TECHNIUM

Technium Vol. 18, pp.47-64 (2023)
ISSN: 2668-778X
www.techniumscience.com

A Deep Learning Based Offline Optical Character Recognition Model for Printed Ottoman Turkish

Ahmed AL-KHAFFAF[1], Ümit ATİLA[2]
[1] Graduate School of Natural and Applied Sciences, Gazi University, Ankara, Turkey
[2] Department of Computer Engineering, Faculty of Engineering, Gazi University, Ankara, Turkey
ahmed.n.khaffaf@gmail.com

Abstract. *Developing efficient optical character recognition (OCR) systems for printed Ottoman text is a problem since current OCR models created for Arabic have restrictions that make it difficult to be performed. The performance of these models has been shown to be low when used for the recognition of Ottoman text. It has also been shown that these models that have been subjected to specialized training on Ottoman text have produced results that are not sufficient. In this study, an analysis of printed Ottoman Turkish documents in the Matbu font is conducted using a deep learning model that is proposed. Through the use of an end-to-end trainable architecture that integrates convolutional neural networks (CNNs) with bidirectional long short-term memory (BiLSTM) units, this study proposes an efficient solution to the Ottoman optical character recognition (OCR) issue. Experimental results show that the proposed model achieved overall scores for accuracy, sensitivity, and precision of 99.6%, 87.1%, and 93.3% on the test dataset respectively.*

Keywords: *Deep Learning, CNN, OCR, BiLSTM, Printed Ottoman Characters*

1. Introduction

1.1 Overview

Academics refer to Ottoman Turkish as the written language that developed within Ottoman borders from the early fourteenth to the early twentieth centuries[1], [2]. Up until the eighteenth century, Turkish, rather than "Ottoman," was the language label used in original texts. The name "Ottoman Turkish" refers to the written and spoken forms of the Turkish language that were used by the Ottoman Empire in Anatolia and the Balkans beginning in the late thirteenth century [3].

The Ottoman script is a cursive writing style characterized by the practice of writing characters in a right-to-left direction and connecting them to form a continuous flow of text. The Ottoman alphabet, derived from the Ottoman script, comprises a total of 34 characters. The Ottoman characters exhibit different forms based on their position inside a word, namely initial, medial, final, or isolated, as illustrated in Figure (1). In specific instances, particular characters may exhibit similar bodily forms while varying in the arrangement of dots [4, 5].

Technium Vol. 18, pp.47-64 (2023)
ISSN: 2668-778X
www.techniumscience.com

Isolated	Final	Medial	Initial
ا	ـا	—	
ء	—		
ب	ـب	ـبـ	بـ
پ	ـپ	ـپـ	پـ
ت	ـت	ـتـ	تـ
ث	ـث	ـثـ	ثـ
ج	ـج	ـجـ	جـ
چ	ـچ	ـچـ	چـ
ح	ـح	ـحـ	حـ
خ	ـخ	ـخـ	خـ
د	ـد	—	
ذ	ـذ	—	

Isolated	Final	Medial	Initial
ر	ـر	—	
ز	ـز	—	
ژ	ـژ	—	
س	ـس	ـسـ	سـ
ش	ـش	ـشـ	شـ
ص	ـص	ـصـ	صـ
ض	ـض	ـضـ	ضـ
ط	ـط	ـطـ	طـ
ظ	ـظ	ـظـ	ظـ
ع	ـع	ـعـ	عـ
غ	ـغ	ـغـ	غـ

Isolated	Final	Medial	Initial
ف	ـف	ـفـ	فـ
ق	ـق	ـقـ	قـ
ك	ـك	ـكـ	كـ
گ	ـگ	ـگـ	گـ
ڭ	ـڭ	ـڭـ	ڭـ
ل	ـل	ـلـ	لـ
م	ـم	ـمـ	مـ
ن	ـن	ـنـ	نـ
و	ـو	—	
ه	ـه	ـهـ	هـ
ى	ـى	ـيـ	يـ

Figure 1: The Ottoman Alphabet [5].

The primary objective of an Optical Character Recognition (OCR) system is to extract individual characters from a document image and transform them into a format that can be comprehended by computers. This is achieved through the utilization of pattern recognition techniques and machine learning algorithms, enabling the system to interpret and process textual information included in various types of documents [6]. However, it is imperative to acknowledge that OCR systems may face several problems, particularly when confronted with low-quality photos and atypical fonts. The accuracy of the OCR process can be influenced by various factors, including image resolution, lighting conditions, and noise levels. Hence, it is imperative to engage in ongoing research and make progress in OCR technology in order to effectively tackle these issues and enhance its overall performance [7]–[9].

1.2 Literature Review

This section presents a brief summary of research conducted on the identification and comprehension of printed texts, with a specific focus on studies relevant to the current study. In order to adopt a thorough methodology, our investigation has integrated research that specifically examines Ottoman alphabet-based scripts that are pertinent to our study. Our research primarily centers on the Matbu font type, and therefore, we exclusively report findings pertaining to the recognition of Matbu and Matbu-like typefaces.

Bilgin introduced a Deep Learning (DL) based character recognition system designed specifically for printed Ottoman script. The researchers begin by creating a synthetic text image dataset using a text corpus and enhancing it through various image processing techniques. They then develop a hybrid model combining Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) to recognize characters, training it using both the original and augmented datasets. Additionally, they employ transfer learning to adapt the system to real image data. The proposed system achieves a Character Error Rate (CER) of 0.11 on synthetic data and 0.16 on real data, which consists of line images extracted from a printed historical Ottoman book. Additionally, the study assesses the usability of two Arabic datasets for recognizing Ottoman documents. The results achieved on standard datasets (APTI and P-KHATT) and Ottoman documents are comparable to existing literature. The research offers a practical solution for handling Ottoman documents and demonstrates the significance of augmentation and language considerations in the recognition process [11]. Tasdemir et al., introduced an automatic transcription system designed for printed Ottoman documents. The system utilizes a CNN-BiLSTM-CTC network architecture and explores various decoding strategies for network output. In testing, the system achieved a 3.68% CER and a 16.61% Word Error Rate (WER) on a 1.4K line image test set, employing the Word Beam Search (WBS) decoder along with a 260K-word lexicon. The study demonstrated that using a recognition lexicon to restrict transcription output improves accuracy, but this approach is not optimal for the agglutinative nature of the Turkish language [12].

Dölek et. al., introduced an OCR tool specifically designed for Ottoman documents in the printed nasikh font. This tool was developed as part of a project called "End-to-End Conversion of Ottoman Documents to Modern Turkish." The OCR tool utilizes deep learning LSTM models trained on a dataset that includes both original and synthetic documents. To evaluate the performance of the tool, an experimental comparison was conducted with five other well-

known OCR tools or models. The test set consisted of 21 pages from various Ottoman documents. Osmanlica.com, the OCR tool presented in the study, outperformed the other tools significantly, achieving an 88.64% raw character recognition accuracy, 95.92% normalized character recognition accuracy, and 97.18% joined character recognition accuracy. Moreover, Osmanlica.com achieved a word recognition accuracy rate of 58%, which was the only rate surpassing 50% among all the compared tools [4]. Dölek et. al., presented a web-based OCR system that uses CNN and Recurrent Neural Network (RNN) based deep neural network models to convert images of Ottoman documents printed with naskh font into editable text. Three datasets were used for training: original (1,000 pages), synthetic (23,000 pages), and a hybrid set containing both. The trained models were compared with other OCR models like Tesseract, Google Docs, Abby FineReader, and Miletos OCR using a 21-page test set and three different texts (raw, normalized, and joined) based on character, ligature, and word recognition criteria. The results showed that the Osmanlica.com Hybrid model outperformed the others, achieving 88.86% raw, 96.12% normalized, and 97.37% joined accuracy in character recognition; 80.48% raw, 91.60% normalized, and 97.37% joined accuracy in ligature recognition, and 44.08% raw and 66.45% normalized accuracy in word recognition. To better understand the impact of the alphabet's characteristics on OCR, the study conducted character, ligature, and word frequency analyses of Ottoman. The characters were grouped based on distinctive features such as connectedness, letter body, position and number of dots, type of character, and source language. Frequencies and recognition accuracies were examined for each group. Additionally, OCR results were reported for each individual character [13].

The authors in [14] introduced the CNN-based Ottoman Document Analyzer project to help historical document researchers learn a new alphabet and understand Ottoman Turkish documents. The platform's tool lets users choose an Ottoman document to translate and rectify it with perspective transformation for further picture processing. A CNN recognizes document characters after dividing the text into lines, words, and characters. The Arabic alphabet and spelling regulations prevent numerous words from using certain characters. Thus, words must be ordered. This procedure uses Zemberek natural language processing plugin to recommend terms that match the content. Users can choose from the Zemberek plugin's suggested words or insert their own. Thus, the algorithm separates lines 97% of the time and words 96% of the time. Additionally, properly separated characters are classified 88.47% correctly. The authors in [15] suggested a trainable CRNN architecture with CNN, LSTM and CTC layers. They used a test set of 21 unique documents to compare their model against the Tesseract Arabic, Tesseract Persian, Abby Finereader, Miletos, and Google Docs OCR tools and models. Character recognition accuracy for their Hybrid model is significantly higher than that of competing models at 88.86% for raw text, 96.12% for normalized text, and 97.37% for joined text. On normalized text, the hybrid model is the only one to achieve word recognition accuracy of 58%.

1.3 Problem statement and Contribution

According to the literature, suggesting systems for distinguishing Arabic characters is considered a challenge and there is a lack in providing standard approaches for this purpose. Therefore, the objective of this study is to recognize the printed Matbu font in Ottoman letters included in documents and subsequently convert them into editable text by introducing an innovative solution to tackle the issue of Ottoman OCR by suggesting a comprehensive trainable framework that integrates CNNs with Bidirectional Long Sort-Term Memory (BiLSTM) units. After the network was set up and ready to be trained, we used the Kaggle website's dataset[10] for printed Ottoman Turkish. The dataset was augmented using augmentation to increase its size and improve the quality of the results. Once the system was prepared for the test phase by completing the training process and achieving high training accuracy, the test phase was applied to a large variety of images.

The remaing sections of this paper is organized as follows: Section 2 describes the proposed system and the dataset used in this study. Section 3 presents the experimental results and discussions about them. Section 4 concludes this paper and provides future directions of the current research.

2. Research Method

2.1. Optical Character Recognition

The OCR refers to the computational procedure of identifying and classifying characters within images through the utilization of image processing and machine learning methodologies. Historically, OCR has been executed by a series of five sequential steps: The academic terms for the mentioned steps are as follows: (i) Preprocessing of images (ii) Segmentation of images (iii) Extraction of features (iv) Recognition of images (v) Post-processing and error correction. In the context of deep neural networks, the processes of feature extraction and recognition are executed simultaneously.

The OCR system's overall flow is as follow:

1. Gather the papers in order to make page image files (page-images).
2. Create the ground truth page text files (page-texts) for the collection's image files.
3. Page data to line data set conversion: [line-texts, line-images] = create-line-data-set (page-texts, page-images)
4. Use the line data set to train the network.

The initial stage was the hand compilation of a diverse array of genuine Ottoman papers into a cohesive collection of pages. The ground truth text for each page within the collected document set is generated during the second stage. Each page image is accompanied by a corresponding page text file that contains the accurate representation of the text found on that page. The page-text files have been generated using semi-automatic processes. The initial pages undergo OCR using a pre-existing tool such as Tesseract OCR. The results generated by the OCR system are thereafter subjected to a human verification and correction process carried out by editors. The page data set, which includes pairings of page images and page texts, is transformed into a line data set. This transformation is achieved by utilizing the create-line-data-set function. The function receives input as a set of page-images files and their matching page-texts files. The photos undergo Otsu thresholding to transform them into binary format, followed by segmentation into lines using OpenCV. The manual segmentation process involved editors performing the task of segmenting page text into lines.

2.2. Dataset

In this study, we use a public dataset comprising a total of 3894 characters [20] categorized into 44 classes. Among these classes, 33 of them are used to label letters in Ottoman Turkish, 10 are used to label digits, and 1 is used to label the character "lamelif". Characters in the dataset belong to 4 different formats. Accordingly, 1371 characters are in Talik format, 411 characters are in Rika format, 1974 characters are in Matbu format, and there are 138 characters that are a combination of other formats. The dimension of each image is 32x32 pixels presented in a binary format. Figure 2 represents sample images from used dataset.

Figure 2: Sample images from used dataset.

As a result of the checks carried out on the dataset, the number of images was reduced to 3520 after some images that might cause problems were removed. Of the 3520 images in the dataset, 80% are divided into the training set and 20% into the test set. Accordingly, the training set consists of 2816 images and the test set consists of 704 images.

On the other hand, data augmentation is a method employed to expand the training set by generating modified replicas of a dataset through the utilization of preexisting data [21]. This process encompasses the implementation of slight modifications to the dataset or the utilization of deep learning techniques to generate novel data instances. Augmentation is performed in response to the limited quantity of data within the dataset to expand the variety of training images to achieve the desired level of accuracy in the OCR system. Several transformations including rotation, width shift, height shift, shear and zoom is applied to create different variations of training images. Consequently, the augmentation process applied on 2816 training images yielded a total of 44982 images in the training dataset. The effect of rotation with range 10 is shown in Figure 3 (b-c). The resulting images obtained from the transformations are reduced in size by a factor of 0.7 as shown in Figure 3(d).

(a) original letter "ع". b) clockwise rotation. c) counterclockwise rotation. (d) scaling with factor 0.7

Technium Vol. 18, pp.47-64 (2023)
ISSN: 2668-778X
www.techniumscience.com

Figure 3: Transformed images with data augmentation.

While the deep neural network is trained with 44982 images in the training dataset, it is tested using 704 images in the test dataset. The final organization of the dataset is given in Table 1.

Table 1: Organization of the dataset.

Number of samples	3520
Training set	2816
Augmented dataset	44982
Test set	704
Color	Binary
Resolution	32x32 pixels

2.3. Proposed Deep Learning Model

Deep neural network models, such as CNN and RNN, have demonstrated considerable potential in achieving favorable outcomes in the domain of text recognition problems [16], [17]. The CNN architectures are extensively employed in object recognition applications, demonstrating notable performance in character identification. One notable limitation of CNN designs pertains to their limited proficiency in accurately recognizing sequential patterns, particularly in the context of object sequences such as character sequences within textual data. In contrast to CNNs, RNNs demonstrate a notable proficiency in handling sequences that are highly prevalent within natural language processing applications. While RNNs exhibit superior performance compared to CNNs in the domain of sequence recognition, their efficacy diminishes notably when confronted with lengthy sequences. In alternative terms, individuals often have a tendency to overlook or experience difficulty in the process of encoding contextual information inside extended sequences. The limitations of RNNs are addressed by a modified variant known as LSTM networks. LSTMs which possess the capability to retain and utilize relevant information through the utilization of memory blocks, have the ability to effectively handle sequences of varying lengths [18]. Nevertheless, the utilization of these elements inside the feature extraction layer of a deep neural network design leads to an increase in the dimensionality of the feature space. Moreover, these models assign equal importance to various features.

In order to address these challenges, we suggest the implementation of a CNN and BiLSTM framework [19]. This design has a BiLSTM layer, which enables the extraction of contextual information from both preceding and succeeding sequences. This is achieved by connecting two hidden layers with opposite directions to a shared context. The utilization of the group-wise enhancement mechanism has been applied to the features that have been extracted by bidirectional layers. This mechanism involves the division of characteristics into different classes, with the objective of strengthening the significant traits within each group and simultaneously reducing the prominence of the less important ones. The strategy provided in this study utilizes convolution and pooling layers to extract high-level features and minimize the dimensionality of the feature space. A diagram for the proposed CNN-BiLSTM architecture for Ottoman Turkish OCR task is shown in Figure 4 and Table 2 shows detailed information for each layer of the model.

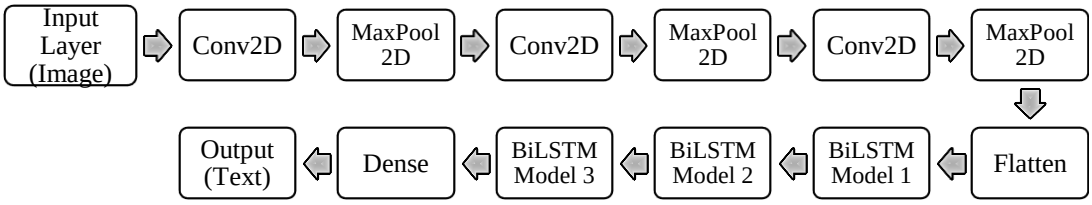


Figure 4: Proposed CNN-BiLSTM model.

The initial layer is a 2D convolution layer (Conv2D) employing 32 filters with a kernel size of 3x3 to the input data. To ensure consistent output dimensions, a padding setting of "same" is applied. The next two subsequent convolutions layers exhibit similarities to the initial convolutional layer, with the sole distinction being the alteration in the filter size to 64 and 128 respectively. Additionally, MaxPooling2D layers with a pooling size of 2×2 are utilized alongside the convolution layers. This pooling operation aids in simplifying the information obtained from the preceding convolutional layer by reducing the height and width of the feature maps by half. The activation function

used in all 2D convolution layers is ReLU (Rectified Linear Unit), which introduces non-linearity into the model. The next layer is the flatten layer which converts the 3D feature maps from the previous layer into a 2D tensor. This is necessary before feeding the data into the BiLSTM network layers. This layer does not introduce any new parameters.

After the flatten layer, the architecture has three consecutive BiLSTM layers. First BiLSTM layer processes the flattened input data in both forward and backward directions. Each direction has 128 units, meaning there are a total of 256 units in the layer. All the units of this layer are processed along the y-axis of the input data. The second BiLSTM layer also processes the data in both directions but the units in the forward direction are processed along the x-axis of the input data. The backward direction remains y-axis processing. Each direction has 64 units, for a total of 128 units in the layer. The final BiLSTM layer has the same configuration as the previous layer with x-axis processing for the backward direction. This layer has 32 units per direction meaning there are a total of 64 units. The final layer of the model is a fully connected (Dense) layer with a Softmax activation function which serves as the output layer with a total size of 44 units. This output layer projects the output of the last LSTM layer onto a 44-dimensional vector, where each dimension corresponds to one of the 44 classes in the classification problem. In this layer, 34 units are used for Ottoman letters and 10 units for Ottoman numerals.

Table 2: Proposed model summary.

LAYER NUM.	LAYER TYPE	SIZE	KERNEL SIZE	PADDING	ACTIVATION
0	Input	32x32x1	-	-	-
1	Conv2D	32x32x32	3x3	same	RELU
2	MaxPooling2D	16x16x32	2x2	-	-
3	Conv2D	16x16x64	3x3	same	RELU
4	MaxPooling2D	8x8x64	2x2	-	-
5	Conv2D	8x8x128	3x3	same	RELU
6	MaxPooling2D	4x4x128	2x2	-	-
7	Flatten	2048	-	-	RELU
8	BiLSTM	256	-	-	-
9	BiLSTM	128	-	-	-
10	BiLSTM	64	-	-	-
11	Output	44	-	-	Softmax

The proposed network that consists of 905,632 parameters was trained 100 epochs using Adam optimizer. The model was trained with the 90% of training data and validated with the rest. The best value for learning rate was determined as 0.0001 by trial and error method. Shuffling was performed on training data so that the model was able to use different images at each epoch and prevent overfitting. During the training process of the proposed model, the target vectors were created using one-hot-encoding method to identify the true class of each character image. In the feed forward calculation of the network, error rates relative to target vectors were calculated by using categorical cross-entropy function. In current distributed deep learning networks, the batch size parameter is regarded as a crucial hyper-parameter that necessitates adjustment in training of CNN. Batch size determines the number of images to be handled by the model in each iteration. Multiple experimental findings indicate that the appropriate batch size is mostly influenced by the number of images utilized during training, the number of repetitions, and the hardware employed. In our experiments the batch size is defined as 64. In order to avoid overfitting problems, model weights belonging to the epoch in which the best classification accuracy was achieved during training were selected for the testing process.

2.4. Character Segmentation from Ottoman Paper Images for Creating Test Dataset

Once the system has been trained and the necessary weights have been obtained, it undergoes a diagnostic process by conducting a test. This involves a series of stages applied to the input images within the system to process the test and present the most optimal outcomes. The input images undergo preprocessing procedures aimed at enhancing their

visual quality, which involves the removal of noise and the correction of mistakes. The significance of image quality is consistently paramount in object identification systems. The detection rate of an image is positively correlated with its picture quality, thus images with higher picture quality exhibit a greater detection rate compared to unprocessed noisy images. Consequently, the process of extracting features from unprocessed images poses challenges due to its impact on the efficiency of OCR [22].

To ensure a fair and transparent evaluation of the system, a diverse range of Ottoman manuscripts and documents written in Matbu script were gathered from multiple sources for the purpose of picture collection. The collected dataset comprises of around 26 pages, 396 lines, 4912 words, and 22450 characters gathered from several source documents. The dataset's lines are roughly 1349 x 120 pixels in size and have a resolution of 96 dpi.

In order to include high-quality images into the framework, a preprocessing step was conducted on all acquired images to rectify any anomalies. The illustration illustrates the workflow for preparing picture collections. The images pre-processing phases were illustrated and explained in Figure 5:

Figure 5: Workflow of preprocessing images datasets[23].

2.4.1 Region of Interest (ROI)

The ROI describes a region or subset of an image or dataset that is of particular interest for processing, analysis, or study [24]. Image processing, computer vision, medical imaging, and remote sensing are just a few examples of fields that regularly employ this technique to zero in on the data that matters most. Determining a return-on-investment ROI is a vital notion for effective and targeted data analysis and interpretation since it allows researchers and professionals to extract useful information, improve the precision of algorithms, and lessen computational load. At this point, the margins around the text are removed so that it may be handled independently and its features can be extracted in an effective and appropriate manner, as shown in Figure 6.

(a) Original Image. (b) Extracted ROI

Figure 6: Image region of interest for processing.

2.4.2 Binarization

The procedure of binarization is commonly employed as the initial step in the segmentation process[25]. In the first step, all photos were transformed into grayscale format. In order to convert pixel values that exhibit different shades of gray to binary values represented by 1's and 0's, the utilization of a threshold value is implemented [26]. The threshold binarization approach can be elucidated in the following manner:

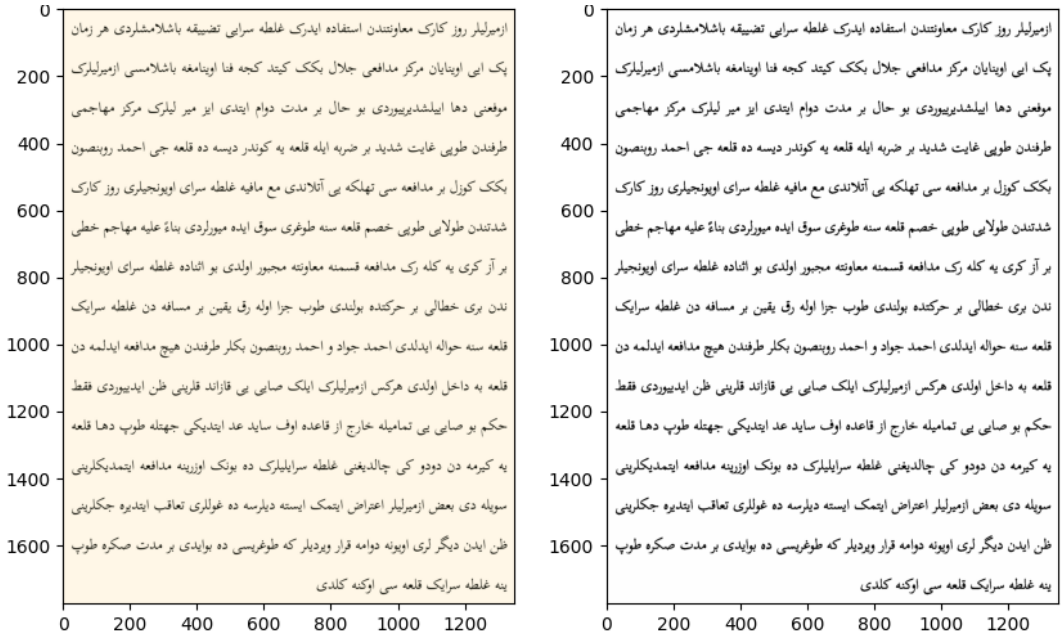
$$F(x,y) = \begin{cases} 1, & if\ f(x,y) \geq T \\ 0, & otherwise. \end{cases} \quad (1)$$

Technium Vol. 18, pp.47-64 (2023)
ISSN: 2668-778X
www.techniumscience.com

where T is the overall value of the threshold (Converting a Grayscale Image to a Binary Image Using Thresholding). The binary representation of the image shown in Figure 7.

2.4.3. Segmentation of Lines

The process of segmenting textual images into lines, words, and individual characters is a critical and formidable task within an OCR system. In the context of writings written in the Latin alphabet, it is common for relationships between characters to primarily manifest in Ottoman forms [27]. Nevertheless, when it comes to Ottoman records, it is worth noting that even machine-printed texts have the potential to have interwoven scripts. The segmentation of individual characters in an Ottoman script presents a considerable difficulty, and any error in this process might detrimentally affect the overall efficacy of the recognition system.



(a) Grayscale image (b) Binary image

Figure (7): Grayscale and binary representations of the image.

In the binary image, black pixels are represented by the value 1, while white pixels are denoted by the value 0. Higher peaks in the image correspond to rows with a larger number of black pixels and a smaller number of white pixels. This is due to the representation of black pixels as binary value 1, and white pixels as binary value 0. According to the peak points seen in Figure 8, it may be inferred that the document illustrated in Figure 8(a) consists of a total of 19 lines. This phenomenon becomes apparent upon examination of the figure.

Let the picture $\beta[i,j]$ be a binary one, and its horizontal projection can be explained as follows:

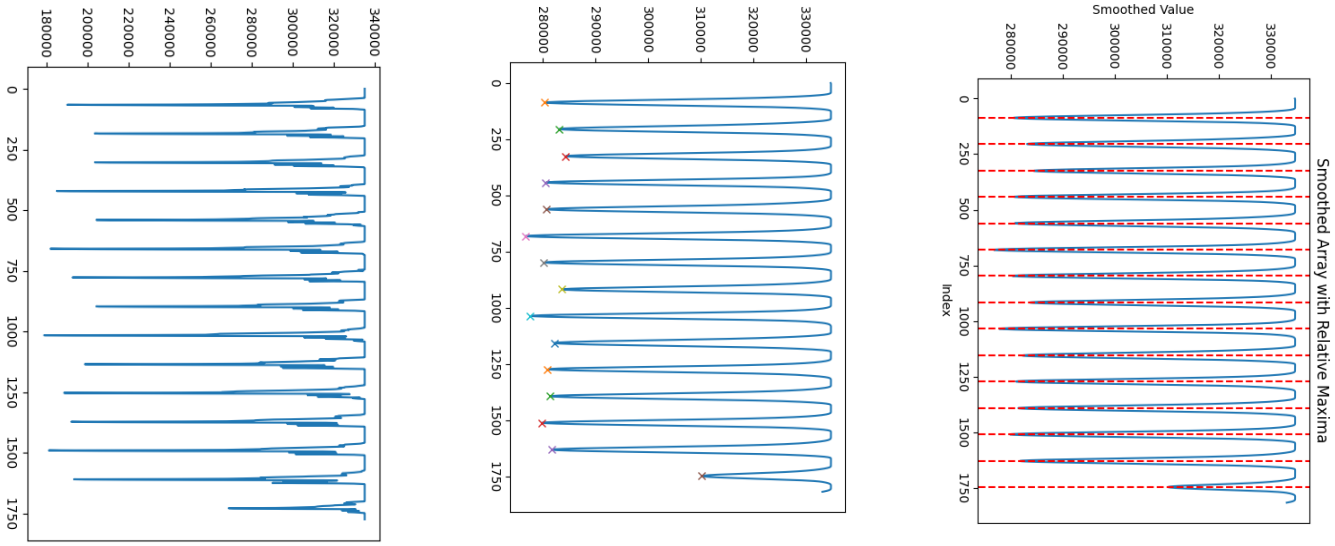
$$\Gamma[i] = \sum_j \beta[i,j] \qquad (2)$$

$$\Omega[i] = \begin{cases} 1, & if\ \Gamma[i] \geq T \\ 0, & otherwise. \end{cases} \qquad (3)$$

The assignment of values to each row is determined by comparing the value of Γ[i] to a threshold T. Rows that include text are assigned the value 1, whilst rows without text are assigned the value 0. Subsequently, the alterations in the values of individual elements inside the set Ω[i], specifically the transition from 0 to 1 and the transition from 1 to 0, are emphasized. As a result of this, the lines are extracted from the original image based on the rows that were previously indicated, as depicted in Figure 8.

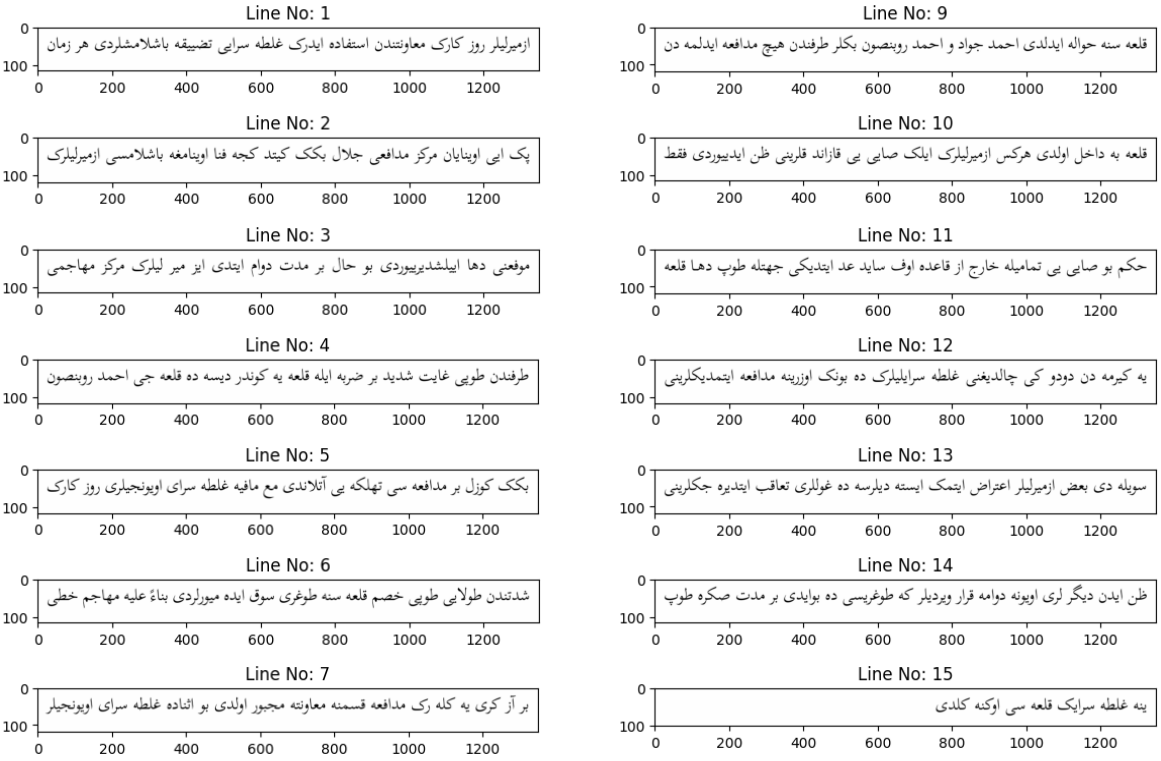
Technium Vol. 18, pp.47-64 (2023)
ISSN: 2668-778X
www.techniumscience.com

Following the initial step of segmenting the image into individual lines, each line is subsequently processed independently. Figure 9 illustrate the line extraction. Then, thorough examination of the spaces between the words within a given line is conducted, with the aim of determining a representative average for each line. This enables the line to be further segmented into individual words by leveraging the largest inter-word spaces, thereby dividing them into multiple components. The resulting segmented words are then outputted in a divided format, resembling an image. Figure 10 represent the visual concept of cutting line into words.



(a) (b) (c)

Figure 8: Line segmentation.



Technium Vol. 18, pp.47-64 (2023)
ISSN: 2668-778X
www.techniumscience.com

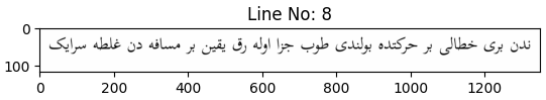
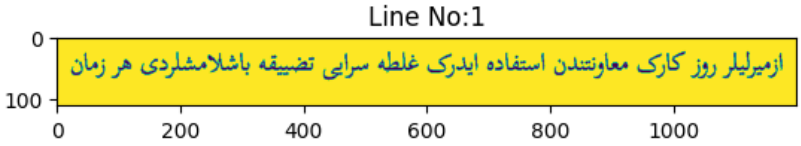
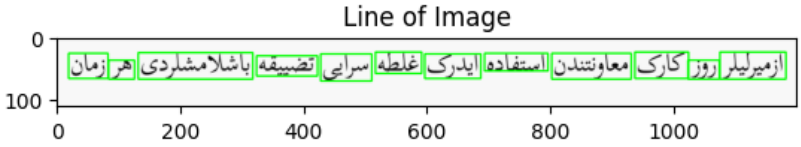


Figure 9: Extracting Lines from the image.



(a) The line that was extracted from the original image



(b) Visual concept of cutting line into words

(c) Individual words after extraction

Figure 10: Extracting and cutting the line form image.

2.4.4. Segmentation of Characters

Once the process of line segmentation has been completed, character segmentation is carried out on the retrieved lines. In instances where texts are generated by machines, certain characters may be written in a connected manner due to the distinctive characteristics of the Ottoman alphabet. Hence, in contrast to line segmentation, character segmentation is a complex task.

Initially, the line undergoes a process of segmentation, wherein it is divided into its constituent characters and groups of characters that are connected by lines. This segmentation approach employs vertical projections, similar to the method employed in line segmentation. There are several methods by which vertical projection can be defined.

$$\Psi[i] = \sum_j \beta[i,j] \qquad (4)$$

Figure 12 illustrates the vertical projection of Figure 11, and displays some of the segmented character groups that are both independent and interrelated.

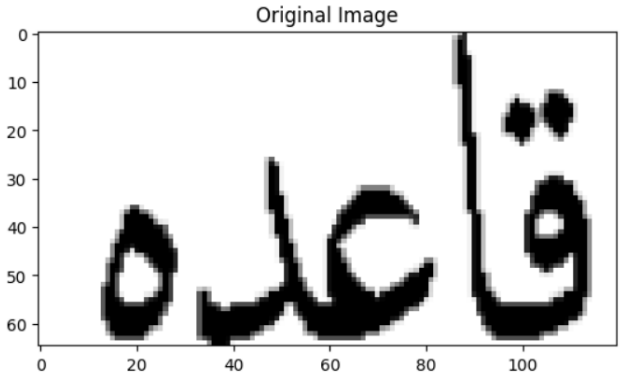


Figure 11: Original image of a chracter group.

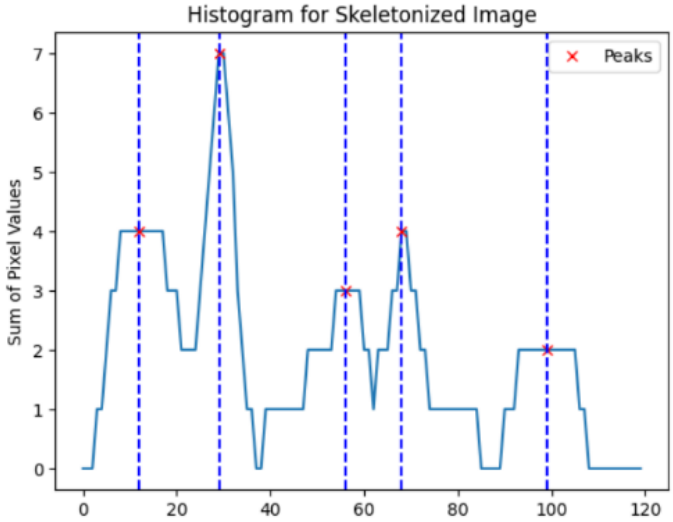
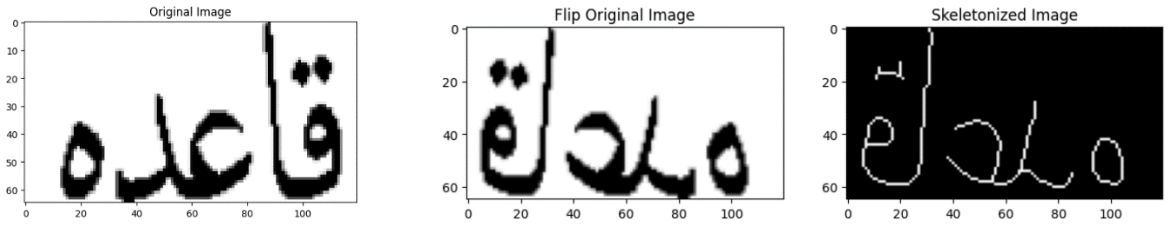


Figure 12: Isolated characters of the word from image.

The initial step is the application of a skeletonization method to the binary image, followed by the segmentation of interconnected characters. The objective of this particular stage is to segregate the shape component that serves as a representation of the overall form of the characters.

In Figure 13 (b), the skeletonized image displays a connection between two characters by a slender line. This line aligns with the columns that possess a value of 1 in the vertical projection depicted in Figure 13 (c). The relationship between the characters can be observed by referring to Figure 13 (b).

Subsequently, the local minima on the vertical projection plot are employed to identify the sites of intersection between the columns with a value of 1 and those specific places. The point of intersection that establishes the link between two characters depicted in the image will match to one of these points. Consequently, the image is partitioned into distinct portions by utilizing the points of intersection, followed by the selection of the most relevant segmentation from these sections. A contiguous set of characters, as depicted in Figure 14, can be partitioned starting at the second point of intersection.



(a) (b) (c)

Figure 13: An illustration of interconnected characters in the context of segmentation. (a) original image (b) image after skeletonization (c) projection of a skeletonized image vertically.

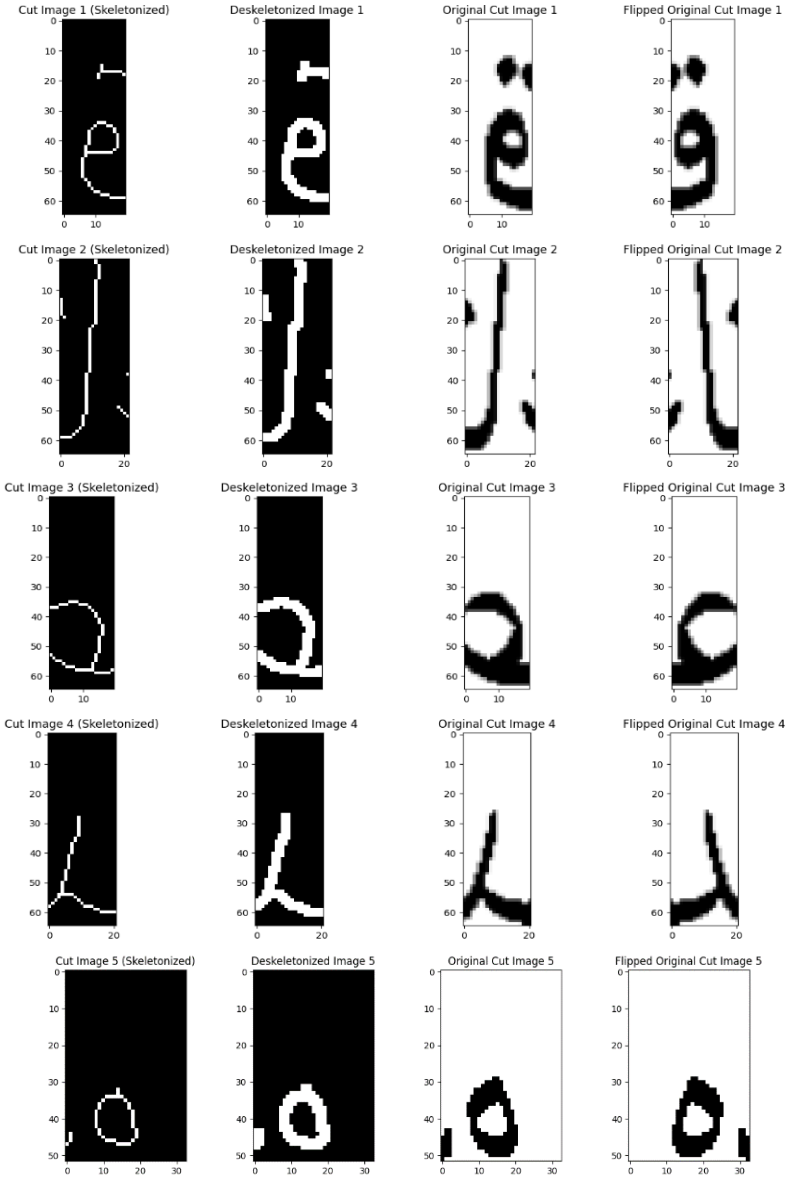


Figure 14: Character segmentation of connected individuals.

The letter and its associated dots are treated as a single entity for the purposes of identification and organization. After that, a low pass filter is used to resize it into an array using bilinear interpolation. By default, the filter has a dimension of 3 by 3. Finally, the variables are transformed into the range [0,1].

3. Results and Discussion

This section presents the experimental results and discussion conducted to assess the veracity of the proposed OCR model. All models used in this study are compiled with GPU support. All experimental studies were conducted in Google cloud environment and codes were realized with Keras framework; an open source-deep neural network library written in Python language.

3.1. Performance Metrics

To assess the effectiveness of the proposed model, the performance efficiency was evaluated using the different metrics such as accuracy, sensitivity, and precision defined in Eq (5-7). Since this study solves a multi-class classification problem, average calculations of the given performance metrics are required as given in Eq (8-10). While

accuracy defines the ratio of correctly classified samples out of all samples, sensitivity is the ratio of correctly classified positives out of all true positives and precision is the ratio of correctly classified positives out of all positives.

For a class k,

$$Acc(k) = \frac{\#TP(k)+\#TN(k)}{\#TP(k)+\#FN(k)+\#TN(k)+\#FP(k)} \quad (5)$$

$$Sen(k) = \frac{\#TP(k)}{\#TP(k)+\#FN(k)} \quad (6)$$

$$Pre(k) = \frac{\#TP(k)}{\#TP(k)+\#FP(k)} \quad (7)$$

$$Average\ Acc = \frac{1}{\#classes} \sum_{k=1}^{\#classes} Acc(k) \quad (8)$$

$$Average\ Sen = \frac{1}{\#classes} \sum_{k=1}^{\#classes} Sen(k) \quad (9)$$

$$Average\ Pre = \frac{1}{\#classes} \sum_{k=1}^{\#classes} Pre(k) \quad (10)$$

3.2. Experimental Results

The measure of accuracy is determined by the training data, but the validation accuracy (Val_Acc) is determined by the validation data. Relying on the Val_Acc metric is recommended for obtaining an accurate representation of model performance. This is due to the fact that a proficient neural network would inevitably achieve a perfect fit to the training data. However, the Val_Acc metric provides insights into the model's performance on unfamiliar data, i.e., data that has not been encountered during the training phase, and may yield suboptimal results in such cases. During the process of deep learning, a neural network endeavors to minimize a metric referred to as the loss (Loss). This metric quantifies the discrepancy between the actual data and the predictions generated by the network. To minimize the distance, the neural network will acquire the ability to adjust its weights and biases, hence reducing the loss. The distinction between Loss and validation loss (Val_Loss) lies in their respective applications to the training set and the test set. Consequently, the latter serves as a strong indicator of the model's performance on unseen data. The accuracy/loss curves obtained in the training and validation sets were presented in Figure 15. Overfitting/ Overfitting is a phenomenon that arises when a model excessively conforms to the training data, resulting in a continuous decrease in Loss while the Val_Loss remains stagnant or increases. Therefore, in attempting to mitigate overfitting, it is advisable to prioritize the Val_Loss metric. As seen in accuracy/loss curves there was no discernible evidence suggesting that the model is experiencing overfitting.

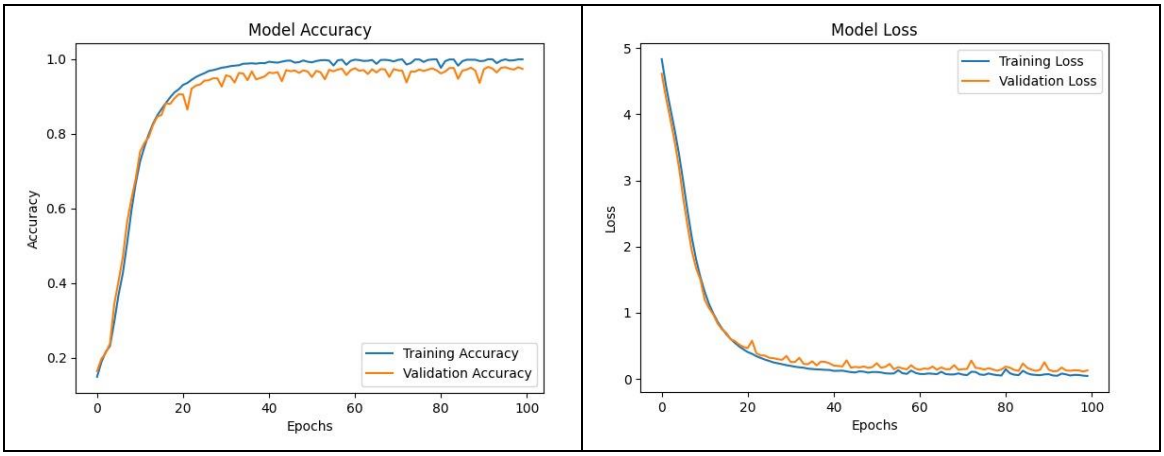


Figure 15: Accuracy / loss curves of the proposed model.

Figure 16 depicts the confusion matrix, which is a (44×44) matrix and contains the number of correct and incorrect classified images for each class of the Ottoman OCR problem. Using this confusion matrix it is also possible to calculate performance metrics such as accuracy, sensitivity, and precision.

Technium Vol. 18, pp.47-64 (2023)
ISSN: 2668-778X
www.techniumscience.com

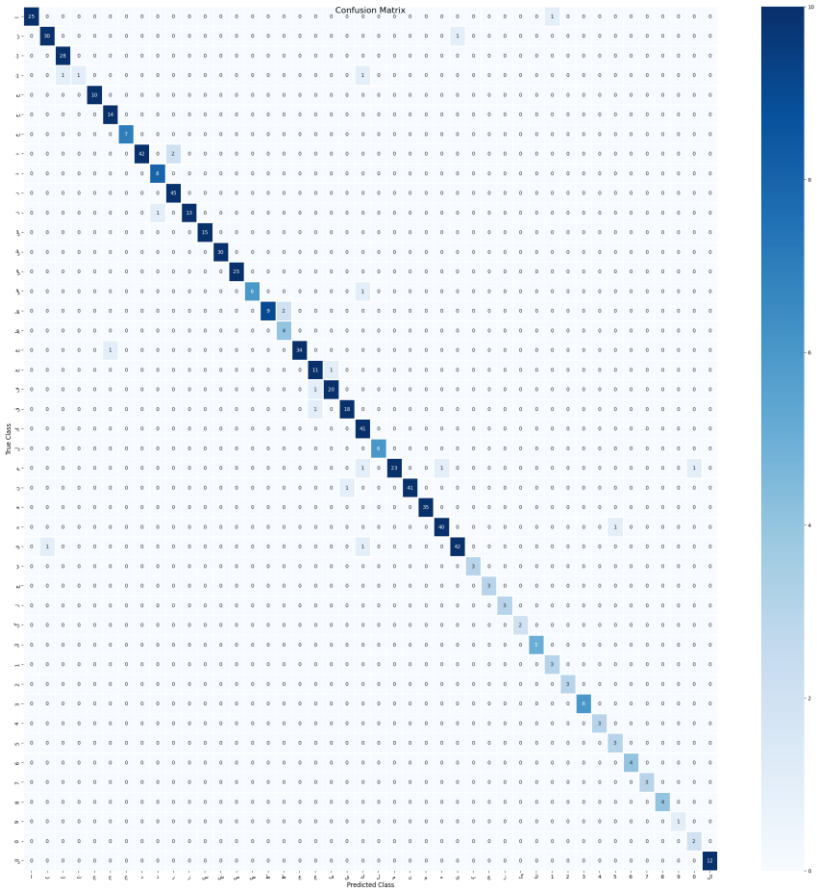


Figure 16: Confusion matrix.

Table 3 summarizes the obtained accuracy, sensitivity, and precision scores for each class of the Ottoman OCR problem. Consequently, the proposed model achieved overall scores for accuracy, sensitivity, and precision of 99.6%, 87.1%, and 93.3% on the test dataset respectively.

Table 3: Classification results of the proposed model by class

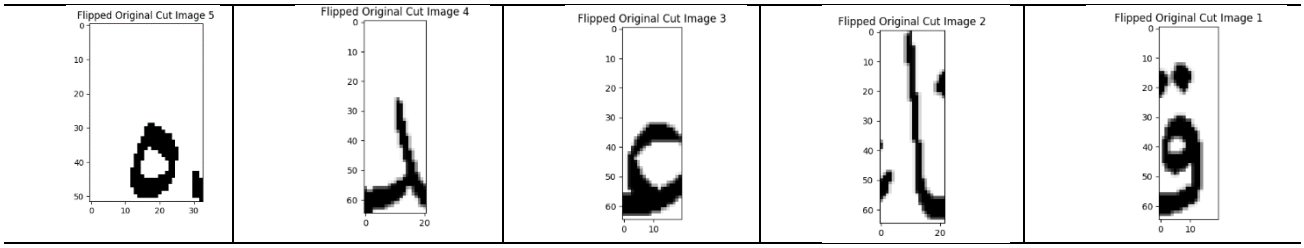
Letters	Precision	Recall	Accuracy	Letters	Precision	Recall	Accuracy	Letters	Precision	Recall	Accuracy	Letters	Precision	Recall	Accuracy
ا	1.000	1.000	1.000	س	1.000	1.000	1.000	ل	1.000	1.000	1.000	1	1.000	0.667	0.999
ب	1.000	0.903	0.996	ش	0.906	0.967	0.994	م	1.000	0.692	0.989	2	1.000	1.000	1.000
ت	0.711	0.964	0.983	ص	1.000	1.000	1.000	ن	1.000	0.762	0.986	3	0.750	1.000	0.997
ث	1.000	0.333	0.997	ض	0.833	0.714	0.996	و	1.000	0.971	0.999	4	0.750	1.000	0.999
ج	0.833	1.000	0.997	ط	1.000	0.818	0.997	ه	0.932	1.000	0.996	5	1.000	1.000	1.000
ح	1.000	0.929	0.999	ظ	0.800	1.000	0.999	ي	1.000	0.886	0.993	6	1.000	1.000	1.000
خ	0.778	1.000	0.997	ع	0.971	0.971	0.997	پ	1.000	0.667	0.999	7	1.000	1.000	1.000
د	0.880	1.000	0.991	غ	0.500	1.000	0.983	چ	1.000	1.000	1.000	8	1.000	1.000	1.000
ذ	1.000	0.625	0.996	ف	0.800	0.190	0.974	ژ	1.000	0.333	0.997	9	1.000	1.000	1.000
ر	1.000	0.889	0.993	ق	1.000	0.737	0.993	گ	1.000	0.500	0.999	0	0.667	1.000	0.999
ز	1.000	0.786	0.996	ك	0.953	1.000	0.997	ڭ	1.000	1.000	1.000	لا	1.000	1.000	1.000

3.3. Evaluation of Proposed Model on Printed Ottoman Documents

The trained CNN-BiLSTM model proposed in this study was applied to the test dataset, which was obtained by applying the document digitizing procedure explained in subsection 2.4 on documents containing scanned printed Ottoman texts. The characters predicted by the model were then collected and linked together to become a complete word and placed where they should be in the sentence. Figure (17) depicts the process of letter recognition and combining them into one-word and Table-4 shows error rates in predicting each letter of digitized Ottoman documents.

Technium Vol. 18, pp.47-64 (2023)
ISSN: 2668-778X
www.techniumscience.com

Segmented Characters



Predicted Characters in Text Format

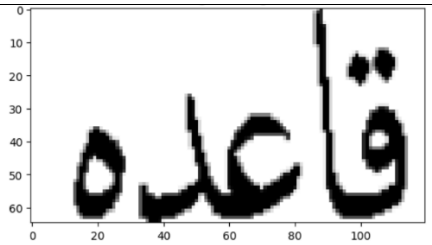
ه	د	ع	ا	ق
Combined Letters in a word		**Original**		
قاعده				

Figure 17: Process of letter recognition and combining them into one word

Table 4: The errors rates in predicting each letter of digitized Ottoman document

Char		Total	Error Ratio
alif	ا	2446	0.04
ba	ب	951	0.26
te	ت	772	0.22
the	ث	30	0.12
cim	ج	339	0.13
ha	ح	149	0.27
xa	خ	105	0.23
dal	د	1333	0.28
zal	ذ	12	0.17
ra	ر	1690	0.29
ze	ز	285	0.32
sin	س	548	0.34
shin	ش	286	0.19
sad	ص	139	0.12
dad	ض	57	0.17
ta	ط	271	0.17
za	ظ	43	0.29
ayn	ع	274	0.32
gayn	غ	257	0.34
fe	ف	284	0.19

Char		Total	Error Ratio
qaf	ق	551	0.12
kaf	ك	963	0.17
lam	ل	1596	0.22
mim	م	1111	0.34
nun	ن	1309	0.23
he	ه	1599	0.13
vav	و	1605	0.12
ye	ي	2797	0.22
pe	پ	119	0.17
chim	چ	117	0.14
je	ژ	14	0.07
gef	گ	13	0.09
nef	ڭ	15	0.08
lamalif	لا	120	0.13
1	١	20	0
2	٢	18	0
3	٣	13	0
4	٤	9	0
5	٥	8	0
6	٦	9	0
7	٧	11	0
8	٨	14	0
9	٩	12	0
0	٠	16	0

4. Conclusion

This work presented a deep learning model to facilitate the optical character recognition of Ottoman documents printed in the Matbu font. The proposed model was based on CNN model that incorporates a recurrent neural network with long short-term memory. For better recognition of long sequences of characters in a line, we combined CNN architecture and BiLSTM because this combination can efficiently identify and summarize characteristics in images. The fact that the CNN and BiLSTM do not require any extra processing or conversion in order to be trained is a significant benefit of this design. Our proposed model was trained and tested on a public dataset and achieved %99.6 overall accuracy. Then, the trained model was tested on 26 pages, 396 lines, 4912 words, and 22450 characters that was gathered from several source documents. The dataset's lines are roughly 1349 x 120 pixels in size and have a resolution of 96 dpi. Moreover, for the purpose of gaining a clearer picture of the types of mistakes made by the proposed model, we calculated the character recognition errors for each character in the test document. The experiments showed that, more errors were made by joiners and dotted letters than by non-joiners and dot-free letters in the experiments. The findings showed that the proposed model achieved overall scores for accuracy, sensitivity, and precision of 99.6%, 87.1%, and 93.3% on the test dataset respectively using both real and simulated data. The limitations of this work could be around the rare characters that cannot be trained because the dataset was too small. The future works and directions can be extending this study to include more models with a more datasets aiming to have more reliable results in terms of accuracy of detection.

References

[1] R. Keleş and others, “Osmanl Türkçesinde Kâf Harfi: Tasnif ve Seslendirme Meselesi,” Cumhuriyet Ilahiyat Dergisi, vol. 25, no. 1, pp. 195–216, 2021.
[2] Ibtisam Uraibi Abdullah and M. F. Jihankheer, “‘ Kabusnâme ve Mezâki Divan Örneğinde’ 14. ve 17. Yüzylarda Osmanlı Türkçesinde Harf-i Ta’rifin Kullanmas Üzerine Bir Inceleme.,” Journal of College of Languages, no. 47, 2023.
[3] D. Steel, “Iron Cross and Crescent Press Discussion of the Ottoman Empire in the United Kingdom, 1914-1918,” 2019.
[4] I. Dolek and A. Kurt, “Ottoman OCR: Printed naskh font,” in 2021 International Conference on INnovations in Intelligent SysTems and Applications (INISTA), 2021, pp. 1–5.
[5] K. Karthick, K. B. Ravindrakumar, R. Francis, and S. Ilankannan, “Steps involved in text recognition and recent research in OCR; a study,” International Journal of Recent Technology and Engineering, vol. 8, no. 1, pp. 2277–3878, 2019.
[6] N. A. M. Isheawy and H. Hasan, “Optical character recognition (OCR) system,” IOSR Journal of Computer Engineering (IOSR-JCE), e-ISSN, pp. 2278–2661, 2015.
[7] A. A. A. Ali, M. Suresha, and H. A. M. Ahmed, “A survey on arabic handwritten character recognition,” SN Comput Sci, vol. 1, no. 3, p. 152, 2020.
[8] N. Lamghari, M. E. H. Charaf, and S. Raghay, “Template matching for recognition of handwritten Arabic characters using structural characteristics and Freeman code,” The International Journal of Computer Science and Information Security, vol. 14, no. 12, pp. 31–40, 2016.
[9] C. Clausner, A. Antonacopoulos, and S. Pletschacher, “Efficient and effective OCR engine training,” International Journal on Document Analysis and Recognition (IJDAR), vol. 23, pp. 73–88, 2020.
[10] A. Özer, “Ottoman Turkish Characters.” 2020. [Online]. Available: https://www.kaggle.com/dsv/1443328
[11] E. F. Bilgin Tasdemir, “Printed Ottoman text recognition using synthetic data and data augmentation,” International Journal on Document Analysis and Recognition (IJDAR), pp. 1–15, 2023.
[12] E. F. B. Tasdemir et al., “Transcription of Ottoman Machine-Print Documents,” 2022.
[13] Ishak Dölek and A. Kurt, “Ottoman Optical Character Recognition with deep neural networks Derin sinir ağlaryla Osmanlca optik karakter tanma,” Journal of the Faculty of Engineering and Architecture of Gazi University, vol. 38, no. 4, 2023.
[14] U. Alp, Ö. Alperen, and H. I. TURKMEN, “Evrişimsel Sinir Ağ Tabanl Osmanlca Belge Çözümleyici,” International Journal of Advances in Engineering and Pure Sciences, vol. 33, no. 4, pp. 581–591, 2021.
[15] Ishak Dölek and A. Kurt, “A deep learning model for Ottoman OCR,” Concurr Comput, vol. 34, no. 20, p. e6937, 2022.
[16] Mahmood, Basim, Marcello Tomasini, and Ronaldo Menezes. "Estimating memory requirements in wireless sensor networks using social tie strengths." In *2015 IEEE 40th Local Computer Networks Conference Workshops (LCN Workshops)*, pp. 695-698. IEEE, 2015.
[17] A. Abdi, S. M. Shamsuddin, S. Hasan, and J. Piran, “Deep learning-based sentiment classification of evaluative text based on Multi-feature fusion,” Inf Process Manag, vol. 56, no. 4, pp. 1245–1259, 2019.
[18] J. Brownlee, Long short-term memory networks with python: develop sequence prediction models with deep learning. Machine Learning Mastery, 2017.
[19] H. Li, J. Li, X. Lin, and X. Qian, “Pancreas segmentation via spatial context based u-net and bidirectional lstm,” arXiv preprint arXiv:1903.00832, 2019.
[20] A. Özer, “Ottoman Turkish Characters.” 2020. [Online]. Available: https://www.kaggle.com/dsv/1443328
[21] Mahmood, Basim Mohammed, and Marwah M. Dabdawb. "The pandemic COVID-19 infection spreading spatial aspects: A network-Based software approach." *AL-Rafidain Journal of Computer Sciences and Mathematics* 14, no. 1 (2020): 159-170.
[22] S. Singh, P. K. Sarangi, C. Singla, and A. K. Sahoo, “Odia character recognition system: A study on feature extraction and classification techniques,” Mater Today Proc, vol. 34, pp. 742–747, 2021.
[23] Sultan, Nagham A., Basim Mahmood, Karam H. Thanoon, and Dheyaa S. Khadhim. "Network Centralities-Based Approach for Evaluating Interdisciplinary Collaboration." In *2020 6th International Engineering Conference “Sustainable Technology and Development"(IEC)*, pp. 216-221. IEEE, 2020.
[24] J. A. Etzel, V. Gazzola, and C. Keysers, “An introduction to anatomical ROI-based fMRI classification analysis,” Brain Res, vol. 1282, pp. 114–125, 2009.

Technium Vol. 18, pp.47-64 (2023)
ISSN: 2668-778X
www.techniumscience.com

[25] A. LaTorre, L. Alonso-Nanclares, S. Muelas, J. M. Peña, and J. DeFelipe, “Segmentation of neuronal nuclei based on clump splitting and a two-step binarization of images,” Expert Syst Appl, vol. 40, no. 16, pp. 6521–6530, 2013.
[26] K. Naresh, K. A. Khan, R. Umer, and W. J. Cantwell, “The use of X-ray computed tomography for design and process modeling of aerospace composites: A review,” Mater Des, vol. 190, p. 108553, 2020.
[27] M. Badry, M. Hassanin, A. Chandio, and N. Moustafa, “Quranic script optical text recognition using deep learning in IoT systems,” CMC-Comput. Mater. Contin, vol. 68, pp. 1847–1858, 2021.
[28] I. Vasilev, D. Slater, G. Spacagna, P. Roelants, and V. Zocca, Python Deep Learning: Exploring deep learning techniques and neural network architectures with Pytorch, Keras, and TensorFlow. Packt Publishing Ltd, 2019.