An OCR Engine for Printed Receipt Images using Deep Learning Techniques

Çağrı Sayallar[1], Ahmet Sayar[2], Nurcan Babalık[3]
Nacsoft, Kocaeli University Technopark[1,3]
Computer Engineering, Kocaeli University[2]
Kocaeli, Turkey

***Abstract*—The digitization of receipts and invoices, and the recording of expenses in industry and accounting have begun to be used in the field of finance tracking. However, 100% success in character recognition for document digitization has not yet been achieved. In this study, a new Optical Character Recognition (OCR) engine called Nacsoft OCR was developed on Turkish receipt data by using artificial intelligence methods. The proposed OCR engine has been compared to widely used engines, Easy OCR, Tesseract OCR, and the Google Vision API. The benchmarking was made on English and Turkish receipts, and the accuracies of OCR engines in terms of character recognition and their speeds are presented. It is known that OCR character recognition engines perform better at word recognition when provided word position information. Therefore, the performance of the Nacsoft OCR engine in determining the word position was also compared with the performance of the other OCR engines, and the results were presented.**

***Keywords*—Optical Character Recognition (OCR); image processing; deep learning; benchmarking; receipt**

I. Introduction

With the introduction of computers into our lives, important documents for the user began to be stored in the computer environment. Although most documents are stored electronically, there are still printed-paper documents that we frequently use in daily life. Invoices and receipts which are among such documents printed on nondurable paper contains information that needs to be saved. When it comes to saving and storing a paper document, the first thing that comes to mind is to scan the document and store the document image in electronic environment. With this method, documents can be stored properly by gaining space. Nevertheless, operations such as listing, sorting and processing these document images are carried out by people, which means a loss of time and resources, especially for companies. In order to meet this need, Optical Character Recognition (OCR) engines have been developed to automate the processing of document images. OCR engines have been developed to read the images containing the text and convert them into processable text outputs.

The most obvious examples of digitized printed documents are receipts and invoices. Receipts and invoices carry data on them that may be important to the user, such as amount, tax, date. The user may wish to store or process this information. Applications have been developed to meet these needs of the user. The common purpose of these applications is to enable the user to track his or her or someone else's spending, store and rank their spending. These applications may encounter many problems when reading the receipt data. Since receipts and invoices are made of paper, they are nondurable, so they wrinkle very quickly, wear out in a short time and the writing on them can be easily erased. In addition, factors such as different fonts, images, shapes, presentation of information in the form of tables, background of the receipt image, and oblique withdrawal of the receipt make the situation more difficult. These factors are examples of problems that make it difficult to read a receipt. In the applications mentioned, it is necessary to use the OCR engine, which gives the best result despite these problems. The field of OCR is a field that attracts a lot of attention and is competitive. For this reason, there are many OCR engines produced. Since OCR engines are mostly useful for large companies, most OCR engines produced are for commercial purposes and the methods used are not shared. The most well-known of the commercially produced OCR engines are Amazon Textract and Google Vision Api. Unlike commercially produced OCR engines, there are also a small number of open source OCR engines. Examples of open source OCR engines are Tesseract [1], Easy OCR [2] and OCRopus [3].

The aim of this study is to develop an open source OCR engine using artificial intelligence methods. The developed OCR engine is trained using receipt data. The receipt dataset is difficult to find because the receipt images contain personal information belonging to the user. For this reason, we need to create the dataset to be used for training ourselves. Since all of the data obtained by us consisted of Turkish receipts, Turkish receipt images were used for the training. The success of the developed OCR engine is compared to other existing OCR engines. For comparison, both Turkish receipt dataset and English receipt dataset were used. Since the training dataset consists of Turkish receipt images, the comparison of the achievements of OCR engines was also made on Turkish receipts. As mentioned earlier, these receipts contain the personal information of the owner on them. For this reason, the Turkish dataset and the results of the comparison cannot be shared. Both for the sharing of the comparison results and to measure the success of the developed OCR engine in different datasets, the comparison is also made on the English receipt data. Test images from the ICDAR-2019 SROIE [4] competition were used for the English receipt dataset.

Although this study focuses on only one document type, the methods used can be applied to other document types. The method used in the study consists of three main headings. These; pre-preparation, word detection and word reading. In the preliminary preparation stage, the receipt areas are determined in the visual by passing through the visual masking and

segmentation model that comes first. Then, the corner detection algorithm is operated on the obtained mask and the perspective process is applied by detecting the four corners of the receipt. Thus, the receipt image is freed from unnecessary background and more focus is provided on the regions containing text. In the next step, the object detection algorithm is applied to the image obtained and the locations of the words in the receipt are determined. In the final stage, the words detected are passed through the Convolutional Recurrent Neural Network (C-RNN) [5] model and the reading process is performed. In this way, the incoming receipt image is OCR'd and the text output is produced. The success of the OCR engine developed in this study is compared to Tesseract, Easy OCR and Google Vision Api. When making the comparison, the test dataset consisting of Turkish receipt images selected by us and not used during the training phase and also English receipt data from the ICDAR2019-SROIE competition is used. Comparisons are evaluated on the basis of speed and accuracy, and the results from each dataset are shown in separate tables.

The flow in this article can be summarized as follows. In the following part of the study, the OCR engine developed will be explained. The method and dataset used in this regard will be detailed. In Section III, the success of the proposed OCR engine will be detailed from the dataset used to compare it with other commonly used OCR engines and the benchmark metrics used. Section IV contains the results from the comparison, interpretations and analyses of these results. Section V briefly summarizes our work and talks about the work to be done in the future.

II. Related Work

OCR engines have been a topic of interest for a long time. Even though studies have been carried out for a long time, the OCR problem is still not a solved problem. However, there are OCR engines that can achieve high success compared to others. Applications such as Amazon Textract and Google Vision Api, which are among these OCR engines, are chargeable to use. Since OCR engines are generally produced for commercial purposes, very few open source OCR engines are available. The most well-known open source OCR engine in the literature is Tesseract. Tesseract OCR engine differs from our study in terms of the methods it uses. While deep learning methods are used for OCR in our study, Tesseract performs OCR with pixel operations. The Tesseract OCR engine first determines the text areas in the image by performing page layout analysis. Blobs are obtained by applying connected component analysis in the specified text fields. The detected blobs are then separated into lines and words. After the words are divided into characters with two different methods, the text recognition process is performed using the two-pass adaptive classifier.

Apart from the Tesseract OCR engine, there is another open source OCR engine called OCRopus. This OCR engine, first determines the text fields by page layout analysis like Tesseract. Then the text fields determined by the page layout are sent to the Text Line Recognition stage, and the language of the text and the writing direction of the text (right to left, left to right) are determined. It uses dynamic programming algorithm for character detection and multi-layer perceptrons (MLPs) for character recognition. There is also a study [6], which uses slightly more modern methods and whose text recognition stage is similar to our work. In the text detection phase, all contours are detected with the Canny algorithm and then lines are determined using these contours. Then, preprocessing operations such as Noise removal, perspective correction, baseline correction are applied to improve the reading process before the text recognition stage. The lines detected in the text detection stage are divided into smaller pieces by the sliding window method for the reading process. The image pieces obtained from the lines are then sent to the text recognition model. First, features are obtained by passing the incoming image pieces through the encoder consisting of Convolution layers. Then, these features are passed through the Bidirectional Long-Short Term Memory (Bi-LSTM) and Connectionist Temporal Classification (CTC) layer, which is the decoder part of the model, and text outputs are produced.

Unlike these [1], [3], [6] studies, [7] study describes a new method for reading clipped words from document data. In this method, the incoming word image is first passed through an encoder called the Gated Recurrent Convolution Neural Network (GRCNN). The features obtained in the encoder section are then sent to the decoder consisting of Bi-LSTM and CTC layers, and text outputs are produced. The aforementioned [1], [3], [6], [7] studies were developed using document data, similar to our study. In the OCR field, there is also OCR in Natural Scene data, in addition to document data. Although the OCR issue in Natural Scene data is similar to the OCR issue in document data, it is a much more difficult problem. While some studies on Natural Scene data focused only on text recognition [8], [9], [10], some others [11], [12], [13], [14] tried to solve the text detection and recognition issue together.

III. Methodology

In this section, all stages of the developed OCR engine are examined in detail. The OCR engine mentioned in this study was developed through three basic processes. In the preliminary preparation stage, which is the first stage; the incoming receipt image is subjected to multiple processes for the detection of text fields. In the second stage, word positions are determined on the full-screen receipt image, which is the output of the first stage. In the final stage, the detected words are clipped and sent to the "word reading model" and the reading process is performed. The diagram given in Fig. 1 summarizes the process mentioned. The details of the processes are given in the following subheadings.

A. Preliminary Preparation

Receipt images sent to the OCR engine and expected to be read often come with background images and taken from different angles. The input image can also contain a background image, as shown in Fig. 2. In such a case, the detection of text fields is even more difficult. Factors such as the constant change of the background, the position of the receipt in the image, the obliquity of the receipt are just some of the reasons that will make it difficult to determine the text field. For this reason, the receipt image is passed through the preliminary preparation stage before the text fields on it are detected. The purpose of the preliminary stage is to obtain a full-screen and vertical image of the receipt in the image, free from the background image, and to send the clean receipt image to the text detection stage.

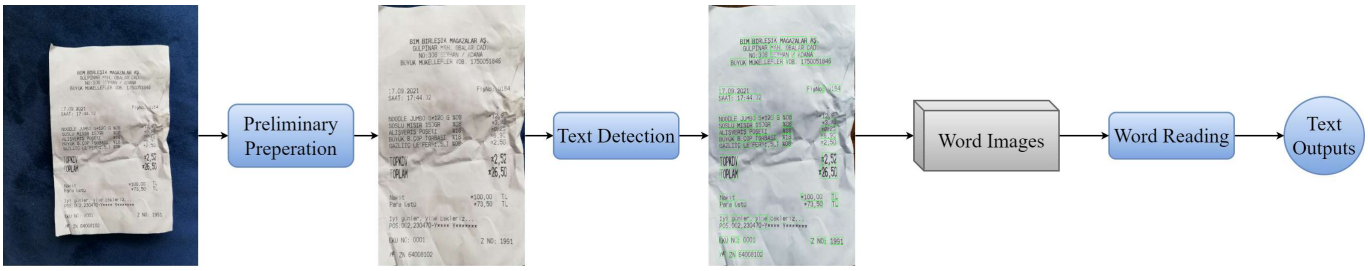

Fig. 1. The diagram that shows all the stages and outputs of the OCR model

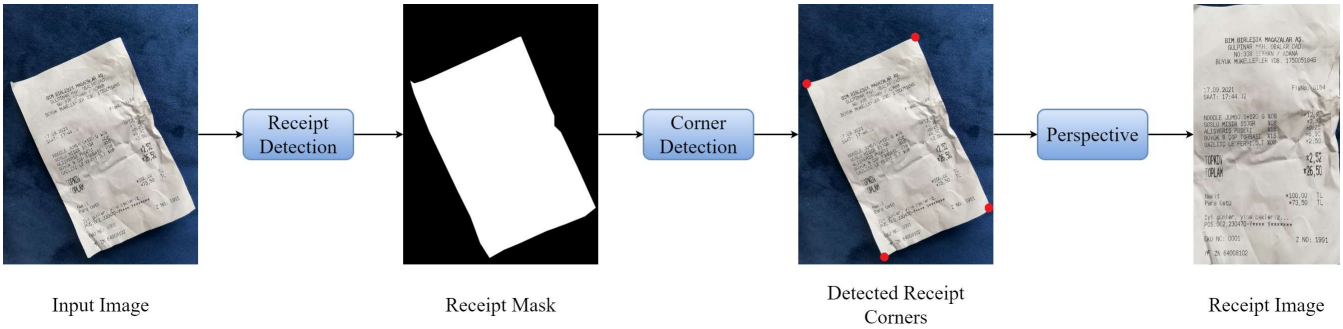

Fig. 2. The processes and outputs of the preliminary stage

Fig. 2 also appears to detail the inside of the box called Pre-porocessing in Fig. 1. The preliminary preparation phase consists of three steps. These can be listed as (i) determination of the receipt area (receipt mask formation), (ii) determination of receipt corners, and (iii) application of the perspective process. Masking model is used to determine the receipt area. The purpose of the masking model is to produce a mask of the objects in the visual and labeled in the training data. In the training of the masking model, the input visual for the input and the mask of the same resolution as the expected output are required. An example of an input image is the Input Image in Fig. 2. While the mask of this image is produced, the receipt field in the image is labeled with the help of a tool called LabelMe [15] . The tagged receipt field has a value of 1, while the background without a receipt is 0. When this matrix is converted to a image, the Receipt Mask given in Fig. 2 is obtained. This is how the dataset for the training of the masking model is prepared.

The masking model is based on the Convolutional Encoder-Decoder method. The resolution of the model input is reduced as the layers progress. This part is known as the Encoder and is the part where the Model draws information from the image. Then, in the Decoder stage, the image that is reduced in the Encoder stage is restored again. Finally, the mask of the input image is produced by passing through the classification layer. Depending on the number of classes in the training dataset, the activation function of this classification layer can be Sigmoid or Softmax. Since the OCR engine mentioned in this study has a single class of receipt field, the last layer of the masking model used is passed through the Sigmoid activation function. As an output, a matrix consisting of numbers between 0 and 1 in the input image resolution is obtained. In this matrix, pixels that are close to zero are considered as backgrounds, while pixels with a value closer to one are considered as receipt areas. The receipt area in the receipt image is thus estimated.

There are multiple Encoder-Decoder methods for the masking model. As the masking model used in this study, the most appropriate Encoder-Decoder method was selected among the methods presented in the segmentation_models [16] library by trying. Among the methods available in this library, only the achievements of Unet [17] and Linknet [18] Encoder-Decoder methods are compared. After selecting the Encoder-Decoder method for the masking model, it is necessary to select a Backbone for the Encoder. As skeletal model, VGG16 [19] and Resnet34 [20] models with similar parameter numbers are compared. A mobilenetv2 [21] model with a lower number of parameters has also been added to the comparison. As a result, the most appropriate masking model was selected by comparing two different Encoder-Decoder methods and three different Skeletons. The results obtained are shown in Table I.

When comparing, dataset and training parameters were kept constant. The dataset used in the comparison consists of the same data as the dataset of the masking model used in the OCR engine mentioned in this study. The dataset contains a

TABLE I. COMPARISON OF COMPARING THE SUCCESSES OF MASKING MODELS. (BATCH SIZE: 8, EPOCHS: 50, METRICS: IOU, SAVEBESTVAL_IOU_ACC, INPUT RESOLUTION: 512,512)

Model	Mobilenetv2 Unet	VGG16 Unet	Resnet34 Unet	Mobilenetv2 Linknet	VGG16 Linknet	Resnet34 Linknet
Duration (Sec)	0.0926	0.1070	0.0908	0.0880	0.1004	**0.0870**
IoU%	98.28	98.42	98.40	98.17	**98.44**	98.31

total of 1352 background receipt images. 10 percent of this data is reserved for testing. The results given in Table I were obtained using this training and testing dataset. In training, bce_jaccard_loss was used as a loss function. The remaining parameters are found in the description of Table I. Each model is trained by up to 50 epochs and the best Val_IoU values are used for success comparison. Then the average elapsed time for a receipt estimate was used to compare the model speed. Test data were used for duration and success measurement. When the results given in Table I are examined, it is understood that there is not much difference between them. However, if it is necessary to choose the most suitable model, the Resnet34 Unet should be chosen due to its proximity to the highest success and lowest speed. In this study, Resnet34 is used as Skeleton and Unet is used as Encoder-Decoder method in the masking model.

Using the masking model, the receipt areas in the incoming receipt image are determined. This process alone is not enough. The detected receipt area needs to be separated from the unnecessary background. Perspective process is used in this study both to get rid of unnecessary background and to correct oblique receipt images. The perspective process converts images that are oblique, such as the input image given in Fig. 2, into full-screen images such as the Receipt Image in Fig. 2. For this, the four corners of the region to be applied to the perspective process (in this case, the receipt area in the image) must be determined correctly. The receipt mask obtained from the Masking model is used for corner determination. The edges of the receipt image are determined using the Canny function in the OpenCV library on the receipt mask. Then, the output consisting of black and white images with edges is sent to the findContours function and the corner points in the receipt mask are determined. Using the positions of the obtained corner points, the best four corners to represent the receipt mask are selected. Finally, the positions of these four designated verticals are sent to the getPerspectiveTransform. The output of this function is sent to the warpPerspective function and the Receipt Image in Fig. 2 with perspective applied is generated.

B. Text Detection

At this stage, the receipt visual, which is the output of the preliminary preparation stage, is taken as input. In order to read the incoming receipt image, the text fields must first be determined. In this study, object detection algorithm is used for the detection of text fields. Object detection algorithms are used to determine the position of pre-labeled objects in the image or video. In this study, the task of the object detection algorithm is to detect the text fields on the receipt image that comes as a full screen. The dataset used for the object detection algorithm is labeled using a tool called LabelMe. The data used in the dataset are the receipt images from the preliminary preparation stage. When this data is labeled, each word in the receipt image is labeled as belonging to the same object class. The dataset consists entirely of Turkish receipt images and consists of 552 data in total. 10 percent of this dataset is reserved for test data.

In this study, You Only Look Once (YOLO) model is used due to its speed and success. For word detection, YOLO V3 [22] and YOLO V4 [23] models were compared and the most appropriate model was selected. For the training, a dataset consisting of Turkish receipt data used in Text Detection was used. For success comparison, 10 percent of the dataset was used as a test dataset. Accuracy was taken into account as a benchmark. In comparison, the batch size and model resolution are kept the same. Average Precision, IoU, F1 Score are used as comparison metrics. The darknet [24] library, which is also used in YOLO training, was used to calculate the mentioned metrics of YOLO models. The results from the comparison are shared in Table II. In Fig. 3, the words detected by both YOLO models in the sample Turkish receipt image are seen. Although the difference in success is not clearly seen in Fig. 3, when Table II is examined, it is seen that the YOLO V4 model gives the best result in all categories. For this reason, YOLO V4 model is used in the word detection model.

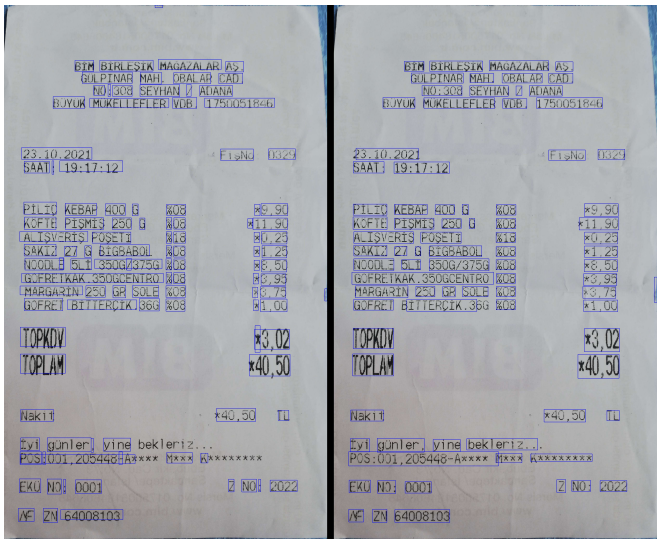

Fig. 3. The Words Detected by YOLO Models on the Sample Turkish Receipt (Blue Boxes), the Output of the YOLO V3 (Left), the Output of the YOLO V4 (Right)

C. Word Reading

The word reading stage is reached with the word positions determined during the text detection phase. At this stage, words are clipped from the receipt image using word positions. The cropped words are read by means of the reading model. C-RNN is used as a word reading model in this study. The dataset used for the training of the model consists of words clipped from Turkish receipt images. Words are manually tagged. Although Turkish data are used in the training dataset, the words are labeled according to the Latin alphabet. The reason for this is that there are letters in Turkish that are very similar to each other (ç-c, ö-o, i-ı ...). These letters create ambiguity due to their similarities, and labeling them as a single class increases success by eliminating uncertainty. For this reason, the training dataset is labeled according to the characters in the Latin alphabet. The dataset consists of 68,000 Turkish words

TABLE II. COMPARISON OF THE ACHIEVEMENTS OF YOLO MODELS

Model	Average Precision	IoU	F1 Score
YOLO V3	84.31	59.50	0.80
YOLO V4	**86.61**	**65.5**	**0.83**

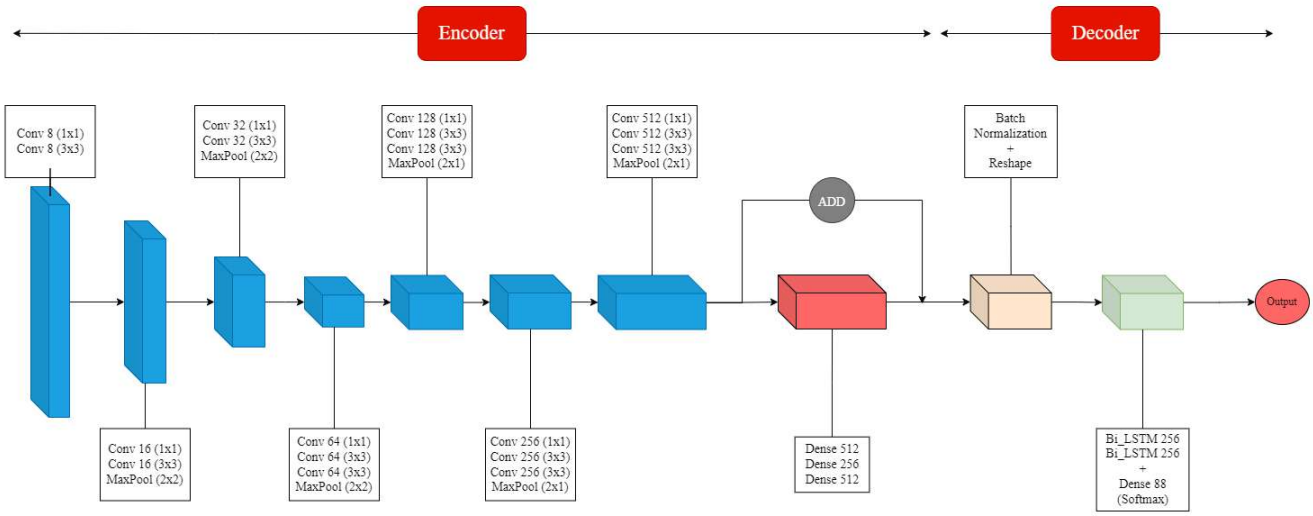

Fig. 4. The C-RNN model scheme used as a word reading model

that are clipped and labeled from approximately 1,100 receipt images. 10 percent of this Turkish data is reserved for the test dataset.

The C-RNN model used for the word reading model is based on the principle of Convolutional Encoder and Recurrent Decoder. As seen in Fig. 4, the incoming word image is first passed through the Convolutional layers. In this section, the information of the letters in the word is obtained. Then, wordprintouts are produced by looking at this information in the Bi-LSTM layer. CTC-Loss is used as the loss function in the model. The model takes words with resolution (256,64,1) as input. During the model prediction, word predictions are produced by passing the word outputs through the ctc_decode function in the [25] Keras library. After this stage, the words on the receipt and the positions of these words are known. Using this information, the lines are determined. Thus, the text output with specified lines that are the output of the OCR engine is produced.

The word reading model consists of two parts: Encoder and Decoder. While creating the word reading model seen in Fig. 4, various methods were tried on the Encoder and Decoder layers. If we need to talk about the methods used in the Encoder section, some of them can be listed as follows: (i) The stride value was made 2 by using the Convolution layer instead of the pooling layer. (ii) If there are three consecutive Convolution layers, the first layer output is summed up with the third layer output. (iii) The number of Convolution layers before the pooling layer was reduced to one in order to create a simpler model and to increase success. (iv) The activation function used and the output dimension were changed. (v) A feed forward layer has been added to the end of the convolution layer, as shown in Fig. 4. Multiple encoder models were developed using such methods, and the most successful Encoder model seen in Fig. 4 was selected among them. When changing the encoder model, the parameters used in the training and the Decoder model were kept completely constant.

After the encoder model is selected, the Decoder model should be selected as the most successful. The methods used for this can be listed as follows: (i) The activation function and output dimension used were changed. (ii) Instead of using bidirectional LSTM, two forward directional and two backward directional LSTMs were used and their outputs were combined. (iii) It was attempted to transfer the LSTM hidden state to the next LSTM layer, but this method was not used because it reduced the success as expected. (iv) LSTM outputs were collected with each other. (v) A feed forward layer has been added after the LSTM output. While trying these methods, the training parameters and the encoder model were kept constant. Among the methods tried, the Decoder model in Fig. 4 was selected that gave the most successful result.

IV. Evaluations

A. Comparison Dataset

While the success of the OCR engine mentioned in this study was measured, the accuracy of the outputs obtained on the same dataset was compared with other OCR engines. The comparison was performed both in the dataset consisting of Turkish receipts and in the dataset consisting of English receipts. For the success comparison, the production of the OCR engine mentioned in this study requires a dataset that is not used in the preliminary preparation, text detection or word reading stages. A test dataset consisting of Turkish receipts was created for comparison. This dataset consists of 66 Turkish receipt images. These receipt data are labeled on a per-word basis. In Fig. 4, a sample image from the Turkish test dataset is given. When the receipt image is examined, there are Turkish letters such as ”I, Ş, Ü, Ç” in it. Since the OCR engine developed could only read the Latin alphabet, these Turkish letters were converted to similar letters in the Latin alphabet (”I, S, U, C”) and tagged. For the sake of equality in comparison, the outputs of all compared OCR engines are also translated into the Latin alphabet. This process was applied only in the Turkish dataset.

As mentioned before, the Turkish dataset used in the comparison cannot be shared because it contains personal information. For this reason, in addition to the Turkish dataset,

the ICDAR2019-SROIE dataset, which consists of English receipt data, is also used. Since the OCR engines were also compared in the receipt dataset in the ICDAR2019-SROIE competition, test images of the same dataset were used to compare the OCR engine produced in this study with other OCR engines. Since the text positions are given on a word-by-word basis in the outputs of the compared OCR engines, the dataset to be compared must also be labeled as words. Since the ICDAR2019-SROIE dataset where the comparison will be made is labeled on a sentence-by-sentence basis, these tags need to be broken down and converted into word tags. When performing this operation, if there are spaces in the labeled text in the dataset, this text is divided into more words than the number of spaces. The process of shredding this tagged text was carried out by us. This English dataset is labeled by the contest holders with all letters capitalized. For this reason, the comparison was made after the outputs of all OCR engines were converted to capital letters. This process was applied to the English dataset only.

The application called LabelMe was used when labeling words. Thanks to the LabelMe application, words and word positions are saved in a file by manually labeling the data. Each word saved in the file is saved in the (LeftTopCornerX LeftTopCornerY RightLowerCornerX RightLowerCornerY Word) format. Only a single word is stored in each line of the file and only the words in a receipt are stored in each file. The order in which words are labeled is not important because then each tagged word is matched to the words read by OCR engines.

B. Comparison Method

The comparison of OCR engines is based on speed and accuracy. The accuracy category includes two main issues. These are the success of Word Reading and Word Position Detection. These categories are the same as task 1 and task 2 in the ICDAR2019-SROIE study. The first task in the competition measures how accurately the contestants determine the positions of the words on the receipt. In the second task of the competition, it is measured how accurately the competitors read the text in the receipt image. In the ICDAR2019-SROIE competition, they used Precision, Recall and F1-Score to measure success in both tasks. These metrics are not enough to measure the success of an OCR engine. For this reason, while comparing OCR engines in this study, in addition to these metrics, Character Error Rate (CER) metric was used in word reading success and IoU metric was used in word position determination. As mentioned in the Comparison Dataset section of the article each receipt is labeled mixed on a word basis. Since the words and positions obtained by reading the receipt image with the OCR engine are also mixed, it is necessary to match the labeled data with the data read by the OCR engine to measure the success of the OCR engine.

When matching words, it is necessary to match them correctly. OCR engines may have guessed more or less than the words tagged. In order to prevent this situation, it is aimed to achieve the highest success in matching. Compared the tagged words with all the predicted words and recorded CER achievements. The achievements are recorded in the list and converted into a table in the size of (Number of Tagged Word X Number of Words Read). Then, starting from the first row of the table, the word with the highest achievement in the row is determined. If this word also has the highest achievement in the column and is higher than the specified threshold value, the labeled word in the index of the row it is in is matched to the read word in the index of the column in which it is located. To prevent the matched words from being re-matched, the achievement values in the table where CER achievements are recorded are equalized to zero. Calculations are made after all the words are matched to each other. When performing calculations, the paired words are taken as True Positive (TP), while the predicted but unmatched words are taken as False Positive. In addition, words that are not predicted but are labeled are considered False Negative (FN). The total number of words matched in a receipt is shown as TP_n. With these definitions, formulas to be used in success comparison are produced. The formulas used are shown as Eq. (1) and Eq. (2).

V. Results

Two different categories are taken into account when comparing OCR engines. These are Speed and Accuracy. The accuracy category is similar to the ICDAR2019-SROIE competition and is divided into two as Word Position Detection Success and Word Reading Success. The speed of OCR engines depends on the system being tested. In particular, offline OCR engines such as Easy OCR and Tesseract OCR depend on the power of the system. For this reason, all OCR engines were tested from the same computer and success measurements were made. The processor of the system in which the tests are carried out is Intel i5-11400H 2.7 GHz, the amount of Ram is 16GB, the Graphics Card is NVIDIA RTX 3050TI (Mobile) and CUDA Version 11.7.

A. Speed Achievement

For companies that use the OCR engine in a product, the document reading speed of the OCR engine is important. The use of a slow OCR engine in a project involving the use of the OCR engine leads to the accumulation of work, thus wasting time for the customer, the project user, and therefore dissatisfaction. For this reason, the speeds of OCR engines are compared in this section. When measuring the speed of the OCR engine, the reading speeds of all the receipts in the dataset used were measured. The sum of these speeds is then averaged by dividing them by the total number of receipts. This results in the average time spent by the OCR engine on a receipt. While the speed of Easy OCR, Tesseract OCR and Nacsoft OCR depends on the system specifications, the speed cannot be measured on the computer where the test results are obtained because the Google Vision Api does not work on the local computer. For this reason, Google Vision Api is not included in the speed measurement. The results obtained in Table III are given.

TABLE III. Comparison of the Duration Achievements of OCR Engines in the English Dataset

OCR Engine	Easy OCR	Tesseract OCR	Nacsoft OCR
Mean of Time (sec)	0.89	1.88	**0.45**

TABLE IV. COMPARISON OF THE WORD READING ACHIEVEMENTS OF OCR ENGINES IN THE TURKISH DATASET

OCR Engine	Easy OCR	Tesseract OCR	Vision Api	Nacsoft OCR
Word Reading Success (CER)	79.49	85.10	91.48	**93.89**
Precision	71.57	82.67	84.12	**88.04**
Recall	78.26	39.88	**92.71**	90.07
F1 Score	74.66	52.68	88.15	**89.01**

TABLE V. COMPARISON OF THE WORD READING ACHIEVEMENTS OF OCR ENGINES IN THE ENGLISH DATASET

OCR Engine	Easy OCR	Tesseract OCR	Vision Api	Nacsoft OCR
Word Reading Success (CER)	88.50	92.90	**94.63**	90.77
Precision	84.90	**92.58**	86.30	83.92
Recall	83.11	75.80	**92.41**	80.91
F1 Score	83.75	82.72	**89.13**	81.88

B. Word Reading Success

Another important metric of OCR engines is Word Reading Success. The accuracy of the information contained in the document read by the OCR engine directly depends on the success of reading words. The success of reading words also affects the success in extracting information from the text read. Although there is no OCR engine that reads all documents correctly, it is desired to choose the OCR engine that gives the highest success possible. In this section, the success of reading the words on the receipts is compared. When measuring reading success, the word matching method mentioned in the Method section was used and then the CER value was measured among the matched words. The achievements are added together and divided by the total number of matched words (TP_n). Thus, the Word Reading Success of a receipt as defined in Eq. (1) is measured. The achievements of these receipts are then summed up and divided by the number of receipts in the dataset. As a result, the Word Reading Success of the dataset is obtained. Table IV shows the success results in the Turkish receipt dataset, while Table V shows the success results obtained from the English receipt dataset.

$$\frac{\sum_{i=1}^{TP_n} CER_i}{TP_n} \tag{1}$$

When Table IV and Table V are examined, Tesseract OCR has high precision and low recall accuracy. This means that the Tesseract OCR engine predicts a small fraction of the words it needs to guess, but the words it predicts are mostly the words that are on the receipt. This difference between the precision and recall categories is only visible in the Tesseract OCR engine. When the results in Table IV are examined, it is seen that the Nacsoft OCR engine, whose methods are described in this study, has the best accuracy rate in almost all categories. When Table V is examined, it is seen that Nacsoft OCR engine gives close results to other OCR engines. Judging by the results of the Vision api in Table IV and Table V, it is seen that it gives the best success in the dataset consisting of English receipts, and in the Turkish dataset, it comes after the Nacsoft OCR engine with a close difference.

TABLE VI. COMPARISON OF THE WORD POSITION DETECTION ACHIEVEMENTS OF OCR ENGINES IN THE TURKISH DATASET

OCR Engine	Easy OCR	Tesseract OCR	Vision Api	Nacsoft OCR
Word Position Detection Success (IoU)	65.34	77.04	**84.03**	78.75
Precision	85.55	80.02	88.82	**92.68**
Recall	93.37	38.85	**97.90**	94.79
F1 Score	89.16	51.26	93.08	**93.69**

TABLE VII. COMPARISON OF THE WORD POSITION DETECTION ACHIEVEMENTS OF OCR ENGINES IN THE ENGLISH DATASET

OCR Engine	Easy OCR	Tesseract OCR	Vision Api	Nacsoft OCR
Word Position Detection Success (IoU)	66.87	**77.73**	75.94	76.86
Precision	92.10	**93.13**	90.01	90.74
Recall	89.92	76.44	**96.38**	86.88
F1 Score	90.72	83.40	**92.96**	88.16

C. Word Position Detection Success

Determining the position of the word is important in extracting information from the text. IoU is used when calculating the success of word positioning. IoU is measured between the matched words in the word matching section mentioned in the method section. By adding up the achievements and dividing them by the total number of matched words, the Word Position Determination success of a receipt data as defined in Eq. (2) is measured. In this way, the IoU success of all receipts in the dataset is measured and the results obtained are collected. Then, the total value obtained is divided by the number of receipts in the dataset, measuring the Word Position Detection Success of the OCR engine in the dataset. Table VI shows the success results in the Turkish receipt dataset, while Table VII shows the success results obtained from the English receipt dataset.

$$\frac{\sum_{i=1}^{TP_n} IoU_i}{TP_n} \tag{2}$$

An examination of the results of the Tesseract OCR engine in Table VI and Table VII shows high precision and low recall as mentioned earlier. When the results of the Easy OCR engine are examined, Table VI and Table VII have the lowest accuracy of word position detection success. The Vision Api has the best success in both Turkish and English datasets. A review of the results of the Nacsoft OCR engine in Table VI shows that it has the best F1-Score accuracy compared to other OCR engines, but it surpasses the Vision Api in word position detection achievement.

VI. CONCLUSION

In the study, OCR engine was developed on Turkish receipt data by using artificial intelligence methods. The success of the developed Nacsoft OCR engine was compared with the success of Tesseract OCR, Easy OCR and Google Vision Api on English and Turkish receipt data. When the results obtained are examined, Nacsoft OCR gives better results in Turkish receipt data than other open source OCR engines, but cannot

reach the same result in English receipt data. This may be due to the fact that the training dataset consists only of Turkish receipts. In addition, the low number of data used in the training of the Nacsoft OCR engine may also adversely affect success. According to this situation, success can be increased by increasing the number of data used in the training of the Nacsoft OCR engine and adding English data to the training dataset. In the training of the Nacsoft OCR engine in future studies, other document types besides the receipt data can be included in the training.

ACKNOWLEDGMENTS

We would like to thank Nacsoft for providing their anonymous receipt data.

DATA AVAILABILITY

The Turkish datasets used in this study to train the models cannot be shared due to they contain personal data. English receipt dataset that used for comparison in our study and also in ICDAR-2019-SROIE competition can be found here.

CONFLICT OF INTEREST

We have no conflicts of interest to disclose.

FUNDING DECLARATION

This work is supported by Turkey's National Small and Medium Enterprises Development Organization (KOSGEB). The project's title is "The development of machine learning software enabling the reading, analysis, and administration of invoices and receipts".

REFERENCES

[1] R. Smith, "An Overview of the Tesseract OCR Engine," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 2, Sep. 2007, pp. 629–633, iSSN: 2379-2140.

[2] "EasyOCR," Sep. 2022, original-date: 2020-03-14T11:46:39Z. [Online]. Available: https://github.com/JaidedAI/EasyOCR

[3] T. M. Breuel, "The OCRopus Open Source OCR System."

[4] Z. Huang, K. Chen, J. He, X. Bai, D. Karatzas, S. Lu, and C. V. Jawahar, "ICDAR2019 Competition on Scanned Receipt OCR and Information Extraction," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, Sep. 2019, pp. 1516–1520, arXiv:2103.10213 [cs]. [Online]. Available: http://arxiv.org/abs/2103.10213

[5] B. Shi, X. Bai, and C. Yao, "An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition," Jul. 2015, arXiv:1507.05717 [cs]. [Online]. Available: http://arxiv.org/abs/1507.05717

[6] H. El Bahi and A. Zatni, "Text recognition in document images obtained by a smartphone based on deep convolutional and recurrent neural network," *Multimedia Tools and Applications*, vol. 78, no. 18, pp. 26 453–26 481, Sep. 2019. [Online]. Available: https://doi.org/10.1007/s11042-019-07855-z

[7] J. Wang and X. Hu, "Gated Recurrent Convolution Neural Network for OCR," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/c24cd76e1ce41366a4bbe8a49b02a028-Abstract.html

[8] R. Atienza, "Vision Transformer for Fast and Efficient Scene Text Recognition," May 2021, arXiv:2105.08582 [cs]. [Online]. Available: http://arxiv.org/abs/2105.08582

[9] O. Alsharif and J. Pineau, "End-to-End Text Recognition with Hybrid HMM Maxout Models," Oct. 2013, arXiv:1310.1811 [cs]. [Online]. Available: http://arxiv.org/abs/1310.1811

[10] X. Tang, Y. Lai, Y. Liu, Y. Fu, and R. Fang, "Visual-Semantic Transformer for Scene Text Recognition," Dec. 2021, arXiv:2112.00948 [cs]. [Online]. Available: http://arxiv.org/abs/2112.00948

[11] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 1457–1464, iSSN: 2380-7504.

[12] M. Busta, L. Neumann, and J. Matas, "Deep TextSpotter: An End-to-End Trainable Scene Text Localization and Recognition Framework," in *2017 IEEE International Conference on Computer Vision (ICCV)*. Venice: IEEE, Oct. 2017, pp. 2223–2231. [Online]. Available: http://ieeexplore.ieee.org/document/8237504/

[13] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun, "An End-to-End TextSpotter with Explicit Alignment and Attention," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA: IEEE, Jun. 2018, pp. 5020–5029. [Online]. Available: https://ieeexplore.ieee.org/document/8578625/

[14] Y. Liu, H. Chen, C. Shen, T. He, L. Jin, and L. Wang, "ABCNet: Real-Time Scene Text Spotting With Adaptive Bezier-Curve Network," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, Jun. 2020, pp. 9806–9815. [Online]. Available: https://ieeexplore.ieee.org/document/9156344/

[15] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "LabelMe: A Database and Web-Based Tool for Image Annotation," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 157–173, May 2008. [Online]. Available: http://link.springer.com/10.1007/s11263-007-0090-8

[16] P. Iakubovskii, "qubvel/segmentation_models," Sep. 2022, original-date: 2018-06-05T13:27:56Z. [Online]. Available: https://github.com/qubvel/segmentation_models

[17] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," May 2015, arXiv:1505.04597 [cs]. [Online]. Available: http://arxiv.org/abs/1505.04597

[18] A. Chaurasia and E. Culurciello, "LinkNet: Exploiting Encoder Representations for Efficient Semantic Segmentation," in *2017 IEEE Visual Communications and Image Processing (VCIP)*, Dec. 2017, pp. 1–4, arXiv:1707.03718 [cs]. [Online]. Available: http://arxiv.org/abs/1707.03718

[19] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Apr. 2015, arXiv:1409.1556 [cs]. [Online]. Available: http://arxiv.org/abs/1409.1556

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Dec. 2015, arXiv:1512.03385 [cs]. [Online]. Available: http://arxiv.org/abs/1512.03385

[21] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," Mar. 2019, arXiv:1801.04381 [cs]. [Online]. Available: http://arxiv.org/abs/1801.04381

[22] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 2018, arXiv:1804.02767 [cs]. [Online]. Available: http://arxiv.org/abs/1804.02767

[23] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," Apr. 2020, arXiv:2004.10934 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2004.10934

[24] Alexey, "Yolo v4, v3 and v2 for Windows and Linux," Oct. 2022, original-date: 2016-12-02T11:14:00Z. [Online]. Available: https://github.com/AlexeyAB/darknet

[25] "Keras: Deep Learning for humans," Sep. 2022, original-date: 2015-03-28T00:35:42Z. [Online]. Available: https://github.com/keras-team/keras