IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 4 August 2023, accepted 26 August 2023, date of publication 7 September 2023, date of current version 20 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3312730

RESEARCH ARTICLE

Real Word Spelling Error Detection and Correction for Urdu Language

ROMILA AZIZ[1], MUHAMMAD WAQAS ANWAR[1,2], MUHAMMAD HASAN JAMAL[1], USAMA IJAZ BAJWA[1], ÁNGEL KUC CASTILLA[3,4,5], CARLOS UC RIOS[3,4,6], ERNESTO BAUTISTA THOMPSON[3,4,7], AND IMRAN ASHRAF[8]

[1]Department of Computer Science, COMSATS University Islamabad, Lahore Campus, Lahore 54000, Pakistan
[2]Department of Computer Science, Government College University, Lahore 54000, Pakistan
[3]Universidad Europea del Atlántico, 39011 Santander, Spain
[4]Universidad Internacional Iberoamericana, Campeche 24560, Mexico
[5]Universidad Internacional Iberoamericana, Arecibo 00613, Puerto Rico
[6]Universidade Internacional do Cuanza-Cuito, Bié 46703, Angola
[7]Fundación Universitaria Internacional de Colombia, Bogotá 111611, Colombia
[8]Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, South Korea

Corresponding author: Imran Ashraf (ashrafimran@live.com)

This work was supported by the European University of the Atlantic.

ABSTRACT Non-word and real-word errors are generally two types of spelling errors. Non-word errors are misspelled words that are nonexistent in the lexicon while real-word errors are misspelled words that exist in the lexicon but are used out of context in a sentence. Lexicon-based lookup approach is widely used for non-word errors but it is incapable of handling real-word errors as they require contextual information. Contrary to the English language, real-word error detection and correction for low-resourced languages like Urdu is an unexplored area. This paper presents a real-word spelling error detection and correction approach for the Urdu language. We develop an extensive lexicon of 593,738 words and use this lexicon to develop a dataset for real-word errors comprising 125562 sentences and 2,552,735 words. Based on the developed lexicon and dataset, we then develop a contextual spell checker that detects and corrects real-word errors. For the real-word error detection phase, word-gram features are used along with five machine learning classifiers, achieving a precision, recall, and F1-score of 0.84,0.79, and 0.81 respectively. We also test the proposed approach with a 40% error density. For real-word error correction, the Damerau-Levenshtein distance is used along with the n-gram model for further ranking of the suggested candidate words, achieving an accuracy of up to 83.67%.

INDEX TERMS Real-word errors, spelling correction, spelling detection, spell checker.

I. INTRODUCTION

Writing is an effective and important way of communication for expressing thoughts and views as it helps in keeping and preserving records and disseminating information through media [1]. Writing a high-quality article requires it to be devoid of spelling and grammatical errors. Spell-checking plays an integral part to ensure the quality of the content as well as its readability. Additionally, an article with no spelling errors is easier to crawl and index for search engines.

Non-word and real-word errors are generally two types of spelling errors. A misspelled word that does not exist in the lexicon is known as a non-word error and occurs at the word level; for instance, misspelling *apple* as *appll*. A real-word error is a word that exists in the lexicon but is used out of context in a sentence. Real-word spelling errors occur at the sentence level. For instance, the word *hole* is a real-word error in the sentence *We all hole that you will recover swiftly*. Although the word *hole*' exists in the lexicon, the sentence is contextually incorrect and the correct word should be the word *hope*.

Context-sensitive spelling errors or real-word errors are more complex as compared to non-word errors as these errors

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

cause semantic inconsistencies [2]. These errors result in valid words but are misfits in a sentence as the context of these intended words is incorrect [3].

As per the survey on spelling errors [2], out of the total errors, 35% errors are phonetic errors, 25% are non-word errors and 40% are real-word errors. Multiple studies report that real-word errors account for 20-40% of the total spelling errors [4], [5]. Furthermore, the efficiency of the spell checker is significantly based on correcting these errors.

Detection of real-word errors, and providing appropriate suggestions for these misspelled words, is difficult for a spell checker, especially for low-resourced languages like the Urdu language which is spoken by 100 million native speakers in the Indian Subcontinent and around the world [6]. Millions of people use social media platforms like Twitter, Facebook, and WhatsApp daily to communicate with each other using the Urdu language. Numerous books, journals, newspapers, magazines, and articles are being published every day in Urdu.

Existing work on the Urdu language is mainly based on the detection and correction of non-word errors [7], [8], [9], [10] that use a lexicon-based lookup approach. For spelling error detection, each word in the written text is compared against the words in the lexicon and is marked as a misspelled word if it is not found in the lexicon. For error correction, various string-matching distance algorithms, i.e., edit distance, Jaro distance, Levenshtein distance, etc., are used to determine all possible candidate words that can be replaced with the misspelled word. The candidate word with the least distance is selected as a replacement word for the misspelled word.

Reportedly, no attempt has been made to solve the problem of real-word error detection and correction for the Urdu language. This is because, in the Urdu language, finding real-word errors and suggesting an appropriate candidate correction word is challenging as it requires contextual information at a very high level rather than syntax and morphology. Urdu is a linguistically complex language having 11 vowels and 41 consonants. It has pairs of alphabets that are phonetically similar. Nouns and verbs in Urdu can have more than 40 forms which make it difficult and complicated to process [11].

Since real-word errors are contextually incorrect words that do exist in the lexicon, the existing lexicon-based lookup approach, used for non-word error detection and correction, would fail to detect any real-word error as lexicon lookup-based approach cannot capture the context of the word. To capture the context, the surrounding words of the error word in the sentence need to be considered. To address this challenge, a contextual spell checker for the Urdu language is developed and an approach for real-word error detection and correction is proposed in this paper. The following points summarize the contributions of this study:

- A dataset for real-word error is developed and manually labeled. The lexicon consists of 125562 sentences and 593738 words and extracts n-gram features at the word level.
- A real-word error detection and correction model is proposed for Urdu. For error detection, different machine learning (ML) classifiers are used. For error correction, confusion sets are generated. The N-gram model is used for the ranking of candidate words.
- The performance of the proposed approach is evaluated using precision, recall, and F1-score evaluation measures.

The rest of this paper is organized as follows. Section II describes the related work. Section III explains the entire methodology. Section IV presents the experimental results and discussion. Section V concludes the entire research work.

II. LITERATURE REVIEW

This section describes the methods explored for real-word error detection and correction for various languages. Rana et al. propose a method that classifies and corrects homophone errors in real-word errors for the Bangla language [12]. For error detection, they use an n-gram language model with the candidate word. To check the validity, they use homophone words to create the bigram and trigram and then calculate the frequency of the bigram and trigram with homophone words to make the final decision.

Sharma et al. design an intelligent system for correcting real-word errors in text using contextual information for the English language [1]. The proposed system corrects those words that belong to the set of confusion words and are contextually wrong. A two-phase algorithm is proposed that identifies and corrects real-word errors. Firstly, the trigram is used to correct real-word errors and secondly, the Bayesian approach is used to fix these errors. Brown corpus with confusion words set is used as a training set. The proposed system gives higher accuracy for contextual error detection and correction for commonly confusion words.

Mridha et al. propose an approach to detect and correct multiple semantic errors in Bengali text [13]. A confused word list is built with the help of edit distance. For error detection and correction, a Naïve Bayes classifier is used. For a candidate word from the sentence, a set of confusion words are picked up. All the other neighbor words are used as a feature for each word from the confusion words. To evaluate the proposed approach, 28,057 sentences are used. The proposed approach achieves an accuracy of more than 90%.

Faili et al. propose a method for real-word error correction for the Persian language [14]. A confusion set is generated to find all possible candidate words. Using mutual information, their proposed algorithm assigns scores to target words as well as words in the confusion set, and the word having the highest score is selected. The proposed method achieves an accuracy of 80.5%.

Candel et al. propose an approach for correcting real-word errors in clinical text [15]. A sequence-to-sequence neural machine translation method is implemented which maps the misspelled sentences to correct them. Various types of errors are created in the correct sentences using different rules. This is done by repeating a sentence several times by just

modifying the error word in the sentence. Every rule produces a unique real-word error in the sentence. The model is trained on two corpora namely wiki corpus and clinical dataset. To extract contextual information, pre-trained word embedding is employed. For this reason, Word2Vec and GloVe embeddings are also used as input data for the model. The medicine corpus is smaller in size as compared to the wiki corpus but the context is limited to a certain domain, and the sentences extracted are uniform, which leads to the overfitting of the model. Additionally, the size of the pertained word embeddings used is larger than the word vectors resulting in poor performance. In this study, no ranking mechanism is used for selecting the best-corrected candidate word.

Kassa et al. adopt sentence-level n-gram features for real-word error detection and correction [16]. Five high-level modules are incorporated to solve the problem of spell checking, (1) language selection, (2) sentence segmentation, (3) n-gram extraction, (4) error detection, and (5) error correction. They collected and used a large corpus of domain languages including Afaan Oromo, Amharic, and Tigrigna languages for training the proposed context-aware spell checker. The corpus is first segmented into a set of sentences and then possible n-gram features are extracted. The validity of each n-gram is checked along with the target n-gram language model. When all possible n-grams are not found in the target language, the last word of the text unit is considered a misspelled word. For error correction, minimum edit distance along with the context feature is used. The proposed model achieves an F1-score of 90.03% for Amharic, 85.95%, for Afaan Oromo, and 84.24% for Tigrigna.

Wang et al., using a confusion set and generalization model, propose a model for the detection and correction of real-word errors in the Chinese language [17]. The proposed model generates a confusion set and trigram. The N-gram language model in combination with the Bayesian model is used to detect and correct the real-word error. The proposed model effectively detects real-word errors in Chinese text. It has a higher recall, detecting accuracy, and correcting accuracy rate.

Roy et al. propose an unsupervised approach for context-aware spell-checking for the Bangla language [18]. Cosine similarity along with the contextual features is used to find the best candidate word for the misspelled word. Character n-gram embedding has been used for generating the embedding of unknown words.

Sakuntharaj et al. develop an approach for correcting real-word errors in the Tamil language by constructing a Bigram probabilistic model to detect real-word errors [19]. For error correction, lexically similar words are found using minimum edit distance. To find similar words quickly, a hash map with word length is used. The hash map is used to search those words whose lengths differ by not more than two from the length of the misspelled word. The proposed model gives an accuracy rate of 98% in the case of appropriate suggestion generation.

Hossain et al. developed a comprehensive spell checker in the Bangla language with the necessary resources [20]. A generalized 100 million words Bangla monolingual corpus is developed. Then, distinct one million words are extracted to form a lexicon. Using the lexicon and the corpus, a Bangla spell checker is developed to detect and correct non-word, real-word errors, and grammatical errors. A double Metaphone encoding and edit distance approach based on distributed lexicons and numerical suffix datasets is used to detect all types of non-word errors with an accuracy rate of 97.21%. A combination of bigram and trigram language models is used to detect the real-word and grammatical errors. For generating suggestions for the misspelled word, the cosine similarity measure is used which gives an accuracy rate of 94.29%.

Jahan et al. propose a method for real-word spelling error detection and correction using bidirectional LSTM and RNN with bigram for the Bangla language [21]. The proposed model only handles real-word errors generated through homophones. Additionally, the proposed system only checks each sentence using bigram probability but in many sentences, more context is required to comprehend a text. Moreover, the proposed system uses word length matching to ensure that the output word length stays consistent with the original word. This approach works only on homophonic errors that are generated through substitution operation.

Huang et al. propose a method for real-word error correction by implementing a real-word confusion set [22]. It combines the binary statistical model and the Glove vector model for correcting real-word errors. The correction method depends heavily on the predefined set as it compares the error word in the predefined confusion set and then calculates the longest common subsequence with the confusion set words. The drawback of using a predefined confusion set is that most of the time the correct candidate word does not exist in the predefined confusion set and a wrong suggestion word is selected, thereby decreasing the system performance.

Toleu et al. propose a method for real-word error correction for the Kazakh language [23]. They only rank the real-word error correction using the noisy channel model which does not capture enough contextual information. Therefore, in some cases, it gives the wrong suggested candidate word by using the Bayes rule.

In existing studies on real-word error detection and correction, the lexicon sizes are comparably inadequate for providing precise results and recommendations and the models are based on small corpora which are usually related to a specific domain. In this study, firstly, we develop a large monolingual corpus that covers different domains e.g., sports, religion, news, education, etc., and then propose a real-word error detection and correction model that makes use of a sizable lexicon, taken from our developed corpus, to identify a variety of spelling errors and provide precise word replacement options.

TABLE 1. Summary of the literature review on real-word error detection and correction.

Language	Methodology		Accuracy
	Error Detection	Error Correction & Suggestion	
Bangla [12]	Bigram and Trigram	N/A	96%
English [1]	N/A	Trigram with Bayesian Approach	89.83%
Bangla [13]	Naïve Bayes with Laplace Smoothing	Edit Distance	90%
Persian [14]	N/A	Bayesian Approach with Mutual Information	80.5 %
Spanish [15]	N/A	Seq2Seq	50%
Bangla [18]	N/A	Cosine Similarity with Contextual Features	80.90
Tamil [19]	Bigram Probabilistic Model	Minimum Edit Distance	98%
Bangla [20]	Double Metaphone Encoding & Edit Distance	Bigram and Trigram	97.21%
Bangla [21]	Bigram and BiLSTM	Bigram and BiLSTM	82.86%
English [22]	N/A	Binary Statistical Model & GloVe model	77.9%
Kazakh [23]	N/A	Noisy Channel Model with Bayes Rule	93%

Most studies either detect real-word errors or correct real-word errors but do not perform both tasks together. Some approaches only detect real-word errors without providing any correct candidate suggestion which helps the user to immediately correct the real-word error, thus reducing the time for correcting real-word errors. Existing real-word error correction approaches usually use predefined real-word confusion sets. These approaches heavily depend on the predefined set as it compares the error word in the predefined confusion set and then calculates the longest common subsequence with the confusion set words. The drawback of using predefined confusion sets is that most of the time the correct candidate word does not exist in the predefined confusion set and a wrong suggestion word is selected, thereby decreasing the system performance. Our proposed model does not rely on predefined confusion set as we generate confusion sets of different lengths from the dictionary using Levenshtein distance and Damerau-Levenshtein distance to calculate the distance of the error word with these confusion sets to generate a list of suggested candidate words. This study further ranks the suggested candidate words for a real-word error based on the contextual information of the language. To the best of our knowledge, no study has been performed on ranking the suggested candidate words for real-word error correction in terms of context, and this is a novel contribution of this study.

Additionally, existing studies do not cover all types of real-word errors. In this study, our model handles all types of real-word errors that are phonetically similar, visually similar, different word lengths, and grammatically similar. The system can handle all types of real word errors that are generated by inserting characters, deleting characters, and substituting characters.

Our approach not only detects and corrects real-word errors by providing different suggestions but also ranks those suggestions with the help of our proposed approach which ranks the best candidate word according to its context. Moreover, to the best of our knowledge, no study has been conducted to address the problem of real-word error detection and correction for the Urdu language and this study is the first effort in this regard. Table 1 presents the summary of the literature review for real-word error detection and correction across various languages.

III. METHODOLOGY

This section describes the proposed methodology for real-word error detection and correction for Urdu. Figure 1 illustrates the architecture of the proposed work.

A. DATA COLLECTION AND PREPROCESSING

Data collection of real-word errors is quite difficult since there is no available benchmark corpus for Urdu. Therefore, we generated a real-word error dataset from two existing corpora namely English-Urdu parallel corpus[1] and Urdu monolingual corpus.[2] We extracted 96240 and 29,322 sentences from the Urdu monolingual and Urdu-English parallel corpus respectively. The collected corpus is a mix of different domains e.g., sports, religion, news, education, etc. The statistics of both corpora are given in Table 2.

TABLE 2. Statistics of the collected data.

Corpus	Sentences	Tokens	Vocabulary
Urdu monolingual corpus	96,240	1,692,948	44,812
Urdu-English parallel corpus	29,322	859,787	39,947

To preprocess the dataset, (i) unwanted characters, (ii) special characters like @, #, $, !, (iii) hyperlinks, (iv) extra spaces, (v) numbers, and (vi) English words are removed. In Urdu, some letters are usually misspelled using multiple variants and it is useful to make these variants into a single canonical form. In our study, it is usually done through the following: (a) substitute {آ → ا}, (b) {ئ،ي → ی}, (c) {ۀ،ة → ہ}, and (d) {ؤ → و}.

B. LEXICON BUILDING

The core component of a spell checker is the lexicon. We create our lexicon by extracting words from different corpora: (1) Urdu monolingual corpus (2) Urdu wordlist[3] (3) English-Urdu parallel corpus (4) Urdu Summary corpus.[4] We perform the preprocessing using the above-mentioned steps and extract all the distinct words from the corpora.

[1] https://ufal.mff.cuni.cz/umc/005-en-ur/
[2] https://ufal.mff.cuni.cz/urmonocorp/
[3] https://www.cle.org.pk/software/ling_resources/wordlist.htm
[4] https://github.com/humsha/USCorpus

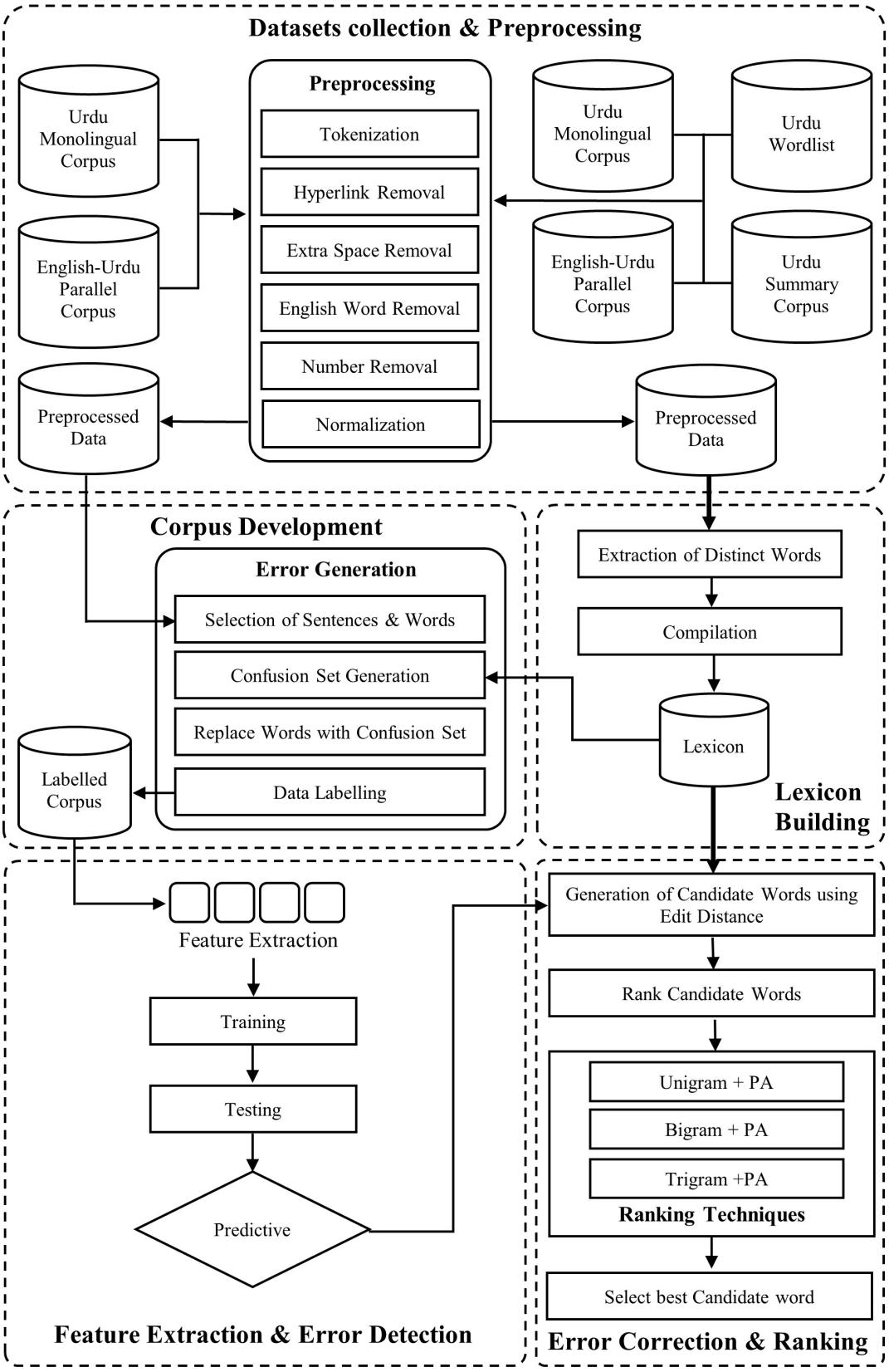


FIGURE 1. Architecture of the proposed work.

Next, the words are sorted in ascending order, and then we combine all these dictionaries by finding out distinct words. The total size of our lexicon is 593,738. Table 3 shows the statistics of the dictionaries.

TABLE 3. Statistics of corpora used for lexicon building.

Corpus	Tokens	Vocabulary
Urdu monolingual corpus	95,411,827	582,795
Urdu-English parallel corpus	859,787	39,947
Urdu summary corpus	29,889	1,527
Urdu wordlist	–	149,466

C. CORPUS DEVELOPMENT

In the literature, the detection and correction problem of a real-word error is generally solved by taking the original text and generating context errors in the text randomly. Islam et al. [24] randomly induced real-word errors in the text at a rate of around one error per every 200 words. Most studies in the literature adopt a limit of one error per sentence, as another constraint. For this study, we also follow the same constraint of inducing one error per sentence. We experimented with different error densities i.e., 30% and 40% which means that 30% to 40% sentences in a corpus will have real-word errors. A collection of confusion sets is normally used to induce real-word errors. A confusion set is specifically used to address the problem of real-word errors. A confusion set is a set of words that are confused with one another either in terms of sound or letter as shown in Table 4.

TABLE 4. Sound and letter-based confusion set examples.

Sound based Confusion Set	Letters based Confusion Set
قمر/کمر	کبیر/کثیر
ہلکا/حلقا	علم/قلم
حال/ہال	شراب/سراب
جعلی/جالی	مصر/مضر
شعر/شیر	اصرار/اسرار
صدا/سدا	اعداد/امداد

The confusion sets can be obtained using a variety of methods. One way is to find the words in the lexicon that differ from other words by just one letter. Jennifer et al. [25] state that more than 50% of the context errors change by just one edit distance from the correct term. It is also observed that almost 80% of the spelling errors contain a single instance of the following four error types (1) insertion, (2) deletion, (3) substitution, and (4) transposition [26]. In the literature review, most studies use single edit distance to generate confusion sets. Levenshtein distance or Damerau-Levenshtein distance (DL) has been used for confusion sets generation. DL is a string metric that is used to calculate the edit distance between two strings. Informally DL between two numbers is the minimum number of editing operations namely insertion, deletion, substitutions, and transposition required to transform one word into the other word.

For this study, we also generate confusion sets for inducing real-word errors in the corpus using DL. We generate two confusion sets, one confusion set with edit distance one, and one confusion set with edit distance two. DL takes $O(n \times m)$ time where n is the length of one word and m is the length of the other word. Usually, we want to find the closest matching word from the whole lexicon of thousands of words. Therefore, for efficient searching, we use a trie data structure. We have built a trie with all the distinct words in the lexicon. A trie is a prefix tree of strings where each branch consists of strings with the same prefix representing a partial or complete word. With a trie, all shared prefixes in the lexicon are collated into a single path, so we can process them in the best order for building up the DL. With trie, the searching time is reduced to $O(m)$ where m is the maximum string length. We select 30% of the sentences from the whole corpus randomly for context error generation. From the 30% of the selected sentences, we use 80% of sentences for context error generation with an edit distance of one and 20% of sentences for context error generation with an edit distance of 2. From each sentence, we select a random word and generate its confusion set. Then, we replace the original word with one of the words in the confusion set and label it as a real-word error and the remaining words in the sentence are labeled as correct words. Table 5 shows the distribution of misspellings by an edit distance of 1 and 2 from the correct word in the corpus.

TABLE 5. Edit distance distribution in the corpus.

Edit Distance	Word Count
1	30,134
2	7,534

D. FEATURE EXTRACTION

For feature extraction, we use term frequency and inverse document frequency (TF-IDF) technique which evaluates the importance of a word in a given text. TF represents the frequency of a word in each sentence and IDF represents the importance of a word in the given text. TF sometimes may count less important words more frequently which usually decreases the performance of the model. To solve this problem, IDF is used that can analyze higher and less relevant words. Thus, by applying this vectorizer, the average weight is raised. Equation 1 is used to calculate the TF-IDF score:

$$W_{t,d} = tf_{t,d} \times idf_t = tf_{t,d} * log\frac{N}{df_t} \quad (1)$$

where $W_{t,d}$ is the weight of TF-IDF, $tf_{t,d}$ is the number of word frequencies, idf_t is the inverse document frequency per word, df_t is the number of document frequency per word and N is the total number of documents. In this work, we extract three different feature sets namely (i) unigram, (ii) bigram, and (iii) trigram for error classification. Additionally, a combination of these three features is also extracted as a different

feature set for error classification. We also include the word frequencies that capture contextual information [27]. TF-IDF assigns a score to the extracted n-grams.

E. REAL-WORD ERROR CLASSIFICATION

By considering the surrounding context in the sentence, extracted using different n-gram features, we classify the real-word errors after training the extracted n-gram features on different ML models. For error classification, we use five ML classifiers namely, support vector machine (SVM), Naïve Bayes (NB), random forest (RF), logistic regression (LR), and K-nearest neighbor (KNN).

1) SUPPORT VECTOR MACHINE

SVM classifier is based on supervised learning and is used for regression analysis and binary classification. SVM has been used to solve various pattern recognition problems because of its well-known high generalization performance and good reported accuracy. For text classification problems, SVM works well due to its advantages, such as its potential to handle large features since it uses overfitting protection [28]. SVM finds the best hyperplane which separates the two input classes in input space. $x^k = f^k$ and w, b is acquired by minimizing the loss function [29]. Its final equation is given below:

$$L(w, b) = w_t w + c \sum \max(0, 1 - y^i(w^T F^{(i)} + b))^2 \quad (2)$$

2) NAÏVE BAYES

NB classifier is the statistical classifier that can predict class membership probabilities. The adjective naïve comes from the assumption that features in the dataset are independent of each other. Using the class c and document d, it has the following form:

$$P(d) = \sum_{j=1}^{|c|} p(c_j)p(d|c_j) \quad (3)$$

where, c_j corresponds to the possible classes, and $P(c_j)$ is the prior probability. Using the Bayesian theorem, the model can be inverted to obtain the posterior probability as:

$$P(c_j|d) = \frac{P(c_j)(d|c_j)}{P(d)} \quad (4)$$

For document classification, the classifier selects a class with maximum posterior probability. The most likely class is given by the following equation:

$$c^*(d) = \underset{j}{argmax} P(c_j) \quad (5)$$

3) RANDOM FOREST

RF classifier is used for classification and regression problems. It helps in decision-making tasks by forming trees for them. It works for categorical and numerical features whereas for our classification problem, it gives us the probability according to specific classes. RF efficiently runs on a large amount of data. It generates a different subset of training data. It focuses to train many decision trees and lets them select the most popular class. The idea of combining many decision classifiers gives special features to the random forest that significantly differentiated it from other traditional classifiers. A single decision tree may impair the performance of the overall model. To overcome such a problem random forest provides randomness due to its robustness to outliers and noise [30]. RF has low bias and high variability which helps learn irregular patterns.

4) LOGISTIC REGRESSION

LR classifier takes the input vector and finds the coefficient for the input expression, thereby determining the class of the text as a word vector. LR determines several linear functions, expressed as:

$$logit(P) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k \quad (6)$$

where P represents the probability of the occurrence of the features. $X_1, X_2 \ldots . X_k$ and $\beta_1, \beta_2 \ldots \beta_k$ represent the value of the predictor and the model intercept respectively [31].

5) K NEAREST NEIGHBOR

KNN is a well-known classifier that predicts the label using a set of training data points that are nearest in distance to the new point. To predict with KNN, a distance metric needs to be defined that will calculate the distance between the two points; a query point and a training data point [32], [33]. To calculate the distance, Euclidean distance is one of the most common distance metrics which is defined as:

$$d(p, q) = d(q, p) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \ldots + (q_n - p_n)^2} \quad (7)$$

$$dist(x, y) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \quad (8)$$

where q represents the query data point and p represents the training data point.

F. REAL-WORD ERROR CORRECTION

Once a misspelled word is detected, the candidate correction phase is started in which different candidate words are generated to correct the error. At this stage, we do not consider the context as we first need to determine the candidate words for correction that can be replaced with the error word. The context of the candidate words, selected in this phase, is determined in the ranking stage for selecting the best suggested candidate word that should be replaced with the error word in the sentence.

In the literature, numerous algorithms have been used to find candidates for correction. The minimum edit distance algorithm is by far the most popular one. For this study, we also have used a minimum edit algorithm for generating

TABLE 6. Example of spelling error correction.

SL	Input	Error word	Suggested words	Sentences with the suggested word
1	جیتے رہو سدا جوش رہو	جوش	ہوش	جیتے رہو سدا ہوش رہو
			دوش	جیتے رہو سدا دوش رہو
			خوش	جیتے رہو سدا خوش رہو
			گوش	جیتے رہو سدا گوش رہو
			پوش	جیتے رہو سدا پوش رہو
2	پھول کی ضرورت میں بھی یہ جلوہ گر ہوا ہے۔	ضرورت	عورت	پھول کی عورت میں بھی یہ جلوہ گر ہوا ہے۔
			صورت	پھول کی صورت میں بھی یہ جلوہ گر ہوا ہے۔
			سورت	پھول کی سورت میں بھی یہ جلوہ گر ہوا ہے۔
			مورت	پھول کی مورت میں بھی یہ جلوہ گر ہوا ہے۔
			عورت	پھول کی عورت میں بھی یہ جلوہ گر ہوا ہے۔
			بصورت	پھول کی بصورت میں بھی یہ جلوہ گر ہوا ہے۔
3	آپ نے اپنی شاعر میں جس شاہین کا تذکرہ کیا ہے وہ شاہین یہی ہے	شاعر	شاعری	تم نے اپنی شاعری میں جس شاہین کا تذکرہ کیا ہے وہ شاہین یہی ہے۔
			شعری	تم نے اپنی شعری میں جس شاہین کا تذکرہ کیا ہے وہ شاہین یہی ہے۔
			شعر	تم نے اپنی شعر میں جس شاہین کا تذکرہ کیا ہے وہ شاہین یہی ہے۔
			شاعرہ	تم نے اپنی شاعرہ میں جس شاہین کا تذکرہ کیا ہے وہ شاہین یہی ہے۔

the list of candidate words. After that, new sentences are formed by replacing the erroneous word with all its correct variations.

While correcting real word errors, we use different confusion sets, e.g., first, we obtain the maximum word length from the dictionary words and generate different confusion sets of length 1 up to the maximum length of the dictionary word. In the first step, we calculate the error word length and select the three confusion sets with respect to the error word length, i.e., confusion set with the same length, one length smaller confusion set, and one length greater confusion set with the error word. Then we calculate the distance of the error word with the selected confusion set. If no candidate word with minimum edit distance is found, then we go to the next confusion sets of two lengths smaller and two lengths greater with respect to the error word. We repeat this process until we reach the maximum limit, i.e., the maximum dictionary word length. Table 6 shows examples of spelling error correction. Algorithm 1 summarizes the error correction phase.

G. CANDIDATE RANKING

Distance algorithms during candidate generation find similar words for the misspelled word without context consideration. Therefore, for selecting the best candidate word according to the context, we rank the candidate words using the language model. To determine the context we use the n-gram language model which gives us the best suggested word. Shannon was the first to use n-grams in natural language processing [34]. An n-gram is a sequence of n number of contiguous elements. They are referred to as unigrams, bigrams, and trigrams when $n = 1, 2$, and 3 respectively. The greatest advantage of using n-gram is that they are language-independent. For candidate ranking, we use unigram, bigram, and trigram separately with our proposed approach (discussed below) to see which n-gram model along with our proposed approach gives the best-suggested candidate word according to their context. As the size of the n-gram increases, more contextual information has been obtained. The trigram is good for extracting the context of its neighbors' words, although trigram has fewer occurrences than bigram and unigram, it extracts more contextual information. Assuming we have a sentence S containing n words, $S = w_1, w_2, , w_3, \cdots, w_n$ and a set of candidate words CS consisting of z words

Algorithm 1 Real-Word Error Correction

Input: Urdu text T with real-word errors
Output: Urdu Text T with error corrections

Begin:
Let W_e be the error word, L_e be the list of all real word errors, $max(Len_d)$ be the maximum length of a dictionary word and Len_n be the length of the error word W_e. L_w is the list of candidate words
Find a word from dictionary with $max(Len_d)$
Generate different confusion sets C_s of Len_1 up to the $max(Len_d)$ from the dictionary
for each $W_e \in L_e$ **do**
  Find candidate words with distance d from C_s of Len_n, Len_{n-1} and Len_{n+1}
  Add candidate word c to L_w
  if $L_w = \emptyset$ **then**
    Find candidate words from C_s of Len_{n-2} and Len_{n+2}
    Repeat the process of finding candidate word up to the $max(Len_d)$
    if $Len_{n+1} > max(Len_d)$ **then**
      break;
    end if
  end if
  for each $c \in L_w$ **do**
    Replace W_e with c designate the resultant text T'
    Calculate the new text T' frequency using the language model and proposed approach.
  end for
end for
Pick candidate c with the highest frequency score
Replace W_e with c in the text T
end:

TABLE 7. Unigram example.

Real-word error	Candidate words	Occurrence Frequency
قادری	قادر	4214
	قاری	1790
	پادری	335
	مادری	305
	نادری	12

$CS(CW_i^j) = CW_i^1, CW_i^2, \cdots, CW_i^Z$, then CW_i^j is the j^{th} candidate word from the n number of words in the sentence which will be replaced on the i^{th} the place to generate n-gram. For each CW_i^j, we find trigram, bigram, and unigram as shown in Equations 9, 10, 11.

$$Trigram = W_{i-2}\ W_{i-1}\ CW_i^j \tag{9}$$
$$Bigram = W_{i-1}\ CW_i^j \tag{10}$$
$$Unigram = CW_i^j \tag{11}$$

where CW_i^j is the i^{th} word in the sentence which will be replaced with the j^{th} candidate word and the range is $1 \leq i \leq n$ and $1 \leq j \leq Z$. After finding the n-gram we find the frequency of different n-grams and create trigram, bigram, and unigram as shown below:

$$Trigram = freq(W_{i-2}\ W_{i-1}\ CW_i^j) \tag{12}$$
$$Bigram = freq(W_{i-1}\ CW_i^j) \tag{13}$$
$$Unigram = freq(CW_i^j) \tag{14}$$

We select the candidate word as a suggested word for the misspelled word whose frequency of occurrence is high. Using unigram ranking, the candidate words are ranked based on their frequency of occurrence in the corpus and the most occurring candidate word is selected as the most suitable word. Consider for example the following sentence containing real-word error:

کسی قوم کے لیے اظہار کا سب سے بہترین ذریعہ اُس کی قادری زبان ہوتی ہے۔

In the sentence above, the word "قادری" (*Qadri*) is a real-word error because it is contextually incorrect. After applying the word correction technique, the candidate words obtained are "پادری" (*Padri*) (*The priest*), "قاری" (*Qari*) (*Reader*), "قادر" (*Qadir*) (*Capable*), "مادری" (*Madri*) (*maternal*), and "نادری" (*Nadri*) (*A rarity*).

After replacing the erroneous word with each of its candidate words, the following sentences are formed:

کسی قوم کے لیے اظہار کا سب سے بہترین ذریعہ اُس کی قادر زبان ہوتی ہے۔
کسی قوم کے لیے اظہار کا سب سے بہترین ذریعہ اُس کی قاری زبان ہوتی ہے۔
کسی قوم کے لیے اظہار کا سب سے بہترین ذریعہ اُس کی پادری زبان ہوتی ہے۔
کسی قوم کے لیے اظہار کا سب سے بہترین ذریعہ اُس کی مادری زبان ہوتی ہے
کسی قوم کے لیے اظہار کا سب سے بہترین ذریعہ اُس کی نادری زبان ہوتی ہے۔

After applying unigram ranking, the candidate words are ranked based on their occurrences in the corpus as shown in Table 7.

Unigram ranking gives us the word "قادر" (*Qadir*) (*Capable*) instead of the word "مادری" (*Madri*) (*maternal*) as a suggested candidate word of the erroneous word "قادری" (*Qadri*) because the frequency of occurrence of word "قادر" (*Qadir*) (*Capable*) is greater than the word "مادری" (*Madri*) (*maternal*). After candidate ranking, the following sentence is obtained.

کسی قوم کے لیے اظہار کا سب سے بہترین ذریعہ اُس کی قادر زبان ہوتی ہے۔

While the correct sentence should be:

کسی قوم کے لیے اظہار کا سب سے بہترین ذریعہ اُس کی مادری زبان ہوتی ہے۔

The best means of expression for a nation is its mother tongue.

Since the unigram language model only considers the frequency of occurrence of a word individually without considering the neighboring words, it fails to consider the context of the word.

Using the bigram language model, we rank the candidate words based on their occurrence with their previous words. The candidate word with the highest bigram ranking is selected as the suggested candidate word. For the above sentence, bigram ranking gives "مادری" (*Madri*) (*maternal*) as the suggested candidate word because it occurred 72 times with its previous word "کی" while the candidate words "قادر" (*Qadir*) and "پادری" (*Padri*) occurred 13 times and 7 times respectively, with the previous word "کی". Meanwhile, there where zero occurrances of the candidate words "قاری" (*Qari*) and "نادری" (*Nadri*) with the previous word "کی". Therefore, we select "مادری" (*Madri*) as our suggested candidate word based on its highest-ranking score.

Using the trigram language model, we rank the candidate words based on their occurrence with the previous two words. The candidate word with the highest trigram ranking is selected as the suggested candidate word. For the above sentence, trigram ranking gives "مادری" (*Madri*) as the suggested candidate word because it occurred 5 times with the previous two words "اس کی" while the candidate word "قادر" (*Qadir*) only occurred once, and the remaining suggested candidate words did not occur, with the previous two words "اس کی". Therefore "مادری" (*Madri*) is selected as the suggested candidate word due to its highest-ranking score.

Since bigram and trigram models consider the neighboring words, they can determine the context by using the words present in the surroundings of the misspelled word.

Consider applying bigram ranking approach to another example sentence below:

میری طرف سے بھی سب لوگوں کو عید تبارک ہو اور میں تو چلی مہندی لگوانے۔

In the above sentence, the word "تبارک"(*Tabarak*) is a real-word error. After applying the word correction technique, the candidate words obtained are "مبارک"(*Mubarak*), "یبارک"(*Yabarak*), and "بارک"(*Barak*). with each of its candidate words, the following sentences are formed:

میری طرف سے بھی سب لوگوں کو عید مبارک ہو اور میں تو چلی مہندی لگوانے۔
میری طرف سے بھی سب لوگوں کو عید یبارک ہو اور میں تو چلی مہندی لگوانے۔
میری طرف سے بھی سب لوگوں کو عید بارک ہو اور میں تو چلی مہندی لگوانے۔

Applying bigram ranking gives us the suggested candidate word "مبارک"(*Mubarak*) because it occurs 317 times with its previous word "عید"(*Eid*). Therefore, the word "مبارک"(*Mubarak*)(*Congratulations*) is selected as the suggested word. The following sentence is obtained after applying Bigram ranking which is also a correct sentence.

عید مبارک میری طرف سے بھی سب لوگوں کو عید مبارک ہو اور میں تو چلی مہندی لگوانے۔

Eid Mubarak from my side to everyone and off I go to get mehndi applied.

Consider applying the trigram ranking approach to another example sentence below:

کھومبے جادو گر سب سے الگ تھلگ اداس کاڑا سوچ رہا تھا۔

In the above sentence, the word "کاڑا" (*Kara*) (*A piece of cake*) is a real-word error. After applying correction technique, the candidate words obtained are "کیڑا" (*Kera*) (*Worm*), "کوڑا" (*Kura*) (*Garbage*), "کھڑا" (*Khara*) (*Stand*), and "کپڑا" (*Kapra*) (*Cloth*). We obtain the following sentences by replacing the erroneous word with each of its correction variations:

کھومبے جادو گر سب سے الگ تھلگ اداس کپڑا سوچ رہا تھا۔
کھومبے جادو گر سب سے الگ تھلگ اداس کھڑا سوچ رہا تھا۔
کھومبے جادو گر سب سے الگ تھلگ اداس کوڑا سوچ رہا تھا۔
کھومبے جادو گر سب سے الگ تھلگ اداس کیڑا سوچ رہا تھا۔

When trigram ranking is applied to these candidate words, it gives the suggested word "کھڑا" (*Khara*) (*Stand*) because its trigram frequency is higher than the other candidate words. The following sentence is obtained after applying trigram ranking which is also a correct sentence.

کھومبے جادو گر سب سے الگ تھلگ اداس کھڑا سوچ رہا تھا۔

Khombe the magician stood isolated and sad, thinking.

When we apply the language model for ranking candidate words, if some suggested words have the same unigram, bigram, or trigram frequencies, then for further ranking we propose another approach.

1) PROPOSED APPROACH FOR FURTHER RANKING OF SUGGESTED WORDS

In the proposed approach, we first find three trigrams for each suggested word (1) trigram with the backward and forward word, (2) backward trigram, and (3) forward trigram as shown in Equation 15.

$$Tri = \{W_{i-2}\ \ W_{i-1}\ \ CW_i^j\ , W_{i-1}\ \ CW_i^j\ W_{i+1}\ , \\ CW_i^j\ W_{i+1}\ W_{i+2}\ \} \tag{15}$$

where, the trigram with the backward and forward words is represented as $W_{i-1}\ \ CW_i^j\ W_{i+1}$, the backward trigram is represented as $W_{i-2}\ \ W_{i-1}\ \ CW_i^j$, and the forward trigram is represented as $CW_i^j\ W_{i+1}\ \ W_{i+2}$. Then, the frequency of these trigrams is calculated and added to get the final score as shown in Equation 16. The addition is used instead of multiplication to save computation time and avoid underflow. We select the suggested word whose total trigram score is the highest.

$$freq(Tri) = \{(W_{i-2}\ \ W_{i-1}\ \ CW_i^j) \\ + (W_{i-1}\ \ CW_i^j\ W_{i+1}) \\ + (W_i^j\ W_{i+1}\ W_{i+2}\ )\} \tag{16}$$

If trigram is not found, then bigram back off is used as shown in Equations 17 and 18. Algorithm 2 summarizes the proposed approach for candidate ranking.

$$Bi = (W_{i-1}CW_i^j), (W_i^jW_{i+1}) \tag{17}$$

$$freq(Bi) = (W_{i-1}CW_i^j) + (W_i^jW_{i+1}) \tag{18}$$

Algorithm 2 Candidate Ranking Using Proposed Appraoch

Input: Urdu text T with Error Correction
Output: Urdu Text T with suggested Candidate Word

```
Begin:
Let S be the list of sentences and T' is a sentence obtained
after applying the language model. freq(Tri) is the trigram
frequency and freq(Bi) is the bigram frequency.
for each T' ∈ S do
    if freq(Tri)inT then
        Pick the suggested candidate word whose trigram
        frequency score is high
    else
        Calculate the bigram frequency freq(Bi)
        Pick the suggested word with the highest bigram
        frequency
    end if
end for
end:
```

Unigram + Proposed approach: In the case of unigram ranking, some candidate words have the same frequency. Consider the following sentence:

مقامی سعودی لوگ بھی اپنی فیملیوں کے ساتھ پاکستانی مطاع میں پاکستانی پکوان بہت زیادہ شوق سے کھاتے ہوئے ملتے ہیں

In the sentence above, "مطاع" (*Mataa*) (*Obedience*) is a real-word error. After candidate correction, the candidate words obtained are "مطاعم" (*Mataam*) (*Restaurants*), "مطاف" (*Mutaf*), "مطاعہ" (*Mataaia*) (*A favor*), "مطاعن" (*Mutaaian*) (*The curse*) and "مطاعی" (*Matai*). Candidate words "مطاعم" (*Mataam*) (*Restaurants*) and "مطاف" (*Mutaf*) occurred two times in the corpus, while the remaining candidate words occurred only one time in the corpus. Therefore, we use our proposed approach for further ranking the two suggested candidate words so that the best word is selected. We first apply the three-trigram approach to the suggested candidate word "مطاعم" (*Mataam*) (*Restaurants*) as shown in Table 8.

Then for the suggested candidate word "مطاف" (*Mutaf*) we apply three trigrams as shown in Table 9.

We select "مطاعم" (*Mataam*) (*Restaurants*) as our suggested word because its trigram score is the highest. After replacing

TABLE 8. Trigrams for suggested candidate word مطاعم.

	Trigram	Frequency
Previous Trigram	ساتھ پاکستانی مطاعم	1
Prev + Fwd Trigram	پاکستانی مطاعم میں	1
Forward Trigram	مطاعم میں پاکستانی	1
Total trigram score=3		
Prev = Previous, Fwd = Forward		

TABLE 9. Trigrams for suggested candidate word مطاف.

	Trigram	Frequency
Previous Trigram	ساتھ پاکستانی مطاف	0
Prev + Fwd Trigram	پاکستانی مطاف میں	0
Forward Trigram	مطاف میں پاکستانی	0
Total trigram score=0		
Prev = Previous, Fwd = Forward		

the misspelled word with the suggested word, we obtain the following correct sentence.

مقامی سعودی لوگ بھی اپنی فیملیوں کے ساتھ پاکستانی مطاعم میں پاکستانی پکوان زیادہ شوق سے کھاتے ہوئے ملتے ہیں

Local Saudi people also enjoy eating Pakistani dishes with their families in Pakistani restaurants.

Bigram + Proposed approach: The candidate words ranked by the Bigram language model can also have the same frequency. Consider the following sentence:

اس موقع پر یوراج سنگھ اور سریش رائنا نے اسکور کو سنبھالنے کی کوشش کی تاہم جنوری افریقی بالرز بھارتی بلے بازوں کو کوئی بھی موقع دینے کے موڈ میں نہ تھے۔

In the above sentence, the word, "جنوری" (*January*) is a real-word error. After applying the word correction technique, the candidate words obtained are "جنونی" (*Janooni*) (*The fanatic*), "نوری" (*Noori*) (*Light*), "جیوری" (*jury*) (*The jury*), "جنوبی" (*janoobi*) (*south*). When bigram ranking is applied to these candidate words, the bigram frequency of suggested candidate words "جنوبی" (*janoobi*) (*south*) and "نوری" (*Noori*)(*Light*) with their previous word "تاہم" (*Taham*) (*However*) is equal as both "تاہم جنوبی" (*Taham Janoobi*) and "تاہم نوری" (*Taham Noori*) occurred three times in the corpus. "تاہم جیوری" (*Taham Jury*) occurred one time and "تاہم جنونی" (*Taham Jannoni*) never occurred in the corpus. Therefore, for selecting the best-suggested word we perform further ranking for both candidate words whose bigram frequency is equal. First, we calculate the three trigrams frequency for the candidate word "جنوبی" (*janoobi*) (*south*) as shown in Table 10.

TABLE 10. Trigrams for suggested candidate word جنوبی.

	Trigram	Frequency
Previous Trigram	کی تاہم جنوبی	2
Prev + Fwd Trigram	تاہم جنوبی افریقی	1
Forward Trigram	جنوبی افریقی بالرز	1
Total trigram score=4		
Prev = Previous, Fwd = Forward		

Now we calculate the three trigrams for the candidate word "نوری" (*Noori*) (*Light*) as shown in Table 11.

TABLE 11. Trigrams for suggested candidate word نوری.

	Trigram	Frequency
Previous Trigram	کی تاہم نوری	1
Prev + Fwd Trigram	تاہم نوری افریقی	0
Forward Trigram	نوری افریقی بالرز	0
Total trigram score=1		
Prev = Previous, Fwd = Forward		

We select the suggested word "جنوبی" (*janoobi*) (*south*) because of its highest trigram score. After replacing the misspelled word with the suggested word, we obtain the following sentence which is also a correct sentence.

اس موقع پر یوراج سنگھ اور سریش رائنا نے اسکور کو سنبھالنے کی کوشش کی تاہم جنوبی افریقی بالرز بھارتی بلے بازوں کو کوئی بھی موقع دینے کے موڈ میں نہ تھے۔

On this occasion, Yuvraj Singh and Suresh Raina tried managing the score, but the South African bowlers were in no mood to give any chance to the Indian batsmen.

Trigram + Proposed approach: We also apply the three-trigram approach when some candidate words have the same trigram frequency.

کہیں یہ زندگی اور روان کے حسین نظاروں سے عبارت ہے۔

For example, in the sentence above, the word "روان" (*Rawaan*) (*Running*) is a real-word error. After applying the word correction technique, the candidate words obtained are "رومان" (*Romaan*) (*Romance*), "رومن" (*Roman*), "رواں" (*Rawan*)(*live*), "روا" (*Rua*). The trigram frequency of the candidate words "رومان" (*Romaan*) (*Romance*) and "رواں" (*Rawan*)(*live*) is equal as both "زندگی اور رومان" (*Zindagi our romaan*) and "زندگی اور رواں" (*Zindagi our rawan*) occurred one time in the corpus. The remaining candidate word with their previous words never occurred in the corpus. Therefore, for further ranking of both candidate words, we apply the three-trigram ranking approach for suggested candidate words "رومان" (*Romaan*) (*Romance*) and "رواں" (*Rawan*) (*live*) as shown in Table 12 and Table 13 respectively.

TABLE 12. Trigrams for suggested candidate word رومان.

	Trigram	Frequency
Previous Trigram	زندگی اور رومان	1
Prev + Fwd Trigram	اور رومان کے	1
Forward Trigram	رومان کے حسین	1
Total trigram score=3		
Prev = Previous, Fwd = Forward		

We select the suggested candidate word "رومان" (*Romaan*)(*Romance*) because of its highest trigram score. After replacing the misspelled word with the suggested word, we obtain the following correct sentence.

کہیں یہ زندگی اور رومان کے حسین نظاروں سے عبارت ہے۔

Somewhere it is composed of beautiful views of life and romance.

TABLE 13. Trigrams for suggested candidate word رواں.

	Trigram	Frequency
Previous Trigram	زندگی اور رواں	1
Prev + Fwd Trigram	اور رواں کے	0
Forward Trigram	رواں کے حسین	0
Total trigram score=1		
Prev = Previous, Fwd = Forward		

In some cases, when unigram, bigram, and trigram ranking is applied, sometimes when we further perform ranking on the suggested candidate words who have the same frequency, we do not find three trigrams and their score is 0. Therefore, in that case, we calculate the two bigrams (1) forward bigram and (2) backward bigram, as shown in Equation 19.

$$freq(CW_i^j) = (W_{i-1}CW_i^j) + (W_i^j W_{i+1}) \tag{19}$$

Consider the following sentence:

حکومت کی کوشش ہے کہ اسی مانی دنیا میں سونے کی ترسیل کا اہم مرکز بن سکے۔

In the sentence above, the word "مانی" (*Mani*) (*Admit*) is a real-word error. After applying correction, we obtained the following candidate words; "ماضی" (*Mazi*) (*Past*), "ماری" (*Mari*) (*Beat*), "ماہی" (*Mahi*), "مامی" (*Mami*) (*Aunt*), "ماسی" (*Masi*) (*Aunty*),"مافی" (*Mafi*) (*Sorry*), "مائی" (*Mati*) (*The soil*), "مالی" (*Mali*) (*financial*), "مادی" (*Madi*)(*Material*). Candidate words "مادی" (*Madi*) (*Material*) and "مالی" (*Mali*) (*financial*) have the highest bigram frequency with its previous word "اسی" (*Isi*) (*That's it*). Both of the suggested candidate words "اسی مالی" (*Isi Mali*) and "اسی مادی" (*Isi Madi*) have occurred 3 times in the corpus. When we apply the three-trigram approach for further ranking, the trigram score of both suggested candidate words score is 0 as shown in Table 14 and Table 15.

TABLE 14. Trigrams for suggested candidate word مالی.

	Trigram	Frequency
Previous Trigram	اسی مالی کہ	0
Prev + Fwd Trigram	اسی مالی دنیا	0
Forward Trigram	مالی دنیا میں	0
Total trigram score=0		
Prev = Previous, Fwd = Forward		

TABLE 15. Trigrams for suggested candidate word مادی.

	Trigram	Frequency
Previous Trigram	اسی مادی کہ	0
Prev + Fwd Trigram	اسی مادی دنیا	0
Forward Trigram	مادی دنیا میں	0
Total trigram score=0		
Prev = Previous, Fwd = Forward		

Therefore, for further ranking we back off to two bigrams as shown in Tables 16 and 17, and based on the highest bigram score, we select "مالی" (*Mali*) (*financial*) as our suggested word.

After the misspelled word is replaced with the suggested word, the following correct sentence is obtained.

TABLE 16. Bigrams for suggested candidate word مالی.

	Bigram	Frequency
Backward Bigram	اسی مالی	3
Forward Bigram	مالی دنیا	4
Total brigram score=7		

TABLE 17. Bigrams for suggested candidate word مادی.

	Bigram	Frequency
Backward Bigram	اسی مادی	3
Forward Bigram	مادی دنیا	2
Total brigram score=5		

حکومت کی کوشش ہے کہ اسی مالی دنیا میں سونے کی ترسیل کا اہم مرکز بن سکے۔

The government is trying to become the main center of delivery of gold in this financial world.

IV. RESULTS AND DISCUSSION

A. EXPERIMENTAL SETUP

For this study, the Scikit-learn toolkit is used for experimentation and the context errors are classified into correct or misspelled words. TF-IDF vectorizer is used to extract features and five ML classifiers i.e., SVM, RF, NB, LR, and KNN, are used for context error classification. These classifiers are trained and tested on our prepared dataset. A split ratio of 80-20% is used for the training and testing dataset. For error detection, we perform four experiments using the unigram, bigram, trigram, and combined feature sets separately by applying various machine learning classifiers to see which feature set works well with machine learning classifiers for error classification. For error correction with candidate ranking, we use unigram, bigram, and trigram separately with our proposed approach to see which n-gram model along with our proposed approach gives the best-suggested candidate word. For all the experiments, a default set of parameters is defined. (1) DL is used for the edit distance measure and is set to $d_{DL} = 1$ or 2, (2) the error generation density is set to $E = 0.30$, and (3) $n = 1$ to 3 for word n-gram model.

B. EVALUATION MEASURES

Precision, recall, F1-score, and accuracy are used for the evaluation of the results. Precision is the proportion of correctly positive instances to the total positively classified instances, recall is the proportion of positively classified instances to the total truly positive instances, the F1-score is simply the harmonic mean of precision and recall, and accuracy is the percentage of correctly classified instances in the corpus [3]. To evaluate real-word error correction performance, we slightly redefine accuracy as shown in Equation 23, where N_{SCW} represents the number of the top suggested candidate words intended for correction and N_{DE} represents the number

TABLE 18. Results of Word gram features with error density of 30%.

Feature Set	Classifier	Precision	Recall	F1-score
Unigram	SVM	0.80	0.75	0.77
	RF	0.75	0.65	0.70
	NB	0.62	0.51	0.56
	LR	0.84	0.77	0.80
	KNN	0.84	0.72	0.78
Bigram	SVM	0.85	0.74	0.79
	RF	0.74	0.65	0.69
	NB	0.65	0.53	0.58
	LR	0.83	0.75	0.79
	KNN	0.81	0.80	0.80
Trigram	SVM	0.85	0.74	0.79
	RF	0.73	0.61	0.67
	NB	0.30	0.46	0.36
	LR	0.83	0.76	0.79
	KNN	0.81	0.79	0.80
Combined Features	SVM	0.82	0.71	0.76
	RF	0.73	0.63	0.68
	NB	0.71	0.52	0.60
	LR	0.84	0.79	0.81
	KNN	0.83	0.71	0.77

TABLE 19. Results of word gram features with error density of 40%.

Feature Set	Classifier	Precision	Recall	F1-score
Unigram	SVM	0.77	0.73	0.75
	RF	0.73	0.64	0.68
	NB	0.58	0.49	0.53
	LR	0.79	0.75	0.77
	KNN	0.79	0.73	0.76
Bigram	SVM	0.83	0.74	0.78
	RF	0.72	0.60	0.65
	NB	0.60	0.50	0.55
	LR	0.80	0.77	0.78
	KNN	0.81	0.79	0.80
Trigram	SVM	0.83	0.74	0.79
	RF	0.72	0.63	0.67
	NB	0.30	0.36	0.33
	LR	0.82	0.77	0.79
	KNN	0.80	0.76	0.78
Combined Features	SVM	0.78	0.75	0.76
	RF	0.72	0.61	0.66
	NB	0.65	0.53	0.5
	LR	0.84	0.79	0.81
	KNN	0.78	0.73	0.75

of detected real-word errors.

$$Precision = \frac{T_P}{T_P + F_P} \tag{20}$$

$$Recall = \frac{T_P}{T_P + F_N} \tag{21}$$

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{22}$$

$$Accuracy = \frac{N_{SCW}}{N_{DE}} \tag{23}$$

C. RESULTS

This section discusses the details of the experimental results of real-word error detection and correction. To detect context errors, we run four experiments using (i) unigram, (ii) bigram, (iii) trigram, and (iv) combined feature sets separately, by applying various ML classifiers to measure the performance of detecting context errors. Table 18 shows the results by considering the different order n-grams features. LR with combined word n-gram features (n = 1 to 3) performed well giving the best precision, recall, and F1-score of 0.84,0.79, and 0.81 respectively. A possible reason for high performance is that LR sets all the features jointly allowing for taking into account possible correlations between features. Another reason for the better performance of LR is that there is no noisy feature in our dataset as all noisy features have been removed during the preprocessing step. NB performs poorly having a precision of 0.30, recall of 0.46, and F1-score of 0.36 for the trigram features. This is because NB works on independent assumptions. It performs poorly because features are not independent given the class label. Features are dependent on one another.

While preparing the dataset, the error density (E) was set to 0.30. To see the impact of higher error density on the performance of context error detection, we conducted experiments with an error density of 0.4. Table 19 shows the results with an error density of 0.4.

At 40% error density, LR outperforms other classifiers with the precision, recall, and F1-score of 0.84,0.79, and 0.81, respectively for the combined features. Its detection performance remains steady at an F1-score of 0.81 in the case of 30% of error density and 40% of error density. Again, NB performs the worst having precision of 0.30, a recall of 0.36, and an F1-score of 0.33 for trigram features.

D. IMPACT ON ERROR CORRECTION

Upon detection of a context error, it is replaced with the correct word. We use accuracy as an evaluation measure to determine the performance of error correction with ranking, using the same default setting mentioned in section IV-A. We perform experiments on a mixture of real-word errors having edit distances of 1 and 2. 80% errors have an edit distance of 1 from the correct word while the remaining errors have an edit distance of 2. To evaluate which approach gives the best-suggested candidate word, we perform three experiments for error correction using a combination of correction and ranking approaches namely (i) unigram with DL and the proposed approach, (ii) bigram with DL and the proposed approach, and (iii) trigram with DL and the proposed approach. Table 20 shows the result of error correction with the three ranking approaches. Trigram with DL and the proposed ranking approach outperform bigram and unigram because it captures contextual information more accurately and considers the previous two words. 83.67% of the detected context errors are appropriately corrected using trigram. The correction performance decreases when we apply bigram and unigram with DL and the proposed ranking approach because they capture less contextual information as compared to trigram. Bigram considers the previous one word while unigram does not consider any contextual information. Bigram and

unigram give us an accuracy rate of 76.33% and 52.65% respectively.

TABLE 20. Error correction results.

Correction Approach	Accuracy
DL + Unigram + PA	52.65%
DL + Bigram + PA	76.33%
DL+ Trigram +PA	83.67%

PA = Proposed Approach

V. CONCLUSION AND FUTURE WORK

This study presents a contextual spell checker for real-word errors in the Urdu language. The entire process of creating the corpus, lexicon, and contextual spell checker is illustrated. For real-word error detection, word n-gram models and five ML classifiers namely SVM, RF, NB, LR, and KNN are used. Additional test for error classification is performed using error densities of 30% and 40%. LR shows the best performance having a precision of 0.84, recall of 0.79, and F1-score of 0.81 for both error densities of 30% and 40%. This shows that error detection performance remains steady even after increasing the error density. For error correction, candidate words are generated using DL and ranked using the n-gram language model. An additional approach based on three trigrams or two bigrams is also proposed if suggested candidate words obtained after ranking have the same unigram, bigram, or trigram frequencies. The best correction accuracy is obtained when DL, trigram, and the proposed approach are applied collectively. For correcting context errors, the average accuracy is 83.67%. In the future, we plan to detect and correct multiple contextual errors in a single sentence. Additionally, We plan to use state-of-the-art deep learning models for real-word error detection and correction for the Urdu language.

REFERENCES

[1] S. Sharma and S. Gupta, "A correction model for real-word errors," *Proc. Comput. Sci.*, vol. 70, pp. 99–106, Jan. 2015.

[2] S. Singh and S. Singh, "Review of real-word error detection and correction methods in text documents," in *Proc. 2nd Int. Conf. Electron., Commun. Aerosp. Technol. (ICECA)*, Mar. 2018, pp. 1076–1081.

[3] S. Singh and S. Singh, "HINDIA: A deep-learning-based model for spell-checking of Hindi language," *Neural Comput. Appl.*, vol. 33, no. 8, pp. 3825–3840, Apr. 2021.

[4] J.-H. Lee, M. Kim, and H.-C. Kwon, "Deep learning-based context-sensitive spelling typing error correction," *IEEE Access*, vol. 8, pp. 152565–152578, 2020.

[5] P. Samanta and B. B. Chaudhuri, "A simple real-word error detection and correction using local word bigram and trigram," in *Proc. 25th Conf. Comput. Linguistics Speech Process. (ROCLING)*, 2013, pp. 211–220.

[6] 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.

[7] T. Naseem, "A hybrid approach for Urdu spell checking," M.S. thesis, Dept. Comput. Sci., Nat. Univ. Comput. Emerg. Sci., Lahore, Pakistan, 2004.

[8] S. Iqbal, W. Anwar, U. I. Bajwa, and Z. Rehman, "Urdu spell checking: Reverse edit distance approach," in *Proc. 4th Workshop South Southeast Asian Natural Lang. Process.*, 2013, pp. 58–65.

[9] R. Aziz and M. W. Anwar, "Urdu spell checker: A scarce resource language," in *Proc. 2nd Int. Conf. Intell. Technol. Appl. (INTAP)*, Bahawalpur, Pakistan. Singapore: Springer, 2020, pp. 471–483.

[10] R. Aziz, M. W. Anwar, M. H. Jamal, and U. I. Bajwa, "A hybrid model for spelling error detection and correction for Urdu language," *Neural Comput. Appl.*, vol. 33, no. 21, pp. 14707–14721, Nov. 2021.

[11] A. Naseer and S. Hussain, "Supervised word sense disambiguation for Urdu using Bayesian classification," Center Res. Urdu Lang. Process., Lahore, Pakistan, Tech. Rep., 2009.

[12] M. M. Rana, M. T. Sultan, M. F. Mridha, M. E. A. Khan, M. M. Ahmed, and M. A. Hamid, "Detection and correction of real-word errors in Bangla language," in *Proc. Int. Conf. Bangla Speech Lang. Process. (ICBSLP)*, Sep. 2018, pp. 1–4.

[13] M. F. Mridha, M. A. Hamid, M. M. Rana, M. E. A. Khan, M. M. Ahmed, and M. T. Sultan, "Semantic error detection and correction in Bangla sentence," in *Proc. Joint 8th Int. Conf. Informat., Electron. Vis. (ICIEV), 3rd Int. Conf. Imag., Vis. Pattern Recognit. (icIVPR)*, May 2019, pp. 184–189.

[14] H. Faili, "Detection and correction of real-word spelling errors in Persian language," in *Proc. 6th Int. Conf. Natural Lang. Process. Knowl. Engineering(NLPKE)*, Aug. 2010, pp. 1–4.

[15] D. Bravo-Candel, J. López-Hernández, J. A. García-Díaz, F. Molina-Molina, and F. García-Sánchez, "Automatic correction of real-word errors in Spanish clinical texts," *Sensors*, vol. 21, no. 9, p. 2893, Apr. 2021.

[16] T. M. Kassa and K. E. Andargie, "Sentence level n-gram context feature in real-word spelling error detection and correction: Unsupervised corpus based approach," *J. Inf. Eng. Appl.*, vol. 10, no. 4, pp. 12–20, 2020.

[17] H. Wang, X. Cao, L. Liu, and D. Gu, "Chinese real-word error automatic detection and correction based on confusion set and generalization model," in *Proc. IEEE/WIC/ACM Int. Joint Conf. Web Intell. Intell. Agent Technol. (WI-IAT)*, Dec. 2020, pp. 632–635.

[18] S. Roy and F. B. Ali, "Unsupervised context-sensitive Bangla spelling correction with character N-gram," in *Proc. 22nd Int. Conf. Comput. Inf. Technol. (ICCIT)*, Dec. 2019, pp. 1–6.

[19] R. Sakuntharaj and S. Mahesan, "Detecting and correcting real-word errors in Tamil sentences," *Ruhuna J. Sci.*, vol. 9, no. 2, p. 150, Dec. 2018.

[20] N. Hossain, S. Islam, and M. N. Huda, "Development of Bangla spell and grammar checkers: Resource creation and evaluation," *IEEE Access*, vol. 9, pp. 141079–141097, 2021.

[21] M. N. Jahan, A. Sarker, S. Tanchangya, and M. A. Yousuf, "Bangla real-word error detection and correction using bidirectional LSTM and bigram hybrid model," in *Proc. Int. Conf. Trends Comput. Cognit. Eng. (TCCE)*. Singapore: Springer, 2020, pp. 3–13.

[22] G. Huang and M. Li, "An errors correction model for the errors of non-word and real-word in English composition," *J. Comput.*, vol. 33, no. 1, pp. 139–150, 2022.

[23] A. Toleu, G. Tolegen, R. Mussabayev, A. Krassovitskiy, and I. Ualiyeva, "Data-driven approach for spellchecking and autocorrection," *Symmetry*, vol. 14, no. 11, p. 2261, Oct. 2022.

[24] A. Islam and D. Inkpen, "Real-word spelling correction using Google web 1T n-gram with backoff," in *Proc. Int. Conf. Natural Lang. Process. Knowl. Eng.*, Sep. 2009, pp. 1241–1249.

[25] J. Pedler, "Computer correction of real-word spelling errors in dyslexic text," Ph.D. dissertation, School Comput. Math. Sci., Birkbeck, Univ. London, London, U.K., 2007.

[26] F. J. Damerau, "A technique for computer detection and correction of spelling errors," *Commun. ACM*, vol. 7, no. 3, pp. 171–176, Mar. 1964.

[27] H. E. Wynne and Z. Z. Wint, "Content based fake news detection using N-gram models," in *Proc. 21st Int. Conf. Inf. Integr. Web-Based Appl. Services*, Dec. 2019, pp. 669–673.

[28] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Proc. Eur. Conf. Mach. Learn.* Berlin, Germany: Springer, Apr. 1998, pp. 137–142.

[29] W. Bourequat and H. Mourad, "Sentiment analysis approach for analyzing iPhone release using support vector machine," *Int. J. Adv. Data Inf. Syst.*, vol. 2, no. 1, pp. 36–44, Apr. 2021.

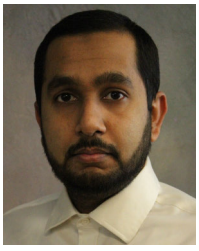
[30] A. Isied and H. Tamimi, "Using random forest (RF) as a transfer learning classifier for detecting error-related potential (ErrP) within the context of p300-speller," in *Proc. Bernstein Conf.*, 2015.

[31] H. M. Ahmed, M. J. Awan, N. S. Khan, A. Yasin, and H. M. F. Shehzad, "Sentiment analysis of online food reviews using big data analytics," *Elementary Educ. Online*, vol. 20, no. 2, pp. 827–836, 2021.

[32] Y. D. Setiyaningrum, A. F. Herdajanti, C. Supriyanto, and Muljono, "Classification of Twitter contents using chi-square and K-nearest neighbour algorithm," in *Proc. Int. Seminar Appl. Technol. Inf. Commun. (iSemantic)*, Sep. 2019, pp. 1–4.
[33] A. Bayhaqy, S. Sfenrianto, K. Nainggolan, and E. R. Kaburuan, "Sentiment analysis about E-commerce from tweets using decision tree, K-nearest neighbor, and Naïve Bayes," in *Proc. Int. Conf. Orange Technol. (ICOT)*, Oct. 2018, pp. 1–6.
[34] C. E. Shannon, "Prediction and entropy of printed English," *Bell Syst. Tech. J.*, vol. 30, no. 1, pp. 50–64, Jan. 1951.

ROMILA AZIZ received the M.S. degree in computer science from COMSATS University Islamabad (Lahore Campus), where she is currently pursuing the Ph.D. degree in computer science. From 2019 to 2022, she worked as a Visiting Lecturer at COMSATS University Islamabad, Lahore Campus. She has also worked as a Software Engineer at Visio Spark, one of the leading software houses in Pakistan. Her research interests include natural language processing, computational linguistics, and data mining.

MUHAMMAD WAQAS ANWAR received the M.S. degree in computer science from Hamdard University, Pakistan, in 2001, and the Ph.D. degree in computer application technology from the Harbin Institute of Technology, China, in 2008. From 2008 to 2012, he was a Lecturer and an Assistant Professor with the Department of Computer Science, COMSATS Institute of Information Technology, Lahore, Pakistan (currently COMSATS University Islamabad, Lahore campus). From 2012 to 2023, he was an Associate Professor with the Department of Computer Science, COMSATS University Islamabad (Lahore campus). Since May 2023, he is working as a Professor and is the Chairperson of the Department of Computer Science, Government College University, Lahore, Pakistan. His research interests include natural language processing, computational intelligence, and bioinformatics.



MUHAMMAD HASAN JAMAL received the B.S. degree in computer and information engineering from the International Islamic University Malaysia, in 2005, the M.S. degree in computer engineering from the University of Engineering and Technology, Lahore, Pakistan, in 2008, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2015. From 2006 to 2010, he was a Research Associate and a Senior Research Associate with the Al-Khawarizmi Institute of Computer Science, UET. He was a Graduate Technical Summer Intern at Sandia National Laboratory, Albuquerque, NM, USA, in Summer 2015. Since 2016, he has been an Assistant Professor with the Department of Computer Science, COMSATS University Islamabad, Lahore Campus. His research interests include parallel and distributed systems, scientific computing, data analytics, and natural language processing. He was a recipient of the Fulbright Scholarship for his Ph.D. studies.

USAMA IJAZ BAJWA received the M.S. and Ph.D. degrees from the Center for Advanced Studies in Engineering, Islamabad, Pakistan, in 2006 and 2013, respectively. He is an aggressively enterprising person who is currently an Associate Professor, a Research Supervisor, a Failed Entrepreneur, the Program Chair of the international conference on FIT, a CO PI of a funded project, the Head of the Machine Perception and Visual Intelligence Research Group, and a Polyglot. Previously, he was an Associate Head of the Computer Science Department, CUI Lahore; an Assistant Professor and a Graduate Program Coordinator at CUI Abbottabad; the Head of the Information Security and Image Processing Research Group, CUI Abbottabad; a Team Lead at ICT research and development funded project; a Co-Founder at Technology Nucleus Pvt. Ltd. and Agriasan Pvt. Ltd.; and a Visiting Researcher at the Medical Imaging Laboratory, University of South Wales. He has authored/coauthored around 60 international conference and journal research papers. His research interests include medical image analysis, video analytics, natural language processing, and pattern classification.

ÁNGEL KUC CASTILLA is working as a Professor at Universidad Europea del Atlántico, Spain. His research interests include process modeling, machine learning, and deep learning.

CARLOS UC RIOS is working as a Professor at Universidad Europea del Atlántico, Spain. His research interests include wireless communication, software-defined networks, and artificial intelligence.

ERNESTO BAUTISTA THOMPSON is working as a Professor at Universidad Europea del Atlántico, Spain. His research interests include data science, business intelligence, and time series analysis.

IMRAN ASHRAF received the M.S. degree in computer science from the Blekinge Institute of Technology, Karlskrona, Sweden, in 2010, and the Ph.D. degree in information and communication engineering from Yeungnam University, Gyeongsan, South Korea, in 2019. He was a Postdoctoral Fellow at Yeungnam University. Currently, he is working as an Assistant Professor at the Information and Communication Engineering Department, Yeungnam University. His research interests include indoor positioning and localization, indoor location-based services in wireless communication, smart sensors (LIDAR) for smart cars, and data mining.