Borderless Table Detection and Extraction in Scanned Documents

Nakul Aggarwal, Snehlata Sinha, Abhilasha Lodha,
Debasmita Ghosh, Shalini Singh, Dharmendra Chaurasia

Advance Technology Centers in India, Accenture
Email: {nakul.aggarwal, snehlata.sinha}@accenture.com, abhilashalodha0101@gmail.com,
{debasmita.ghosh, shalini.am.singh, dharmendra.chaurasia}@accenture.com

***Abstract*—**
Automated detection and extraction of tabular data from scanned documents is a difficult task. Detecting and extracting information from tables without any borders becomes more challenging even with the state-of-the-art OpenCV based solutions. There have been few approaches suggesting the solutions based on deep learning models but none of the open-source solutions performed well when table boundaries are not visible. In this paper, we are proposing a novel end-to-end solution that works well with borderless tables of different layouts and templates. The proposed method consists of two stages. First stage is automated table detection irrespective of the format, border and layout of the scanned documents using custom trained deep learning model. Second stage is a novel OpenCV based custom algorithm which utilize blob detection to define row-column relationship of each cell. This information is used to extract the data into .csv/.xlsx format. Our proposed algorithm was evaluated on more than 100 images and has achieved an average table extraction accuracy of 98.4% outperforming many state of the art models.

I. Introduction

Human society in today's world has an ever increasing wealth of information with millions of paper documents being produced everyday.Tables are extensively used for exhibiting structural and functional information [1]. They are effective and compact means of displaying information such as numeric values [2] to human readers as they make such information easy to search, compare and understand. Therefore, table detection from documents is considered an essential part of a document analysis through the extraction of summarized and well-organized information [3]. However, extraction of tabular data in scanned documents using Optical Character recognition (OCR) is a challenging task due to the presence of a wide variety of table formats and layouts.

A lot of OpenCV and OCR based solutions have been proposed for tables with visible lines and borders. However, they are not able to perform well on the borderless tables i.e. tables without horizontal and vertical lines for demarcation. Our paper proposes an end to end pipeline which takes an input scanned document and extracts the tabular data in an editable format(.csv/.xlsx). We have collected borderless table dataset of 300 images, 150 from TableBank [4], 75 from internet and 75 are custom scanned invoice images. This data contains variations like different backgrounds, brightness, angles, etc to accommodate real-life scenarios.

The input image is pre-processed and sent for table detection and extraction. For table detection, a custom trained deep learning model is used to detect the tables irrespective of borders, format or layout. For table extraction, a novel algorithm is designed to define the row column relationship of each cell as depicted in the section III. Tesseract [5] is used to extract raw data from the identified cells. The detected cell ROI are pasted on an empty image to decrease the amount of time it takes for Tesseract extraction as explained further in section III-E. Once cell extracts are retrieved, they are arranged back in the same tabular structure to the spreadsheets.

In order to check the performance of our solution, the parameters taken into consideration are time taken for extraction, precision, recall and F1-Score as shown in section IV.

II. Literature Review

Jihu Kim and Hyoseok Hwang [6] presented a rule based method for detecting tables in website images. They proposed a pipeline in two stages, feature extraction based on constraint rules followed by grid pattern recognition using rules and tree structures. They were able to reduce number of false positives for non-text blocks in images. Their method outperforms previous methods for website images. However, the performance is not good for document images.

Sahoo et al. [1] have proposed the Auto- Table-Extract system to automatically detect and extract information from the table in a pdf file into an excel file.It performs well on scanned and searchable files. However, p erformance is not good for partially bordered and borderless tables.

Prasad et al. [7] proposed instance segmentation based CascadeTabNet to recognize structures within tables by predicting table cell masks while using the line information as well. It performs good on public datasets and suggests improving the post-processing modules for increasing accuracy.

Fisher et al. [8] presented a multistage pipeline for table detection and table structure recognition with document alignment and color invariance pre-processing methods. Their solution does not generalize for complex tables with cells or entire table within a cell.

Shafait and Smith [9] proposed an approach using a layout analysis of Tesseract [17] with tab-stops used as an indication of where a text block starts and ends. Their table detection approach competes well with many commercial OCR systems. The detection accuracy of their algorithm for the UNLV dataset achieved a precision of 86% and a recall of 79%.

Hassan et al. [10] detect tables in PDF documents utilizing both ruling lines and content layout. However, their method utilizes these two sources separately. And lots of false positive tables are detected because the detected graphic lines are not verified first in their method.

Similarly, Fang et al. [11] propose a method via both visual separators and tabular structures of contents. The separators refer to not only graphic lines but also white spaces to handle borderless tables. This method detects page columns in the first place to assist table detection in multi-column pages, and achieves a satisfactory accuracy rate.

Liu et al. [12] [13] proposed a method in PDF based on the notice that almost all the table rows are sparse lines. Therefore, the table boundary detection problem could be simplified into the sparse line analysis problem with much less noise. The authors designed eight line label types and applied two machine learning techniques, Conditional Random Field (CRF) and Support Vector Machines (SVM), on the table boundary detection field.

III. Methodology

Our solution offers an end to end flow for borderless table detection and extraction. Fig. 1 illustrates the pipeline for the same. The document uploaded is passed to pdf to jpg module in case it's a pdf. The input images are then passed to the skewness correction module. This is followed by a deep learning based object detection module to detect bordered and borderless table layouts present in the image. In case of images consisting of borderless tables, the detected table crop is then pre-processed using OpenCV techniques and passed for detection of cells in the table. The cells are detected using blob-detection technique which is then followed by cell row-column relationship module to define the position of cells in rows and columns. Text extraction module extracts the data present in table cells followed by data writing to support output formats. Data writing module has the capability of writing the detected table extracts to a spreadsheet table which is a CSV file. All these different modules are packed together to create the borderless table detection and extraction pipeline.

A. Data Acquisition

Our research is focused on extracting borderless tables of both uniform and non uniform types. Our model consists of 2 classes, bordered and borderless tables as shown in Fig. 2. Around 300 images were collected, 150 per class from TableBank, 75 from internet and 75 are custom scanned invoice images. The dataset has varied angles, layout, background and quality to accommodate multiple variations. The dataset included invoices, scanned contract document images, receipts etc. The input images are in .jpg, .png and .jpeg format. The images are then annotated for tables using the annotation tool LabelImg [14] and json files gets created for each of them.

B. Object detection model for table detection

An object detection model is trained on custom dataset to identify bordered and borderless tables in the image. The architecture used is SSD Mobilenet V1 FPN [15] pretrained on coco dataset [16]. The trained model was converted to TensorFlow Lite (TF Lite) [17] model for improved performance.

The main advantage of using the SSD architecture is its ability to detect multiple objects in the image with a high MAP score. It also provides best accuracy tradeoffs within the fastest detectors.

In our pipeline, the input image is initially checked for skewness. An OpenCV based module is used to calculate and adjust the angle of the input image. The skew-corrected image is then passed through the custom trained object detection TF Lite model. The custom model was trained using x2gd.xlarge ec2 instance with 4 vCPU and 64GiB memory. If the class of detected table in the image is 'bordered table', it is passed to the bordered table pipeline. If the class of detected table is 'Borderless table', coordinates of the detected table are used to extract the table region from the image. A table crop is generated and sent for further processing to extract table data as explained in section III-C.

C. Image Preprocessing

To handle table with varied backgrounds, adaptive thresholding was done on the input image. To get better cell detection, any horizontal or vertical lines present in the image were removed using the line removal module. The cropped table RoI detected image was further passed to the pre-processing stage i.e. conversion to gray scale image by applying denoising filters. The resultant image after pre-processing is shown in the Fig. 4

D. Cell RoI Detection and Extraction

After the table image was pre-processed, cells were detected using the OpenCV based cell blob detection technique. A blob is a group of connected pixels that share a common property and is widely used technique in OpenCV based applications.

Cells blobs were detected by doing contour detection on the vertically dilated image as shown in the Fig. 5 and detected cells were shown in Fig. 6 on table image.

To detect the position of the cells, a list of the coordinates (x,y,w,h) of the contours was extracted and were sorted from top-to-bottom. These coordinates were used to define the row-column relationship of each cell in the table with accurate starting and ending row-column cell information.

Our custom algorithm can be used to extract data from all types of borderless tables. Here we used minimum x and y values, to trace along the table in ascending order to detect each row and column. The row-column relationship for each cell is defined using five parameters namely cell coordinates,

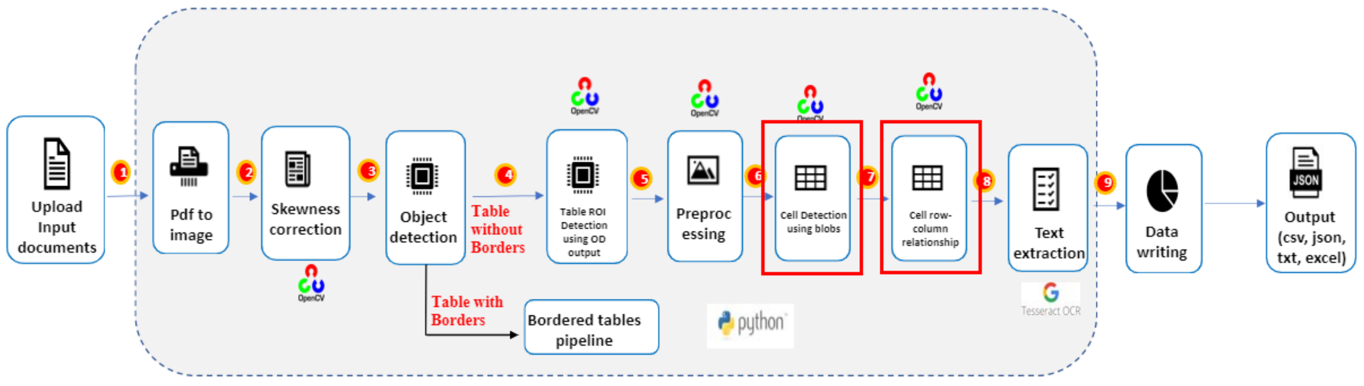


Fig. 1: Borderless table detection and extraction pipeline

start row number, end row number, start column number and end column number. To check if the table is uniform or non-uniform, we also calculated average height of cells in the same row. If cell height is less than or equal to average height, then table is tagged as a uniform borderless table, else it is tagged as a non-uniform borderless table.

In case of non-uniform tables, we used maximum y values of merged cell and traced along the table until its value matched with minimum y value of an another cell. Once done, the end row number of the merged cell would be equal to the start row number of the identified cell.

Finally a refined table content was extracted and the cell row-column information was sent to a csv writer module to interpret the formation and provide a csv/json output as shown in Fig. 7.

E. Performance Optimization

Reduction of pyTesseract API calls using OpenCV Techniques - OpenCV processing is known for being very fast with images. When cell RoI crops are detected, each crop image is then converted into text using Tesseract OCR. The pyTesseract API takes about a second approximately when each cell RoI crop are sent to it. This way, the computational time increased in the cell extraction module. In order to resolve this issue, a new approach is introduced in this paper.

A background mask is created by calculating the height and width of all cell RoI crops. In in the background mask, all the crops are arranged one below the other, having a delimiter in between, and then the image is sent to pyTesseract API for the text extraction. Fig. 8 shows the way the RoI crops were arranged with delimiter RoI. Fig. 8 (a) shows the first round of implementation where we used a white background mask for arranging the ROI crops and then sending it for Tesseract extraction. With this approach, as the RoI crop background wasn't necessarily being white in color all the time, the OCR extraction had some faults. Then in Round 2 (shown in Fig. 8 (b)) the RoI crop's background pixel was detected and then that pixel was used as a background mask color. This reduced the OCR faults while doing text extraction.

TABLE I: Time taken for Table Data Extraction once cell RoIs are detected

Table	Basic Approach	Background mask approach
Table with 24 cells	24 seconds	3 seconds
Table with 50 cells	50 seconds	4 seconds

The cell data which is extracted from Tesseract, is segregated back to rows and columns using the delimiter and coordinates or the RoI crops. This approach reduces the time taken by the pyTesseract API to extract the textual data out of the tables.

Table I describes the time differences in processing text extraction from RoI crops using the default approach of sending each RoI crop for text extraction vs sending one image with arranged RoI crops.

IV. Result and Conclusion

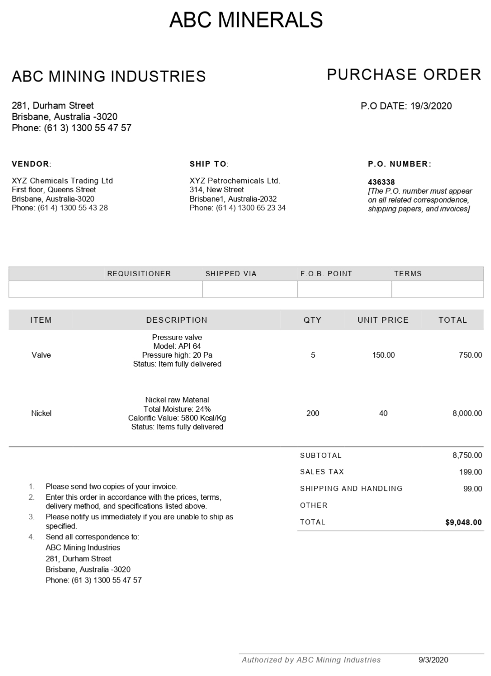
Our proposed solution offers multiple novelties over the other state of the art algorithms. Being a two-step approach, our solution aids in accurate table detection and row and column data extraction.

At the first step, our custom trained object detection module identifies borderless tables in scanned input documents explicitly. Then at the next step, an OpenCV based novel algorithm identifies cells (rows and columns) from the table extracted in the previous step. This algorithm defines row-column relationship for each cell of the table. Accuracies for both steps are mentioned in Table II.

TABLE II: Accuracy

Table Detection Accuracy	96.4%
Data Extraction Accuracy	98.8%

The performance of our system is evaluated based on the three model evaluation metrics termed as precision, recall and F1 score.



ABC MINERALS

ABC MINING INDUSTRIES

PURCHASE ORDER

281, Durham Street
Brisbane, Australia -3020
Phone: (61 3) 1300 55 47 57

P.O DATE: 19/3/2020

VENDOR:
XYZ Chemicals Trading Ltd
First floor, Queens Street
Brisbane, Australia-3020
Phone: (61 4) 1300 55 43 28

SHIP TO:
XYZ Petrochemicals Ltd.
314, New Street
Brisbane1, Australia-2032
Phone: (61 4) 1300 65 23 34

P.O. NUMBER:
436338
[The P.O. number must appear on all related correspondence, shipping papers, and invoices]

REQUISITIONER	SHIPPED VIA	F.O.B. POINT	TERMS

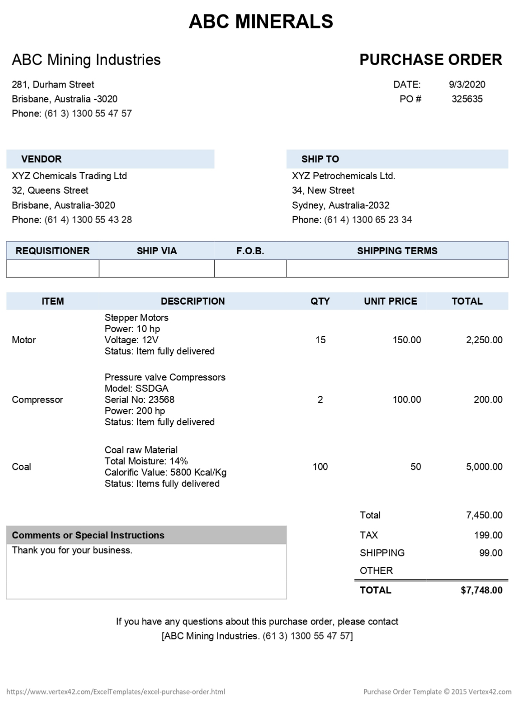
ITEM	DESCRIPTION	QTY	UNIT PRICE	TOTAL
Valve	Pressure valve Model: API 64 Pressure high: 20 Pa Status: Item fully delivered	5	150.00	750.00
Nickel	Nickel raw Material Total Moisture: 24% Calorific Value: 5800 Kcal/Kg Status: Items fully delivered	200	40	8,000.00

SUBTOTAL	8,750.00
SALES TAX	199.00
SHIPPING AND HANDLING	99.00
OTHER	
TOTAL	**$9,048.00**

1. Please send two copies of your invoice.
2. Enter this order in accordance with the prices, terms, delivery method, and specifications listed above.
3. Please notify us immediately if you are unable to ship as specified.
4. Send all correspondence to:
   ABC Mining Industries
   281, Durham Street
   Brisbane, Australia -3020
   Phone: (61 3) 1300 55 47 57

Authorized by ABC Mining Industries 9/3/2020

(a)



ABC MINERALS

ABC Mining Industries

PURCHASE ORDER

281, Durham Street
Brisbane, Australia -3020
Phone: (61 3) 1300 55 47 57

DATE: 9/3/2020
PO # 325635

VENDOR
XYZ Chemicals Trading Ltd
32, Queens Street
Brisbane, Australia-3020
Phone: (61 4) 1300 55 43 28

SHIP TO
XYZ Petrochemicals Ltd.
34, New Street
Sydney, Australia-2032
Phone: (61 4) 1300 65 23 34

REQUISITIONER	SHIP VIA	F.O.B.	SHIPPING TERMS

ITEM	DESCRIPTION	QTY	UNIT PRICE	TOTAL
Motor	Stepper Motors Power: 10 hp Voltage: 12V Status: Item fully delivered	15	150.00	2,250.00
Compressor	Pressure valve Compressors Model: SSDGA Serial No: 23568 Power: 200 hp Status: Item fully delivered	2	100.00	200.00
Coal	Coal raw Material Total Moisture: 14% Calorific Value: 5800 Kcal/Kg Status: Items fully delivered	100	50	5,000.00

Total	7,450.00
TAX	199.00
SHIPPING	99.00
OTHER	
TOTAL	**$7,748.00**

Comments or Special Instructions
Thank you for your business.

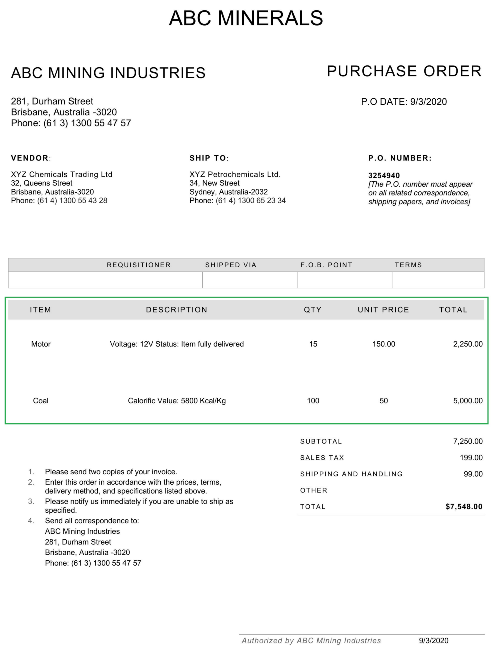
If you have any questions about this purchase order, please contact
[ABC Mining Industries. (61 3) 1300 55 47 57]

https://www.vertex42.com/ExcelTemplates/excel-purchase-order.html

Purchase Order Template © 2015 Vertex42.com

(b)

Fig. 2: Dataset sourced from Tablebank, custom and internet



ABC MINERALS

ABC MINING INDUSTRIES

PURCHASE ORDER

281, Durham Street
Brisbane, Australia -3020
Phone: (61 3) 1300 55 47 57

P.O DATE: 9/3/2020

VENDOR:
XYZ Chemicals Trading Ltd
32, Queens Street
Brisbane, Australia-3020
Phone: (61 4) 1300 55 43 28

SHIP TO:
XYZ Petrochemicals Ltd.
34, New Street
Sydney, Australia-2032
Phone: (61 4) 1300 65 23 34

P.O. NUMBER:
3254940
[The P.O. number must appear on all related correspondence, shipping papers, and invoices]

REQUISITIONER	SHIPPED VIA	F.O.B. POINT	TERMS

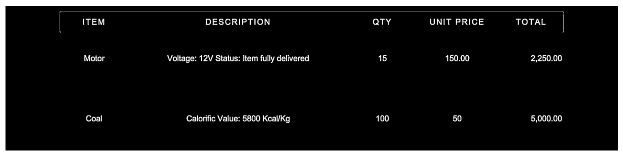
ITEM	DESCRIPTION	QTY	UNIT PRICE	TOTAL
Motor	Voltage: 12V Status: Item fully delivered	15	150.00	2,250.00
Coal	Calorific Value: 5800 Kcal/Kg	100	50	5,000.00

SUBTOTAL	7,250.00
SALES TAX	199.00
SHIPPING AND HANDLING	99.00
OTHER	
TOTAL	**$7,548.00**

1. Please send two copies of your invoice.
2. Enter this order in accordance with the prices, terms, delivery method, and specifications listed above.
3. Please notify us immediately if you are unable to ship as specified.
4. Send all correspondence to:
   ABC Mining Industries
   281, Durham Street
   Brisbane, Australia -3020
   Phone: (61 3) 1300 55 47 57

Authorized by ABC Mining Industries 9/3/2020

Fig. 3: Table detection in the input image



ITEM	DESCRIPTION	QTY	UNIT PRICE	TOTAL
Motor	Voltage: 12V Status: Item fully delivered	15	150.00	2,250.00
Coal	Calorific Value: 5800 Kcal/Kg	100	50	5,000.00

Fig. 4: Processed image after adaptive thresholding

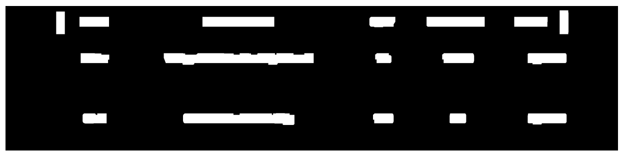
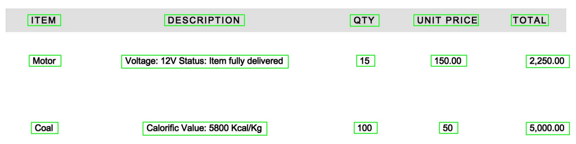


Fig. 5: Horizontally and vertically dilated cells

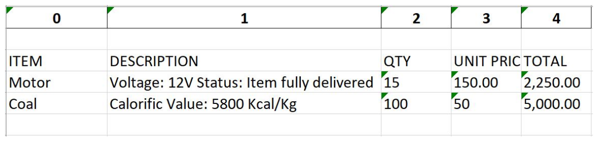


ITEM	DESCRIPTION	QTY	UNIT PRICE	TOTAL
Motor	Voltage: 12V Status: Item fully delivered	15	150.00	2,250.00
Coal	Calorific Value: 5800 Kcal/Kg	100	50	5,000.00

Fig. 6: Cells detected in the table

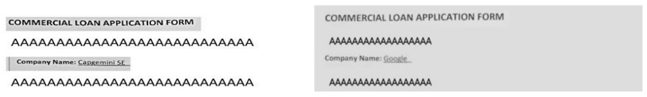
A. Precision - Precision can be evaluated by finding out the number of cell crops that were correctly identified to the total number of identified cell crops by our OpenCV module.

$$\text{Precision} = \frac{\text{Number of correctly identified cell crops}}{\text{Total number of identified cell crops}} \tag{1}$$



0	1	2	3	4
ITEM	DESCRIPTION	QTY	UNIT PRIC	TOTAL
Motor	Voltage: 12V Status: Item fully delivered	15	150.00	2,250.00
Coal	Calorific Value: 5800 Kcal/Kg	100	50	5,000.00

Fig. 7: Extracted table data in CSV format



COMMERCIAL LOAN APPLICATION FORM

AAAAAAAAAAAAAAAAAAAAAAAAAAA

Company Name: Capgemini SE

AAAAAAAAAAAAAAAAAAAAAAAAAAA

(a)

COMMERCIAL LOAN APPLICATION FORM

AAAAAAAAAAAAAAAAA

Company Name: Google

AAAAAAAAAAAAAAAAA

(b)

Fig. 8: (a) Round 1 with white background (b) Round 2 with input RoI background color

B. Recall - Recall can be evaluated by finding the number of cell crops that were correctly identified by our OpenCV module to the total number of cell crops present.

$$\text{Recall} = \frac{\text{Number of correctly identified cell crops}}{\text{Total number of cell crops in the document}} \quad (2)$$

C. F1-score - F1-score can be evaluated with the help of precision and recall which were calculated in the equations (1) and (2).

$$\text{F1-score} = 2 * \frac{(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})} \quad (3)$$

The Fig.3 represents accurate detection of borderless tables from scanned documents by our custom trained object detection module. The precise identification and row and column data extraction from the detected table is depicted in Fig. 5,6 and 7.

In order to identify the effectiveness of our solution, we have evaluated our 2-step method with a dataset comprising of more than 100 PDFs irrespective of the format, layout and border of the scanned documents. Table III represents the model evaluation metrics (precision, recall and F1 score) used for performance evaluation of our novel solution.

TABLE III: Performance Evaluation

Type of table	Precision	Recall	F1-score
Bordered table	0.97	1.00	0.98
Borderless table	0.95	0.98	0.96

V. Future Works

The performance of our solution is comparable with some state-of-the-art commercial and open-source tools available in the market and overcomes the limitations offered by them.

However, there are certain challenges offered by our solution that we would inculcate in our future works. Firstly, our solution can't process handwritten text. Also, our object detection module used to extract tables out of images can be improvised by training the model on more train data. Lastly, the horizontal and vertical dilation-erosion techniques can be automated.

References

[1] R. Sahoo, C. Kathale, M. Kubal, and S. Malik, "Auto-table-extract: A system to identify and extract tables from pdf to excel," vol. 9, pp. 217–221, 2020.

[2] M. Ohta, R. Yamada, T. Kanazawa, and A. Takasu, "A cell-detection-based table-structure recognition method." New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3342558.3345412

[3] K. A. Hashmi, M. Liwicki, D. Stricker, M. Afzal, M. Afzal, and M. Z. Afzal, *IEEE Access*, vol. PP, pp. 1–1, 06 2021.

[4] M. Li, L. Cui, S. Huang, F. Wei, M. Zhou, and Z. Li, "Tablebank: Table benchmark for image-based table detection and recognition," *CoRR*, vol. abs/1903.01949, 2019. [Online]. Available: http://arxiv.org/abs/1903.01949

[5] R. Smith, "An overview of the tesseract ocr engine," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 2, 2007, pp. 629–633.

[6] J. Kim and H. Hwang, "A rule-based method for table detection in website images," *IEEE Access*, vol. 8, pp. 81 022–81 033, 2020.

[7] D. Prasad, A. Gadpal, K. Kapadni, M. Visave, and K. Sultanpure, "Cascadetabnet: An approach for end to end table detection and structure recognition from image-based documents," *CoRR*, vol. abs/2004.12629, 2020.

[8] P. Fischer, A. Smajic, A. Mehler, and G. Abrami, "Multi-type-td-tsr - extracting tables from document images using a multi-stage pipeline for table detection and table structure recognition: from OCR to structured table representations," *CoRR*, vol. abs/2105.11021, 2021.

[9] F. Shafait and R. Smith, "Table detection in heterogeneous documents," 01 2010, pp. 65–72.

[10] T. Hassan and R. Baumgartner, "Table recognition and understanding from pdf files," in *Proceedings of the Ninth International Conference on Document Analysis and Recognition - Volume 02*, ser. ICDAR '07. USA: IEEE Computer Society, 2007, p. 1143–1147.

[11] J. Fang, L. Gao, K. Bai, R. Qiu, X. Tao, and Z. Tang, "A table detection method for multipage pdf documents via visual seperators and tabular structures," in *2011 International Conference on Document Analysis and Recognition*, 2011, pp. 779–783.

[12] Y. Liu, K. Bai, P. Mitra, and C. L. Giles, "Improving the table boundary detection in pdfs by fixing the sequence error of the sparse lines," in *2009 10th International Conference on Document Analysis and Recognition*, 2009, pp. 1006–1010.

[13] Y. Liu, P. Mitra, and C. L. Giles, "Identifying table boundaries in digital documents via sparse line detection," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, ser. CIKM '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 1311–1320. [Online]. Available: https://doi.org/10.1145/1458082.1458255

[14] Tzutalin, "Labelimg," Free Software: MIT License, 2015. [Online]. Available: https://github.com/tzutalin/labelImg

[15] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017. [Online]. Available: http://arxiv.org/abs/1704.04861

[16] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015.

[17] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/