiers and context dependent words. Asghar *et al.* [23] created the Urdu sentiment lexicon collection SentiUrduNet by translating English opinion words to equivalent Urdu words. Similarly, Urdu language modifiers were translated to English to obtain sentiment scoring. Manual-driven scoring was applied to words where the sentiment score was not provided or was incorrect. Hassan and Shoaib [24] presented a SEGMODEL that probed sentence sentiment when a sub-opinion was considered. The polarity of the overall sentence was computed by utilizing the polarity of each subsentence. Their experimental results showed that their technique performed better than the BOW technique by achieving 75.8 accuracy.

B. MACHINE LEARNING (ML) APPROACH

Mukhtar *et al.* [25] proved that Lib SVM, J48, and IBK were the best classifiers for classification of Urdu text sentiment. In [19] Mukhtar *et al.* applied supervised ML methods for sentence level SA. They extracted 154 features such as neutral, positive, negative, negation and intensifier to improve the classification through SVM, KNN and decision tree. Authors claimed that KNN performed better than rest of the classifiers in terms of accuracy. Awais and Shoaib [26] identified sub-opinions by using discourse information which they then fed to rule-based and supervised methods. In their observation, the rule-based classifier performed better than BOW and the model based on ML techniques trained with discourse features performed significantly better than the model trained without discourse features. They discovered that ML methods outperform rule-based methods if training data is available; otherwise, rule-based methods are a better choice for sentiment classification. Nasim and Ghani [27] developed a 3-class (positive, negative, and neutral) sentiment classification model for tweets in Urdu language using Markov Chains. As compared to lexicon-based and other ML methods their approach gained better accuracy.

C. DEEP LEARNING BASED APPROACH

In recent years, deep learning methods have been explored for Urdu text classification. Akhter *et al.* [28] applied deep learning for Urdu document classification in product manufacturing. They eliminated rare words as well as stop words, which increased accuracy for small and medium-sized datasets but decreased accuracy for large datasets. Their experiments revealed that CNN with multiple filters (3,4,5) achieved the highest rank, while BilSTM outperformed LSTM and CLSTM. In [27], the authors used a single layer CNN with multiple filters for document-level text classification and found it superior to the baseline methods. Asim *et al.* [29] assessed performance of state-of-the-art ML, DL, and hybrid model for document classification. Their experiments showed that the normalized difference measure-based feature selection approach improved the performance of all models. LSTM is explored by [30] for sentiment analysis of roman Urdu text and found that their DL model outperformed baseline ML methods.

By looking into Table 1, we can conclude that although lexicon-based sentiment classification performed better than ML methods for sentiment analysis of Urdu text, however, the development of a sentiment lexicon necessitates a large number of manual operations and is highly dependent on dictionary size. Rules are made in accordance with domain knowledge for better classification, but data especially on social media deviate a lot from linguistic rules so they are less effective. At the same time ML approaches require labelled data and lack the domain knowledge. This can be improved by adding positive words, negative words, and negation as features.

As DL methods use word embedding as inputs, which already lists the semantic data and learns the relation between data elements in the sentence. So, there is no need to include rules or special features along with the text. Word embeddings can be created through embedding models such as fasttext [31], word2vec [16] or pre-trained embeddings can be used that provide transfer learning for the languages that lack such resources. We selected DL methods due to their effectiveness proven in sentiment classification as they have yet to be investigated for Urdu text. There are also Urdu language embeddings available, such as fasttext [32], Urdu CoNLL17 [33], and Samar [34].

TABLE 1. Work done in literature for sentiment analysis.

Method	Author	Accuracy Reported
Lexicon-Based	Syed et al. 2010 [18]	78%
	Syed et al. 2014 [20]	82.5%
	Mukhtar et al. 2019 [22]	89%
	Asghar et al. 2019 [23]	92.4%
Machine Learning	Mukhtar et al. 2018 [19]	67%
	Awais et al. 2019 [26]	86%
	Nasim et al. 2020 [27]	69%

III. METHODOLOGY

This section first discusses preprocessing strategies, different text representations, and different DL models with implementation details. Fig. 1 describes the proposed classification model.

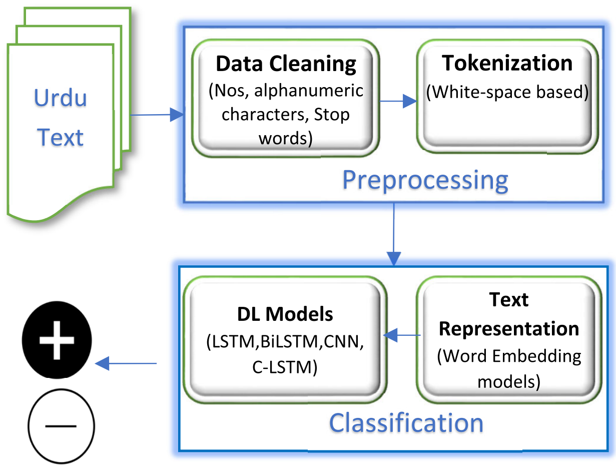

FIGURE 1. Proposed sentence-level sentiment classification model.

A. PREPROCESSING

Steps performed in preprocessing are described in Fig. 2.

1) In the first step, the data is cleaned up by removing punctuation marks, numbers, alphanumeric characters, and characters from other languages other than Urdu.
2) In the second step, useless and unwanted data items such as stop words that do not contribute to sentiment classification are removed. There is a list of 254 Urdu stop words used.
3) Extra spaces are removed, and sentences are tokenized based on white spaces.

B. TEXT REPRESENTATION

Word Embeddings (WE) convert each indexed word into fixed size vectors of continuous numbers. The embedding matrix is formed by vectors for each unique word in the vocabulary. Embedding matrix is produced by applying embedding models. It has a dimension of $s*d$, as shown in (1) where s represents the size of vocabulary and d is the dimension of dense vector.

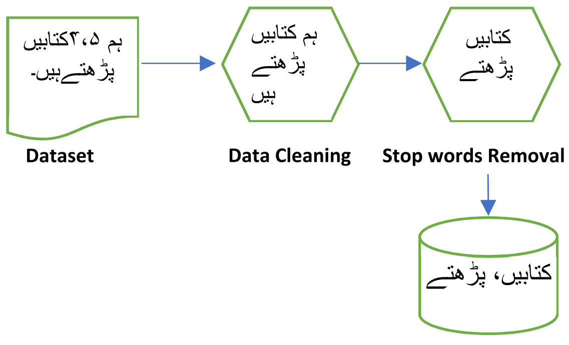

FIGURE 2. Visualization of preprocessing steps for Urdu sentences.

TABLE 2. Parameters use to train fasttext model.

Dimension	Method	Word ngram	Min ngram	Context window
300	Skipgram	3	2	5

There are numerous word representation models. This study examined four word-embeddings, including CoNLL (Word2Vec), fastText (pre-trained & self-trained), and Samar models, to determine which embeddings reflect better classification association.

$$M = \begin{array}{ccc} f_{1,1} & \cdots\cdots & f_{1,d} \\ f_{2,1} & \cdots\cdots & f_{2,d} \\ f_{3,1} & \cdots\cdots & f_{3,d} \\ . & \cdots\cdots & . \\ . & \cdots\cdots & . \\ . & \cdots\cdots & . \\ f_{s,1} & \cdots\cdots & f_{s,d} \end{array} \tag{1}$$

1) FASTTEXT

In fasttext model, each word is represented as a bag of character n-grams. It enables to use subword-level information for creating embeddings. Such as word ‘بہترین’ can be broken down into ‘بہت’, ‘بہتر’, ‘بہتری’, and ‘بہترین’. So, it can take care of words that are not part of the dataset/dictionary. It is especially helpful for languages like Urdu where words can have no space markers and a sufficient number of compound words.

a: FASTTEXT(SELF-TRAINED)

To learn word vectors through fasttext unsupervised model [35], we selected skipgram model. Parameters applied for embedding model are provided in Table 2. We observed that raising the context window size to more than 5 reduces classification accuracy. We considered the minimum single occurrence of the word because the vocabulary was insufficient. Model is trained for 20 Epochs.

E.g., the self-trained model returned the most similar words for “نوازا” include the following.

,(’نوازی’, 0.7989), (’نواز’,0.85142), (’نوا’, 0.85947)]

,0.6900), (’نواسن’, 0.7086), (’نواح’, 0.7735), (’نوازشریف’, 0.7770)

[(’نواسہ’, 0.66678), (’بنوا’, 0.68011), (’نوازتے’, 0.6836), (’نوازتی’.

b: FASTTEXT (PRE-TRAINED)

Second variation used is fasttext pre-trained Urdu embedding to obtain word vectors, which are available online [36]. It provides word vectors of 300 dimensions trained using CBOW model.

2) SAMAR EMBEDDING

Another embedding model we used to initialize our word vectors was produced by Samar Haider. It is publicly available [37]. It is trained over more than 140 million Urdu words using skipgram model. It also represents words in 300-dimension vectors.

3) WORD2VEC(CONLL

We also used NLPL word embeddings repository to get embeddings for Urdu [38]. It provides vectors of 100 dimensions trained over 108310 from the CoNLL17 Urdu corpus using the Word2Vec Continuous Skipgram model.

The parameters for embedding layer are the input data shape, embedding matrix, and maximum sentence length. For all the DL models implemented, this layer has the same parameters.

C. CLASSIFICATION MODEL IMPLEMENTATION

This section explains the DL classification models and their application in this study.

Four different model representations are constructed based on LSTM, BiLSTM-ATT, CNN, and C-LSTM techniques. Except for CNN, stacked layers in our models since they result in a deeper network that can predict more accurately. After multiple attempts, we were able to establish combination of deep layers, which yielded the best results. At the end of this section, Table 3 shows the parameter specifications applied to the models.

1) LONG SHORT-TERM MEMORY (LSTM)

An LSTM consists of a set of recurrently connected blocks, known as memory blocks. These blocks can be thought of as a differentiable version of the memory chips in a digital computer. Each one contains one or more recurrently connected memory cells and three multiplicative gates: input, output, and forget as shown in Fig. 3. These gates can figure out which data in a sequence should be kept and which should be discarded. Equations for gates in LSTM cell are:

$$i_t = \sigma\left(w_i\left[h_{t-1}, x_t\right] + b_i\right) \tag{2}$$

$$f_t = \sigma\left(w_i\left[h_{t-1}, x_t\right] + b_f\right) \tag{3}$$

$$o_t = \sigma\left(w_i\left[h_{t-1}, x_t\right] + b_o\right) \tag{4}$$

where $\mathrm{i_t}$ represents input gate, σ sigmoid function, $\mathrm{w_i}$ weight for the respective gate, $\mathrm{h_{t-1}}$ output from the previous LSTM gate at time stamp t-1, $\mathrm{x_t}$ input at the current time stamp, $\mathrm{b_i}$ biases for the respective gates(x).

Two sequential layers are employed with each layer of 128 units.

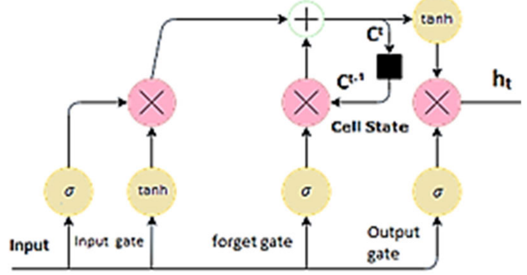

FIGURE 3. A segment in LSTM with interacting layers.

2) ATTENTION-BASED BIDIRECTIONAL LSTM (BILSTM-ATT)

BiLSTM allows signal to propagate forward as well backward in time. BiLSTM uses two different LSTM units, one for forward motion and the other for backward motion as shown in Fig. 4. It preserves not only data from long-term but also with two hidden states able to preserve data from both past and future in any point in time.

$$h_t^{bi} = h_t^{forward} + h_t^{backward} \tag{5}$$

where $h_t^{forward}$, $h_t^{backward}$ represents hidden states and + represents concatenation.

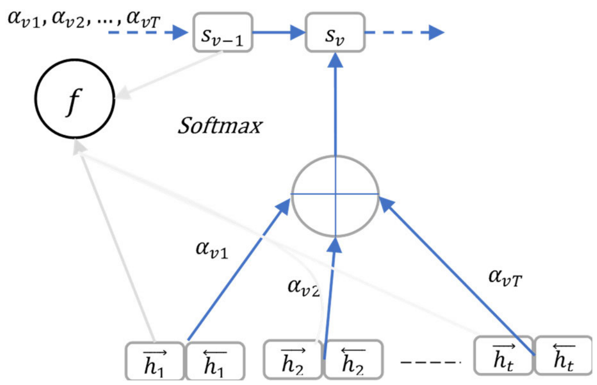

FIGURE 4. Mechanism of attention-based BiLSTM model.

In sentiment classification of sentence, sentiment carrier words are more important than the rest of words. To enhance the weightage of words that play key role in sentiment categorization, the attention mechanism is utilized in combination with BiLSTM. With attention all former states can be retrieved and weighted according to some learned measure of relevance to the current token allowing it to deliver more specific information on distant relevant tokens.

First, attention weights are determined, and a context vector is formed utilizing attention values and BiLSTM output:

$$c_v = \sum\nolimits_{t=1}^{T_x} \alpha_{vt} h_t \tag{6}$$

where α_{vt} represents attention weights for each h_t is calculated as:

$$\alpha_{vt} = \frac{exp\left(e_{vt}\right)}{\sum_{j=1}^{T_x} exp\left(e_{vj}\right)} \tag{7}$$

$$e_{vt} = f\left(h_t, s_{v-1}\right) \tag{8}$$

e_{vt} is a function of both inputs at position h_t and previous hidden state outputs at position s_{v-1}.

This model is implemented as two BiLSTM layers with 128 units followed by an attention layer.

3) CONVOLUTION NEURAL NETWORK

Convolution Neural Network (CNN) is used to find relationships and patterns between data items according to their relative position. They extract higher level features by convoluting efficiently. CNN learns spatial features of the data, convolutes down to a smaller subset of the data while trying to learn more features from the already learned features. CNNs apply a layer called the pooling layer, which reduces input by combining multiple related inputs while preserving the information. This process is visualized in Fig. 5.

Single layer Conv1D with 128 filters of multi-size kernel(3,4,5) is used to extract features.

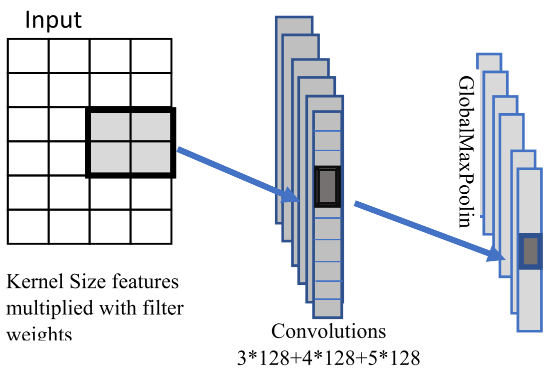

FIGURE 5. Convolution model for text classification.

4) HYBRID CNN & LSTM (C-LSTM)

In this technique, CNN is applied to extract features from data and fed into LSTMs to support sequence prediction. C-LSTM can capture both local features of words as well as whole sequential sentence semantics. CNN captures spatial features much better than others and then it is fed into long-term sequential model to lean sentence classification.

In the hybrid model, the first CNN layer of 128 filters of size 3 with max pooling layer is used to extract features and extracted features are fed to the LSTM layer of size 64 cells.

5) DENSE LAYERS

To complete the DL model, output of previous hidden layers is fed into fully connected dense layer. Dense layer receives input from all nodes of its previous layer as described in Fig. 6. All DL models have 64 units at dense layer except BiLSTM-ATT which has 20 units. Last dense layer in DL models is used to predict the class. It has one neuron per class, so for binary classification, it has two neurons.

D. OVERFITTING REGULARIZATION

Several methods are used to reduce overfitting of DL models. A few of them applied in our study are explained in the following section.

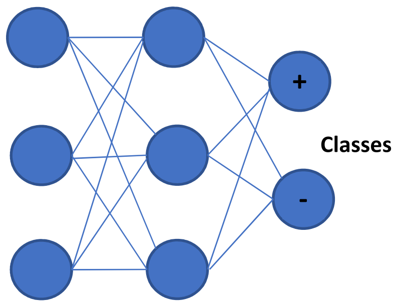

FIGURE 6. An illustration of fully connected dense layers.

1) DROPOU

Dropout randomly removes some nodes by setting them to zero during training. It avoids learning the same values repeatedly by the model in case of large set of parameters.

First layer in LSTM model used 0.5 dropout regularization value. In BiLSTM-ATT, each BiLSTM layer has dropout rate of 0.2 and 0.5 recurrent dropout rate. Dropout ratio of 0.5 is applied after merging multichannel in CNN model. In C-LSTM 0.5 dropout rate applied with LSTM layer and a dropout layer with 0.2 dropout ratio is added after first dense layer.

2) L2 REGULARIZATIO

To overcome overfitting of training data, l2 regularization technique is used to lower the weights. It adds a regularization term to the cost function, causing the values of weight matrices to decrease.

$$Cost\ f = Loss + \frac{\lambda}{2m} * \sum ||w||^2 \tag{9}$$

" λ" is a regularization parameter.

In the proposed study, value for hyperparameter λ is chosen differently for each model based on the performance evaluation. For recurrent models such as LSTM and BiLSTM-ATT, the l2 value "1e-6" is applied to the first LSTM layer. While the l2 value for CNN is "4e-4" and for C-LSTM is "1e-3" in the dense layer.

3) EARLY STOPPING

Early stopping is a technique that stops training once the model's performance on the validation dataset stops improving irrespective of epochs selected to prevent overfitting. In our DL SA system, callback function is used to end the training. The performance values for each epoch are recorded in the callbacks, which include the verification loss, the verification accuracy, the training loss value, and the training accuracy. Two important parameters of this function are monitor and patience. Validation accuracy is chosen as a monitor, which means it keeps track of validation data loss. Patience is set to three to make sure that if the accuracy rate does not improve over the course of 3 epochs, the model will automatically stop training.

IV. EXPERIMENTAL SETUP

To evaluate the deep learning methods for sentiment analysis of Urdu text, an experimental setup is designed in

TABLE 3. Summary of parameters setting for dl models.

Parameters	DL Models			
	BiLSTM-ATT	LSTM	CNN	C-LSTM
Layers	2 BiLSTM, 1 Attention, 2 Dense	2 LSTM, 2 Dense	1 CNN, 2 Dense	1 CNN, 1 LSTM, 2 Dense
Units	128,128,20,1	128,128,64,1	64,1	64,1
Filters & Size	-	-	128, 3,4,5	128, 3

which different deep learning techniques are compared with different embeddings. These approaches are evaluated for sentence-level classification tasks in the domain of views about current affairs, sports, literature, and health. Accuracy is used as our main evaluation criteria. In the following subsections, the details of the experiments and their results are described.

A. DEVELOPMENT ENVIRONMENT

All experiments were performed using TensorFlow library along with the Keras in the Spyder 3.2.6 environment. Matplotlib python library is used for plotting.

B. DATASET

Dataset for this study is collected from Urdu blogs and news websites such as BBC, DW, Express, Dunya and humsub. Dataset belongs to politics, religion, sports, technology, Economy, health, humor, and literature.

The dataset has two classes, positive and negative, comprising 6000 sentences and 117685 words, respectively. Statistics about the dataset are provided in Table 4.

TABLE 4. Statistics of dataset.

Train	Test	Max-Length	Unique Tokens
4800	1200	44	11230

The dataset comes as a CSV file, with each line containing a sentence and its label(‘p’ for Positive, ’n’ for Negative). An imbalanced dataset of 6000 sentences is used and a distribution ratio of 80:20 is applied for train and test as shown in Fig 7.

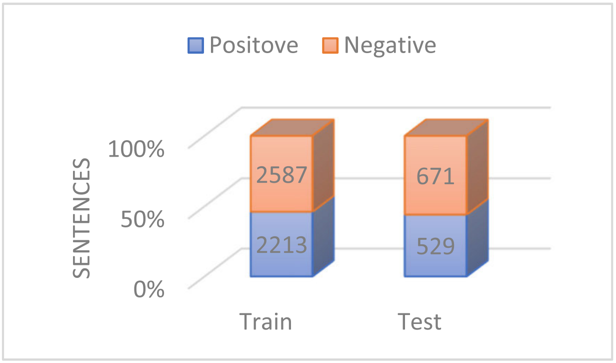

FIGURE 7. Dataset statistics based on sentiment labels.

C. PARAMETER SETS

All of the embeddings used in experiments were of the size 300 except CoNLL. Rest of the parameters setting for experiments is given in Table 5.

TABLE 5. Parameters setting for experiments.

Epochs	Optimizer	Classification Function	Activation Function
15	Adam	Sigmoid	ReLU

As we applied early stopping to prevent overfitting, so the number of epochs varied for different experiments.

D. PERFORMANCE EVALUATION METRICS

The metrics used for the evaluation predictions are accuracy, F1-score, precision, recall and ROC curve.

$$\textbf{Precision} : Precision = \frac{TP}{TP + FP} \tag{10}$$

$$\textbf{Recall} : Recall = \frac{TP}{TP + FN} \tag{11}$$

$$\textbf{F} - \textbf{measure} : F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{12}$$

$$\textbf{Accuracy} : Accuracy = 2 * \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

The measure TP, FP, TN, FN for binary classification (i.e., positive, or negative) can be defined as:

- **True Positive (TP):** For a class Positive, TP is the number of sentences that actually belong to Positive/Negative category and are also correctly predicted as Positive /Negative by classifier.
- **True Negative (TN):** TN is the number of sentences that do not belong to Positive/Negative category and are also not predicted by a classifier.
- **False Positive (FP):** FP denotes the number of sentences whose actual labels do not belong to class Positive but are predicted as Positive by a classifier, and vice versa.
- **False Negative (FN):** FN denotes the number of sentences whose actual labels belong to class Positive but are predicted as Negative by a classifier, and vice versa. **AUC-ROC** It demonstrates the relationship between the proportion of true positive and false positive classification results as its threshold value is varied in a two-class classification task. The Area Under the Curve (AUC) is to quantify ROC Curve. Higher value of AUC shows classifier performance at recognizing distinctive classes. AUC is useful even when there is imbalance class data

V. RESULTS AND DISCUSSION

The results of the performance evaluation of various DL models are presented in this section. Four different word vector representations were used to provide input to DL models, as stated in section IV. As a result, four experiments were carried out for each model based on the embeddings. Performance of models is examined and how different embedding representations affected it. Table 6 provides the comparative

TABLE 6. Comparative analysis of DL models embedding wise.

Embeddings	Samar				CoNLL				fasttext (pre-trained)				fasttext (self-trained)			
DL Models	**Pre**	**Re**	**F1**	**Acc**	**Pre**	**Rec**	**F1**	**Acc**	**Pre**	**Rec**	**F1**	**Acc**	**Pre**	**Rec**	**F1**	**Acc**
LSTM	**85.16**	57.51	68.66	76.41	72.30	63.45	67.59	72.67	78.57	55.10	64.77	73.08	61.16	66.41	63.93	66.33
BiLSTM-ATT	75.40	**69.38**	**72.27**	**77.92**	78.35	64.75	70.34	75.83	72.21	70.87	71.53	74.67	78.64	60.11	68.14	74.75
CNN	81.47	51.39	63.03	72.91	74.31	60.11	66.46	72.75	70.30	63.26	66.60	71.50	62.87	61.59	62.23	66.41
C-LSTM	84.82	52.87	65.14	74.58	69.47	61.22	65.08	70.50	73.75	60.48	66.46	72.58	67.80	55.10	60.79	68.08

detail of results obtained for respective models. The highest values achieved by models among all embeddings are highlighted in bold font.

Model based on BiLSTM-ATT outperformed all others, by achieving the highest recall, F1, and accuracy. LSTM achieved the highest precision. Despite the fact that C-LSTM has been found to be useful for classification in other languages, it has not improved in our experiments.

Based on the results, it can be determined that regardless of embedding, BiLSTM-ATT performed better for sentence level sentiment classification.

It proved to be the preferred model for short text classification. During experimentation it is observed that CNN and C-LSTM model began to overfit after only five Epochs, therefore training was terminated while BiLSTM-ATT and LSTM produced results after 15 Epochs.

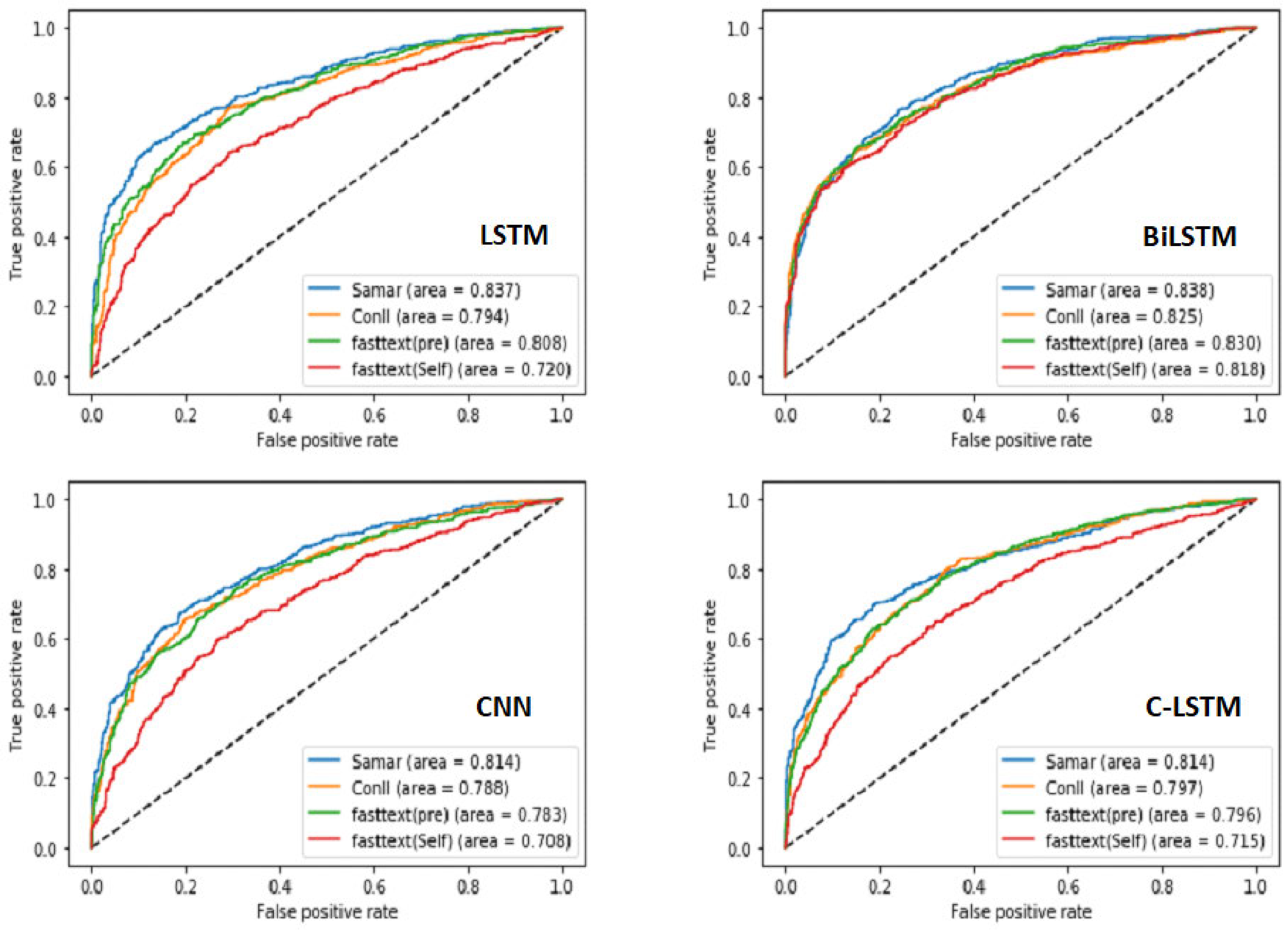

FIGURE 8. ROC curves for sentiment classification by DL models.

All models managed better accuracy, F1, precision, and recall when Samar embeddings were used for word vectors, which is a highly notable observation. It is also clear that when models are fed with fasttext pre-trained word vectors, they produce reasonable outcomes.

Samar embeddings enhanced the model's performance by 2-8 times as compared to the rest of the embedding models. Because of the small dimensions and size of the trained dataset, models that used CoNLL embeddings yielded average results. Similarly, the while utilizing the fasttext self-trained model, the performance of the DL models degraded significantly. Only BiLSTM-ATT performed better with it in experimentation. Still, these are quite encouraging results as the training set used to train this fasttext embedding model is very small.

We gained a better understanding that if we have a large dataset, we may be able to obtain a better grasp of how the self-trained embedding learning paradigm might produce better results.

Figure 8 shows the AUC score for each DL methodology that utilizes different embeddings. It is observed that the ROC of the BiLSTM-ATT model and the other models differ. There is a merge of curves for all embeddings that showed BiLSTM-ATT performed consistently better with all word representations. Overall models scored AUC above 0.70 which is fair enough for our applied classification strategy. The AUC score reveals that all models with Samar embedding had AUC score greater than 0.80, and all models with pre-trained fasttext had score close or above 0.80. So, it can be inferred that each model scaled maximum sentences correctly when word embedding is extracted from Samar embedding model.

Although it has been found that DL models better classified the input sentences when provided in Samar embeddings, this cannot be generalized for all Urdu data. It has shown its effectiveness for selected dataset as it is not experimented with data from different domains such as reviews.

Several limitations were faced during the course of this study. Experimentation was done with a relatively smaller dataset, which may not have helped DL models perform at their best. The data is also unbalanced in terms of sentiments as the number of negative sentences exceeds the number of positive sentences. Also, Urdu language contains a large number of compounds that are not considered due to tokenization restriction and variability of writing style such as word 'خوش دامن' can be written with space and can also be written without space such as 'خوشدامن'. Since words are tokenized based on space, tokenizer can split compound words into two separate words, reducing their effectiveness as compound words. In the future, this problem will be solved by applying multiword tokenization.

VI. CONCLUSION & FUTURE DIRECTION

In this research, for Urdu text sentiment analysis, four separate deep learning models are investigated. Since DL methods are less explored methods for Urdu text document classification related tasks, it revealed several issues related to language complexities, lack of resources and solutions are explored for its application. Another investigation is the effect of using different pre-trained and self-trained word embeddings on these deep learning architectures. The accuracy of each model was computed using four different embeddings. BiLSTM-ATT demonstrated to be the most efficient model, exceeding all other models irrespective of embedding. In the experiments, we focused on the performance of the fasttext (pre-trained, self-trained), CoNLL, and Samar embeddings by utilizing them in order to determine the best word embedding representation for a specific data set. It has been observed that by using the Samar embedding model, DL models performance improved. It also highlights the importance of embedding on classification task. Models attained acceptable classification performance irrespective of imbalanced dataset.

In future, DL models with more embeddings that provide better contextual information along with the increased and balanced dataset will be explored. Role of preprocessing techniques such as lemmatization and stemming needs to be investigated. Sentiment lexicons can also be utilized to aid in the sentiment classification accuracy. We intend to explore compound words in Urdu text for SA in future to attain better classification performance.

ACKNOWLEDGMENT

The authors would like to thank the editor and anonymous reviewers for their perceptive comments and suggestions.

REFERENCES

[1] *Why the Coronavirus Lockdown is Making the Internet Stronger Than Ever | MIT Technology Review*. Accessed: Feb. 1, 2021. [Online]. Available: https://www.technologyreview.com/2020/04/07/998552/why-the-coronavirus-lockdown-is-making-the-internet-better-than-ever/

[2] *Internet Users in the World 2021 | Statista*. Accessed: Jul. 15, 2021. [Online]. Available: https://www.statista.com/statistics/617136/digital-population-worldwide/

[3] *Alexa–Top Sites in Pakistan–Alexa*. Accessed: Jul. 15, 2021. [Online]. Available: https://www.alexa.com/topsites/countries/PK

[4] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Eng. J.*, vol. 5, no. 4, pp. 1093–1113, 2014, doi: 10.1016/j.asej.2014.04.011.

[5] T. Wilson, J. Wiebe, and P. Hoffman, "Recognizing contextual polarity in phrase-level sentiment analysis," *ACL*, vol. 7, no. 5, pp. 12–21, 2005, doi: 10.3115/1220575.1220619.

[6] T. Bos and F. Frasincar, "Automatically building financial sentiment lexicons while accounting for negation," *Cognit. Comput.*, to be published, doi: 10.1007/s12559-021-09833-w.

[7] R. Kashyap and A. V. S. Kumar, *Challenges and Applications for Implementing Machine Learning in Computer Vision*. Hershey, PA, USA: IGI Global, 2020.

[8] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang, "Machine learning and deep learning methods for cybersecurity," *IEEE Access*, vol. 6, pp. 35365–35381, 2018, doi: 10.1109/ACCESS.2018.2836950.

[9] A. Ramdan, A. Heryana, A. Arisal, R. B. S. Kusumo, and H. F. Pardede, "Transfer learning and fine-tuning for deep learning-based tea diseases detection on small datasets," in *Proc. Int. Conf. Radar, Antenna, Microw., Electron., Telecommun. (ICRAMET)*, Nov. 2020, pp. 206–211, doi: 10.1109/ICRAMET51080.2020.9298575.

[10] A. Anaby-Tavor, B. Carmeli, and E. Goldbraich, "Do not have enough data? Deep learning to the rescue!," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 5, pp. 7383–7390. Accessed: Feb. 15, 2021. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/6233

[11] B. Barz and J. Denzler, "Deep learning on small datasets without pre-training using cosine loss," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2020, pp. 1371–1380. Accessed: Feb. 25, 2021. [Online]. Available:

[12] S. Feng, H. Zhou, and H. Dong, "Using deep neural network with small dataset to predict material defects," *Mater. Des.*, vol. 162, pp. 300–310, Jan. 2019, doi: 10.1016/j.matdes.2018.11.060.

[13] *Examining Attention Mechanisms in Deep Learning Models for Sentiment Analysis | Enhanced Reader*,

[14] W. Hou, H. Suominen, P. Koniusz, S. Caldwell, and T. Gedeon, "A token-wise CNN-based method for sentence compression," in *Proc. Int. Conf. Neural Inf. Process.*, 2020, pp. 668–679.

[15] S. Seo, C. Kim, H. Kim, K. Mo, and P. Kang, "Comparative study of deep learning-based sentiment classification," *IEEE Access*, vol. 8, pp. 6861–6875, 2020, doi: 10.1109/ACCESS.2019.2963426.

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: http://arxiv.org/abs/1301.3781

[17] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," 2016, *arXiv:1607.04606*. [Online]. Available: http://arxiv.org/abs/1607.04606

[18] A. Z. Syed, M. Aslam, and A. M. Martinez-Enriquez, "Lexicon based sentiment analysis of Urdu text using SentiUnits," *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6437. 2010, pp. 32–43, doi: 10.1007/978-3-642-16761-4_4.

[19] N. Mukhtar and M. A. Khan, "Urdu sentiment analysis using supervised machine learning approach," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 32, no. 2, Feb. 2018, Art. no. 1851001, doi: 10.1142/S0218001418510011.

[20] A. Z. Syed, M. Aslam, and A. M. Martinez-Enriquez, "Associating targets with SentiUnits: A step forward in sentiment analysis of Urdu text," *Artif. Intell. Rev.*, vol. 41, no. 4, pp. 535–561, Apr. 2014, doi: 10.1007/s10462-012-9322-6.

[21] A. Z. Syed, A. M. Martinez-Enriquez, A. Nazir, M. Aslam, and R. H. Basit, "Mining the Urdu language-based web content for opinion extraction," in *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10267. 2017, pp. 244–253, doi: 10.1007/978-3-319-59226-8_24.

[22] N. Mukhtar, M. A. Khan, and N. Chiragh, "Lexicon-based approach outperforms supervised machine learning approach for Urdu sentiment analysis in multiple domains," *Telematics Informat.*, vol. 35, no. 8, pp. 2173–2183, Dec. 2018, doi: 10.1016/j.tele.2018.08.003.

[23] M. Z. Asghar, A. Sattar, A. Khan, A. Ali, F. Masud Kundi, and S. Ahmad, "Creating sentiment lexicon for sentiment analysis in Urdu: The case of a resource-poor language," *Expert Syst.*, vol. 36, no. 3, 2019, Art. no. e12397, doi: 10.1111/exsy.12397.

[24] M. Hassan and M. Shoaib, "Opinion within opinion: Segmentation approach for Urdu sentiment analysis," *Int. Arab J. Inf. Technol.*, vol. 15, no. 1, pp. 21–28, 2018.

[25] N. Mukhtar, M. A. Khan, and N. Chiragh, "Effective use of evaluation measures for the validation of best classifier in Urdu sentiment analysis," *Cognit. Comput.*, vol. 9, no. 4, pp. 446–456, Aug. 2017, doi: 10.1007/s12559-017-9481-5.

[26] D. M. Awais and D. M. Shoaib, "Role of discourse information in Urdu sentiment classification: A rule-based method and machine-learning technique," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 18, no. 4, pp. 1–37, Aug. 2019, doi: 10.1145/3300050.

[27] Z. Nasim and S. Ghani, "Sentiment analysis on Urdu tweets using Markov chains," *Social Netw. Comput. Sci.*, vol. 1, no. 5, Sep. 2020, doi: 10.1007/s42979-020-00279-9.

[28] M. P. Akhter, Z. Jiangbin, I. R. Naqvi, M. Abdelmajeed, and M. Fayyaz, "Exploring deep learning approaches for Urdu text classification in product manufacturing," *Enterprise Inf. Syst.*, pp. 1–26, May 2020, doi: 10.1080/17517575.2020.1755455.

[29] M. N. Asim, M. U. Ghani, M. A. Ibrahim, W. Mahmood, A. Dengel, and S. Ahmed, "Benchmarking performance of machine and deep learning-based methodologies for Urdu text document classification," *Neural Comput. Appl.*, vol. 33, pp. 5437–5469, Sep. 2020, doi: 10.1007/s00521-020-05321-8.

[30] H. Ghulam, F. Zeng, W. Li, and Y. Xiao, "Deep learning-based sentiment analysis for Roman Urdu text," *Procedia Comput. Sci.*, vol. 147, pp. 131–135, 2019, doi: 10.1016/j.procs.2019.01.202.

[31] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov. (2018). *Learning Word Vectors for 157 Languages*. Accessed: Apr. 9, 2021. [Online]. Available: https://fasttext.cc/docs/en/crawl-vectors.html

[32] *Word Vectors for 157 Languages ? fastText*. Accessed: Jul. 5, 2021. [Online]. Available: https://fasttext.cc/docs/en/crawl-vectors.html

[33] M. Fares, A. Kutuzov, S. Oepen, and E. Velldal, "Word vectors, reuse, and replicability: Towards a community repository of large-text resources," in *Proc. 21st Nordic Conf. Comput. Linguistics*, Gothenburg, Sweden, 2017, pp. 271–276. Accessed: Jul. 5, 2021. [Online]. Available: https://ep.liu.se/ecp/131/037/ecp17131037.pdf

[34] S. Haider, "Urdu word embeddings," in *Proc. 11th Int. Conf. Lang. Resour. Eval. (LREC)*, 2018, pp. 964–968. Accessed: Apr. 9, 2021. [Online]. Available: https://ethnologue.com/language/urd

[35] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 135–146, Dec. 2017, doi: 10.1162/tacl_a_00051.

[36] *Word Vectors for 157 Languages ? fastText*. Accessed: Jul. 19, 2021. [Online]. Available: https://fasttext.cc/docs/en/crawl-vectors.html

[37] *GitHub–Samarh/Urduvec*. Accessed: Jul. 19, 2021. [Online]. Available: https://github.com/samarh/urduvec

[38] *NLPL Word Embeddings Repository*. Accessed: Jul. 19, 2021. [Online]. Available: http://vectors.nlpl.eu/repository/

UZMA NAQVI received the M.S. (S.E.) degree from Bahria University, Islamabad, Pakistan. She is currently pursuing the Ph.D. degree with the Department of Computer Sciences and Information Technology, The University of Azad Jammu & Kashmir, Muzaffarabad. She published various research articles in the area of machine learning.

ABDUL MAJID received the Ph.D. degree in computer science from the College of Computer Science, Zhejiang University, Hangzhou, China, in 2012. He is currently working with the Department of Computer Science and Technology, The University of Azad Jammu & Kashmir, Muzaffarabad, Pakistan, as an Assistant Professor. His research interests include spatiotemporal data mining, pervasive computing, and recommender systems.

SYED ALI ABBAS received the B.Sc. degree in mathematics from The University of Azad Jammu & Kashmir, the master's degree in computer science from the University of the Punjab, Pakistan, and the Ph.D. degree in computer science from Chongqing University, China. He is currently an Assistant Professor with the Department of Computer Sciences and Information Technology, The University of Azad Jammu & Kashmir. His research interests include software engineering, data mining, and machine learning.