applied sciences

MDPI

Article

Persian Optical Character Recognition Using Deep Bidirectional Long Short-Term Memory

Zohreh Khosrobeigi [1], Hadi Veisi [2], Ehsan Hoseinzade [3] and Hanieh Shabanian [4,*]

[1] School of Computer Science and Statistics, Trinity College Dublin, D02 YY50 Dublin, Ireland
[2] Faculty of New Sciences and Technologies, University of Tehran, Tehran P.O. Box 14399-56191, Iran
[3] School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada
[4] Computer Science Department, School of Computing and Analytics, Northern Kentucky University, Highland Heights, KY 41076, USA
* Correspondence: shabanianh1@nku.edu

Abstract: Optical Character Recognition (OCR) is a system of converting images, including text,into editable text and is applied to various languages such as English, Arabic, and Persian. While these languages have similarities, their fundamental differences can create unique challenges. In Persian, continuity between Characters, the existence of semicircles, dots, oblique, and left-to-right characters such as English words in the context are some of the most important challenges in designing Persian OCR systems. Our proposed framework, Bina, is designed in a special way to address the issue of continuity by utilizing Convolution Neural Network (CNN) and deep bidirectional Long-Short Term Memory (BLSTM), a type of LSTM networks that has access to both past and future context. A huge and diverse dataset, including about 2M samples of both Persian and English contexts,consisting of various fonts and sizes, is also generated to train and test the performance of the proposed model. Various configurations are tested to find the optimal structure of CNN and BLSTM. The results show that Bina successfully outperformed state of the art baseline algorithm by achieving about 96% accuracy in the Persian and 88% accuracy in the Persian and English contexts.

Keywords: Optical Character Recognition (OCR); Long Short-Term Memory (LSTM); Bidirectional LSTM (BLSTM); Convolution Neural Network (CNN); Persian language

Citation: Khosrobeygi, Z.; Veisi, H.; Hoseinzade, E.; Shabanian, H. Persian Optical Character Recognition Using Deep Bidirectional Long Short-Term Memory. *Appl. Sci.* **2022**, *12*, 11760. https://doi.org/10.3390/app122211760

Academic Editor: Hui Yuan

Received: 12 October 2022
Accepted: 16 November 2022
Published: 19 November 2022

1. Introduction

Optical Character Recognition (OCR) is the process of extracting words and characters from an image and converting them to their corresponding editable text. The input images include PDF files, images captured by cell phone or camera, and scanned documents [1]. OCR's applications can be found in many areas, such as reading forms and cheques; converting hard copies of archived documents to digital files, separating written parts of the documents such as written addresses on letter envelopes; reading books and papers for blind or illiterate people, and so on. An accurate OCR system speeds up the processes mentioned above by removing the time-consuming user tasks [2]. Each OCR system contains several modules illustrated in Figure 1.

First, an analysis is done on the document image, which is fed into the system to be divided into several segments [3]. In the next step, a pre-processing module removes noises and deviations and converts the input image into a binary one. Then, inner segmentation decomposes text lines into characters. In the feature extraction step, beneficial features are generated and given into the recognition module to recognize the corresponding character to the passed features. In the optional post-processing step, the accuracy is improved by using the dictionary to correct the recognized words. In the end, the final output is shown to the user.

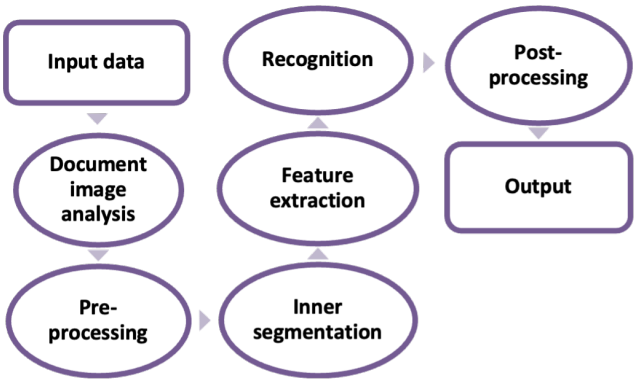

Figure 1. Different modules of an OCR system.

The quality of an OCR system can be improved by various factors, including understanding the existing challenges of the target language, choosing an appropriate training algorithm for an OCR system, and finally covering, different styles of the characters in the target language by utilizing a thorough and diverse dataset.

Persian is one of the most challenging languages in terms of developing an accurate OCR system due to its characteristics. It is a right-to-left language that has 32 characters and ten numbers. Some of the most important challenges are as follows [4]:

1. In Persian, a word is made by the connection of continuous characters that overlap each other. So, it makes it difficult to divide them into different characters. For example, in the word "شکلات", separation of /Sh/ "ش" and "ک" or /L/ "ل" and "ا" in "لا" is a difficult challenge since they cannot be separated by a simple vertical line [5]. The mentioned issue is illustrated in Figure 2.
2. Some characters are formed by two elements like /U/ "آ" or /G/ "گ", existence of space between these two elements makes segmentation to be difficult. The reason is their body shape is similar to some other characters but different in their oblique. Since their oblique is too smaller than their body, the error rate increases. For instance, "آ" is similar to /A/ "ا", which does not have the oblique.
3. The difference between some of the Persian characters, such as /P/ "پ", /B/ "ب", /Gh/ "ژ" and /Z/ "ز" is that they are only limited to their dot(s). So, OCR might find it difficult to recognize these characters [5]. In Persian language, there are 18 characters with dot(s).
4. A character is allowed to have different shapes based on the place it appears in a word [6]. For example, shapes of /H/ "ه" are /Guidance/ "مهتاب" at the beginning of the word, /Moonlight/ "شکوفه" in the middle of the word, /Blossom/ "شکوفه" at the end of the word, which is connected to another character, and /Simple/ "ساده" at the end of the word which is not connected to another character [7].
5. Using English characters, words, and numbers is a common phenomenon in the Persian text. This feature brings complexity to the segmentation process in a text containing both.

Figure 2. Segmentation of a word in the Persian context.

Researchers have tested various algorithms to improve the performance of the segmentation and recognition sections of the OCR system [8–10]. Deep learning [11] is the newest branch of machine learning algorithms that have been applied in this field [12]. Among different deep learning algorithms, CNNs are renowned for feature extraction from images and are widely used in the recognition section of an OCR system to extract beneficial features from an input image [11,13]. Recurrent Neural Networks (RNN) [14], and especially LSTM networks [15] are designed in a way that can save useful information in the sequential data because of their internal memory [16]. Due to the sequential nature of textual data, some researchers have taken advantage of LSTM to design a better OCR system [3,16–23], and it can be used in Persian OCR as well. While in languages with continuous characters like Arabic and Persian getting insight into both previous and future contexts is useful, LSTM has access to only previous context which leads to missing beneficial information about the future context. To address this issue, BLSTM networks are utilized. BLSTM network is a modified version of LSTM that can simultaneously get information from past and future contexts. This process in an Arabic OCR system, have shown some improvements in the accuracy of the system [16,17]. This investigation is also necessary to improve the accuracy of the Persian OCR systems. To the best of the authors' knowledge, this process has not been performed yet. Hence, one of the contributions of this paper is specified to address the current issue of the Persian OCR systems in accessing information from past and future contexts simultaneously. To improve performance of an Persian OCR system, it is vital to have a thorough and diverse dataset covering different forms of Persian words including those mentioned challenges. It is even more important to have such a dataset when we are dealing with deep learning-based algorithms like CNN, LSTM, and BLSTM because these algorithms need plenty of sample data for training. However, none of the previous works [24–26] were able to introduce a thorough and diverse dataset considering the mentioned challenges. As a result, having an appropriate training dataset is a demanding issue in Persian OCR systems. We collected a huge dataset to cover not only situation mentioned as challenges, but also the need of deep learning algorithms to plenty of sample data.

In this paper, we introduce a Persian OCR system called Bina. The core of the feature extraction and recognition sections in Bina are chosen to be a combination of CNN and BLSTM. More specifically, CNN acts as a trainable feature detector for the spatial signal. It learns powerful convolutional features which operates on a static spatial input while the BLSTM processes a sequence of such high-level representations to map them to one of the classes of outputs. BLSTM's access to previous and future contexts comes handy in addressing above mentioned challenges. To thoroughly train and test the performance of Bina, a comprehensive and massive dataset with about 2 million samples including various fonts and styles of the words and characters in both Persian and English in the same context is used. The main contributions of this paper can be summarized as follows:

1. To our knowledge, this is the first work that applies CNN and Deep BLTSM to a Persian OCR system to address the issue of continuity of characters in this language.
2. Gathering a huge and diverse dataset for the Persian OCR system that covers different fonts, sizes, styles of characters, and words. The proposed dataset includes English and Persian words in the same context for the first time. We collected a dataset of 2 million samples that not only improve the accuracy of our system but also can be used by other scientists in their future research.

The paper is organized as follows: Section 2 explains related works, the proposed method is illustrated in Section 3, Section 4 summarizes the experimental setup and results, and Section 5 analyzes the results of the reference model. A comprehensive discussion of the proposed method is presented in Section 6. Section 7 summarizes the conclusions and future work.

2. Related Works

We shortly review some related works to the segmental script writing system. A part of this writing system that is focused on in this paper can be categorized into two general groups, the true alphabetic system (i.e., Latin/English), which is left to right, and the Abjads writing system (i.e., Arabic and Persian) that is right to left. OCRs developed for Latin languages like English in which characters are written isolated, fall in the first category, and OCRs that cover languages like Arabic and Persian belong to the second category. The Abjads scripts are more complex than Latin scripts due to context-sensitive shape, cursive, and overlapping of word parts. In this section, we review the two mentioned categories of OCRs.

To thoroughly cover previous relevant works to our research about the Persian language, we also review Persian OCRs and Persian datasets that have been utilized to build OCRs in two separate subsections.

2.1. Alphabet Languages

Researchers have started developing OCR systems for the left to right languages since 1900 [27]. Of all the left-to-right languages English is the most explored language in developing an OCR system.

One of the most famous OCR methods belonged to R. Smit [3], in which a line-finding algorithm used to find the rows of text was followed by baseline fitting. Then, fixed pitch detection, chopping, and proportional word finding were implemented. In the word recognition phase, chopping joined characters was improved by chopping the blob with the worst confidence from the character classifier. In the character classifier step, features were extracted using segments of the polygonal approximation. After that, the classification was done in two steps. First, a class pruner created a shortlist of character classes that the unknown might match. Second, each feature of the unknown looked up a bit vector of prototypes of the given class that it might match, and then the actual similarity between them is computed.

English OCR systems can perform with the accuracy of about 100% in recognizing of typed and digital texts. However, recognition of handwritten texts is still challenging and considered as one of the trending topics for several researchers in this field. In 2009, an approach based on training a BLSTM network on unsegmented sequence data was introduced for recognizing unconstrained English handwritten text [22]. Word recognition accuracy on online and offline data was 79.7% and 74.1%. In another research [23], the authors suggested an approach for online recognition of English handwritten whiteboard notes. In this work, the BLSTM with a Connectionist Temporal Classification (CTC) output layer was applied to unsegmented labels directly. By having a 74.0%-word recognition rate, this model showed an improvement over the previously developed Hidden Markov Model (HMM)-based recognition system with a recognition rate of 65.4%.

Researchers presented an LSTM-based model for multilingual OCR systems that covered English, French, and German without any language model or dictionary correction [28]. Possessing only 1% recognition error showed its extraordinary performance. In another work [19], scientists used an RNN-LSTM model to recognize French words with 99% accuracy. The utilized data set involved the top 5000 French words in six fonts with the size 14.

2.2. Abjad Languages

The Persian language is similar to Arabic, Urdu, and Pashto languages in terms of being an Abjad right-to-left languages and having several common characters. As a result, they deserve to be reviewed more than other right-to-left scripts.

A Multi-Dimensional LSTM was developed to build an Arabic offline handwriting recognition system [16]. The input data was raw images without manual feature extraction, and the results proved that the suggested algorithm outperformed baseline algorithms. Authors in another work [17] have successfully developed an Arabic OCR system by using

a Deep BLSTM in the task of automatic discretization, which resulted in an 18% error rate reduction compared to the best-published algorithms. In the Urdu Nastaliq language, applying multi-dimensional LSTM with a right-to-left sliding window has led to achieving 94.97% accuracy [20]. Researchers also applied a multi-dimensional LSTM to the Pashto's image data and their OCR system was able to recognize out of raw image data with an accuracy of 98.9% [21]. Recently, LSTM-RNN was successfully utilized to extract high-level features from an initial scale-invariant feature set that covered different font sizes of Urdu's characters [29]. Two benchmark datasets, Centre for Language Engineering Text Images (CLETI) and Urdu printed text images (UPTI) were used. The achieved accuracy was 99%.

Persian Language

The most relevant works to our research are Persian OCR systems. So, several of them are reviewed in this subsection. There are two new feature extraction methods of "improved gradient" and "gradient histogram" were used by researchers to enhance the quality of Persian OCR systems [30]. Features are based on the gradient feature for brightness and for the two-level and gray-scale images. Through a neural network classifier and new features, the recognition rate was 99.02% by using the improved gradient method, and 98.8% by using the gradient histogram method on the HODA dataset. In 2018, researchers created a Sub-Word Image Dictionary (SWID) that was applied to sub-word based on Persian OCR methods [31]. To recognize the sub-words, Support Vector Machine (SVM) classifiers were trained by SWID. The K-means algorithm clustered data and initial classifier learned the result of this clustering as well as the entire shape of the sub-words. To find the same text equivalent of the sub-word image in a determined cluster, another SVM classifier was trained. Accuracy in word level was 61.14%. In another work in 2016 [32] the locations of the characters were extracted based on the baselines, the locations of dots, and the number of dots. Then, features of the image were extracted and decomposed into eight primary elements. Elements of each character were decomposed into simple shapes, which were called strokes. The generated Stroke Identification Vector (SIV) contained several elements, including the number of dots and location of dots of characters based on the baseline. The SIV was used to determine primary elements. In the recognition phase, another vector, Character Identification Vectors (CIV), which was similar to SIV in terms of structure was used to represent a determined character. The CIV was compared with the SIV, and character was found with a decision tree. The accuracy was 98.8% for digits, 88.7% for one sub-word, 81.4% for two characters sub-word, 73.6% for three characters sub-word, and 69.7% for four characters sub-word. CNN was used by researchers to automatically extract features [33]. In their work, the Convolution layer of CNN with size 5*5 was convolved with the input, and the output size was (H-5-1)*(W-5-1), where 1 is stride. Next, the pooling layer with 2*2 filters and stride 1 decreased the size of the output of the previous step. The last layer was a fully connected layer, which outputs a 32-dimensional vector, the number of Persian character classes. The accuracy of character recognition was 97%. In another work [34] researchers used famous CNN-based frameworks like DenseNet [35], ResNe [36], and VGG [37] in a Persian OCR systems and compared their performance. Authors have found that DenseNet is more suitable for a Persian OCR system because it has less learnable parameters which makes it less prone to overfitting. Another usage of a well known CNN-based network in Persian OCR system can be found in [38] where authors have used a LeNet [39] optimized by meta heuristic training in their system. Authors have utilized four algorithms of firefly algorithm, ant colony optimization, chimp optimization algorithm (ChOA), and particle swarm optimization to optimize LeNet's weights and concluded that ChOA serves this purpose better than the other algorithms.

According to our knowledge, Tesseract [18,40] is the only Persian OCR system that uses deep LSTM. It utilizes a CNN to extract features. Extracted features act as input data to an LSTM with four layers of 64 forward, 96 forward, 96 backward, and 192 forward. The output layer consists of 90 neurons. It can detect most of the Persian characters, but not all of them, and no English character. The accuracy of Tesseract when it is dealing with clean

images of the Persian context is higher than 93%; however, it cannot recognize contexts including English characters or some other Persian characters, punctuation that are not involved in 90 output nodes. This model was used as a reference model in this work.

Dataset is also a significant factor; having an inappropriate one could decrease the accuracy of an OCR system considerably. Several Persian datasets are used in the literature, and some of them are briefly introduced in this paper. However, there is no freely available dataset for Persian formal written text, but there are some for handwritten.

The first one is a dataset of handwritten Persian digits [21] containing binary images of 102,352 digits, which are extracted from about 12,000 registration forms in 200 dpi.

Another Persian dataset [25] is the Persian Handwritten Text Dataset (PHTD), including 140 handwritten documents of three different categories written by 40 individuals. The total number of text lines and words/sub-words in the dataset are 1787 and 27,073. The average number of text lines in PHTD is 13. By having two types of ground truths that are based on pixels and content information, PHTD can be used in some areas, such as sentence recognition/understanding, text-line segmentation, word segmentation, word recognition, and character segmentation.

PHCWT [26] is a Persian dataset containing Persian handwritten characters, words, and texts that can be used in various fields, such as offline handwritten optical character recognition, word segmentation, and text extraction. It includes a total of 51,200-character images, 3600-word images, and 400 texts collected by 400 contributors.

Table 1 illustrates all the presented information in this section in terms of algorithms, language, and accuracy (more Persian OCR systems [41]). As can be seen, none of the previously introduced OCR systems can be successful in both the Persian and English contexts.

Table 1. A comparison between explained algorithms in the related work section.

Ref.	Language	Architecture	Accuracy
[23]	On-line English Handwriting	BLSTM	74.00
[23]	On-line English Handwriting	HMM	65.40
[22]	Off-line English	BLSTM	74.10
[22]	On-line English	BLSTM	79.70
[19]	French	LSTM	99.00
[20]	Urdu	MLSTM	94.97
[21]	Pashto	MLSTM	98.90
[29]	Urdu	LSTM-RNN	99.00
[30]	Persian	Improved Gradient & Gradient Histogram	98.80
[31]	Persian	SVM	61.14
[33]	Persian	CNN	97.00
[32]	Persian	Stroke Identification Vector Character Identification Vectors	82.20
[41]	Persian	BSVM 2-D Wavelet Transform & PCA	82.51

3. Proposed Method: Bina

Like the other OCR systems, Bina consists of several modules, including segmentation, feature extraction, and recognition. The core module of Bina that make it distinctive compared to the other Persian OCRs is recognition.

In the segmentation phase, various characters inside the input image are separated from each other, a frame of the input image is selected with a constant high of 48 pixels and a variable width and is fed into the next step. This step's output is a matrix processed in the feature extraction step using CNN layers. In the feature extraction step, the features of each frame are extracted by applying a CNN on that frame. Then, the outputs of the last layer of the CNN are concatenated to build an embedding vector. Finally, the extracted features (feature vector) of the frame are fed to a deep BLSTM as input data to recognize which character this frame is showing.

Figure 3 shows a graphical view of the structure of the proposed model.

Different modules are explained thoroughly in the following subsections.

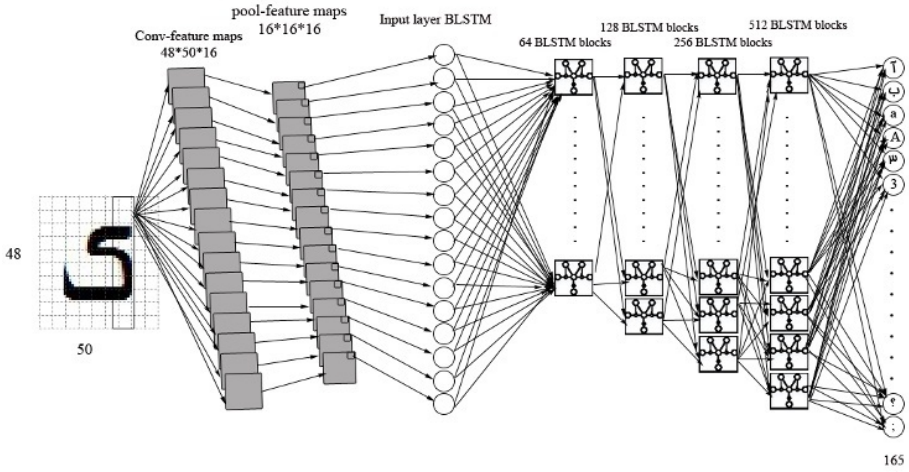

Figure 3. A graphical view of Bina.

3.1. Segmentation

The output of the segmentation section presents the input for the training of Bina. So, the quality of segmented words plays an important role in the success of the OCR system. To get high-quality segmented words inside the image, slid windowing [18,40,42], a precise and suitable method for the segmentation of long unstructured words, was chosen. This method is used for languages such as English, Arabic, and Persian [40]. This approach reads an image, and the height-normalized text lines are decomposed into various frames. Height normalization preserves the aspect ratio of the ligature image and ensures that the computed features are not dependent upon the size of the ligature [43]. The width is variable since each character of a word has a different width. For text lines with a fixed pitch, the algorithm uses fixed pitch detection and chopping [18]. Then, proportional word-finding is applied to non-fixed-pitch texts with the aim of defining characters of each word because the length of characters is different and there is no horizontal gap between bounding boxes of characters. For non-fixed-pitch text, candidate chop points are found from concave vertices of a polygonal approximation of the outline [18]. Then, an associator enhances the quality of words by making an A* (best first) search of the segmentation graph of possible combinations of the maximally chopped blobs into candidate characters. Finally, frames with a constant height of 48 pixels and variable width are ready to be used in the feature extraction module. Width is variable since each frame does not include an exact character; there is no gap between the characters, and they overlap. So, each frame includes a character with its overlap with other characters.

3.2. Feature Extraction

The main role of feature extraction is to eliminate the extra data from the sequences of feature vectors and maintain both useful and important information.

CNN, a renowned artificial neural network used for feature extraction from various types of data [44–46] is utilized to extract features from each frame. The advantage of CNN lies in the connection of each neuron only to a small local receptive field instead of fully connected to all the existing neurons of the previous layer [47]. Consequently, the number of parameters the network needs to learn is much lower than the other types of artificial neural networks which makes CNN less prone to overfitting and by extension generating better and general features. CNNs usually consist of three main layers of convolutional, pooling, and fully connected [48]. Generated frames in the segmentation module are modified by applying zero paddings, adding two columns and rows of zeros on them, and they are fed to a CNN. Zero padding is done to make sure that the image contains all the characters without clipping [42]. Inspired by previous works in image processing, the size of the filters in the convolution and pooling layers are chosen to be 3*3. In addition, convolutional filters of this size cover frame of each character. In the first convolutional

layer, 16 filters with the size of 3*3, followed by the ReLU activation function are applied to the input frame. The width and length of the output of this layer are the same as the input frame because of applying zero padding. However, the depth is equal to the number of applied filters, which are 16 in this layer. The next step is applying a 3*3 max-pooling layer, which takes the maximum value inside every 3*3 window. Outputs of the pooling layer are converted into a feature vector and then fed to an LSTM network to recognize what character it is showing.

3.3. Recognition

This module takes extracted features in the feature extraction module and predicts their representing character. When dealing with textual data, gaining insight into the future context is equally important as the past data. So, BLSTM [49] is chosen for this module because of its ability in both handling sequential data and two side coverage of the input. It is a type of recurrent neural network, especially LSTMs, that can scan both sides of the context by splitting the neurons of hidden layers into two parts that work independently on each side of the input. However, the next hidden layer receives the output of both mentioned groups of neurons and has information from both sides of the context [50].

The recognition module of Bina takes advantage of a deep BLSTM. The outputs of the feature extraction module are fed to the BLSTM one by one while covering the depth of each element completely. Assuming the output is an m*n matrix with d feature maps that are represented in the depth of the output. The first input to BLSTM is all the d elements with a width and length of 1. The next step is to feed all the d elements with a width of 2 and a length of 1. This process stops whenever all the data are fed to BLSTM. While the activation function of the input and output layers is tanh, the hidden layers use sigmoid. Utilized BLSTM in the Bina incorporates four hidden layers that contain 64, 128, 256, and 512 BLSTM blocks correspondingly. The output layer is selected to be connectionist temporal classification (CTC) [50] and it consists of 165 neurons to cover all the unique characters in the training set.

4. Experimental Settings and Results

In this section, experimental settings and results are reported and discussed. First, our dataset is explained, followed by an introduction to the evaluation methodologies used to measure Bina's performance and details about the implementation of the proposed system. Then, performances of various tested structures of CNN and BLSTM are reported. Finally, we compare the performance of Bina with the baseline algorithm.

It is worth mentioning that the volume of all data is 130 GB and a machine with an Intel Core i7-405U processor, 16 GB RAM, 100 GB SSD, and 500 GB HDD is used. Linux (Ubuntu) is installed as the operating system.

4.1. Dataset

Two datasets are used in this research. First, a synthetic dataset is created and used for training and testing of Bina. Another dataset of real images is also used for testing performance of Bina.

4.1.1. Synthetic Dataset

A large and massive dataset containing all the Persian and English characters, numbers and punctuation is essential for a successful OCR system. Due to the lack of such a dataset, we have created a dataset called Persian and English Contexts and Images (PECI). The dataset includes more than 4,000,000 text lines out of Persian News websites which are formal Persian language. Text lines are normalized by eliminating low-use and rare characters, which are not common in Persian texts. The final dataset includes 165 unique characters of all Persian and English alphabet, numbers, and punctuations. Also, each line involves approximately 20 Persian words and between one to four English words with an Enter at the end. The most 2000 prevalent English words are also added between Persian

words randomly. Of more than 4,000,000 text lines, 80% of lines are selected randomly for the training set, and 20% is considered as two categories of the test set. In addition, out of the 80% training set, 10% was used as a validation set. Texts were converted into images by tiff type in 150 DPI resolution. PECI was divided into two parts of training and test dataset, as follows:

1. Training dataset: About 4,000,000 text lines including 15,000,000 words and 200,000 unique words. A total of 2,000,000 images, each including 70 lines of text, are created consisting of 11 various fonts with sizes of 12, 14, and 18.
2. Test dataset: Test set was divided into two categories of only Persian words and a combination of English and Persian words. The reason is that the reference model (Tesseract) which we used does not recognize English words in the Persian context and inserts Persian numbers. However, our proposed model can recognize both Persian and English words in the Persian context. Therefore, to have a reasonable comparison between two models, we categorized the test set into two categories. In each category, 4000 lines are converted into images, in which 4600 images are generated. The fonts and sizes are similar to the training dataset.

Table 2 demonstrates generated PECI dataset.

Table 2. Details of the generated PECI dataset for the Persian OCR.

Dataset	Train Data of Persian-English	Test Data of Persian	Test Data of Persian-English
Number of Lines	4,000,000	4000	4000
Number of Images	2,000,000	4600	4600
Size	12-14-18	12-14-18	12-14-18
Fonts	1—Arial. 2—Homa. 3—Calibri. 4—Lotus. 5—Mitra. 6—Nazanin. 7—Tahoma. 8—Time New Roman. 9—Traffic. 10—Yagut. 11—Zar.		

4.1.2. Real Dataset

To evaluate the proposed model on the real image, which is not synthetic, the real images with their corresponding text were needed. However, books were written with different fonts, and most importantly, their corresponding text was not available. Nevertheless, Microsoft Office was used to type the context of different and random books, some of which included figures, and then the typed contests were printed. Afterward, we took an image of each printed paper with a cell phone Samsung A5 in an ordinary room with natural light. Since the images' original size was too large, for instance, 109/22*145/63, they were scaled to height 27/9, with different width sizes. Besides, the resolution of images is 300 dpi. Finally, the model was applied to the collected images to extract their text. Since collecting a dataset in this manner is time-consuming and costly, only 20 images were collected [51]. The utilized fonts were Times New Roman, Arial, Mitra, Nazanin, Calibri, and Tahoma, with size 14. The collected dataset was uploaded on Kaggle websites [51], and its link is available on the Data and Signal Processing (DSP) webpage [52] for student use.

Figures 4–6 show examples of the dataset for three different fonts and sizes.

Figure 4. An example of font Homa-size 12.

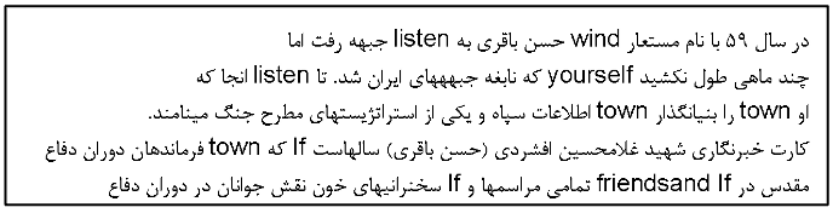

Figure 5. An example of font Nazanin-size 14.

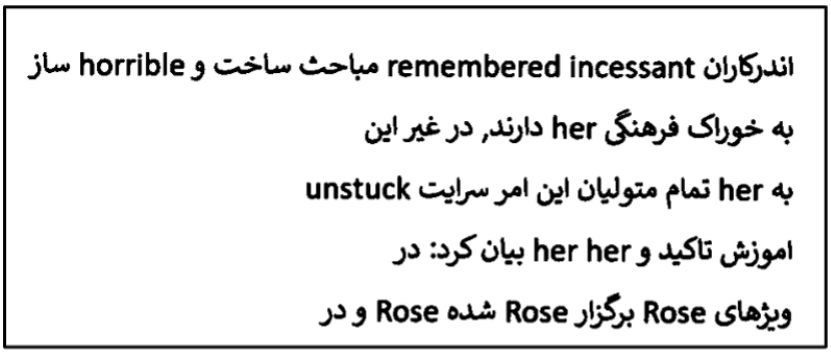

Figure 6. An example of font Calibri-size 18.

4.2. Evaluation Methodology

Accuracy and correctness are the two standard metrics used by researchers to evaluate the effectiveness of the proposed approach. Accuracy (Equation (1)) is a measure indicating correctly recognized words. A word is considered correct if all its characters are true. If one of the characters of a word is wrong, it is considered a wrong word. There are three types of wrong words: wrongly inserted words that are not in the real dataset, deleted words which are in the real dataset but are deleted wrongly, sub-words which are substitutions of the true word. Punctuations like "«" or "?" are considered as a single character.

Correctness (Equation (2)) also indicates correctly recognized words but it does not consider inserted words.

$$Accuracy = \frac{\#AllWrds - (\#InsWrd + \#DelWrd + \#SubWrd)}{\#AllWrd} * 100\,. \quad (1)$$

$$Correctness = \frac{\#AllWrds - (\#DelWrd + \#SubWrd)}{\#AllWrd} * 100\,. \quad (2)$$

In Equations (1) and (2), *AllWrds* is the number of all words in the text, *InsWrd* is the number of wrongly added words to the text. *DelWrd* is the number of wrongly deleted words from the text, and *SubWrd* is the number of wrongly replaced words in the text. We have used (Equation (1)) for accuracy of character recognition, while characters are considered instead of words.

4.3. Implementation and Computational Power

Various libraries can be used for the implementation of deep learning algorithms. We use Tensorflow to model CNN and BLSTM, and Adam optimization algorithm. The learning rate of 0.001 is used to train deep neural networks. Initial weights of all the layers are randomly selected to be a number in $[-1, 1]$ and the network is trained for 300,000 epochs. The number of input neurons of BLSTM is 16. The number of output neurons is 165 equal to the number of Persian characters, English characters, punctuations, and numbers.

4.4. Results of LSTM

Deep learning algorithms, including LSTMs and BLTMS have various parameters such as the number of hidden layers, the number of cells in each hidden layer, etc. To find the

best configuration of the used BLSTM in the recognition module of Bina, several settings of LSTM and BLSTM were tested and reported in this section. CNN setting is explained in Section 3.2, which is fixed in all the experiments.

The character "r" means backward layers, the character "f" shows forward layers, and the character "b" illustrates bidirectional layers, all of the reported results are generated by testing Bina on the Persian and English test set that involves 11 fonts with three different sizes.

Eight experiments were done to find the best structure of the recurrent neural network used in the recognition section of Bina. All the experiments are summarized in Table 3 in terms of the network structure and their performances. The eight experiments show the best performance among all the experiments. Table 4 demonstrates more details about this experiment by showing the performance of the tested network on various fonts and sizes of our dataset. Besides, Table 5 illustrates the character recognition accuracy on the best model. Additionally, the best model was also applied to the Persian-only test set. The reason is that the reference model (Tesseract) we used does not recognize English words in the Persian context and inserts Persian numbers instead of them. The model we proposed, however, recognizes both Persian and English words. Due to this reason, the best model is applied to the Persian-only test set, and reasonable comparison can be made between the proposed results and the reference results. The result of this experiment is illustrated in Table 6.

Table 3. Accuracy of experiments with different LSTMs on the Persian and English test set.

Font	1 Layer		2 Layer		4 Layer			Best Model
	512f	512b 512r,	64f, 512b,	64b, 96r, 192r,	64f, 96f, 96r, 512f,	64f, 96f, 96b, 192b,	64b, 96b, 256b, 512b,	64b, 128b,
Arial	46.56	50.78	72.63	84.13	81.39	83.38	84.27	83.36
Calibri	43.15	47.38	71.38	82.65	79.38	81.21	82.91	80.68
Homa	39.72	44.43	66.01	76.25	44.47	44.45	75.02	88.81
Lotus	39.25	44.27	72.94	90.51	77.70	89.19	92.44	93.91
Mitra	37.44	41.64	66.52	75.83	58.78	74.50	79.71	80.64
Nazanin	42.99	45.74	72.51	91.34	75.88	89.04	93.42	94.21
Tahoma	43.07	47.09	72.94	84.04	80.71	82.23	82.90	83.15
Times New Roman	41.21	44.03	72.10	82.82	80.61	82.91	84.31	81.40
Traffic	38.18	43.24	73.02	92.22	72.46	87.62	93.68	94.22
Yagut	37.96	41.41	72.14	93.69	82.43	89.49	94.33	94.55
Zar	37.49	42.03	70.02	88.00	75.30	86.35	91.20	92.43
Average Accuracy	40.46	44.75	71.10	85.59	73.55	80.94	86.74	87.94

Table 4. Deep BLSTM accuracy on Persian and English test set with four layers and 64, 128, 256 and 512 BLSTM blocks.

Font	12	14	18	Average of Fonts
Arial	92.25	94.65	63.18	83.36
Calibri	88.41	92.34	61.29	80.68
Homa	94.13	93.60	78.69	88.81
Lotus	95.43	95.50	90.79	93.91
Mitra	95.05	94.98	51.88	80.64
Nazanin	95.02	95.85	91.76	94.21
Tahoma	91.89	94.40	63.15	83.15
Times New Roman	87.95	93.41	62.83	81.40
Traffic	95.98	94.36	92.33	94.22
Yagut	95.32	94.89	93.44	94.55
Zar	93.47	93.78	90.03	92.43
Average of Accuracy	93.17	94.34	76.30	87.94

Table 5. Character level accuracy of Deep BLSTM with four layers of 64, 128, 256 and 512 BLSTM blocks on Persian and English test set.

Font	12	14	18	Average of Fonts
Arial	96.78	96.92	68.17	87.29
Calibri	96.42	96.54	68.09	87.00
Homa	92.16	97.87	96.95	95.66
Lotus	97.51	98.13	96.01	97.21
Mitra	96.66	98.01	72.39	89.00
Nazanin	96.53	98.29	96.58	97.13
Tahoma	96.72	96.90	68.34	87.32
Times New Roman	96.29	96.84	68.06	87.00
Traffic	97.79	97.00	96.58	97.12
Yagut	98.37	98.16	96.81	97.78
Zar	96.53	97.8	95.95	96.76
Average of Accuracy	96.52	97.49	83.99	92.66

Table 6. Deep BLSTM accuracy on Persian only test set with four layers and 64, 128, 256 and 512 BLSTM blocks.

Font	12	14	18	Average of Fonts
Arial	93.97	96.90	97.74	96.20
Calibri	92.21	94.16	94.81	93.73
Homa	94.92	95.23	97.53	95.89
Lotus	93.89	97.30	97.86	96.35
Mitra	94.27	97.33	97.65	96.42
Nazanin	95.85	97.62	97.73	97.07
Tahoma	92.09	96.30	97.74	95.38
Times New Roman	93.82	96.25	97.39	95.82
Traffic	92.99	96.03	97.74	95.59
Yagut	96.36	97.70	97.96	97.34
Zar	95.72	97.29	97.69	96.90
Average of Accuracy	94.19	96.55	97.44	96.06

In addition, Figure 7 demonstrates the average accuracy of different architectures in this paper.

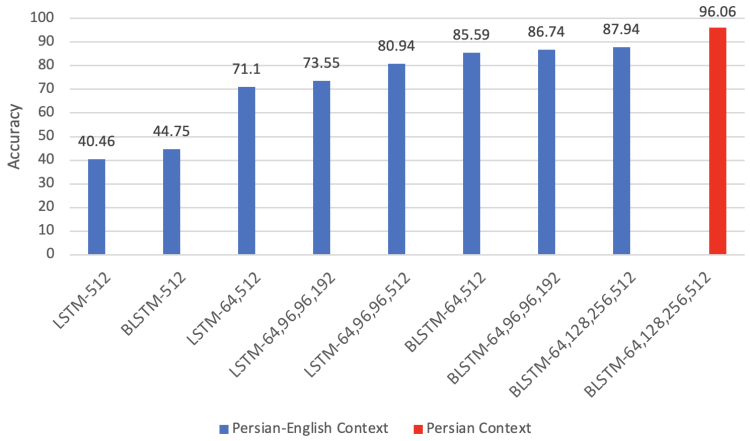

Figure 7. Average of the accuracy in different models.

Table 7 shows a confusion matrix of nearly 4680 characters of test set, font Calibri and size 14 which has low accuracy on the best model. Since the initial table contains around 86 rows, some Persian and English characters with precision and recall of 100 are eliminated. The first row shows information of characters that are deleted, inserted, or

replaced by the model (Section 4.2). The model was able to recognize the character "ت" correctly 129 times and was not able to recognize "ت" one time when it was in the labels. The character "ت" was recognized two times which was not in labels. Finally, the model could not recognize the characters correctly 4533 times.

Table 7. Confusion matrix of nearly 4680 characters of test set, font Calibri and size 14 (the characters with 100% of accuracy are not shown).

Wrongly Inserted, Deleted, and Substituted Characters	TP 0	FN 32	FP 15	TN 4618	Precision 0.0	Recall 0.0
،	23	0	0	4642	100	100
/	4	0	0	4661	100	100
:	10	0	0	4655	100	100
a	60	1	0	4604	100	98.4
c	24	1	0	4640	100	96
i	59	0	1	4605	98.3	100
k	9	0	0	4656	10	100
l	50	1	0	4614	100	98
»	1	0	0	4664	100	100
ا	445	3	0	4217	100	99.3
ب	121	1	0	4543	100	99.2
ت	129	1	2	4533	98.5	99.2
ج	31	1	0	4633	100	96.9
ح	43	2	0	4620	100	95.6
ر	234	1	0	4430	100	99.6
ز	71	1	0	4593	100	98.6
ع	44	0	3	4618	93.6	100
غ	1	1	0	4663	100	50
ق	29	0	1	4635	96.7	100
ل	69	3	0	4593	100	95.8
م	171	1	6	4487	96.6	99.4
ن	188	0	1	4476	99.5	100
ه	162	0	1	4502	99.4	100
و	187	1	1	4476	99.5	99.5
ک	63	0	1	4601	98.4	100
ی	246	0	5	4414	98	100
۰	6	0	1	4658	85.7	100
۱	4	1	1	4659	80.0	80
٤	1	1	2	4661	33.3	50
۷	1	0	1	4663	50	100

The proposed model was tested and evaluated on a dataset [51,52] of real images. The dataset contains images of different books with different fonts. Images were taken by a cell phone in an ordinary condition. The dataset also contains Persian words, punctuation, and numbers. Our model has successfully achieved the accuracy of 90.51%. The results of this experiment are reported in Table 8. Figure 8 illustrates an image of the dataset, and Figure 9 demonstrates its output. The model has difficulty to recognize "ٔ", "آ", "ء".

Table 8. Proposed model accuracy on real test set, size 14.

Font	Number of Images	Size 14
Arial	2	97.03
Calibri	1	92.38
Mitra	12	88.55
Nazanin	2	90.41
Tahoma	1	83.68
Times New Roman	2	91
Average of Accuracy	-	90.51

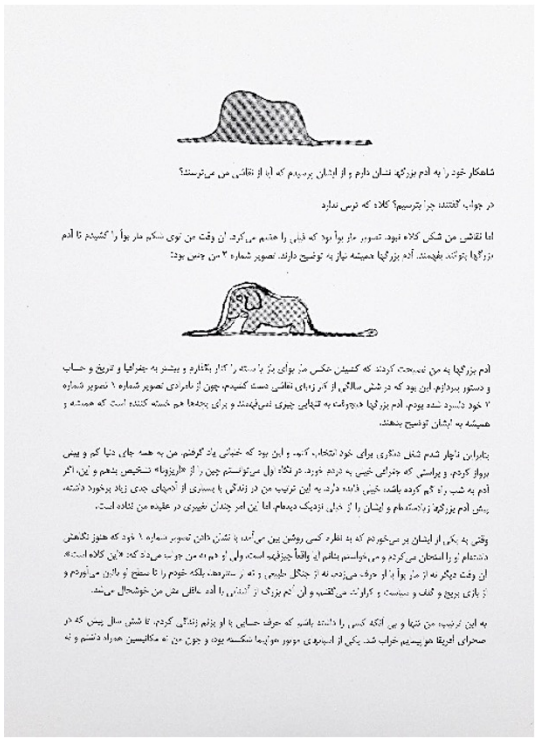

Figure 8. Testing BINA using a real image.

تاهکار خود را به ادم بزرگها نشان دارم و از ایشان پرسیدم که آیا از نقاشی من میترسند؟

در جواب قند: چرا بترسیم؟ کلاه که درس ندارد

اما نقاشی من تکل کلاه نبود. تصویر مار بوا بود که فیلی را هضم میکرد. ان وقت من توی تکم مار بوا را کشیدم تا ادم

بزرگها بتوانند بفهمند. ادم بزرگها همیشه نیاز به توضیح دارند. تصویر شماره ۲ من چنین بود:

ادم بزرگها به من نصیحت کردند که کشیدن عکس مار بوی باز یا بسته را کنار بگذارم و بیشتر به جغرافیا و تاریخ و حساب

و دستور بپردازم. این بود که در تش سالگی از کار زیبای نقاشی دست کشیدم» چون از نامرادی تصویر شماره ۱ تصویر شماره ۲ خود دلسرد شده بودم. ادم بزرگها هیچوقت به تنهایی چیزی نمیفهمند و برای بچهها هم خسته کننده است که همیشه و همیشه به ایشان توضیح بدهند.

بنابراین ناچار شدم شغل دیگری برای خود انتخاب کنم» و این بود که خلبانی یاد گرفتم. من به همه جای دنیا کم و بیش

رروواز کردم» و براستی که جغرافی خیلی به. دردم خورد. در نگاه اول می توانستم چین رااز ۷یزوا؟ نشتیس بدهم واین اگر ادم به تب راه گم کرده باشد» خیلی فایده دارد. به این ترتیب من در زندگی با بسیاری از ادمهای جدی زیاد برخورد داشتم. پیش ادم بزرگها زیادمانده ام و ایشان را از خیلی نزدیک دیده ام. اما این امر چندان تغییری در عقیده من نداده است.

وقتی به یکی از ایشان بر می خوردم که به نظرم کمی روشن بین می امده یا نشان دادن تصویر شماره ۱ خود که هنوز نگاهش داشته ام را امتحان میکردم و می خواستم بدانم ایا واقعا چیزفهم است. ولی او هم به من جواب میداد که: «این کلاه است».

ن وقت دیگر نه از مار بوا یا او حرف میزدم» نه از جنگل طبیعی و نه از ستاره هاء بلکه خودم را تا سطح او پائین میاوردم و

از بازی بریج و گلف و سیاست و کراوات می گفتم» و ان ادم بزرگ از اشنایی با ادم عاقلی مثل من خوشحال میشد.

به این ترتیب» من تنها و بی انکه کسی را داشته باشم که حرف حسابی با او بزنم زندگی کردم. تا شش سال پیش که در

صحرای اقریقا هواپیمایم خراب شد. یکی از اسبابهای موتور هواپیما شکسته بوده و چون من نه مکانیسین همراه داشتم و نه

Figure 9. Output of the Figure 8 using proposed model.

4.5. CNN Optimization

To test and find the best configuration of the used CNN in the feature extraction module of Bina, several settings of CNNs were tested on the Persian and English test set and reported in this section. The BLSTM used in the recognition module is fixed in this experiment (the best tested setting of deep BLSTM is utilized and is described in Section 4.4).

The structure of the whole feature extraction module and average accuracy is mentioned in Table 9. The best performance belongs to a CNN with 16 3*3 filters in the convolution layer followed by a 3*3 max-pooling layer.

Table 9. Different CNN setting with one layer for feature extraction with the best model deep BLSTM setting.

1st Convolution Layer	Number of Filters in Convolution Layer	Pooling Layer	Output	Average of Accuracy of All Fonts and Sizes
3,3	16	3,3	16	87.93
3,3	32	3,3	32	87.59
5,5	16	5,5	16	84.60
5,5	32	5,5	32	85.15

In addition, a CNN with two convolution layers and two pooling layers is tested. The configuration of the CNN is as follow: the size of the first convolution layer is 3,3, with 32 filters and pooling layer of size 3,3. The second convolution layer is 3,3 with 16 number of filters and a pooing layer of size 2,2. The number of outputs is 16. The network did not train after 100,000 iterations, and its accuracy was 63.63% for three sizes and 11 fonts.

5. Results of Reference Model

Tesseract model [18,40] is a model based on recurrent neural networks that use four layers of 64, 96, 96, and 192 LSTM blocks. The first, second, and fourth layers are forward, the third layer is backward, and the number of nodes in the output layer is 90. Tesseract achieved the best results in Persian OCR and because of that, it was considered as a baseline model in this work. Both test data categories are fed to the Tesseract and its performance is compared with Bina. As it is mentioned earlier, Tesseract cannot recognize English words in the Persian context and inserts Persian number instead. As a result, its accuracy on the Persian-English test set is 1.33% since the number of wrongly added words to text is high. So, the correctness evaluation is used instead of accuracy since it does not consider the added words. The results show an accuracy of about 91.61% on the Persian-only test set and the correctness of 71.65% on the Persian-English test set. More information can be found in Table 10.

Table 10. Results of reference model.

Font	Accuracy Persian Only	Accuracy Persian & English	Correctness Persian & English
Arial	94.73	2.00	69.40
Calibri	92.70	2.33	68.17
Homa	84.22	1.00	69.83
Lotus	91.55	1.00	76.35
Mitra	92.59	1.33	68.75
Nazanin	93.67	1.00	76.97
Tahoma	92.04	1.67	66.44
Times New Roman	94.39	1.33	69.17
Traffic	87.86	1.00	72.87
Yagut	94.25	1.00	75.59
Zar	91.68	1.00	74.64
Average	91.61	1.33	71.65

To thoroughly evaluate the proposed model with Tesseract, we trained the Tesseract model with our collected dataset. Since the collected dataset includes 165 unique characters, the number of neurons in the output layer is increased from 90 to 165. Table 11 shows its accuracy.

Table 11. Trained Tesseract accuracy on Persian and English test set with four layers and 64, 96, 96 and 192 LSTM blocks.

Font	12	14	18	Average of Fonts
Arial	86.89	95.32	61.97	81.39
Calibri	81.08	94.53	62.52	79.38
Homa	54.77	54.25	24.38	44.47
Lotus	83.20	81.18	68.72	77.70
Mitra	72.24	71.82	32.27	58.78
Nazanin	80.81	75.03	71.80	75.88
Tahoma	83.89	94.59	63.64	80.71
Times New Roman	83.71	95.20	62.92	80.61
Traffic	75.40	73.38	68.61	72.46
Yagut	87.56	82.89	76.84	82.43
Zar	79.27	75.32	71.32	75.30
Average of Accuracy	78.98	81.23	60.45	73.56

Figure 10 draws an analogy between Bina and Tesseract in terms of accuracy on the Persian-only test set. As it was expected, Bina outperforms Tesseract due to taking advantage of using BLSTM instead of LSTM. Figure 11 demonstrates the correctness of the two models in the Persian and English test set. Also, Figure 12 illustrates the accuracy of the trained Tesseract model and the proposed model.

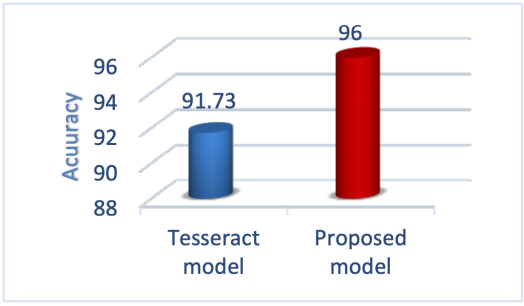

Figure 10. Average of the accuracy of the reference model and the proposed model on Persian only test set.

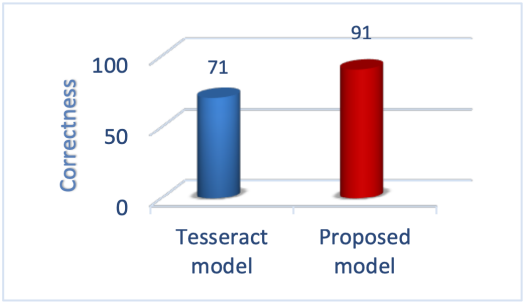

Figure 11. Average of the correctness of reference model and the proposed model on Persian and English test set.

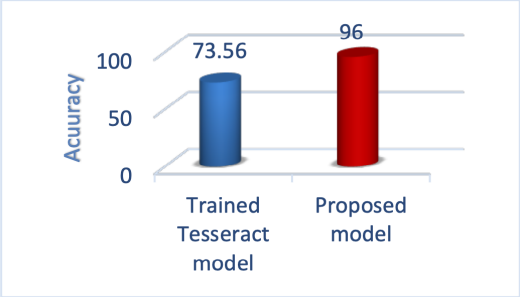

Figure 12. Average of the accuracy of trained reference model and the proposed model on Persian and English test set.

6. Discussion

As mentioned earlier, the best-achieved performance is a CNN with 16 3*3 filters in the convolution layer followed by a 3*3 max-pooling layer. Another CNN with 32 3*3 convolutional filters followed by a 3*3 pooling layer is also tested. While more feature maps result in more computational costs, the accuracy remains the same compared to the previous experiment. In two other experiments, a 5*5 convolution layer with 16 and 32 feature maps before a 5*5 max-pooling layer was tested. The obtained accuracy could have been better than the first experiments due to extracting fewer precision features by 5*5 convolutional filters. To test the effect of the number of hidden layers in CNN to the improvement of the accuracy of the final results, a CNN with two convolutional layers and two pooling layers was tested. The first convolution layer included 32 3*3 convolutional filters followed by a 3*3 max-pooling layer. Then, another 3*3 convolutional layer with 16 feature maps pursued by a 2*2 max-pooling layer was added. Its accuracy was not increased after 100,000 iterations.

In addition, several experiments of Deep BLSTM were implemented. The first experiment involved 512 LSTM blocks in one forward layer. After training the network for 3 h based on the mentioned system configuration in Section 4.4, it achieved an accuracy of 40.46%. The second one involved 512 LSTM blocks in one bidirectional layer. Training the network for 5 h led to an accuracy of 44.75%. Using a bidirectional layer rather than a forward layer resulted in about a 4% improvement in the average accuracy. The third experiment involved two layers with 64 forward and 512 backward LSTM blocks. It took 5 h to train this network and the accuracy was 71.1%. Based on the obtained results, the accuracy was increased remarkably when one hidden layer is added to the previous networks. The fourth experiment involved two layers of 64 and 512 BLSTM blocks. After training the network for 8 h, it achieved an accuracy of 85.59%. About 15% improvement compared to the third experiment is another proof that bidirectional layers are more effective in this area. The fifth experiment involved four layers of 64 forward, 96 forward, 96 backward, and 192 forward LSTM blocks. Training the network for 19 h led to an accuracy of 73.55%. Although the number of hidden layers were increased to 4 layers, the accuracy is decreased around 12% rather than the fourth experiment, which was BLSTM with two layers. The experiment indicated bidirectional blocks are more effective than the number of hidden layers in increasing accuracy. The sixth experiment involved four layers of 64, 96, 96, 512 LSTM blocks, and other settings were similar to the fifth experiment. It took 24 h to train this network and the accuracy was 80.94%. The only difference between this experiment and the previous one was the number of LSTM blocks of the last hidden layer. By comparing the results, it can be concluded that adding more LSTM blocks enhanced the network's performance by about 7%. However, the accuracy is still lower than the structure incorporating two bidirectional layers (fourth experiment). The seventh experiment involved four layers of 64, 96, 96,and 192 BLSTM blocks. After training for 36 h, it achieved an accuracy of 86.74%. The results showed an improvement of up to 13% in the average accuracy over the fifth and sixth experiments that could be interpreted as the superiority of BLSTM over LSTM in an OCR system. The eighth experiment involved four layers of 64,

128, 256, 512 BLSTM blocks while the other settings were similar to the other experiments. Training the network for 50 h led to an accuracy of 87.94%. The accuracy was increased by around 1% compared to the seventh experiment since the number of BLSTM blocks in the last hidden layer was increased to 512. Based on the all performed experiments it can be concluded that BLSTMs are much more powerful tool than LSTMs in the realm of OCR systems. In addition, using more units in the hidden layers and making the network deeper usually results in an accuracy improvement.

To find the best model, different numbers of epochs and a validation set were considered to evaluate the model's accuracy during the training. 10% of the training set was used as the validation set. It was observed, when the epoch number was low, the accuracy was low on the validation set. Moreover, when the epoch number was high, the model was not trained after a point, and the accuracy was not increased. The highest accuracy in the validation set was achieved in epoch 300,000 (Due to the large training set, the model was trained at high epochs). This process prevented the over-fitting. Figure 13 illustrates the loss on training set and validation set.

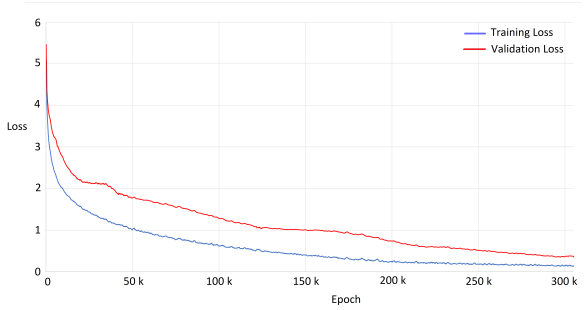

Figure 13. Loss curves for training and validation sets for the best model.

By looking at Tables 4 and 5, one can conclude that the most accurate results belong to font sizes of 12 and 14 which are the most privileged sizes in the Persian contexts. A convincible reason for our system's poor performance in font size 18, which is used as a title in the Persian context, could be the poor performance of the used box file method on segmentation of big sizes English characters when mixed with Persian scripts. In addition, some fonts, such as Homa and Tahoma, have lower accuracy than other fonts.

As it is mentioned in the Section 4.1, removed characters are rarely used in the Persian text. However, our model can handle the presence of these characters by recognizing them as trained characters. For instance, "Ω" is not in the dataset. If it appeared in the real image, the proposed model would recognize it as "(".

As Table 10 shows, when we are dealing with the Persian and English test set, accuracy cannot be an appropriate measure of determining Tesseract performance, the reason is that the dataset involves English words and Tesseract puts Persian numbers when it faces an English word which causes a low accuracy. Therefore, correctness, a measure that does not consider inserted words (words that are inserted wrongly and are not in the real dataset), is also used to measure Tesseract's performance.

According to Tables 10 and 11, after fine-tuning the Tesseract model with our data on the Persian-English training set, its accuracy has been changed from 1.33% (Table 10) to 73.56% (Table 11). The averages of the columns were added to Table 10. It means that training with more data and covering more characters are effective and could be the primary reason behind the increase in accuracy.

7. Conclusions and Future Works

In this paper, we designed a Persian OCR system, Bina, which simultaneously covers English and Persian words. A combination of CNN and deep BLSTM was used in Bina's feature extraction and recognition modules. Many experiments were carried out, and the structure of our proposed model was chosen to have one convolutional layer followed

by a max-pooling layer in the feature extraction module. In contrast, the recognition module took advantage of 64, 128, 256, and 512 Bidirectional LSTM blocks in four hidden layers. Due to the lack of a diverse dataset, we generated a huge dataset containing 2,000,000 images for the training phase and two datasets of only Persian and a combination of Persian and English contexts, each including 4600 images. The overall accuracy of the proposed model on the Persian-only test set was 96.06%, while on the Persian and English test set it was 87.94%. In the experiments in the fonts with the size of 18 and some fonts such as Homa and Tahoma, accuracy could have been higher. In future work, we will focus on the accuracy improvement of this size and various fonts.

Author Contributions: Methodology, Z.K. and E.H.; Validation, Z.K., H.V. and E.H.; Formal analysis, Z.K., H.V. and H.S.; Investigation, Z.K. and H.V.; Data curation, Z.K.; Writing—original draft, Z.K.; Writing—review & editing, Z.K., H.V., E.H. and H.S.; Visualization, E.H.; Supervision, H.V. and H.S.; Project administration, H.V.; Funding acquisition, H.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the Computer Science program in the School of Computing and Analytics at Northern Kentucky University.

Data Availability Statement: The collected dataset was uploaded on Kaggle websites [51], and its link is available on the Data and Signal Processing (DSP) webpage [52] for student use.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, P.; Peng, L.; Wen, J. Rejecting character recognition errors using CNN based confidence estimation. *Chin. J. Electron.* **2016**, *25*, 520–526. [CrossRef]
2. Saeed, S.; Naz, S.; Razzak, M.I. An application of deep learning in character recognition: An overview. In *Handbook of Deep Learning Applications*; Springer: Cham, Switzerland, 2019; pp. 53–81.
3. Smith, R. A simple and efficient skew detection algorithm via text row accumulation. In Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, Canada, 14–16 August 1995; Volume 2, pp. 1145–1148.
4. Khosravi, H.; Kabir, E. A blackboard approach towards integrated Farsi OCR system. *Int. J. Doc. Anal. Recognit. (IJDAR)* **2009**, *12*, 21–32. [CrossRef]
5. Zayene, O.; Touj, S.M.; Hennebert, J.; Ingold, R.; Amara, N.E.B. Multi-dimensional long short-term memory networks for artificial Arabic text recognition in news video. *IET Comput. Vis.* **2018**, *12*, 710–719. [CrossRef]
6. Khosrobeygi, Z.; Veisi, H.; Ahmadi, H.; Shabanian, H. A rule-based post-processing approach to improve Persian OCR performance. *Sci. Iran.* **2020**, *27*, 3019–3033. [CrossRef]
7. Malik, S.A.; Maqsood, M.; Aadil, F.; Khan, M.F. An efficient segmentation technique for Urdu optical character recognizer (OCR). In Proceedings of the Future of Information and Communication Conference, San Francisco, CA, USA, 14–15 March 2019; pp. 131–141.
8. Javed, S.T.; Hussain, S.; Maqbool, A.; Asloob, S.; Jamil, S.; Moin, H. Segmentation free nastalique urdu ocr. *World Acad. Sci. Eng. Technol.* **2010**, *46*, 456–461.
9. Bulan, O.; Kozitsky, V.; Ramesh, P.; Shreve, M. Segmentation-and annotation-free license plate recognition with deep localization and failure identification. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 2351–2363. [CrossRef]
10. Hussain, S.; Ali, S. Nastalique segmentation-based approach for Urdu OCR. *Int. J. Doc. Anal. Recognit. (IJDAR)* **2015**, *18*, 357–374. [CrossRef]
11. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
12. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef]
13. Manikonda, S.K.; Gaonkar, D.N. IDM based on image classification with CNN. *J. Eng.* **2019**, *2019*, 7256–7262. [CrossRef]
14. Gao, F.; Wang, M.; Wang, J.; Yang, E.; Zhou, H. A novel separability objective function in CNN for feature extraction of SAR images. *Chin. J. Electron.* **2019**, *28*, 423–429. [CrossRef]
15. Cheng, G.; Xin, L.; Yonghong, Y. Using highway connections to enable deep small-footprint LSTM-RNNs for speech recognition. *Chin. J. Electron.* **2019**, *28*, 107–112. [CrossRef]
16. Graves, A. Offline Arabic Handwriting Recognition with Multidimensional Recurrent Neural Networks. In *Guide to OCR for Arabic Scripts*; Märgner, V., El Abed, H., Eds.; Springer: London, UK, 2012; pp. 297–313. [CrossRef]
17. Abandah, G.A.; Graves, A.; Al-Shagoor, B.; Arabiyat, A.; Jamour, F.; Al-Taee, M. Automatic discretization of Arabic text using recurrent neural networks. *Int. J. Doc. Anal. Recognit. (IJDAR)* **2015**, *18*, 183–197. [CrossRef]
18. Smith, R. An overview of the Tesseract OCR engine. In Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), Curitiba, Brazil, 23–26 September 2007; Volume 2, pp. 629–633.

19. Sarraf, S. French word recognition through a quick survey on recurrent neural networks using long-short term memory RNN-LSTM. *arXiv* **2018**, arXiv:1804.03683.
20. Naz, S.; Umar, A.; Ahmed, A.; Razzak, M.; Rashid, S.; Sheikh, F. Urdu Nasta'liq Text Recognition Using Implicit Segmentation Based on Multi-Dimensional Long Short Term Memory Neural Networks. *SpringerPlus* **2016**, *5*, 2010. [CrossRef]
21. Ahmad, R.; Afzal, M.Z.; Rashid, S.F.; Liwicki, M.; Breuel, T. Scale and rotation invariant OCR for Pashto cursive script using MDLSTM network. In Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23–26 August 2015; pp. 1101–1105.
22. Graves, A.; Liwicki, M.; Fernández, S.; Bertolami, R.; Bunke, H.; Schmidhuber, J. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *31*, 855–868. [CrossRef]
23. Liwicki, M.; Graves, A.; Fernàndez, S.; Bunke, H.; Schmidhuber, J. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In Proceedings of the 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil, 23–26 September 2007.
24. Khosravi, H.; Kabir, E. Introducing a very large dataset of handwritten Farsi digits and a study on their varieties. *Pattern Recognit. Lett.* **2007**, *28*, 1133–1141. [CrossRef]
25. Alaei, A.; Nagabhushan, P.; Pal, U. A new dataset of Persian handwritten documents and its segmentation. In Proceedings of the 2011 7th Iranian Conference on Machine Vision and Image Processing, Tehran, Iran, 16–17 November 2011; pp. 1–5.
26. Montazeri, M.; Kiani, R. PHCWT: A Persian Handwritten Characters, Words and Text dataset. In Proceedings of the 2017 10th Iranian Conference on Machine Vision and Image Processing (MVIP), Isfahan, Iran, 22–23 November 2017; pp. 235–240.
27. Arica, N.; Yarman-Vural, F.T. An overview of character recognition focused on off-line handwriting. *IEEE Trans. Syst. Man Cybern. Part (Appl. Rev.)* **2001**, *31*, 216–233. [CrossRef]
28. Ul-Hasan, A.; Breuel, T.M. Can we build language-independent OCR using LSTM networks? In Proceedings of the 4th International Workshop on Multilingual OCR, Washington DC, USA, 24 August 2013; pp. 1–5.
29. Naseer, A.; Zafar, K. Meta features-based scale invariant OCR decision making using LSTM-RNN. *Comput. Math. Organ. Theory* **2019**, *25*, 165–183. [CrossRef]
30. Khosravi, H.; Kabir, E. Introduction of two fast and effective features for handwritten Farsi digit recognition. In Proceedings of the 4th Iranian Conference on Machine Vision and Image Processing, Mashhad, Iran, 14–15 February 2006.
31. Pourreza, M.; Derakhshan, R.; Fayyazi, H.; Sabokrou, M. Sub-word based Persian OCR using auto-encoder features and cascade classifier. In Proceedings of the 2018 9th International Symposium on Telecommunications (IST), Tehran, Iran, 17–19 December 2018; pp. 481–485.
32. Moradi, M.; Eshghi, M.; Shahbazi, K. A Trainless Recognition of Handwritten Persian/Arabic Letters using Primitive Elements. *Int. J. Comput. Appl.* **2016**, *975*, 8887. [CrossRef]
33. Alizadehashraf, B.; Roohi, S. Persian handwritten character recognition using convolutional neural network. In Proceedings of the 2017 10th Iranian Conference on Machine Vision and Image Processing (MVIP), Isfahan, Iran, 22–23 November 2017; pp. 247–251.
34. Bonyani, M.; Jahangard, S.; Daneshm, M. Persian handwritten digit, character and word recognition using deep learning. *Int. J. Doc. Anal. Recognit. (IJDAR)* **2021**, *24*, 133–143. [CrossRef]
35. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
36. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
37. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
38. Khosravi, S.; Chalechale, A. Chimp Optimization Algorithm to Optimize a Convolutional Neural Network for Recognizing Persian/Arabic Handwritten Words. *Math. Probl. Eng.* **2022**, *2022*, 4894922. [CrossRef]
39. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
40. Tesseract-Ocr. Release 5.1.0 · TESSERACT-OCR/tesseract.
41. Ghanbari, N. A review of research studies on the recognition of Farsi alphabetic and numeric characters in the last decade. In *Fundamental Research in Electrical Engineering*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 173–184.
42. Ul-Hasan, A. Generic Text Recognition Using Long Short-Term Memory Networks. Ph.D. Thesis, University of Kaiserslautern, Kaiserslautern, Germany, 2016.
43. Ul-Hasan, A.; Shafaity, F.; Liwicki, M. Curriculum learning for printed text line recognition of ligature-based scripts. In Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23–26 August 2015; pp. 1001–1005.
44. Hoseinzade, E.; Haratizadeh, S. CNNpred: CNN-based stock market prediction using a diverse set of variables. *Expert Syst. Appl.* **2019**, *129*, 273–285. [CrossRef]
45. Lee, C.Y.; Osindero, S. Recursive recurrent nets with attention modeling for ocr in the wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2231–2239.
46. Wang, F.; Guo, Q.; Lei, J.; Zhang, J. Convolutional recurrent neural networks with hidden Markov model bootstrap for scene text recognition. *IET Comput. Vis.* **2017**, *11*, 497–504. [CrossRef]

47. Yadav, M.; Purwar, R.K.; Mittal, M. Handwritten Hindi character recognition: A review. *IET Image Process.* **2018**, *12*, 1919–1933. [CrossRef]
48. Graves, A.; Schmidhuber, J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **2005**, *18*, 602–610. [CrossRef]
49. Graves, A. Supervised sequence labelling. In *Supervised Sequence Labelling with Recurrent Neural Networks*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 5–13.
50. Din, I.U.; Siddiqi, I.; Khalid, S.; Azam, T. Segmentation-free optical character recognition for printed Urdu text. *EURASIP J. Image Video Process.* **2017**, *2017*, 62. [CrossRef]
51. Khosrobeigi, Z. Real dataset for Persain OCR-V1. 2020.
52. Data and Signal Processing Laboratory. The University of Tehran. Iran.