scientific reports

OPEN

Correcting spelling mistakes in Persian texts with rules and deep learning methods

Sa. Kasmaiee✉, Si. Kasmaiee & M. Homayounpour

This study aims to develop a system for automatically correcting spelling errors in Persian texts using two approaches: one that relies on rules and a common spelling mistake list and another that uses a deep neural network. The list of 700 common misspellings was compiled, and a database of 55,000 common Persian words was used to identify spelling errors in the rule-based approach. 112 rules were implemented for spelling correction, each providing suggested words for misspelled words. 2500 sentences were used for evaluation, with the word with the shortest Levenshtein distance selected for evaluation. In the deep learning approach, a deep encoder-decoder network that utilized long short-term memory (LSTM) with a word embedding layer was used as the base network, with FastText chosen as the word embedding layer. The base network was enhanced by adding convolutional and capsule layers. A database of 1.2 million sentences was created, with 800,000 for training, 200,000 for testing, and 200,000 for evaluation. The results showed that the network's performance with capsule and convolutional layers was similar to that of the base network. The network performed well in evaluation, achieving accuracy, precision, recall, F-measure, and bilingual evaluation understudy (Bleu) scores of 87%, 70%, 89%, 78%, and 84%, respectively.

Natural Language Processing (NLP) is a field that bridges the gap between human language and computers. This technology enables applications to comprehend, procedure, and interpret human language. The significance of NLP as a tool for comprehending information generated by humans stems from the fact that data depends on the context[1]. NLP encompasses a wide range of topics related to the computational processing and understanding of human languages.

Since the 1980s, data-driven methods such as statistics, probability, and machine learning have become increasingly popular in this field. Recent advances in computing power and the parallelization of graphics processing units have enabled the use of deep learning techniques. Deep neural networks with millions of trainable parameters are used in this learning process. The availability of large datasets collected through complex processes makes it possible to train such deep architectures[1,2].

The growth of text data in recent years highlights the importance of text processors. Research in this field is ongoing because text-based applications have not performed satisfactorily despite their long history and current developments. Text processing is more challenging than processing data such as images, speech, videos, etc., due to complexities such as metaphors, homographs, etc.[3]. Text-based applications are numerous and include information retrieval[4–7], document classification and sentiment analysis[8–12], keyword extraction[13], word embedding[14], handwriting recognition[15], etc. In the fields of education and medicine, deep learning is of great interest to researchers due to its capabilities and the creation of suitable platforms[16–18].

Spelling is a component of word processing that checks the correctness of words and makes necessary corrections if there are errors. This task is one of the simplest applications that involves text processing. Many research studies have been conducted on the automatic correction of misspellings in several languages, especially English, and there are powerful tools available in this field[19–21]. However, there is a lack of research and powerful tools in this field for the Persian language.

In this study, an attempt has been made to identify and correct the misspelled word. This study was the first to conduct a comprehensive investigation of spelling correction in Persian texts using deep learning and a rule-based. A training database was created using the rules approach. According to other studies conducted in Persian language processing, 112 rules were devised for spelling correction and applied to Persian texts in this study. It is discussed in detail in the Rules for Text Correction section. A database containing 1.2 million sentences was obtained. The capsule layer, FastText embedding, and CNN in LSTM encoder decoder were employed for the

Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran. ✉email: saman.kasmaiee@aut.ac.ir

first time in Persian text error correction and their impact was examined. A graphical interface-based system was developed with Django[22]. The results indicated that the designed system had high accuracy in identifying and correcting errors in the Persian language. The background of the research and related studies will be discussed first, followed by the method for detecting and correcting spelling errors using two approaches based on rules and deep networks. Finally, an evaluation of these two approaches and their comparison will be presented.

Related work

Spell correction is a field of research within NLP that has been extensively studied. Numerous spelling correction techniques have been developed for various languages spoken worldwide. This section examines studies related to the topic of this research. As shown in Fig. 1, spelling correction techniques based on previous literature include probabilistic-based, rule-based, and deep-learning-based approaches. These three approaches are discussed in the following sections.

Probabilistic and rule-based methods

This category includes methods that do not need any specific language knowledge. The most well-known techniques in this group are n-gram-based, frequency-based, and finite state automata-based. These techniques rely on the counts and features of the words. N-gram-based techniques operate based on unigram, bi-gram, tri-gram, etc. This method splits each text string into a set of adjacent n-grams[23,24]. In this technique, the error probability of each n-gram should be calculated individually. This method can detect the position of the misspelled words with or without a dictionary. The word-frequency method identifies and corrects errors by using the probability of common mistakes repetition and the distance-editing method[25,26]. Singh et al.[26] proposed a spelling checker based on frequency and a grammar checker based on rules. They used a dictionary and bigram to find spelling errors and chose the best word for spelling correction. In the grammar part, they applied rules that can determine the grammaticality of the sentences based on the verb tense and whether it is negative or positive. In the finite state machine, the computational model of mathematics is used to design natural language. A word accepted by the finite state machine model is considered correct in the language[27,28]. The limitation of these methods is that some spelling errors require specific language knowledge, and these methods cannot detect and correct such errors because they employ features such as frequency, word count, and others.

Rules use heuristics derived from morphology, stemming, and other features of words to correct the spelling. These rules usually correct errors with inserted and deleted letters. Despite the recent advances in applying deep neural networks such as sequence-to-sequence language modeling, rule-based methods are still used for morphologically rich languages like Kurdish due to accuracy and feasibility[29]. Such rules generally work based on probabilities[30]. An improved Morphological Analyzer (MA) has been developed to enhance the Cushitic grammar checker tool[31] and Persian Optical Character Recognition[32]. In the Persian language, some spelling structures can be expressed as rules and studies have been done on this topic[33,34]. The main disadvantage of these methods is the need to explore different heuristic rules for each language. Therefore, rules related to a specific language are not used for other languages. The performance of the error corrector can be enhanced by combining several methods. Aziz et al.[35] could find wrong spellings of an Urdu word using a broad lexicon lookup method and provide a list of candidate words with correct spellings. They used a mixed model that rates the words in

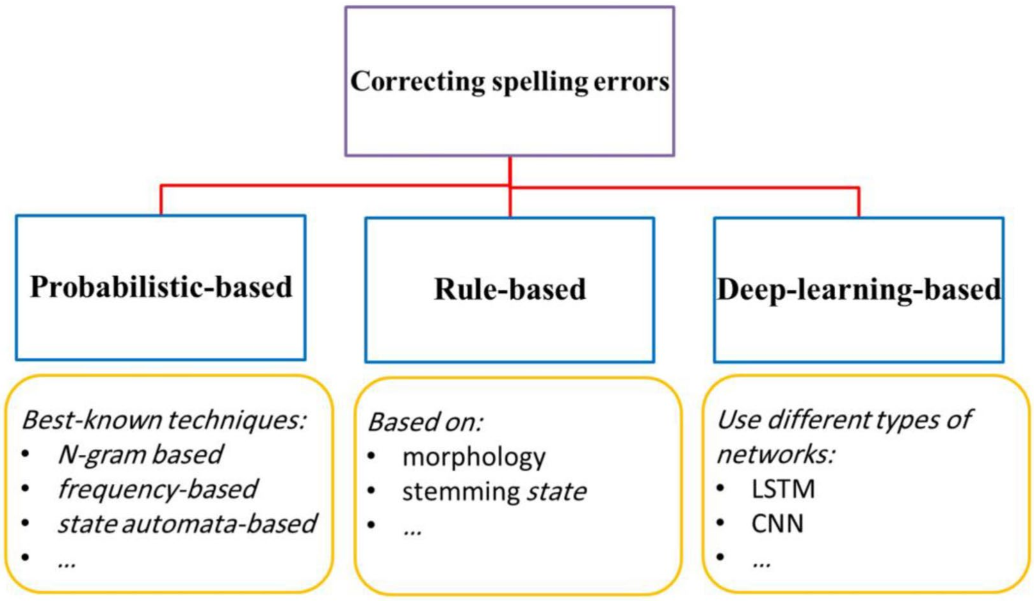


Figure 1. Types of spelling correction techniques in literature.

the nominee word list. They used several ranking techniques such as Soundex[36], Shapex[36], longest common subsequence (LCS)[37], and N-gram, as well as their combinations, to find the finest technique in terms of F1 score.

Deep-learning-based method

In many cases, we need to know and understand the word context in relation to sentence to recognize the error. In these cases, the deep network is helpful. Kaur et al.[38] used the tree data structure to store the dictionary. They used a recurrent neural network that incorporates Long-short term memory (LSTM) unit to reduce the vector distance between the error word and the word in the dictionary to correct errors. They also applied rules that were manually crafted, rules that followed the syntax of the language, and rules that dealt with tokenization to enhance the detection and correction of errors. The error detection method was the absence of words in the database after pre-processing, and the error correction was done by n-gram and rules. Finally, if it was not found in the database, the network suggested the closest sentence. A deep learning-based spell checker for Malayalam was proposed by Sooraj et al.[39]. They trained a neural network that uses LSTM to detect misspelled words and the location of the errors in the error detection phase. They used the F measure to evaluate the accuracy of error detection and they selected the most probable word from the candidate words to correct the errors. Caryappa et al.[40] developed a deep learning model that incorporates LSTM units that could detect grammatical errors in the Dravidian language. They implemented neural network-based word embedding in their model.

Hu et al.[41] used the combination of encoder representations from transformers (BERT) and the algorithm of Levenshtein editing distance to rank and choose the word that corrects the spelling. They showed that if these two are appropriately combined, spelling errors will be edited effectively, although they only examined the accuracy criterion in their study. Beloki et al.[42] employed neural models of sequence-to-sequence transformation to correct grammatical errors in the Basque language. They did not have any training data for this task, so with a rule-based method, they created grammatically incorrect sentences from a set of terms extracted from 500,000 news stories, used them for training and obtained a network with an F criterion of 0.87. Zolzaya et al.[43] conducted the first text normalization study for the Mongolian language. Their goal was to convert the text with errors in social networks into an official one. The original texts were Roman and had to be converted to Cyrillic. They used a two-stage model; the first stage included a neural network to convert sequence to sequence at the character level and its output as input in the second stage—the second stage combined two methods based on LSTM and the editing algorithm. Mandal and Nanmaran[44] presented a new architecture focusing on type normalization. One of the main features of their architecture was that, in addition to normalization, it could be applied to restore text and generate keyword.

Mager et al.[45] normalized social texts with colloquial errors. Using a sequence-to-sequence neural network, they obtained a 5% improvement in results compared to the previous model. Singh and Singh[46] corrected spelling errors in Hindi using deep learning. In this process, the misspelled words were automatically identified by the network and replaced by the closest word. They used a database containing 9 gigabytes of data to train and test their network. They implemented two scenarios to test their neural network. They reported an accuracy of 83%, recall of 72%, and F-measure of 76% as the network results for the test data. Some studies[47,48] have examined the combined effect of rules and neural network-based methods, which shows an improvement in the accuracy of the hybrid model. Sampath and Shanmugavel[47] suggested a hybrid approach that combines the algorithm of Levenshtein's edit distance, the algorithm based on rules, the algorithm of Soundex, and the model of LSTM for the spell checker of the Tamil language. Their proposed machine had an accuracy of 95.67% and a Tamil scholar verified it.

Ethical and informed consent for data used

This paper was done by the authors, and no human participants other than the authors were involved in it, and informed consent was obtained from all authors.

Background

Artificial neural networks

Artificial neural networks (ANNs) are computational models inspired by the human brain and other organisms[49]. The learning procedure of neural networks can be accomplished in three methods: supervised, unsupervised, and reinforced, which depend on the type of network[49]. Neural networks are applied in the automatic correction of texts in two main parts: word representation and vocabulary correction. Different types of networks can be applied in each part to improve performance. Most basic frameworks in natural language processing programs depend on sequence-to-sequence models in which both input and output are sequences. These models are used for different tasks that involve natural language processing, such as translating languages, text summarization, speech-to-text conversion, and text-to-speech conversion. The most common sequence-to-sequence framework has an encoder and a decoder. An example of this framework is shown in Fig. 2. The input data sequence is processed by the encoder, which generates an intermediate output. This output is then used by the decoder to create a series of final outputs[50]. With the development of neural networks, their usage to improve the sequence-to-sequence process was considered[49,50]. The rest of this section mentions the neural networks used in this research. The advantages and reasons for using them are discussed in each part.

Encoder

An encoder takes a sequence of input data and produces an output at an intermediate level. Different types of networks, such as CNN, capsule, RNN, LSTM, and Bidirectional are used for different purposes. The kinds of neural networks that can be useful in this research are described below[1,3,51–67].

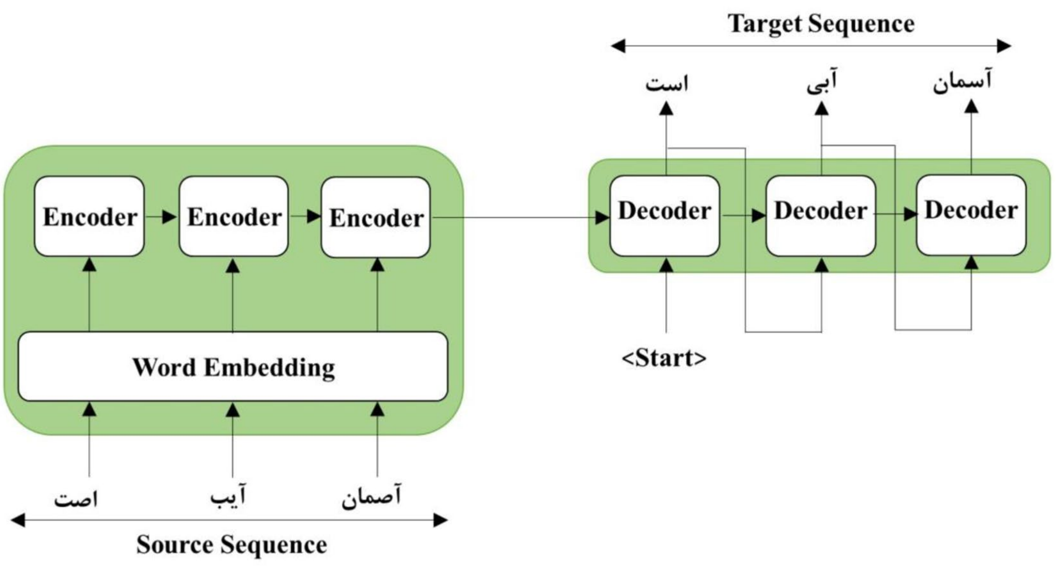


Figure 2. Seq2Seq framework.

Neural network for vocabulary representation
Word embedding is a representation of words as numerical vectors. In fact, by embedding words, vocabulary is mapped to a vector space that different language models can learn. To use the neural network to represent the words, the word-to-vector mapping algorithm is coupled with the learning algorithm. These networks are of the shallow type and are trained for this purpose and typically have between 50 and 300 dimensions[3]. Two common algorithms CBOW and Skip-Gram are used for word embedding. The CBOW model predicts the target word from the surrounding words, while the Skip-Gram model predicts the surrounding words from the target word[51,52]. The general principle of these two models is presented in Fig. 3. The purpose of mapping words to vector space is to group similar words in this space. The following relation determines the criterion of similarity[52]:

$$\cos\left(vec_i, vec_j\right) = \frac{vec_i vec_j}{\|vec_i\|\|vec_j\|} \tag{1}$$

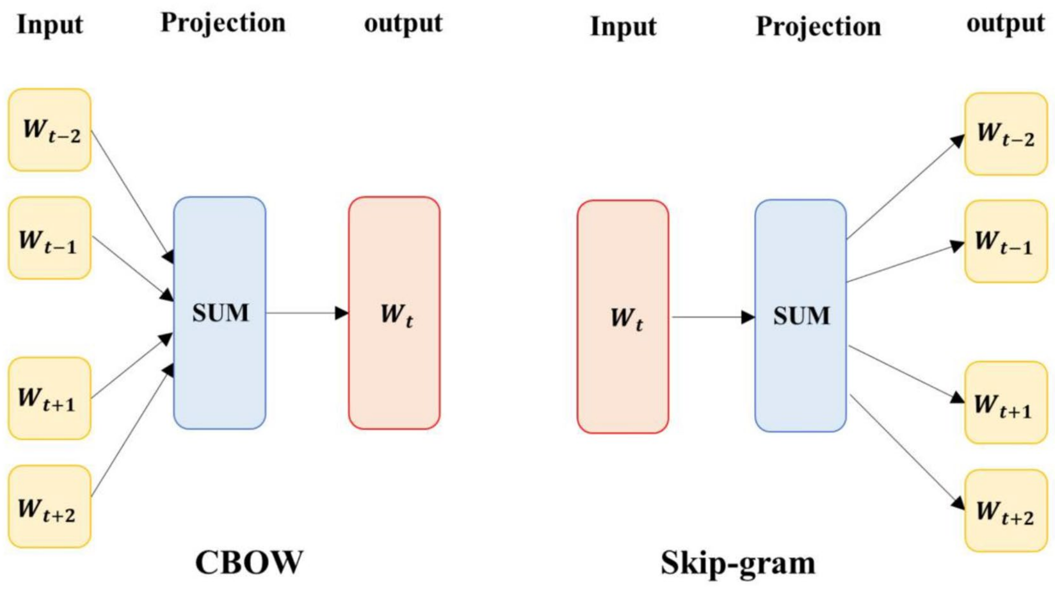


Figure 3. CBOW and Skip-Gram models.

Neural network for applying various filters, removing noise, and extracting main features
Because of their structure, Convolutional networks can apply different kinds of filters on the input. These filters help them to identify the main patterns in the input data. A convolutional neural network includes an input layer, an output layer, and several hidden layers. The hidden layers can be of different types, such as convolutional, pooling, or fully connected. The convolutional layers perform convolutions on the inputs. CNNs use pooling layers that reduce the number of neurons in one layer by combining the outputs of several neurons into one neuron in the next layer. For instance, the max pooling method selects the highest value from a group of neurons in the previous layer and passes it to the next layer. The pooling layer's task is to reduce the matrix's dimensions to extract the main features and reduce the computational cost. Fully connected layers are like a multilayer perceptron responsible for mapping[53–55]. Different parts of this network are shown in Fig. 4. Assuming that the sentence length and size of the word window are n and h, respectively. The $c \in \mathbb{R}^{n-h+1}$ feature vector is obtained by applying the convolution and pooling layers on the input sequence from the following relations[53]:

$$c_i = f\left(wx_{i:i+h} + b\right) \tag{2}$$

$$\widehat{c} = \max(c) \tag{3}$$

where $w \in \mathbb{R}^{hk}$ is the filter and b is a term that adds a constant value to the output. $x_{i:j}$ represents the combination of words i to j. f is also an activation function, which is usually considered as *tanh*.

Neural network for maintaining hierarchical relationships
A capsule neural network is a type of ANN that can model hierarchical relations better. This network adds structures called capsules to the convolutional neural networks to keep the information about the hierarchy. In multilayer perceptron networks, each neuron's output is a scalar. In convolutional networks, each number obtained is a convolution between a kernel and a part of the input data. The convolutional layer produces the output by stacking these matrices on top of each other, which are the results of applying filters to the input image. Then, the pooling is applied to these matrices. In this network, a lot of useful input data is lost. To address this problem, a capsule network is used. In capsule networks, all the crucial features are conserved and stored as a vector, which has size, direction, and position properties. They can be used to learn hierarchical relations[56].

Neural network for encoding wrong words
A recurrent neural network (RNN) is a type of ANN that can process sequential data by connecting a series of feed-forward networks and passing the output of each network as input to the next one. A recurrent network has three types of layers like feed-forward neural networks. In sequential models, the network receives one

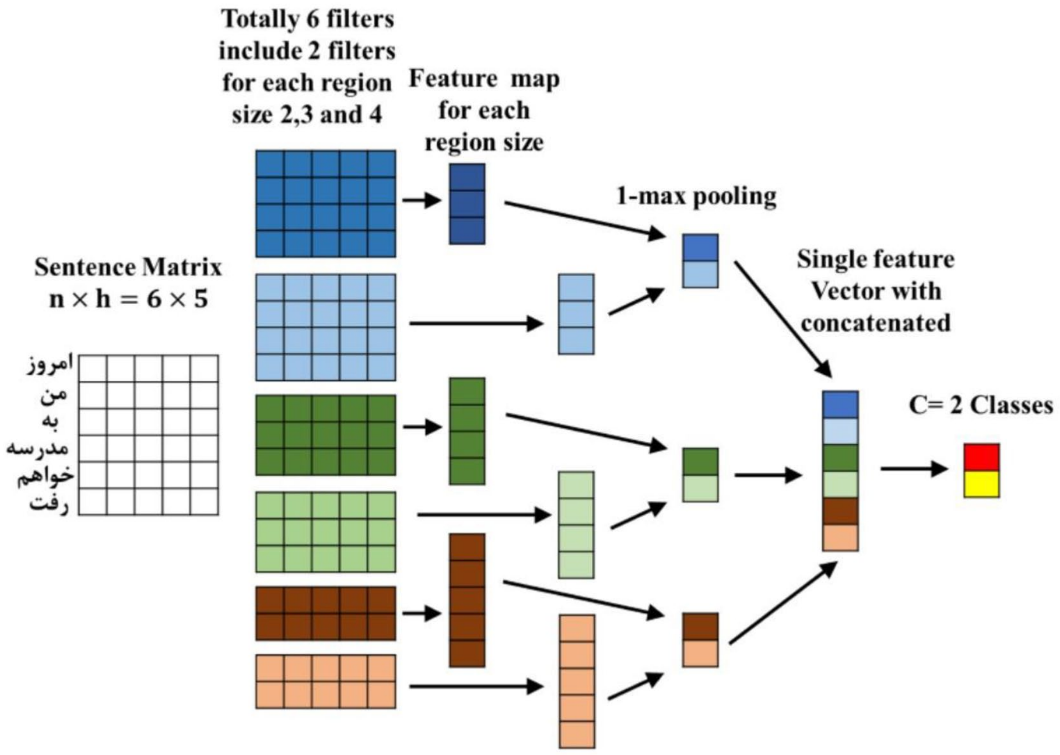

Totally 6 filters include 2 filters for each region size 2,3 and 4
Feature map for each region size
1-max pooling
Sentence Matrix n × h = 6 × 5
امروز
من
به
مدرسه
خواهم
رفت
Single feature Vector with concatenated
C= 2 Classes

Figure 4. CNN architecture.

vector at each time step from the input vectors. The network processes the input batch and updates its weights accordingly, and then takes the next input batch to continue the operation. Therefore, at each time step, the network uses not only the input vector at that time but also the variables of the previous hidden layer as inputs to the current stage. RNNs have hidden layers that can act as memory by storing information from previous inputs. This feature makes them very suitable for applications that deal with sequential inputs, such as language modeling. In NLP and machine translation, sequences are taken from inputs, and sequences are produced from outputs[1]. Figure 5 shows an illustration of it. Let x_t, v_t,and h_t represent the input vector, the current word, and its memory, respectively. Then we have[57]:

$$x_t = v_t + h_{t-1} \tag{4}$$

Assuming that w_t, u_t, and b_t represent the input vector's weight matrix, the recurrent weight matrix, and the bias, respectively[57]:

$$h_t = \tanh(w_t x_t + u_{t-1} h_{t-1} + b_t) \tag{5}$$

The output of the network is obtained from the following relation[58]:

$$y_t = \mathrm{Softmax}(u_t h_t) \tag{6}$$

where the Softmax function maps the output vector to the possible output word. The recurrent network problem is that their ability to remember and use the information they learned from the far past diminishes as the sequence gets longer. In other words, they cannot use information from the far past[57,58].

A special kind of recurrent neural network that can capture dependencies over long time spans is a long short-term memory-based network (LSTM). The problem of long-term dependence was the reason for designing these networks. Remembering information for long periods is the default and normal behavior of these networks, and their internal structure is designed in such a way that they can learn information from far away, so this feature is hidden in their structure. LSTMs also have a sequence or chain-like structure, but the repeating module has a different structure. They have four neural network layers similar to RNN which these layers can be seen in Fig. 5. These layers interact and communicate with each other according to the special structure of these layers. This network introduces new concepts that were not present in the conventional recurrent neural network. Figure 6 illustrates the internal structure of this network. In these networks, there are so-called three gates (forget, update or input, and output gate) that control the data flow within the network[59].

In addition, there is a memory cell, the network also has an input from the cache or h_{t-1} and an input or x and produces two outputs (one output is c_t and the other output is h_t which itself splits into two parts, one part goes to the following stage and another is used to generate output in the present stage if required). The forgetting gate regulates the information flow from the previous stage. This gate decides whether to use or not use the memory information from the former stage and, if something should be entered from the previous time step, how much should it be. The update gateway controls the flow of new information. This gate chooses to accept or reject new information in the present stage and if so, how much. This gate is generally called the input gate. The output gate also decides the amount of information from the former and present stage that will be passed to the following stage[60]. Therefore, LSTM works as follows[59]:

$$c_t = f_t c_{t-1} + i_t \widehat{c_t} \tag{7}$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o) \tag{8}$$

$$h_t = o_t \tanh(c_t) \tag{9}$$

$$y_t = Softmax(o_t) \tag{10}$$

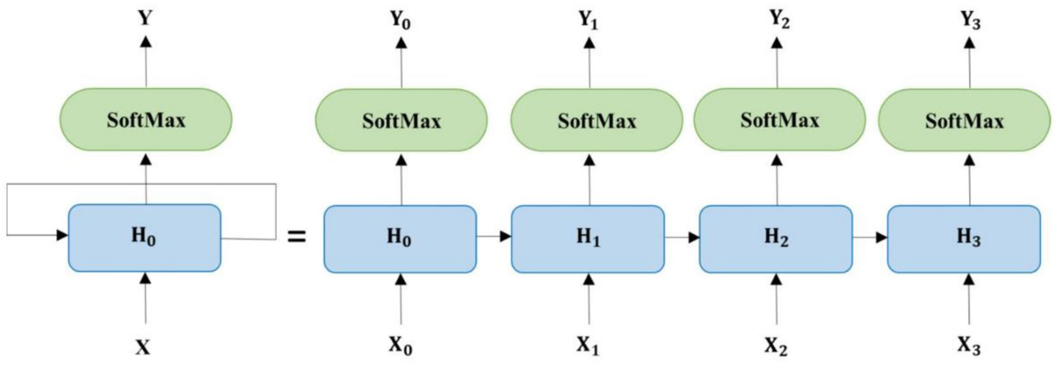


Figure 5. RNN architecture.

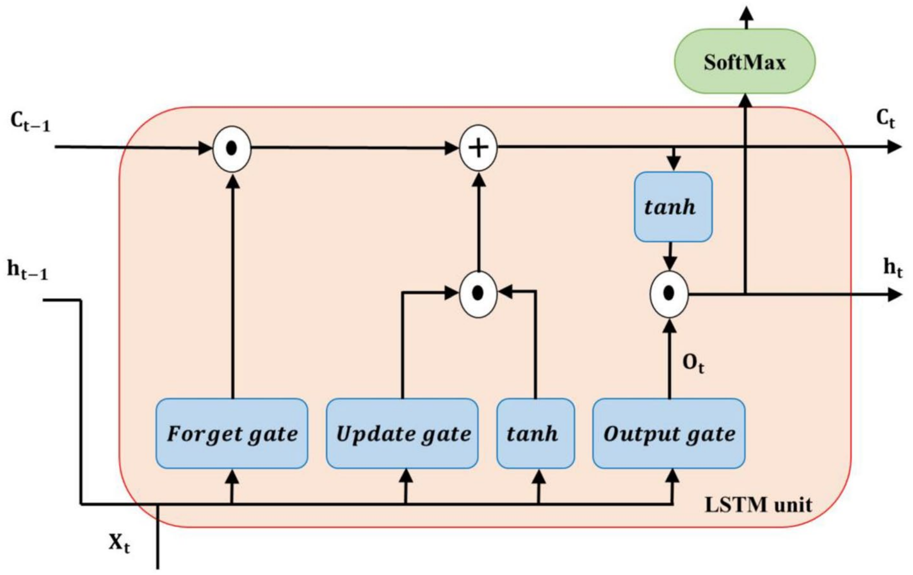


Figure 6. Architecture of LSTM unit.

Where σ represents the sigmoid function and o_t,w_o, b_o, and y_t are output, weight matrix, output gate's bias at time t, and the possible output word, respectively. f_t and i_t are activation vectors of forget and input gates, and $\widehat{c}_t$ is the vector that feeds into the cell. These parameters are obtained from the following relations[60]:

$$f_t = \sigma\left(w_f[h_{t-1}, x_t] + b_f\right) \tag{11}$$

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \tag{12}$$

$$\widehat{c}_t = \tanh(w_c[h_{t-1}, x_t] + b_c) \tag{13}$$

In more advanced and modified LSTM, the gate outputs (f_t, i_t, o_t) are also the current cell state function (h_t)[61].

Bidirectional LSTMs are a more advanced model of this type of network, which, besides considering more distant dependencies, access information in both directions, enhancing their performance[62]. The bidirectional LSTM is an evolved type of LSTM with two separate recursive hidden layers. Each hidden layer has blocks of LSTM memory. There is no connection between these two hidden layers, and the two hidden layers are connected to one output layer. In this research, these types of networks have been used[63,64]. Figure 7 shows how connections

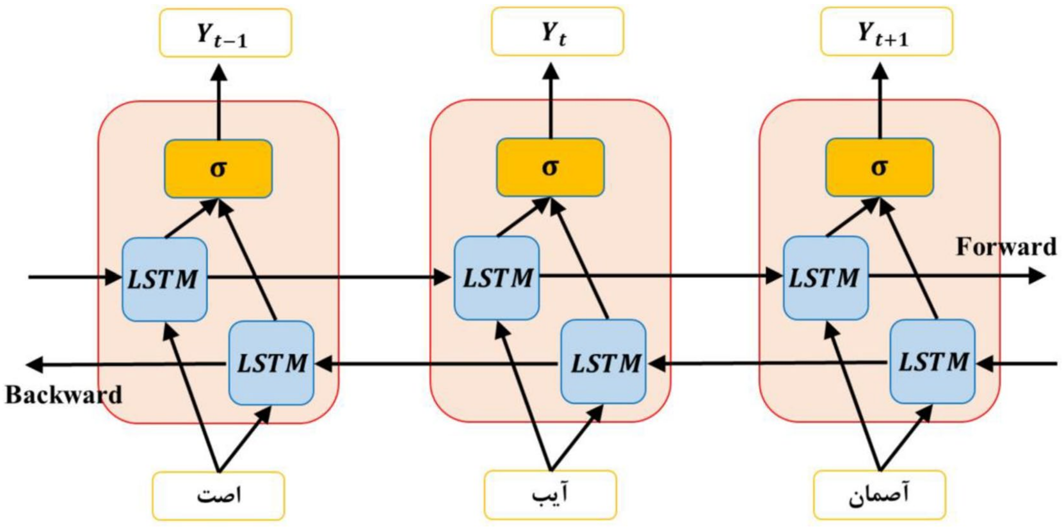


Figure 7. Bidirectional LSTM network architecture.

are made in this type of network. Therefore, in these networks, the output is obtained from combining the forward and backward results, so[63]:

$$h_t = \left[h_t^f, h_t^b\right] \tag{14}$$

where[63]:

$$h_t^f = LSTM\left(x_t, h_{t-1}^f\right) \tag{15}$$

$$h_t^b = LSTM\left(x_t, h_{t-1}^b\right) \tag{16}$$

Decoder

The decoder uses the intermediate data sequence generated in the encoder and produces a set of final outputs. The kinds of neural networks that can be useful for different tasks in decoding are described below[62,65,66].

Neural network for decoding words without mistakes

In this part, a recurrent neural network, LSTM or bidirectional LSTM can be utilized. The advantages and disadvantages of them were presented in the previous section. This research employed a neural network that utilized Bidirectional LSTM. The number of memory units is very important in the learning power and recall of far information. So, its enhancement improves performance and increases the computing cost[62].

Neural network for converting decoder output vectors into correctly spelled words

In the final part of the decoder, multilayer perceptron networks (MLPs) are usually implemented. An MLP consists of a minimum of three layers (input, output, and hidden layers). Each layer contains a group of neurons responsible for converting and transferring information from the preceding layer to the following layer. In the structure of this type of network, the neurons of a layer do not interact with other neurons in the same layer. This network uses nonlinear activation functions. All the neurons in a layer are linked to every neuron in the following layer, forming a fully connected network. They are the most basic kind of feed-forward neural network[65].

Deep learning

A sub-branch of machine learning, deep learning (DL) employs algorithms that aim to represent high-level abstract notions in the data. A deep graph with various processing layers that do linear and nonlinear transformations is used by this process. In other words, it learns knowledge and features in several layers. Deep learning is the term for researching new techniques for ANNs. It means using deep neural networks on huge data sets to learn a procedure to deal with a task. This method can vary from simple classification to complex reasoning. Multiple hidden layers are present in a neural network that is called a deep neural network (DNN). In a DNN, each layer works on a specific data feature. The deeper we go into the network, the more complex features the network will be able to recognize. This process is called feature hierarchy. This neural network feature makes them powerful and processes data with multiple features[66].

Evaluation measures

The performance of a text system is measured using different parameters such as accuracy, recall, precision, and F-criterion. Another parameter called Bleu number is also applied in machine translation. In this study, this parameter was also utilized to check the performance of spelling correction due to the similarity and closeness of spelling correction to machine translation. These evaluation parameters show how well the performance of the developed model is appropriate in textual data analysis. There are four different modes for the model to respond to the actual model. Since the model aims to identify and correct misspelled words, we consider misspelled word identification as positive like Singh and Singh study[46]. In our application, these states are: the real phrase has a misspelling and the network or rules have edited it correctly (TP), the real phrase has a misspelling but the network or the rules have not edited it correctly or it has not recognized incorrectly (FP), the real phrase is without spelling mistakes and the network or rules correctly recognized it without mistakes (TN), and the real phrase is without spelling mistakes but the network recognized it as having a spelling mistake (FN). In the following, various criteria for evaluating the performance of text systems and their calculation methods are described based on these states[46,50,68].

The accuracy criterion expresses the ratio of inputs that the process correctly recognized out of all the predictions that the same process made. In other words, it means[46]:

$$\text{Accuracy}\% = \frac{TP + TN}{TP + TN + FP + FN}\% \tag{17}$$

The recall criterion expresses the proportion of detections that were picked by the model among the total correct detection. Therefore, it is calculated from the following relationship[50]:

$$\text{Recall}\% = \frac{TP}{TP + FN}\% \tag{18}$$

The precision criterion evaluates the proportion of words that had the right classification among the total positive detections forecasted. Therefore, this criterion is calculated through the following relationship[50]:

$$\text{Precision}\% = \frac{TP}{TP + FP}\% \tag{19}$$

The F-measure criterion merges precision and recall factors and evaluates how good a model is in both precision and recall metrics. The harmonic measure of precision and recall elements is another name for this criterion. Calculation of this parameter in terms of two other parameters is as follows[46]:

$$\text{F-Measure}\% = 2\frac{Precision.\text{Recall}}{Precision + Recall}\% \tag{20}$$

Bilingual evaluation understudy criteria (Bleu Criteria) is a method for evaluating the quality of the machine-translated text. In this criterion, in order to compare the machine translation with the original translation, the modified form of precisions is used. For this purpose, the overlap of n-grams of machine output text and reference text is identified and the precisions value of n-grams with different gram sizes from 1 to 4 is calculated. Using these precisions, the Bleu value is calculated from the equation[68]:

$$BLEU = \min\left(1, \frac{output - length}{reference - length}\right) \times \left(\prod_{i=1}^{4} precision_i\right)^{0.25} \tag{21}$$

A number ranging from 0 to 1 is the output of Bleu, which reveals the similarity between the candidate text and the reference texts. The closer the number is to one, the greater the similarity.

Materials and method

Figure 8 illustrates the flowchart of error correction with two methods: rule-based and deep learning. The system receives sentences from the input and performs preprocessing and tokenization on the text. Then, it detects and corrects the error and displays the result in the output. This research proposes a system for correcting spelling errors that consists of three main components: input data preprocessing, error detection, and error correction. Each language has its own preprocessing requirements. The hazm library[69] was used for data preprocessing. In the rule-based method, the system searches for words with misspellings in the database. If a word is not found in the database, it is misspelled and the system corrects it based on the rules. Before that, the system fixes common spelling errors using a list. The system trains a neural network using an artificial database in the deep learning method. The process of creating a synthetic database is explained in "Artificial database construction for learning and deep networks structures" section. The system has a user-friendly UI based on Django, which allows users to enter text and view the output text generated by the system.

Preprocessing and database of Persian words

As mentioned, the hazm library was used for preprocessing. The hazm is a standard library for preprocessing in the Persian language. The three main digestion modules that were used in this research are Stemmer (extract the root of the noun), Lemmatizer (extract the root of the verb), and Normalization (transform a text into a standard form). For Stemmer, a list is used that specifies the prefix and suffix that can be attached to the nouns, and if one of them appears in the words, it is removed. The Lemmatizer is more complex than Stemmer, and besides the unit of tokenization and root finder, several auxiliary verbs such as is, was and other verbs were used. The current suffixes and prefixes are stored in the form of a list and separate auxiliary verbs from the main verb. In the normalization unit, the extra space is removed by suffixes and prefixes. In fact, in Persian, the prefix and suffix of the words include a half space, but a space is mistakenly placed instead, so this extra space should be removed and by identifying this prefix and suffix, this space can be removed.

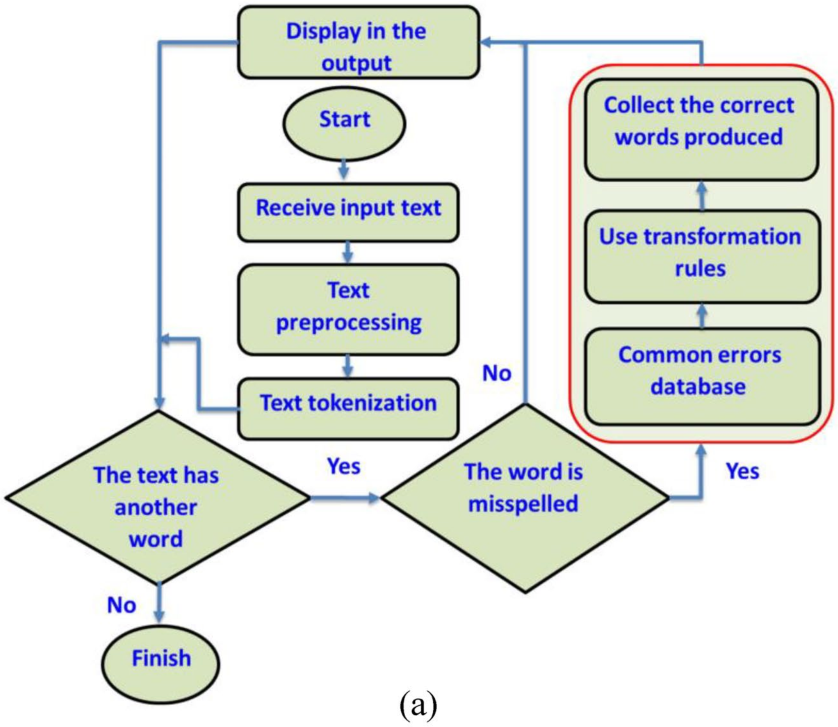
The database of Persian words used is the database of Zaya words of the Persian language[70], a standard library with common and usual words. The vocabulary in this database includes about 55 thousand entries. In each entry, information related to the word's written form in the Persian script, phonetic construction, lexical category, stress pattern and frequency of the word is stored. A text corpus containing 10 million words was used to prepare this database. This corpus included 100 thousand words with different frequencies, and by removing inflectional forms, the number of these words was reduced to about 44 thousand words. Due to the lack of some common words and completely scientific words in this list, Zaya words were compared with Persian culture and about 11 thousand new entries were added to it. And finally, the Zaya database with 55 thousand distinct words was obtained[67].

List of common misspellings

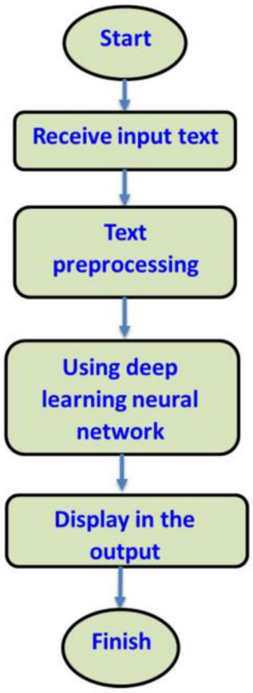
About 700 incorrect words were collected that were used as common misspelled words. This list is derived from several sources, the main one of which is the Wikipedia site, the error finder section. All the words detected as wrong should be checked with these words; if they exist in this list, they should be changed into their correct ones. Some of these words and their common mistake forms are displayed in Table 1.

Rules for text correction

In general, we have two types of errors in texts, one is phonic and the other is typographical. Phonic mistakes are limited and occur in the wrong spelling of homophonic letters in the Persian language. To implement it, we



(a)



(b)

Figure 8. Spelling correction flowchart of (**a**) rule-based, (**b**) deep learning.

tried to consider all vowel letters, which was implemented by replacing each vowel letter with a letter that has the same sound as it. It should be noted that only one vowel letter is considered in each rule.

Correct vocabulary	Misspellings word	Correct vocabulary	Misspellings word
تظاهر (Pretense)	تضاهر	آشکار (Obvious)	اشکار
توسعه (Development)	توصعه	آماج (Purpose)	عاماج
سپاسگزار (Grateful)	سپاسگذار	اطلاق (Applicability)	اتلاق
صفحه (Page)	صحفه	اختراع (Invention)	اخطراع
ضبط (Recording)	ظبط	ازدحام (Congestion)	ازدهام
فضیلت (Virtue)	فظیلت	(Acknowledgment) اذعان	ازعان
مضحک (Ridiculous)	مزحک	انضباط (Discipline)	انظباط
مزاحمت (Nuisance)	مزاهمت	بررسی (Check)	برسی
هراس (Fear)	حراس	بردگی (Slavery)	برده گی
نقض (Violation)	نغض	بی‌محابا (Fearlessly)	بی‌مهابا

Table 1. Examples of words in the common misspelling list.

There are many types of typos, which can be classified according to Table 2 based on the research of Shahmiri et al.[71]. We tried to take into account all these typographical errors, and implement them. In Table 2, the additional distance means inserting a space between the two main letters of the word and removing the distance means removing the space between two words and becoming one mistake word. In the following, the implementation of each of the errors is discussed.

Substitution of the letter
Instead of one of the main letters, another key was pressed on the keyboard. The pressed key is often close to the main key. Therefore, to create this error, the right and left keys were considered for each letter and it was assumed that the right or left key would be pressed instead of that key.

Letter transposition
One of the main letters should be replaced with the next or previous letter in the word itself. For this, each sentence in the database is broken into words and in each word, every time the letters of the word are read from the database, the first letter is replaced with the second, then the second with the third, and in this way. If there are sufficient letters for the word, the first five will be considered for this process. That is, 5 different misspellings were generated for each word.

Letter repetition
One of the main letters should be written twice. To create this error, each sentence was converted into words, then one of the letters of the word was repeated every time the text was read, for example, the first letter of each word was repeated in the first turn. In the next reading, the second letter of the word was repeated and so on until the fifth letter.

Removing the letter
One of the main letters of the word should not be typed, its implementation method is similar to repeating a letter, only here, instead of repeating a letter, that letter is deleted every time the text is read. That means the first letter of each word is deleted in the first round, and so on. It is continued until the fifth letter.

Additional distance
Insert a space between the two main letters of the word. It has been done in the same way as before, that every time reading the text, a space was considered between two consecutive letters of the word, for example, in the first reading, a space was inserted between letters one and two, and so on.

Row	The error type	Percentage error	Example
1	Inserting letters	8	The author writes "منرده" instead of "مرده" (dead)
2	Letter repetition	14.5	The author writes "ممرده" instead of "مرده" (dead)
3	Removing the letter	19	The author writes "مده" instead of "مرده" (dead)
4	Substitution of the letter	39.5	The author writes "مزده" instead of "مرده" (dead)
5	Letter transposition	5	The author writes "رمده" instead of "مرده" (dead)
6	Phonic mistakes	0.5	The author writes "مردح" instead of "مرده" (dead)
7	Additional distance	2.5	The author writes "م رده" instead of "مرده" (dead)
8	Removing the distance	5	The author writes "مردهاست" instead of "مرده است" (is dead)
9	Other things	6.5	The author writes "ممزده" instead of "مرده" (dead)

Table 2. Types of mistakes and their occurrence ratio in the Persian language[71].

Inserting letters
It happens when a letter is written between two main letters during the writing and this letter will be close to the main letters. In order to implement this error, it was done according to the repetition of the letter, only instead of repeating the main letter, based on what the letter was, the letter of the right or left button on the keyboard was pressed when typing.

In this research, we tried to cover all the errors, therefore, by implementing the rules related to these errors, 112 incorrect sentences of different types of errors were created for each correct sentence. In this way, the necessary corpus for neural network training was obtained. Part of the study is error correction with the help of rules, the same rules were used for spelling correction, for example, if a letter has been removed as an error in a word, we must add a letter in the correction section and other rules can also be utilized similar to this. Several examples of error sentences corrected by the rules and database of common errors are indicated in Table 3.

Artificial database construction for learning and deep networks structures

Using a corruptor is one way to create a training database for the correction of misspelled. Alian et al.[72] used a corruptor to error generation in words for the Arabic language. All the spelling mistakes that are considered in the previous section were tried to be made artificially by rules. A database of 10 thousand sentences without spelling mistakes, which was the sum of the official sentences of the standard database of the Persian to English translation[73] and the database of official sentences obtained from the subtitle collection of different movies, was considered. Then according to the conducted studies, 112 rules were applied to each sentence that each rule produce errors in a different way. By doing this, a database with about 1 million and 200 thousand sentences

Correct sentence (Target)	Input (Sentence with errors)	Output (Sentence with correction)
من تو را دوست دارم. (I love you.)	من طو را دوصت دارم.	من (تو) را (دوست) دارم
آدم با دل خود عاشق یار میشود. (A person falls in love with his heart.)	آددم با لد خوددد غاشق یار میشود.	(آدم) با (بد|دل|بلد) (خود) (قاشق|عاشق) یار میشود.
دوست مهربان خیلی خوب است. (Kind friend is very good.)	دوتس محربان خلیی خبو است.	(دوست) (مهربان|مجربان|مخربان|محبان) (خیلی) (خوب|بو|خو|خب) است.
انسان پست ترین موجود زمین میتواند باشد. (Man can be the lowest creature on earth.)	انصان پتس ترین مووجود رمین میتواند باشد.	(انسان|نصان|انان) (پنس|پاس|پست|پس) (ترین) (موجود) (رمی|زمین|رمیم|رمیت) میتواند باشد.

Table 3. Examples of sentence correction by rules.

with misspellings was created. One million sentences of this database were applied in the neural network section for training and testing.

The deep learning method was used to map the sequence to the sequence (transforming incorrect text into correct text). The word embedding method based on FastText[74] was also applied in the neural network. FastText is a library for learning word embedding and text classification, which was developed by Facebook's artificial intelligence laboratory and uses neural networks for word embedding. In this study, the trained Persian language model that was provided by the Facebook artificial intelligence laboratory team was used.

The artificial database that was created was used to train and test the final DNN model. The training and testing data consisted of 800 thousand and 200 thousand sentences, respectively. The parameters that were set in the ANN are shown in Table 4. The default embedding layer was turned off and FastText was activated as the embedding layer. The model was trained by the Adam algorithm. This method can deal with the gradient reduction problem in noisy data. This algorithm is very popular in deep learning because of its fast learning ability. The value that a model should aim to reduce during training is computed by the loss function. The sparse categorical cross entropy was chosen as the loss function, which is defined by the following equation[75]:

$$Loss = \frac{-1}{N}\sum_{s\in S}\sum_{c\in C} l_{s\in c}\log p(s \in c) \tag{23}$$

where S are samples and C are classes and each sample belongs to a class. $l_{s\in c}$ is the function that indicates whether sample s is in class c or not. $p(s \in c)$ is the probability that the model gives for sample s being in class c.

After several times of trials and errors, the batch size was selected to 10. To stop training, two conditions were considered: 100 training epochs or failure to improve in three consecutive epochs. In all cases, the second condition was established.

The architecture of the implemented network types in this research is presented in Fig. 9. The basic encoder-decoder architecture is exhibited in Fig. 9a, and Fig. 9b shows the types of layers added to improve performance. In Fig. 9a, first, all the sentences are converted into integers, then they are given to the encoder-decoder-based network to map this sequence (the text with errors) to the target sequence (corrected text). At the input of this network, a FastText-based word embedding layer was placed, which converted integers into vectors with dimensions of 300 dimensions. After using the word embedding layer in the encoder section, Bidirectional LSTM units have been applied. A repeating vector layer was employed to transform the encoder output into the decoder

Criterion	Value/function
Embedding	FastText
Optimizer	Adam
Loss	sparse_categorical_crossentropy
Metrics	acc
Epochs	100
Batch_Size	8
EarlyStopping	3
Monitor	val_loss

Table 4. Selected characteristics for network training.

input. In fact, with this vector, it is possible to separate different sequences for the decoder. Then the decoding section is started where there is a layer with bidirectional LSTM. After that, a time distribution layer was placed to extract the sequences and map vectors to words. In Fig. 9b, two layers of convolutional and max pooling are utilized to apply various filters, remove noise, and letter repetition errors, and extract the main feature. The capsule layer was employed to maintain relationships between words. After the word embedding layer, these layers were appended to the encoder section, and then a layer of LSTM units followed them.

User interface

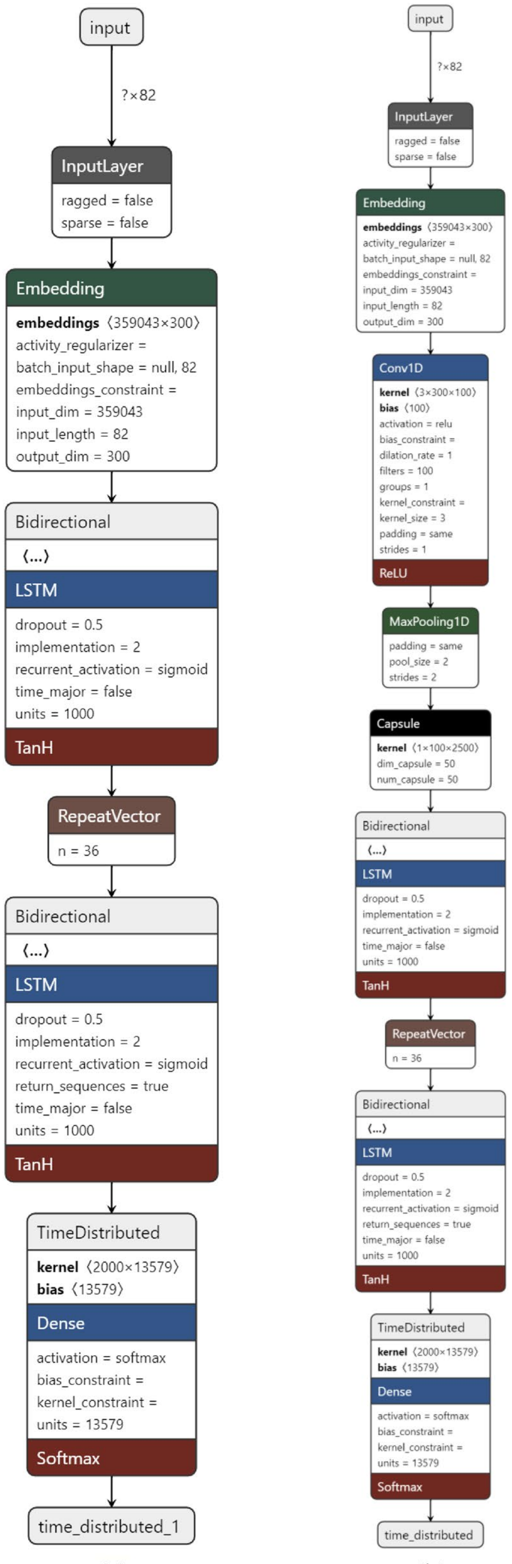
In practice, the system requires the input of sentences with misspellings by the user and the output of corrected sentences. A web-based user interface was implemented by the Django framework of python language. Django[22] is a free and open-source web framework that uses Python and follows the model-view-controller pattern. Django aims to make it easy to build complex and database-driven websites, and it is designed based on the reusability and connect ability of different components, rapid development, and doesn't repeat yourself. The "doesn't repeat yourself" principle in software development focuses on minimizing repetition of information. It achieves this by replacing redundant details with more stable abstractions or by utilizing data normalization to prevent redundancy. Django uses Python throughout, even for configuration, files, and data models. The designed user interface is shown in Fig. 10. This user interface is actually a web page that is connected to our server. Input and output are received and sent by the page on the front-end side, and the necessary processing is performed on the server side, which is the back-end side. The user enters his sentences in the specified box and sees them corrected in the bottom box. The output contains two texts, one text is related to correction using rules and the other is related to modification by deep learning.

Result and discussion

The accuracy results of the trained networks are presented in Table 5. The loss and accuracy diagrams related to these networks are also shown in Figs. 11, 12, 13. It should be noted that an epoch includes a complete training cycle in the training set, in other words, when every sample of the training set has been viewed, an epoch is finished. As can be seen in Figs. 11, 12, 13 and Table 5, in general, the three networks have acted almost identically. According to Fig. 12 and Table 5, by increasing the number of LSTM blocks of the encoder and decoder layers, the performance has improved and it has been able to achieve more accuracy than the previous state with the half number of epochs, which is both in the accuracy of the test and training data and in their loss function is observed. In the diagram of loss and accuracy of Fig. 12, we see graphs were closer to each other in this case compared to the case with less LSTM block number. This subject indicates an improvement in the speed of training, and according to the figure, the number of epochs for training has decreased to 6 for this case. On the other hand, according to Table 5, the accuracy of training and testing for 1000 LSTM blocks is higher than 500 LSTM blocks.

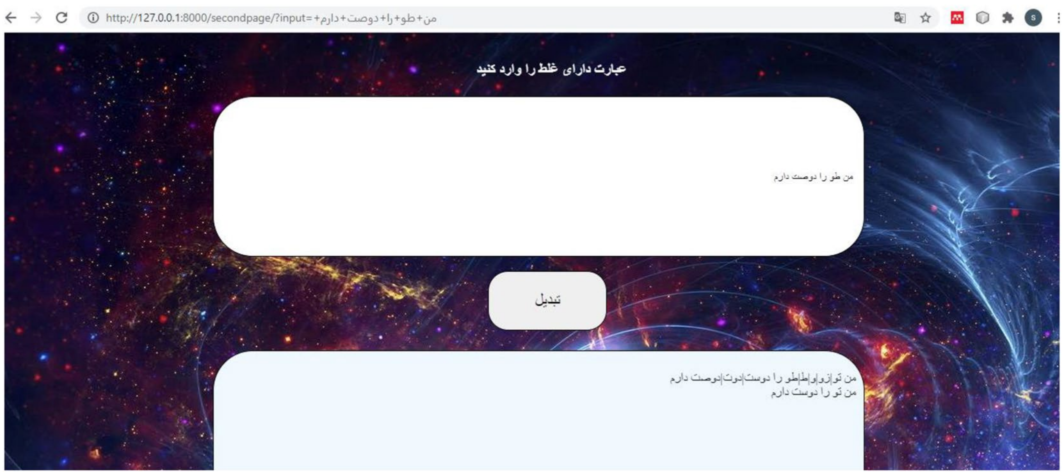
According to Fig. 13 and Table 5, the performance of the network with capsular and convolutional layers is similar to the basic network based on LSTM. This closeness is due to the fact bidirectional LSTM units have the ability to preserve the order of their previous and next words, and to some extent, they can also perform the task of the capsule network. On the other hand, LSTM units can extract features to some extent by memorizing long strings. The basic network with 1000 LSTM units in each encoder and decoder part was considered as the final network topology. Examples of the output of this network in spelling correction are reported in Table 6. According to this, all kinds of errors have been in these sentences. The network has been able to correctly identify these errors and replace the correct phrase due to the high accuracy of the learning.

In Singh and Singh[46], which has done the spelling correction for the Indian language, a part of the corpus was considered as evaluation data. They obtained and reported various evaluation criteria such as accuracy, precision, recall, and F-criterion. We also evaluated the system by calculating all these parameters, and the Bleu criterion. 200 thousand data were used to evaluate the approach based on deep networks. But since it was not possible to apply this number of sentences to the other approaches, we compared and checked the evaluations with 2500 sentences. In the rule-based approach and the list of common errors, as different rules suggest different words for a wrong word, therefore Levenshtein's distance algorithm was used. The suggested words were sorted based on that and the word with the highest ranking was chosen for evaluation. As we know, to transform one string to another string, the least number of operations that are required is the Levenshtein distance between two strings, and an operation can be either inserting or substituting a character, or switching two characters. Therefore, it



(a) (b)

Figure 9. The structure of the neural networks employed in this study (a) basic network (LSTM unit) (b) basic network (LSTM unit) with capsule and convolution layer.



http://127.0.0.1:8000/secondpage/?input=من+طو+را+دوصت+دارم

عبارت دارای غلط را وارد کنید

من طو را دوصت دارم

تبدیل

من تو|زو|و|ط|طو را دوست|دوت|دوصت دارم
من تو را دوست دارم

Figure 10. The user interface implemented in this study.

Network type	Training accuracy	Testing accuracy
Basic neural network with 500 memory units per layer	0.968	0.978
Basic neural network with 1000 memory units per layer	0.984	0.988
Neural network with 1000 memory units per layer along with encapsulation and convolution layer	0.979	0.979

Table 5. Accuracy of training and testing data in trained networks.

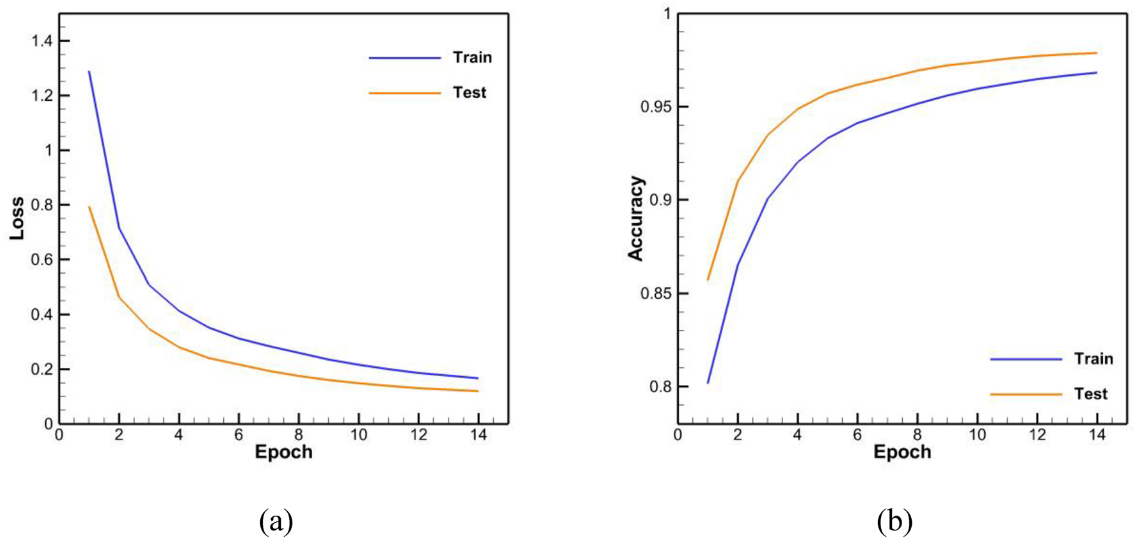


Figure 11. Base network with 500 units of memory per layer (**a**) loss function, (**b**) accuracy.

is a good criterion for choosing one or more words from among several suggested words in error correction. Table 7 displays the results of the evaluation for spelling correction. The table shows that the network's basic architecture had the best performance. The basic network with 200 thousand evaluation data was able to correct 70% of the errors and increase the accuracy of the text to 87%. The weaker performance of the rules is due to the initial database, which does not cover all the words. Arab Surkhi et al.[62] and Yazdani et al.[76] have pointed out the impact of the database on the evaluation result in the rule-based method. In addition, only the lowest Levenshtein distance has been utilized for evaluation, while a suggested list is presented to the user. Hu et al.[41] indicated that when using the suggested list, the actual system performance will become higher than the lowest Levenshtein distance suggested. As one can observe, the performance of the network with capsular and convolutional layers is similar to the basic network, and in the evaluation, both have performed almost the same.

To show the adequacy and goodness of the network's performance, the operation of the neural network-based system was compared with similar systems (spelling correction) for other languages. The purpose of this

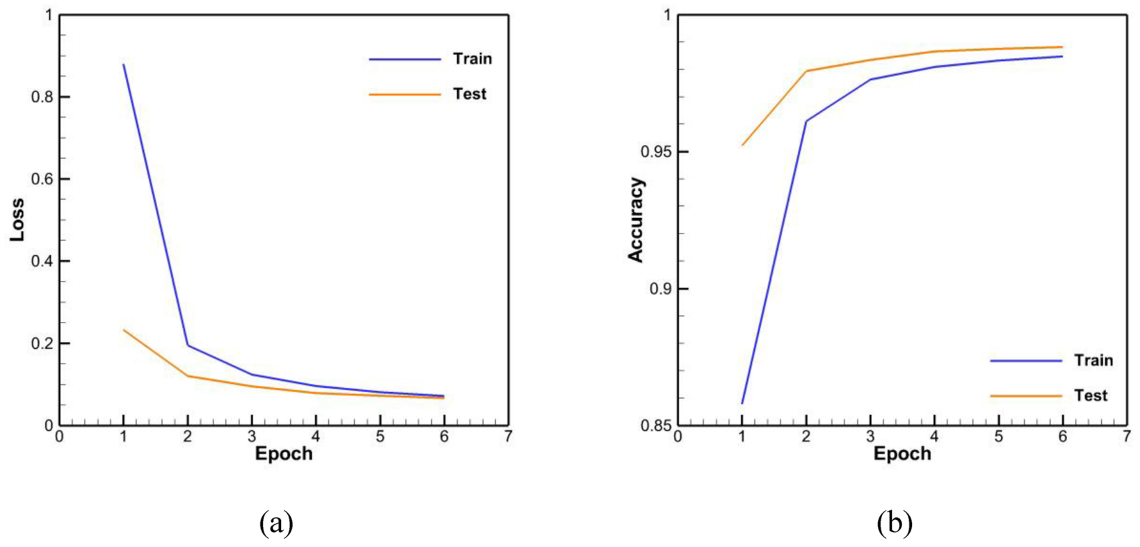


Figure 12. Base network with 1000 units of memory per layer (**a**) loss function, (**b**) accuracy.

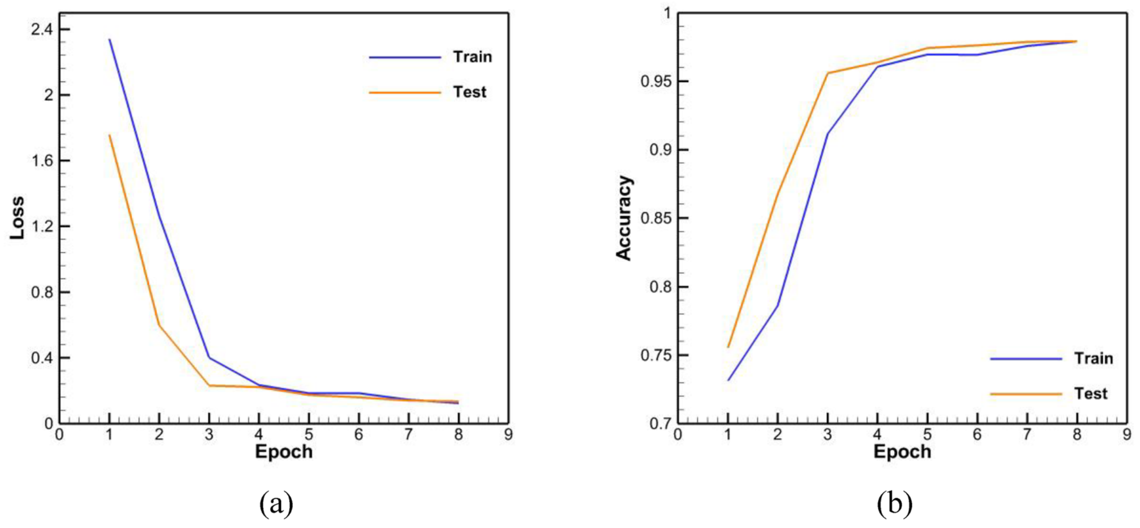


Figure 13. Network with 1000 units of memory per layer with capsule and convolutional layers (**a**) loss function, (**b**) accuracy.

comparison is to validate and ensure the sufficiently of the network activity in the evaluation, the result of which can be seen in Table 8. In this comparison, Singh and Singh study[46] was used, which contrasted its system with similar systems and reported the results. As shown in the table below, the network-based approach is consistent with the results of similar studies for other languages, and the proposed model gives better performance than other current models for spell-checking and correction.

Summary and conclusion

Due to the lack of a comprehensive system for the automatic correction of Persian texts and considering the much study to create such systems in the English language, the creation of an automatic system for spelling correction of Persian texts was investigated in this study. In order to correct the error, the error must first be identified. Therefore, a database that includes the main and more useful Persian words was collected. After the initial pre-processing of the input data, the words were compared with the database. If the word is not found in the database, it is recognized as an error. In the proposed system of this study, error correction is done utilizing three approaches. In this research, a list of commonly misspelled words including 700 words was prepared so that if they are identified, they can be inserted and replaced directly. A database was compiled to detect spelling errors based on rules, and this database included 55 thousand common Persian words. 112 rules were implemented to correct spelling caused by phonetic similarity and typographical errors, which provided suggested words for unrecognized words in the database. In order to evaluate, 2500 spell error sentences were applied and the word with the lowest Levenshtein distance was chosen as the selected word in this method. In the deep learning approach, an encoder-decoder network based on LSTM units was employed along with word embedding in the

Correct sentence (Target)	Input (Sentence with errors)	Output (Sentence with correction)
من یکی را نصیحت نکن. (Do not advise me.)	من یکسی را نصسیحت نکتن.	من یکی را نصیحت نکن.
این بخاطر این است که مجبورت کردم معطل شی. (It's because I made you linger.)	این بخارط این است که مجبروت کردم معطل شی.	این بخاطر این است که مجبورت کردم معطل شی.
به نظرم همیشه یک امر پنهان پنهان است چون به اندازه‌ی کافی محترم قابل دفاع و یا شایسته‌ی آشکار شدن نیست اصلا کلمه پنهان نفرت آور است برایم. (In my opinion, a hidden matter is always hidden because it is not respectable enough to be defended or worthy of being revealed. The word hidden is disgusting to me.)	به نظرمم همیششه یک امر پنهاان پنهاان است چون به انداازه ی کافیی محتررم قابلل دفاعع و یا شایسسته‌ی آشکاار شدن نیستتن اصلاا کلمهه پنهاان نفرتت آور است براییم.	به نظرم همیشه یک امر پنهان پنهان است چون به اندازه ی کافی محترم قابل دفاع و یا شایسته ی آشکار شدن نیست اصلا کلمه پنهان نفرت آور است برایم.
تناسخ وجود دارد. (There is reincarnation.)	تناصخ وجود دارد.	تناسخ وجود دارد.
احتیاط کن. (Be careful.)	احتیات کن.	احتیاط کن.

Table 6. Examples of sentences correction by the trained network.

Type of approach	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)	Bleu (%)
Rules (with 2500 sentences)	62	21	33	25	40
Basic network (with 2500 sentences)	64	50	48	49	65
Network with convolutional layer and capsule (with 2500 sentences)	60	43	40	41	60
Basic network (with 200 thousand sentences)	87	70	89	78	84

Table 7. Evaluation of different methods used in this research.

input and convolutional, max pooling, and capsule layers. An artificial database containing about 1 million and 200 thousand data was constructed to train the deep neural network. 800 thousand sentences were considered for training, 200 thousand sentences were applied for testing, and the others were utilized for evaluation. The basic structure was considered an encoder-decoder network based on LSTM. The results indicated that the effectiveness of the network with capsular and CNN layers is similar to the basic network. The trained network performed well in the evaluation and obtained values of 87%, 70%, 89%, 78%, and 84 for accuracy, precision, recall, F-criterion, and Bleu's criterion, respectively. The results indicated that in spelling correction, the approach

Method type	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)	Bleu (%)
LSTM model (Malayalam)	55	47	97	63	–
HINSPELL (Hindi)	77	–	–	–	–
CNN-Strides (English)	48	46	40	42	–
CNN-Filters (English)	55	–	–	–	–
HINDIA (split dataset)	80	85	72	78	–
HINDIA (testing dataset)	74	81	72	77	–
Our network (Persian dataset)	87	70	89	78	84

Table 8. Comparison of network-based system performance of this research with similar systems expressed in Singh and Singh's study[46].

based on the deep learning is better than rules (The codes written in conducting this research are available in Supplementary Information).

Data availability

Data will be made available on request. If someone wants to request the data from this study, they can contact corresponding author.

Received: 9 July 2023; Accepted: 11 November 2023
Published online: 15 November 2023

References

1. Torfi, A., Shirvani, R. A., Keneshloo, Y., Tavaf, N. & Fox, E. A. Natural language processing advancements by deep learning: A survey. arXiv Prepr. arXiv 2003.01200. (2020) https://doi.org/10.48550/arXiv.2003.01200.
2. Otter, D. W., Medina, J. R. & Kalita, J. K. A survey of the usages of deep learning for natural language processing. *IEEE Trans. Neural Netw. Learn. Syst.* **32**, 604–624. https://doi.org/10.1109/TNNLS.2020.2979670 (2020).
3. Kale, M. *et al.* Deep learning for digital text analytics: Sentiment analysis. arXiv Prepr. arXiv 1804.03673 (2018) https://doi.org/10.48550/arXiv.1804.03673.
4. Sansone, C. & Sperlí, G. Legal information retrieval systems: State-of-the-art and open issues. *Inf. Syst.* **106**, 101967. https://doi.org/10.1016/j.is.2021.101967 (2022).
5. Wu, L.-T., Lin, J.-R., Leng, S., Li, J.-L. & Hu, Z.-Z. Rule-based information extraction for mechanical-electrical-plumbing-specific semantic web. *Autom. Constr.* **135**, 104108. https://doi.org/10.1016/j.autcon.2021.104108 (2022).
6. Kim, Y., Bang, S., Sohn, J. & Kim, H. Question answering method for infrastructure damage information retrieval from textual data using bidirectional encoder representations from transformers. *Autom. Constr.* **134**, 104061. https://doi.org/10.1016/j.autcon.2021.104061 (2022).
7. Yi, J. *et al.* Analysis of stock market public opinion based on web crawler and deep learning technologies including 1DCNN and LSTM. *Arab. J. Sci. Eng.* https://doi.org/10.1007/s13369-022-07444-7 (2022).
8. Jiang, S., Hu, J., Magee, C. L. & Luo, J. Deep learning for technical document classification. *IEEE Trans. Eng. Manag.* https://doi.org/10.1109/TEM.2022.3152216 (2022).
9. Song, D., Vold, A., Madan, K. & Schilder, F. Multi-label legal document classification: A deep learning-based approach with label-attention and domain-specific pre-training. *Inf. Syst.* **106**, 101718. https://doi.org/10.1016/j.is.2021.101718 (2022).
10. Huan, J. L., Sekh, A. A., Quek, C. & Prasad, D. K. Emotionally charged text classification with deep learning and sentiment semantic. *Neural Comput. Appl.* https://doi.org/10.1007/s00521-021-06542-1 (2022).
11. Balyan, R., McCarthy, K. S. & McNamara, D. S. Applying natural language processing and hierarchical machine learning approaches to text difficulty classification. *Int. J. Artif. Intell. Educ.* **30**, 337–370. https://doi.org/10.1007/s40593-020-00201-7 (2020).
12. Khan, L., Amjad, A., Ashraf, N. & Chang, H.-T. Multi-class sentiment analysis of urdu text using multilingual BERT. *Sci. Rep.* **12**, 5436. https://doi.org/10.1038/s41598-022-09381-9 (2022).
13. Kim, Y. *et al.* Validation of deep learning natural language processing algorithm for keyword extraction from pathology reports in electronic health records. *Sci. Rep.* **10**, 20265. https://doi.org/10.1038/s41598-020-77258-w (2020).
14. Zhang, Y., Chen, Q., Yang, Z., Lin, H. & Lu, Z. BioWordVec, improving biomedical word embeddings with subword information and MeSH. *Sci. Data.* **6**, 52. https://doi.org/10.1038/s41597-019-0055-0 (2019).
15. Alrobah, N. & Albahli, S. Arabic handwritten recognition using deep learning: A survey. *Arab. J. Sci. Eng.* **47**, 9943–9963. https://doi.org/10.1007/s13369-021-06363-3 (2022).
16. Bai, X. & Stede, M. A survey of current machine learning approaches to student free-text evaluation for intelligent tutoring. *Int. J. Artif. Intell. Educ.* https://doi.org/10.1007/s40593-022-00323-0 (2022).
17. Rostamzadeh, S. *et al.* A comparative investigation of machine learning algorithms for predicting safety signs comprehension based on socio-demographic factors and cognitive sign features. *Sci. Rep.* **13**, 10843. https://doi.org/10.1038/s41598-023-38065-1 (2023).
18. Kim, Y. *et al.* A pre-trained BERT for Korean medical natural language processing. *Sci. Rep.* **12**, 13847. https://doi.org/10.1038/s41598-022-17806-8 (2022).
19. Azmi, A. M., Almutery, M. N. & Aboalsamh, H. A. Real-word errors in Arabic texts: A better algorithm for detection and correction. *IEEE/ACM Trans. Audio Speech Lang. Process.* **27**, 1308–1320. https://doi.org/10.1109/TASLP.2019.2918404 (2019).
20. Lee, J.-H., Kim, M. & Kwon, H.-C. Deep learning-based context-sensitive spelling typing error correction. *IEEE Access.* **8**, 152565–152578. https://doi.org/10.1109/ACCESS.2020.3014779 (2020).
21. Karthikeyan, S., de Herrera, A. G. S., Doctor, F. & Mirza, A. An ocr post-correction approach using deep learning for processing medical reports. *IEEE Trans. Circuits Syst. Video Technol.* **32**, 2574–2581. https://doi.org/10.1109/TCSVT.2021.3087641 (2021).
22. https://www.djangoproject.com/.
23. Chaabi, Y. & Allah, F. A. Amazigh spell checker using Damerau-Levenshtein algorithm and N-gram. *J. King Saud Univ. Inf. Sci.* **34**, 6116–6124. https://doi.org/10.1016/j.jksuci.2021.07.015 (2022).

24. Sharma, D., Mattu, G. S. & Sharma, S. N-gram based amharic grammar checker. In *Proceedings of the Future Technologies Conference (FTC) 2022*, vol. 3, 622–632 (2022) https://doi.org/10.1007/978-3-031-18344-7_44.
25. Naseem, T. & Hussain, S. A novel approach for ranking spelling error corrections for Urdu. *Lang. Resour. Eval.* **41**, 117–128. https://doi.org/10.1007/s10579-007-9028-6 (2007).
26. Singh, S.P., Kumar, A., Singh, L., Bhargava, M., Goyal, K. & Sharma, B. Frequency based spell checking and rule based grammar checking. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, 4435–4439 (2016) https://doi.org/10.1109/ICEEOT.2016.7755557.
27. Imperial, J. M. R., Ya-On, C. G. V. & Ureta, J. C. An experimental Tagalog Finite State Automata spellchecker with Levenshtein edit-distance feature. In *2019 International Conference on Asian Language Processing (IALP)*, 240–243 (2019) https://doi.org/10.1109/IALP48816.2019.9037687.
28. Manohar, N., Lekshmipriya, P. T., Jayan, V. & Bhadran, V. K. Spellchecker for Malayalam using finite state transition models. In *2015 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, 157–161 (2015) https://doi.org/10.1109/RAICS.2015.7488406.
29. Ahmadi, S. Hunspell for Sorani Kurdish Spell Checking and Morphological Analysis. arXiv Prepr. arXiv2109.06374 (2021) https://doi.org/10.48550/arXiv.2109.06374.
30. Ramasamy, L. & Žabokrtský, Z. Tamil dependency parsing: results using rule based and corpus based approaches. In *Computational Linguistics and Intelligent Text Processing: 12th International Conference, CICLing 2011, Tokyo, Japan, February 20–26, 2011. Proceedings, Part I* 12, 82–95 (2011) https://doi.org/10.1007/978-3-642-19400-9_7
31. Ganfure, G. O. & Midekso, D. Design and implementation of morphology based spell checker. *Int. J. Sci. Technol. Res.* **3**, 118–125 (2014).
32. Khosrobeygi, Z., Veisi, H., Ahmadi, H. R. & Shabanian, H. A rule-based post-processing approach to improve Persian OCR performance. *Sci. Iran.* **27**, 3019–3033. https://doi.org/10.24200/sci.2020.53435.3267 (2020).
33. Dastgheib, M. B. *et al.* Design and implementation of Persian spelling detection and correction system based on Semantic. *Signal Data Process.* **16**, 117–128. https://doi.org/10.29252/jsdp.16.3.128 (2019).
34. Dashtipour, K. *et al.* A hybrid Persian sentiment analysis framework: Integrating dependency grammar based rules and deep neural networks. *Neurocomputing.* **380**, 1–10. https://doi.org/10.1016/j.neucom.2019.10.009 (2020).
35. Aziz, R., Anwar, M. W., Jamal, M. H. & Bajwa, U. I. A hybrid model for spelling error detection and correction for Urdu language. *Neural Comput. Appl.* **33**, 14707–14721. https://doi.org/10.1007/s00521-021-06110-7 (2021).
36. Zobel, J. & Dart, P. Finding approximate matches in large lexicons. *Softw. Pract. Experience* **25**(3), 331–345 (1995).
37. Tahira Naseem, A. Hybrid Approach for Urdu Spell Checking. (Doctoral dissertation, MS Thesis), thesis at the National University of Computer & Emerging Sciences (2004).
38. Kaur, G., Kaur, K. & Singh, P. Spell checker for Punjabi language using deep neural network. In *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, 147–151 (2019) https://doi.org/10.1109/ICACCS.2019.8728369.
39. Sooraj, S., Manjusha, K., Anand Kumar, M. & Soman, K. P. Deep learning based spell checker for Malayalam language. *J. Intell. Fuzzy Syst.* **34**, 1427–1434. https://doi.org/10.3233/JIFS-169438 (2018).
40. Caryappa, B. C., Hulipalled, V. R. & Simha, J. B. Kannada grammar checker using LSTM neural network. In *2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)*, 332–337 (2020) https://doi.org/10.1109/ICSTCEE49637.2020.9277479.
41. Hu, Y., Jing, X., Ko, Y. & Rayz, J. T. Misspelling correction with pre-trained contextual language model. In *2020 IEEE 19th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, 144–149 (2020) https://doi.org/10.1109/ICCICC50026.2020.9450253.
42. Beloki, Z., Saralegi, X., Ceberio, K. & Corral, A. Grammatical Error Correction for Basque through a seq2seq neural architecture and synthetic examples. *Proces. del Leng. Nat.* **65**, 13–20 (2020).
43. Zolzaya, B., Nishimura, R., Altangerel, A. & Kitaoka, N. Normalization of translated words using seq2seq model with spell checker. (2020) https://doi.org/10.1145/3464361.
44. Mandal, S. & Nanmaran, K. Normalization of transliterated words in code-mixed data using Seq2Seq model and Levenshtein distance. arXiv Prepr. arXiv 1805.08701 (2018) https://doi.org/10.48550/arXiv.1805.08701.
45. Mager, M., Rosales, M. J., Çetinoğlu, Ö. & Meza, I. Low-resource neural character-based noisy text normalization. *J. Intell. Fuzzy Syst.* **36**, 4921–4929. https://doi.org/10.3233/JIFS-179039 (2019).
46. Singh, S. & Singh, S. HINDIA: A deep-learning-based model for spell-checking of Hindi language. *Neural Comput. Appl.* **33**, 3825–3840. https://doi.org/10.1007/s00521-020-05207-9 (2021).
47. Sampath, A. & Shanmugavel, V. Hybrid Tamil spell checker with combined character splitting. *Concurr. Comput. Pract. Exp.* **35**, e7440. https://doi.org/10.1002/cpe.7440 (2023).
48. Anbukkarasi, S. & Varadhaganapathy, S. Neural network-based error handler in natural language processing. *Neural Comput. Appl.* https://doi.org/10.1007/s00521-022-07489-7 (2022).
49. Engelbrecht, A. P. *Computational Intelligence: An Introduction* (Wiley, 2007).
50. El Asri, L., He, J. & Suleman, K. A sequence-to-sequence model for user simulation in spoken dialogue systems. arXiv Prepr. arXiv 1607.00070 (2016) https://doi.org/10.48550/arXiv.1607.00070.
51. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. & Dean, J. Distributed representations of words and phrases and their compositionality. *Adv. Neural Inf. Process. Syst.* https://doi.org/10.5555/2999792.2999959 (2013).
52. Wensen, L., Zewen, C., Jun, W. & Xiaoyi, W. Short text classification based on Wikipedia and Word2vec. In *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, 1195–1200 (2016) https://doi.org/10.1109/CompComm.2016.7924894.
53. Collobert, R. *et al.* Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* **12**, 2493–2537 (2011).
54. Mallick, R., Susan, S., Agrawal, V., Garg, R. & Rawal, P. Context-and sequence-aware convolutional recurrent encoder for neural machine translation. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, 853–856 (2021) https://doi.org/10.1145/3412841.3442099.
55. Goldberg, Y. Neural network methods for natural language processing. *Synth. Lect. Hum. Lang. Technol.* **10**, 1–309. https://doi.org/10.1007/978-3-031-02165-7 (2017).
56. Xi, E., Bing, S. & Jin, Y. Capsule network performance on complex data. arXiv Prepr. arXiv 1712.03480 (2017) https://doi.org/10.48550/arXiv.1712.03480.
57. Gumaei, A., Hassan, M. M., Alelaiwi, A. & Alsalman, H. A hybrid deep learning model for human activity recognition using multimodal body sensing data. *IEEE Access.* **7**, 99152–99160. https://doi.org/10.1109/ACCESS.2019.2927134 (2019).
58. Uddin, M. Z., Hassan, M. M., Alsanad, A. & Savaglio, C. A body sensor data fusion and deep recurrent neural network-based behavior recognition approach for robust healthcare. *Inf. Fusion.* **55**, 105–115. https://doi.org/10.1016/j.inffus.2019.08.004 (2020).
59. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural Comput.* **9**, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735 (1997).
60. Xia, C., Zhao, D., Wang, J., Liu, J. & Ma, J. ICSH 2018: LSTM based sentiment analysis for patient experience narratives in E-survey tools. In *Smart Health: International Conference, ICSH 2018, Wuhan, China, July 1–3, 2018, Proceedings* vol. 6, 231–239 (2018) https://doi.org/10.1007/978-3-030-03649-2_23.

61. Gers, F. A., Schraudolph, N. N. & Schmidhuber, J. Learning precise timing with LSTM recurrent networks. *J. Mach. Learn. Res.* **3**, 115–143. https://doi.org/10.1162/153244303768966139 (2002).
62. Arab Surkhi, M., Fili, H. & Azadnia, M. Providing an expert system for automatic correction of Persian language spelling errors. In *The 12th Annual Conference of the Iranian Computer Association* (2006).
63. Xie, J., Chen, B., Gu, X., Liang, F. & Xu, X. Self-attention-based BiLSTM model for short text fine-grained sentiment classification. *IEEE Access.* **7**, 180558–180570. https://doi.org/10.1109/ACCESS.2019.2957510 (2019).
64. Zhang, X. & Gao, T. Multi-head attention model for aspect level sentiment analysis. *J. Intell. Fuzzy Syst.* **38**, 89–96. https://doi.org/10.3233/JIFS-179383 (2020).
65. Xu, T. *et al.* Neural machine translation of chemical nomenclature between English and Chinese. *J. Cheminform.* **12**, 1–6. https://doi.org/10.1186/s13321-020-00457-0 (2020).
66. Ghosh, S. & Kristensson, P. O. Neural networks for text correction and completion in keyboard decoding. arXiv Prepr. arXiv 1709.06429 (2017) https://doi.org/10.48550/arXiv.1709.06429.
67. Eslami, M., Sharifi Ateshgah, M., Alizadeh Lemjiri, S. & Zandi, T. The Zaya (generative vocabulary) of the Persian language. In *Collection of Articles of the First Persian Language and Computer Research Workshop* (2004)
68. Papineni, K., Roukos, S., Ward, T. & Zhu, W. J. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 311–318 (2002) https://doi.org/10.3115/1073083.1073135.
69. https://github.com/roshan-research/hazm.
70. https://www.peykaregan.ir/dataset.
71. Shahmiri, A., Safabakhsh, R. & Dejkam, R. Automatic correction of Farsi typos with the help of hybrid artificial neural network. *Electr. Electron. Eng. Iran.* **5**, 10–16 (2008).
72. Alian, M., Al-Naymat, G. & Ramadan, B. Arabic real time entity resolution using inverted indexing. *Lang. Resour. Eval.* **54**, 921–941. https://doi.org/10.1007/s10579-020-09504-6 (2020).
73. http://www.manythings.org/anki/.
74. https://fasttext.cc/.
75. https://keras.io/api/losses/probabilistic_losses/.
76. Yazdani, A. *et al.* Automated misspelling detection and correction in Persian clinical text. *J. Digit. Imaging.* **33**, 555–562. https://doi.org/10.1007/s10278-019-00296-y (2020).

Author contributions
All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Sa.Kasmaiee, Si.Kasmaiee and M.H. The first draft of the manuscript was written by Sa.Kasmaiee and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Competing interests
The authors declare no competing interests.

Additional information
Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-023-47295-2.

Correspondence and requests for materials should be addressed to S.K.

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

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

© The Author(s) 2023