Realtime Detection of Table Objects and Text Areas for OCR Preprocessing

HANGSEO CHOI, JONGPIL JEONG,
Department of Smart Factory Convergence,
Sungkyunkwan University,
Cheoncheon-dong, Jangan-gu Suwon-si, Gyeonggi-do,
REPUBLIC OF KOREA

Abstract: - OCR (Optical Character Recognition) is a technology that automatically detects, recognizes, and digitally converts text into images. OCR has a variety of uses, including reducing human error when viewing and typing documents and helping people work more efficiently with documents. It can increase efficiency and save money by eliminating the need to manually type text, especially when scanning documents or digitizing images. OCR is divided into text object detection and text recognition in an image, and preprocessing techniques are used during the original document imaging process to increase the accuracy of OCR results. There are various preprocessing techniques. They are generally classified into image enhancement, binarization techniques, text alignment and correction, and segmentation techniques. In this paper, we propose a special-purpose preprocessing technique and application called Table Area Detection. Recently, table detection using deep learning has been actively researched, and the research results are helping to improve the performance of table recognition technology. Table detection will become an important preprocessing technology for text extraction and analysis in various documents, and it requires a lot of research and accuracy. While many previous studies have focused on improving the accuracy of OCR algorithms through various techniques, this study proposes a method to discover and exclude false positives by introducing a factor called Table Area Detection.

Key-Words: - Text Detection, Text Recognition, Table Detection**,** Object Detection, Preprocessing, Segmentation

Received: June 17, 2022. Revised: April 22, 2023. Accepted: May 21, 2023. Published: June 21, 2023.

1 Introduction

OCR is divided into two phases: text detection and text recognition. The text detection stage identifies and extracts the regions with text from the image, while the text recognition stage recognizes the actual text from the detected text regions and converts it into computer-understandable text. These two stages are closely linked to improving the performance of the OCR system, and preprocessing is located at the initial stage of text detection.

The purpose and method of preprocessing is to eventually provide the optimal conditions for detecting and recognizing text in an image, but image quality is not the only thing that can get in the way of identifying text. Many papers and experiments have focused on improving OCR performance by preprocessing documents, but the characteristics of documents vary so much that preprocessing is not the answer. A typical example is table detection. To detect text in a document, it is important to correctly identify the text area, and tables complicate the text structure. Between the rows and columns of a table, there are various lines, grids, borders, etc. that create confusion in the process of identifying text areas. Text detection algorithms use a variety of computer vision techniques to detect boundaries, segment text regions, and recognize text blocks. They analyze the brightness, color, texture, and shape of the image to determine the features of the text and identify regions. Tables need to be separated from text at this stage due to the difficulty in detecting boundaries, text overlap, and interference.

Table detection is a task closely related to text detection and recognition in OCR, [1]. Tables are a form of structured text, consisting of rows and columns, with each cell containing textual information. Therefore, it is necessary to understand the structure of the table and extract the contents of each cell, which involves detecting row and column boundaries, identifying regions in each cell, and recognizing text in each cell, [2]. The recognition rate of OCR depends on the quality of the image, [3], and rather than finding the location and size of

the text in the image, it extracts the text information from the entire area within the image and then processes it. However, targeting OCR based on the table area rather than the entire document, and excluding the text area from recognition if it exceeds the table area, will improve the accuracy of the detection target and reduce boundary interference, [4]. This proposal locates the table in the image, extracts the position and size of each cell, and text information, and excludes areas where it is technically difficult to separate the text. Table recognition techniques typically use object detection techniques to locate tables. Object detection is a technique for finding the location and size of objects in an image, [5]. It compares the respective coordinates of the table and text, measures the distance between the two coordinates, and sets a certain threshold value to exclude the text area from the OCR target when it exceeds the area of the table object, [6].

The algorithm for detecting and separating table and text regions is not different from the technique for extracting text information from images. Therefore, the text recognition rate can be improved by clearly specifying the extraction target. In OCR preprocessing, table positions, and text areas can be detected, and table and text boundaries can be separated to improve recognition rates in the text recognition stage. Alignment and correction of text areas can also be performed. By accurately aligning the rows and columns of the table and adjusting the regular placement of the document, text recognition accuracy can be improved.

The paper is organized as follows: Section 2 provides an overview of the technology and the concept of OCR using deep learning, Section 3 presents the overall architecture of the system, Section 4 describes the implementation process, and Section 5 concludes with future research considerations.

2 Related Work

2.1 Faster R-CNN

Faster R-CNN(Faster Regions with Convolutional Neuron Networks features) is an algorithm proposed in 2015, [7], that can perform fast and accurate object detection by compensating for the shortcomings of R-CNN, [8], and Fast R-CNN. It first processes the input image with a CNN(Convolutional Neural Network) to generate a feature map. It then uses an RPN(Region Proposal Network) to generate candidate regions and performs RoI(Region of Interest) pooling on these candidate regions to extract the features of each object. These features are then utilized to perform object classification and bounding box estimation. Recently, there has been a lot of research in the field of table recognition that utilizes it to detect table regions. By utilizing it, table areas can be detected accurately, and tables of various sizes and shapes can be detected.

2.2 YOLO

YOLO (You Only Look Once) is an algorithm proposed in 2016 that provides fast speed and high accuracy. It divides the input image into a grid and predicts the probability of the bounding box and corresponding object in each grid cell. It uses these predictions to perform object classification and bounding box estimation. It has been widely used in the field of table recognition recently due to its fast speed and high accuracy. YOLOv3, [9], provides both high accuracy and fast speed, and it can detect tables of various sizes and shapes. Recently, various object detection algorithms, including Faster R-CNN and YOLO, have been developed and continue to be used in the field of table recognition to provide high recognition rates and fast speeds.

2.3 Mask R-CNN

Mask R-CNN, [10], is a state-of-the-art object detection and instance segmentation algorithm that has been proven to be effective for a variety of tasks, including table segmentation. It is an extension of Faster R-CNN, which performs object detection and object segmentation simultaneously, [11]. Therefore, a method is proposed to utilize it to detect table regions within document images and perform table recognition based on them. This method is performed in the following steps.

- Step 1: Perform object detection and object segmentation in the image.
- Step 2: Extract table regions from the object segmentation results.
- Step 3: Perform table recognition based on the extracted table regions, [11].

In recent years, there have been many advances in it that have improved its performance on table segmentation tasks. One of the most important advances is the use of attention mechanisms, which allow the model to focus on specific regions of the image when making predictions. They are particularly effective for segmenting tables, as tables are often difficult to distinguish from other objects in an image. Another important development is the use of data augmentation techniques. Data

augmentation techniques are used to artificially increase the size of a training dataset. This can help improve the model's performance on unseen data. Many improvements have been made to its architecture of it, which can be used to segment tables in a variety of documents, including scientific papers, news articles, and legal documents.

2.4 Table Net

End-to-end models that perform table recognition and OCR simultaneously are a relatively new area of research, [12]. Deep learning techniques and large datasets for table recognition, [13], and OCR have become available, and because of these advances, end-to-end models, [14], that perform table recognition and OCR simultaneously are becoming increasingly common. These models have several advantages over traditional approaches. First, they are more accurate. Traditional approaches often perform table recognition and OCR separately. This can lead to errors because the results of one step can affect the results of the other. End-to-end models, on the other hand, perform table recognition and OCR simultaneously, [15], which helps ensure the accuracy of the results. Second, it's more efficient. Traditional methods for table recognition and OCR can be time-consuming. The end-to-end model, on the other hand, is much faster. This is because it performs both tasks simultaneously. Third, it's more flexible. Traditional table recognition and OCR methods are often limited to certain types of tables, while end-to-end models can be used to recognize, [16], and OCR a wide variety of tables. Overall, end-to-end models that perform table recognition and OCR simultaneously are promising new approaches. They are more accurate, efficient, and flexible than traditional approaches, so they are likely to become increasingly common in the future.

2.5 OCR

A typical example is deep learning-based OCR. Deep learning algorithms are used to perform character recognition, which creates a model to classify characters by learning features extracted from the character area. Typically, a convolutional neural network (CNN) is used. CNNs are responsible for extracting features within an image and recognizing characters based on the extracted features. The second is template matching-based OCR. Template matching recognizes characters by calculating the similarity between the input image and the template image. A template image for each character is created in advance, and the most similar character is recognized by measuring the similarity between the input image and the template image. However, this method is vulnerable to variations in size, rotation, and distortion, and requires the preparation of many template images. Finally, statistical-based OCR recognizes characters by analyzing their statistical characteristics. A statistical model learns the frequency, occurrence pattern, probability distribution, etc. of each character, and recognizes the characters in the input image based on the statistical model. This method makes good use of the statistical characteristics of language to improve recognition performance, but it requires a large amount of training data and requires the use of models specialized for a particular language.

3 Proposed Method

3.1 System Process Flow

The main process performs preprocessing on the input image, as shown in Fig. 1, and then separates the table and text areas. If there is text outside the table area or overlapping, it is processed separately, and text recognition is performed on the text that exists in the area.

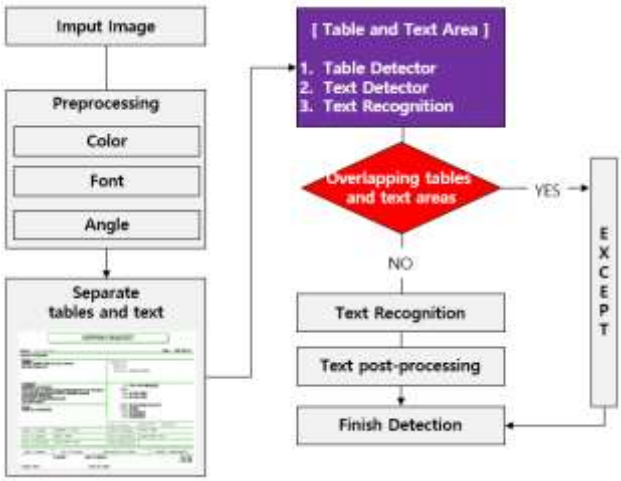

Fig. 1: System Process Flow

After image input and preprocessing, the process is divided into two stages: Table Detection and Text Detection. Each step extracts only table and text information while keeping the document structure intact. In step 1, Faster R-CNN was used to train table detection.

Faster R-CNN combines two main components, RPN and Fast R-CNN, to effectively detect objects and generate bounding boxes. Since table detection

involves identifying rectangular regions of a certain shape within an image or document, we believe that Faster R-CNN, which simultaneously performs object detection and bounding box generation, is suitable for table detection. Fig. 2 shows the structure of Table Detection, including Faster R-CNN.

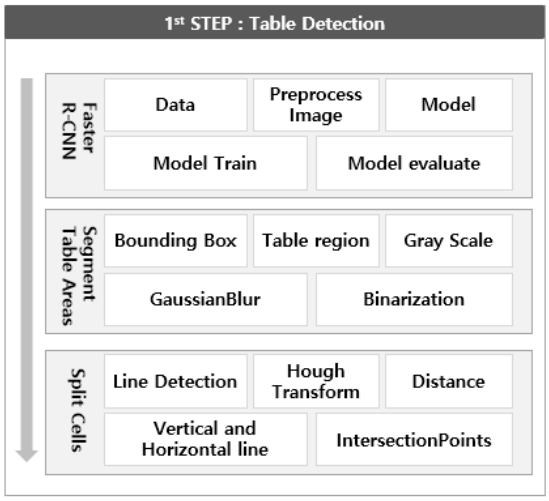

Fig. 2: 1st SETP Table Detection

In step 2, we used CRAFT (Character Region Awareness for Text Detection) to detect text, as shown in Fig. 3. CRAFT is a deep learning-based algorithm for character region detection that specializes in detecting character regions in images containing text, and can accurately identify the location and shape of the text.

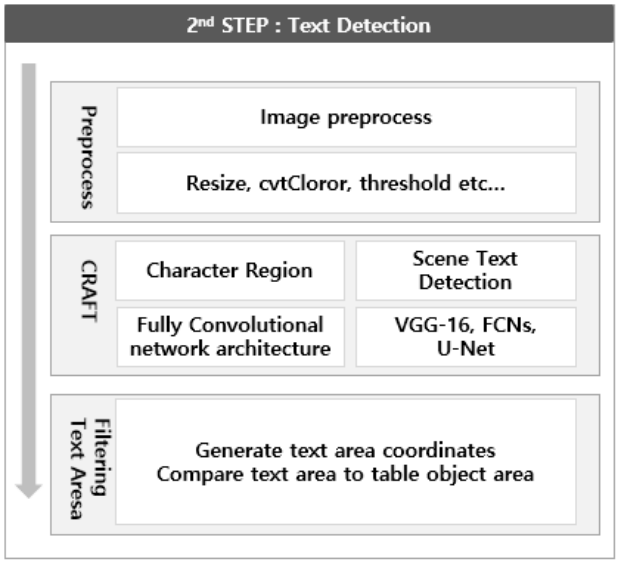

Fig. 3: 2nd STEP Text Detection

The idea of this paper consists of finding a table in an image and extracting the position, size, and text information of each cell, and the main steps are shown below.

- Table recognition with object detection, [17]: Use the image dataset and the table bounding box information of the image to find the table in the image using Faster R-CNN and train the object detection model.
- Segment the table region: segment the table region in the original image using the table bounding box information obtained from the object detection model. The segmented table image is preprocessed for processing in the next step.
- Cell segmentation: segment the cell region by finding the structure of rows and columns in the table image. For this purpose, we used Hough transform, a line detection algorithm, [18], but finally used an object detection model.
- Text extraction via OCR: We use the easy OCR engine to extract the text of each cell region. Preprocess the extracted text to remove noise and extract coordinates for the text area.
- Filter text area: Compares the area information of the table object with the text coordinates of each cell and excludes the text area from OCR if it exceeds the area of the table object based on the threshold value you set.

3.2 Table Detection

To use the Faster R-CNN model, we first define the model architecture, which uses a backbone network based on ResNet-50 and an FPN(Feature Pyramid Network) to extract feature maps. ResNet-50 is a 50-layer residual network, a deep CNN with excellent performance in object recognition and classification, which we used as the basic structure to extract features from images. The FPN detects objects of different sizes by extracting feature maps of different sizes, and it takes the feature maps extracted by ResNet-50 as input and incorporates features from higher levels into lower levels to produce an improved feature representation.

Faster R-CNN consists of RPN, which suggests RoI in an image, and Fast R-CNN, which classifies these regions and refines the bounding box. Model training is prepared by configuring RPN.

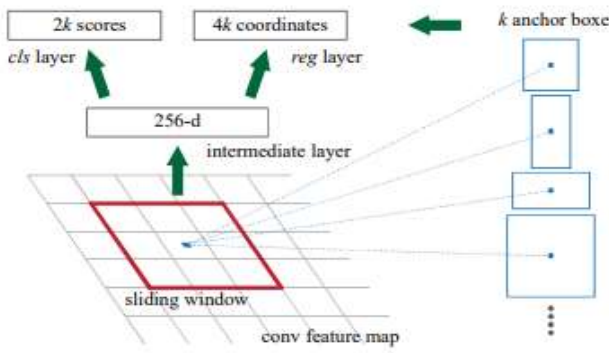

Fig. 4: Towards Real-Time Object Detection with RPNs, [7]

Fig. 4 shows the structure of an RPN that generates bounding box candidates for an object while sliding over an input image or feature map using anchor boxes of different sizes and proportions.

The RPNs are used to generate anchor boxes for bounding box regression and classification. The anchor box generates anchor boxes at each location in the feature map. Anchor boxes are reference Bounding Boxes with different sizes and ratios to correspond to different object sizes and aspect ratios in the image. Multiple anchor boxes are generated at each location with anchor sizes and ratios defined by hyperparameters.

RPNs are trained using a multi-task loss function consisting of a binary classification loss and a bounding box regression loss. The binary classification loss typically uses a binary cross-entropy loss, and the regression loss typically uses a smooth L1 loss. The two losses are combined to compute the final loss, and the weights are updated to minimize it. The loss function formula for this process is as follows. Binary Cross-Entropy Loss The binary cross-entropy loss is used by the RPN to classify whether each anchor box is an object or background. The formula is as follows.

$$L_{cls} = -\frac{1}{N_{cls}}\sum[\mathrm{p}_i \log(p_i') + (1 - \mathrm{p}_i)\log(1 - p_i')]$$

L_{cls} : Binary classification loss
N_{cls} : Mini batch size
p_i: actual label of the i-th anchor box (object: 1, background: 0)
p_i' : Model's predicted probability for the i-th anchor box (probability of being an object)

Bounding Box Regression Loss (Smooth L1 Loss) The Bounding Box Regression Loss is used in the process of aligning the coordinates of the anchor boxes in the RPN with the actual bounding boxes. The formula is as follows.

$$L_{reg} = -\frac{1}{N_{reg}}\sum SMOOTH\_L1\,(\mathrm{t}_i - t_i')$$

L_{reg} : Bounding Box regression loss
N_{reg} : Mini batch size
t_i: Actual Bounding Box coordinates of the i-th anchor box (x, y, w, h)
t_i': Model's predicted Bounding Box coordinates for the i-th anchor box (x, y, w, h)

SMOOTH_L1: Smooth L1 loss function
Now combine the two losses to calculate the final loss.

$$L_{total} = L_{cls} + \lambda L_{reg}$$

L_{total} : Final loss
λ: Weight of the Bounding Box regression loss (set as a hyperparameter)

Update the weights of the RPN model in the direction of minimizing the final loss L_total calculated in this way. Once the object detection results for the image are obtained, filter out the bounding boxes with low confidence among the model inference results, or apply NMS(Non-Maximum Suppression) to remove redundant bounding boxes. From these filtered and refined results, we extract the corresponding bounding box coordinates in a table and prepare them for comparison with the text extraction area.

3.3 Text Detection

CRAFT performs the steps of extracting the features of an image to detect text regions. It mainly uses CNN to generate feature maps at different scales, which refers to the process of extracting structural information about text to identify text regions in an input image. CNN learn regional patterns in an image and uses them to distinguish between text and non-text regions in an image. Passing an image through multiple CNN layers creates a multi-level feature map, which can capture different sizes and details of text areas. A small-scale feature map helps recognize small text areas, while a large-scale feature map helps recognize large text areas. The generated text candidate regions are then adjusted with a bounding box in a subsequent step. Feature maps are also used to extract text structure information. Feature maps are good at capturing text features such as sharp boundaries, strong vertical lines, text orientation, etc., and utilize this

information to accurately detect text regions. Fig. 6 shows an example of text detection from a table in a document.

Fig. 5: Text Detection by CRAFT

3.4 Text Recognition

Text Recognition uses the easyOCR algorithm. easyOCR is a deep learning-based character recognition technique that aims to recognize the bounding box of a text and follows the following steps.

- Preprocessing: The preprocessing step is important to improve the quality of the image and make it easier for the deep-learning model to classify the characters. Common preprocessing techniques include removing noise from the image, such as dust, scratches, and uneven lighting, and adjusting the contrast of the image to make the characters more visible.
- Segmentation: The segmentation step is used to break down the image into individual characters. This is done by identifying connected components in the image. Connected components are groups of pixels that are all connected to each other. Connected components that are the same size as a character are then categorized into characters, [19].
- Classification: The classification stage is used to identify individual characters in an image, [20] This is done using a deep learning model. The deep learning model is trained on a large dataset of images containing labeled characters. The model learns to identify characters by their shape, size, and texture.
- Reconstruction: The reconstruction step is used to combine individual characters into text, [21]. This is done by aligning the characters according to their bounding boxes and then combining them together.

4 Experiment and Results

4.1 Experimental Environments

This experiment was run on Google Corel Pro with the following hardware: GPU NVIDIA T4, CPU Inter(R) Xeon(R) CPU, 16GB RAM Software: Development language: Python 3.8, CUDA version: 11.8, Deep Learning framework: PyTorch 1.8.1, Required Python libraries: OpenCV, NumPy, PIL, etc. To build the environment by installing the necessary Python packages and libraries, we customized the SR (Shipping Request) document, which is used in the field for test data other than training. Colab is an online implementation of Jupiter Notebook that allows you to write and run Python code, and you can use PyTorch to implement the table detection model and text recognition model.

4.2 Data Set

The model was trained using a prepared custom dataset and the Faster R-CNN model. The custom dataset utilized delivery request documents used in the field. The customized documents were collected from actual documents used in the shipping and logistics industry and consisted of 4 templates from 27 companies, totaling 112 files. Each file contains titles, diagrams, text, and images such as company logos. This data was used for testing purposes rather than training purposes and was not pre-trained.

4.3 Table Detection by Faster R-CNN

It uses the Faster R-CNN to find a table in an image. The document image was resized and normalized to preprocess it into a format suitable for the Faster R-CNN model. Although Faster R-CNN does not require a fixed input size, we resized the image to a reasonable size of 800-1000 pixels wide by 1000 pixels high, considering GPU memory limitations and training time, model performance, and computational efficiency, while maintaining the aspect ratio of the original image. The reason for normalizing the image is to stabilize the training process by making the distribution of pixel values constant and to speed up convergence, so we scaled

the image pixel values to the range [0, 1], and then performed normalization. As shown in Fig. 5, you can build a single pipeline to extract surface areas and table areas in real-time.

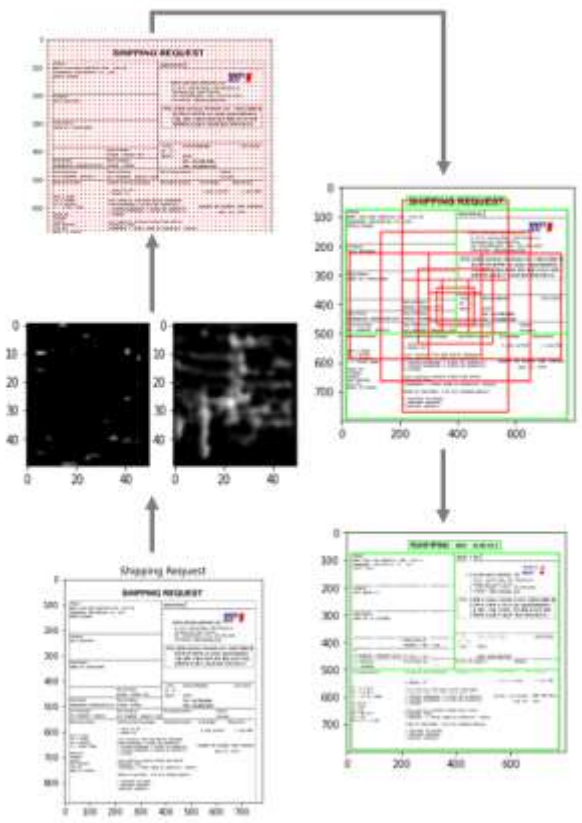

Fig. 6: Detect Table Process and results

4.4 Text Detection by CRAFT

The experiments were divided into 1 and 2: Experiment 1 recognized only text without table detection, and Experiment 2 tried to recognize text in the table area with table detection results. Since CRAFT is a pre-trained model for text detection, we were able to extract text directly from the image without having to train a separate object detection model. However, when tables in the document overlapped with table areas, it sometimes failed to recognize the table structure. The Text Detection Results are presented in Fig. 7.

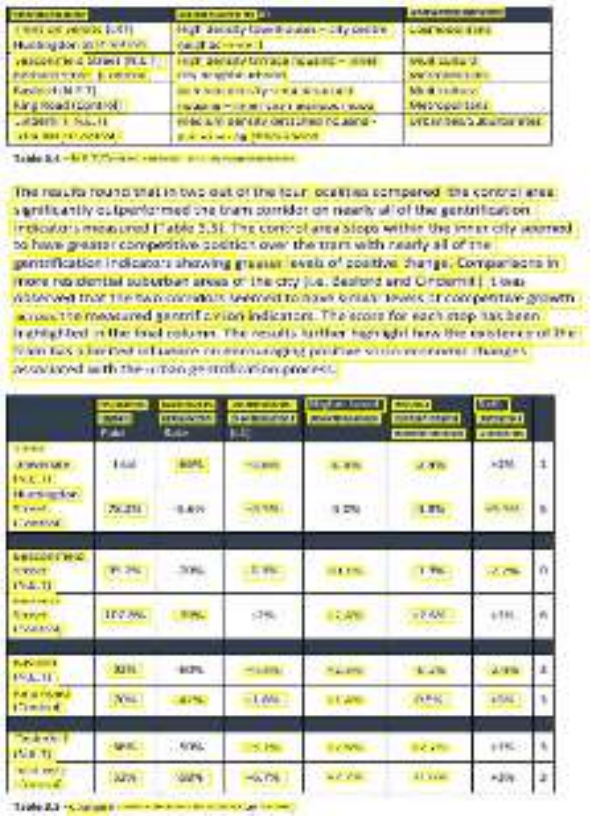

Fig. 7: Text Detection Results

4.5 Results

For Experiment 1, we performed text extraction using only CRAFT and easyOCR to check the text detection rate without table detection. To perform Experiment 2, we prepared a custom dataset to train the Faster R-CNN model, extracted the bounding box to obtain the coordinate information, and then segmented the table area with the table recognition output coordinates and extracted the text information with Faster R-CNN to obtain the coordinates of each text area. To compare the coordinates of the most important table and text regions during the implementation phase, we measured the distance between the two coordinates and set a certain threshold to exclude the text region from OCR if it exceeds the region of the table object. This allowed us to extract only the text of the table and check the text extraction accuracy compared to the steps in Experiment 1, and we found a performance improvement of about 3%. The results in Table 1 show that there are a total of 4 document template types, and the more templates you train, the higher the table detection rate.

Table 1. Table detection rate as training documentation templates increase

Template	Data	Table Detect rate
4	112	82%
3	112	73%
2	112	70%
1	112	65%

The results in Table. 2 show the results of Experiments 1 and 2 separately. Experiment 1 (column name: Text Detect 1) only tried to detect text without detecting tables. Experiment 2 (column name: Text Detect 2) detects the table and text, removes duplicates, and detects the text. As a result, we can see that removing and detecting duplicates in the table can contribute to performance improvement.

Table. 2. Plain text detection rate and table detection rate after excluding duplicate regions

Table Detect rate	Text Detect 1	Text Detect 2
82%	86%	89%

5 Conclusions

In this paper, we presented an approach to detecting tables in images, extracting the location, size, and textual information of each cell, and comparing them to text regions. The efficient table recognition and text extraction pipeline is a fusion of deep learning models and commercial open source. The pipeline includes all the necessary steps to process information related to tables in document images, and we believed that a method that measures the distance between two coordinates and excludes them from OCR if the text area exceeds the area of the table object according to a certain threshold would improve the text detection rate compared to recognizing an unspecified number of documents. In addition, a coordinate-based region overlap filtering method to implement this concept would help improve the accuracy of text extraction and preserve the structure of the table. Nevertheless, we were disappointed that we had to use a custom dataset due to the lack of data. As it may be difficult to ensure the generalization of the model in various situations, we expect that repeating the same training process and experiments with an advanced model trained on a large dataset would lead to improved performance and universality.

Acknowledgment:
This research was supported by the SungKyunKwan University and the BK21 FOUR(Graduate School Innovation) funded by the Ministry of Education(MOE, Korea) and the National Research Foundation of Korea(NRF). And this work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1F1A1060054). Corresponding authors: Professor Jongpil Jeong.

References:

[1] M. Kasem, A. Abdallah, A. Berendeyev, E. Elkady, M. Abdalla, M. Mahmoud, M. Hamada, D. Nurseitov, I. Taj-Eddin, Deep learning for table detection and structure recognition: A survey, arXiv:2211.08469v1 [cs.CV], 2022.

[2] Y. Li, L. Gao, Z. Tang, Q. Yan, Y. Huang, A GAN-Based Feature Generator for Table Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, 03 February 2020.

[3] D. G Lee. CNN-based Image Rotation Correction Algorithm to Improve Image Recognition Rate, *The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)* Vol. 20, No. 1, 2022, pp.225-229, JIIBC 2020-1-32.

[4] C. B. Jang, Implementation of Pre-Post Process for Accuracy Improvement of OCR Recognition Engine Based on Deep-Learning Technology, *Journal of Convergence for Information Technology*, Vol. 12. No. 1, 2022, pp. 163-170.

[5] J. S Choi, Table Detection Scheme based on Deep Learning, Proceedings of the Korean Computer Conference, 2022, 930 – 932.

[6] J. Hu, R. S. Kashi, D. Lopresti, and G. T. Wilfong, Evaluating the performance of table processing algorithms, *International Journal on Document Analysis and Recognition*, Vol. 4, 2022, 140–153,

[7] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards 14 real-time object detection with region proposal networks, *in Neural Information Processing Systems (NIPS)*, 2015, pp 91-99.

[8] R. Girshick, J. Donahue, T. Darrell, J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, 580–587.

[9] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You Only Look Once: Unified, Real-Time Object Detection, *e IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp 7263-7271

[10] K. He, G. Gkioxari, P. Dollar, R. Gir-shick. Mask R-CNN. *IEEE International Conference on Computer Vision (ICCV)*, 2017, 2961–2969

[11] Y.W. Lee, J.Y. Park, CenterMask: Real-time anchor-free instance segmentation, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[12] D. W. Embley, M. Hurst, D. Lopresti, and G. Nagy, Table-processing paradigms: a research survey, *International Journal of Document Analysis and Recognition (IJDAR)*, Vol. 8, No. 2-3, 2016, pp. 66–86.

[13] I. Kavasidis, S. Palazzo, C. Spampinato, C. Pino, D. Giordano, D. Giuffrida, and P. Messina, A saliency-based convolutional neural network for table and chart detection in digitized documents, *arXiv preprint arXiv:1804.06236*, 2018.

[14] S. Appalaraju, Jasani, B. Kota, B.U, X. Y. Manmatha, R. Docformer, End-to-end transformer for document understanding. *In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 993–1003

[15] Y. Baek, B. Lee, D. Han, S. Yun, H. Lee, Character region awareness for text detection. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* 2019. pp. 9357–9366

[16] A.W. Harley, A. Ufkes, K.G Derpanis, Evaluation of deep convolutional nets for document image classification and retrieval, *13th International Conference on Document Analysis and Recognition (ICDAR)*, 2015, pp. 991–995

[17] M. Fan, D.S. Kim, Table region detection on large-scale PDF files without labeled data. *CoRR, abs/1506.08891*, 2015.

[18] B. Gatos, D. Danatsas, I. Pratikakis, and S.J. Perantonis, Automatic table detection in document images. *In Proc. of ICAPR (2005) - Volume Part I, ICAPR*, 05, Berlin, Heidelberg. Springer-Verlag, pages 609–618.

[19] S. A. Oliveira, B. Seguin, F. Kaplan, segment: A generic deep-learning approach for document segmentation. *ICFHR 2018*.

[20] R. Child, S. Gray, A. Radford, I. Sutskever, Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509,* 2019.

[21] D. Deng, H. Liu, X. Li, D. Cai. Pixellink: Detecting scene text via instance segmentation. *In AAAI. 2018*.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

-Hangseo Choi set the research topic and goals, developed the software, conducted the experiments, validated, and wrote the paper.

-Professor Jongpil Jeong conceptualized the idea, presented the methodology, and conducted the review.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

This research was supported by the SungKyunKwan University and the BK21 FOUR(Graduate School Innovation) funded by the Ministry of Education(MOE, Korea) and the National Research Foundation of Korea(NRF). And this work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. 2021R1F1A1060054). Corresponding authors: Professor Jongpil Jeong.

Conflict of Interest

The authors have no conflict of interest to declare.