Predicting Carbon Dioxide Emission Levels

A Project Report

Presented to  
**Dr. Vidhyacharan Bhaskar**

*San Jose State University*

In Partial Fulfillment

Of the Requirements for

CMPE 272

Enterprise Software Platforms

By

Moxank Patel

Nyugen Bui

12/2023

Table of Content

Predicting Carbon Dioxide Emission Levels

|  |  |
| --- | --- |
| Moxank Patel  *Computer Engineering*  *San José State University*  *San Jose, California, USA*  moxankprakashbhai.patel@sjsu.edu | Nguyen Bui  *Software Engineering*  *San José State University*  *San Jose, California, USA*  nguyen.bui@sjsu.edu |

***ABSTRACT - In response to the critical challenge posed by global warming, this study proposes an innovative approach to monitor atmospheric carbon dioxide (CO2) levels. Leveraging advanced machine learning techniques, we utilize a comprehensive dataset derived from the Sentinel-5P satellite, accessed via Kaggle. This methodology circumvents the limitations inherent in traditional Non-Dispersive Infrared (NDIR) sensor-based measurements, such as the need for physical sensor deployment and associated high costs. Our approach employs various predictive models, including Linear Regression, Lasso Regression, and Ensemble Regression Algorithms, with the ensemble model demonstrating superior performance, evidenced by a Root Mean Square Error of 10. This research signifies a significant advancement in environmental monitoring, offering a scalable, cost-effective, and precise method for tracking global CO2 emissions, which is vital for informed decision-making in climate change mitigation efforts.***

***Index Terms -***  Sentinel-5P, NDIR sensor, Linear Regression, Ensemble Learning, Root Mean Square Error.

1. INTRODUCTION

The current state-of-the-art models for table detection are either based on a two-stage approach or use rows and columns to identify the table's cells. As these systems use rows and columns to locate cells, they cannot be relied on because some tables might have merged cells. Not only this, but certain tables in a document are not in alignment with the x-axis, so detectors do not consider them a table. So, these systems are specific to a particular table type and are not usable for every table [1].

Previously table detection was done using a simple algorithm that used morphological operations to find the edges of tables. The approach is feasible and straightforward, but it fails for a borderless table. Mandal, S., and the team developed the abovementioned system [2]. A system developed by D.-D. Nguyen, based on neural networks, could recognize borderless tables and other tables with complex structures [3]. The system relied on four parameters to detect the table, and they are the coordinates of the bounding box of the table. A general assumption here is that tables would always be horizontally aligned and have uniform rows and columns, and due to this, neural networks do not perform well with a document that does not hold this assumption. Moreover, this assumption does not hold in the real world, and the model's accuracy suffers.

In this research, instead of predicting the rows and columns and then merging them to obtain cells, the system developed in this study proposes to identify cells directly. The model designed consists of a Convolution Neural Network (CNN) and Feature Pyramid Network (FPN) [4]. The system inspired by RetinaNet [5] uses a similar architecture. One of the most critical parts of designing neural networks is choosing the loss function. The loss functions used are Focal Loss [5] and a variation of Intersection over Union. The model will also account for additional parameters to detect the elevation angle.

Here, the model will be trained on different datasets consisting of tables with varying structures to obtain an efficient system. There are various datasets available for Table Detection, like ICDAR 2017 [7], TableBank [8], TNCR [9], DocBank, marmot, and many more, and the evaluation metrics used will be mean average precision (mAP).

The structure of the paper is as follows. The second section goes over the work done on this problem. The third section justifies the research and mentions all the problems that the research will deal with, and the fourth section explains the system architecture. The fifth section will go over the results and experiments set up. The sixth section concludes the research and outlines future works.

1. RELATED WORK

One of the most addressed problems in the Document Image Analysis and Recognition (DIAR) community is Table recognition. Recent development in Deep Learning Field has opened the door to the same problem with different approach [10]. Many competitions were held by International Conference on Document Analysis and Recognition [11].

Table recognition mainly involves two tasks: first, table localization, and then detecting the elements of that table. Earlier, they were treated as different tasks, and two separate systems were employed to handle each task individually. Kieninger and the team worked on this approach and developed a two-stage table recognition system [12]. "A Simple and Effective Table Detection System from Document Images" [2] is another research based on the above approach carried out by Mandal and the team. In both studies, the main idea was to use the morphological operation to identify the edges, which helps detect the table's structure. However, this approach does not go well with borderless tables or multi-column tables. Using Tesseract OCR, another system was developed in which columns are detected using edges, and later, the OCR would extract text, and then structuring is done. This approach was developed by Shafait and Smith [13].

As neural networks became computationally feasible, many new architectures for the problem were developed. Gilani and his team, inspired by the idea that neural networks can detect complex patterns, developed a system using neural networks to detect tables. In this system, first, segmentation of the image is done, and then, using a neural network, the segmented image is processed, and finally, a set of the bounding box is generated. However, they used rows and columns to identify the table's elements, making the system useless for multi-row or multi-column tables.

One critical aspect of neural networks is loss function. Loss functions are crucial because they calculate the expected and predicted output error. In one stage object detection model becomes biased towards the background as many boxes contain background. To deal with this, T.Y. Lin and his team developed Focal Loss to prevent the model from becoming biased [5]. Another loss function that helps to localize the error for the bounding box is Complete Intersection over Union (CIoU) [6].

Neural Networks only perform as good as the training data. Table Recognition is a problem, but many researchers are also working on developing a diverse dataset for Table Recognition. "TNCR: Table net detection and classification dataset" [9], "DocBank: A Benchmark dataset for Document Layout Analysis", and "TableBank: A Benchmark for Table Detection and Recognition" [14] are some datasets designed by researchers for this specific problem.

1. PROJECT JUSTIFICATION

Image processing algorithms can single handily detect bordered tables and horizontally aligned tables. However, tables with complex structures cannot be recognized using these algorithms. Another approach is to use neural networks. This research proposes to use a neural network to identify cells directly and predict an additional parameter, the elevation angle. This addition of one parameter makes the system quite robust as it can easily detect a table of any orientation. Not only this, but the prediction of cells instead of tables or rows or columns makes the system far more accurate than other systems. Because of this, here the system here will learn to identify cells instead of rows and columns. The systems developed so far merged rows and columns to identify cells. Although a practical approach, it is not always true, as there are tables that will prove this hypothesis wrong. Hence, a system that accounts for inclination angle and does table predictions using cells will be pretty accurate than the system that does not.

1. PROPOSED PROCEDURE FOR DETECTING TABLES

Tables are complex entities of a document and have varying sizes and orientations. Due to the varying size, the study proposes the use of Feature Pyramid Networks (FPN) to detect the features of the document. After learning the features, a Convolution Neural Network (CNN) tries to identify the elements of the tables by learning the pattern of the elements. Once the CNN is done, the feature generated by CNN is passed to a fully connected layer which is tasked to generate a bounding box, elevation angle, and assign labels. The architecture proposed is illustrated in Fig. 1.

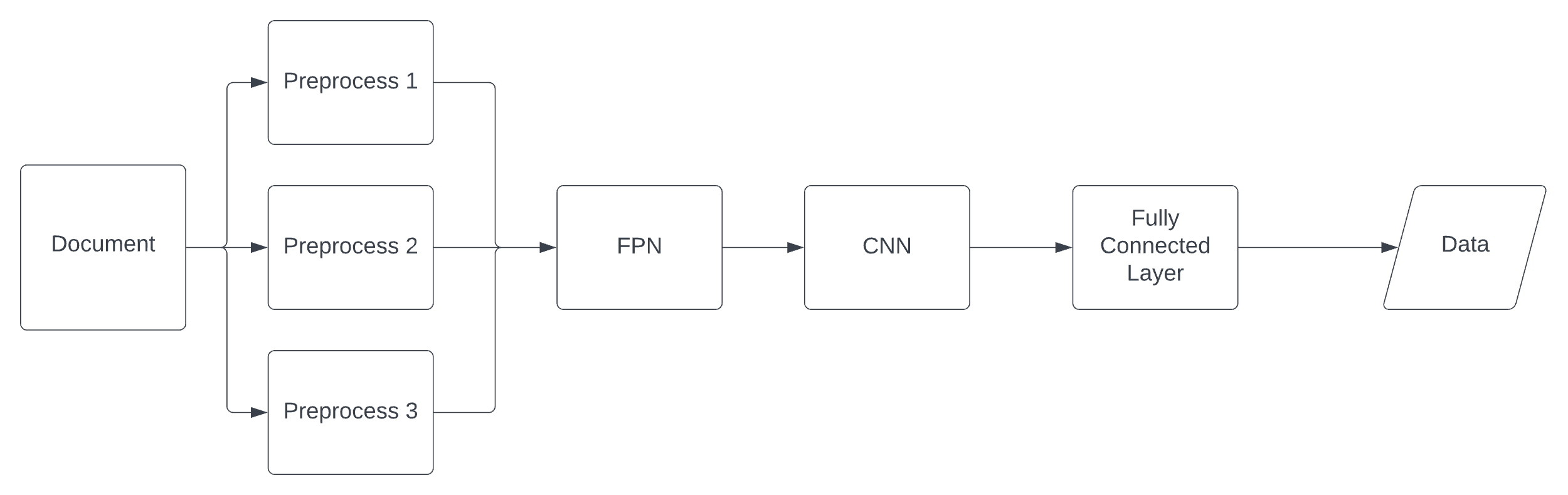


Fig. 1 Architecture of the Model.

Here, three images are produced from the document, as shown in Fig. 1 above. These three images give the model a more descriptive idea about the background information, making object detection easier. Each block of architecture will be explained in subsections further.

* 1. *Preprocessing*

Instead of using the original image, here we use preprocessed images. The FPN uses these preprocessed images to generate the features; later, these features are passed to an average pooling layer. Preprocessing step involves finding edges and smudging; as shown in Fig 2, the first function only converts the given image into a black and white image. The second function generates the edges of the image, and the last function produces a smudged image. The algorithm for the first function is simple; here, we just invert the image after converting it to its grayscale form. Mathematically that is 255 – grayscale(given-image). Before applying any of these preprocessing steps, the dimension of the image is reduced, and the maximum allowed dimension is 1080\*820. This reduction of dimension is done to limit the number of parameters.

The second function first converts the given image into grayscale, and then the image is blurred. On the original image, Sobel's mask is applied to get the edges of the image. Then blurred image and the second image are combined by adding pixel value. The image is then blurred, and then an opening operation is performed.

The third function first converts the document into a black and white image. This black and white image is further eroded to get fine edges from the document, and later on, the Canny algorithm is applied to obtain the final image. These preprocessing techniques improve the localization task for object detection [15]. Combine this with FPN, and detecting the table on different dimensions becomes easier.

Table, calendar

Description automatically generated

A picture containing text

Description automatically generated Diagram

Description automatically generated A black and white image of a person's face

Description automatically generated with medium confidence

Fig. 2 The Original Image (The top one) and Preprocessed images (bottom ones)

Image Source: TableBank: A Benchmark Dataset for Table Detection and Recognition [14]

* 1. *Feature Pyramid Network And Convolution Neural Network*

Feature Pyramid Network is known to detect varying length patterns in an image [4]. They are networks stacked upon each other, and they use up sampling and down sampling to detect varying length patterns. The study proposes that an FPN with three layers be used to detect the table elements, and the parameters used for FPN are the same as FPN used in RetinaNet [5]. CNN is going to use the features that FPN generated. The activation function used in Feature Pyramid Network is ReLU. The features that FPN generates are passed to a CNN as they are known to work well with the task such as object detection [16]. To do the training faster here, we use transfer learning. Instead of initializing parameters randomly for CNN and then training the network will take much time. After hitting different combinations, it was found that ResNet50 [17] is the required backbone for this task. This ResNet version was trained on an object detection dataset, making it a perfect candidate for the table detection task. Here, we are trying to predict the bounding box and determine if that bounding box contains the elements of the table or not.

* 1. *Bounding Box And Loss Functions*

In the study, a bounding box is presented as XC, YC, Width, Height, and Inclination angle (Θ). In a 2D plane, a rectangular box consists of 4 coordinates. Here we represent this rectangle or bounding box for the table's element in the form of a coordinate of centroids, its width, height, and inclination angle.

Equation for:

*XC = (Xmin + Xmax)/2,*

*YC = (Ymin + Ymax)/2* (1)

In the above equation (1), Xmin and Ymin are the upper left coordinate of the bounding box, and Xmax and Ymax are the bottom right coordinate of the rectangle. The dataset used the following convention: (XMIN, XMAX, YMIN, YMAX). Hence, they need to be converted. The final output by the system will consist of coordinates of cells and inclination angle.

This system has two loss functions as the model tries to do two things. First, it tries to generate the bounding box, and second, it classifies it. For the first task, an intuitive loss function would be Intersection over Union (IoU) [18]. IoU indicates how much the prediction overlaps with the ground truth. Mathematically, IoU is the area of the intersection divided by the union area. In the figure below, the area of union and area of intersection is illustrated. Based on IoU, another loss function was developed by X. Wang and his team [6]. They developed a loss function named Improved Complete Intersection over Union (ICIoU). Instead of simply relying on the area of intersection and union, it also compares the distance between the center of two bounding boxes and the dimension of the boxes.

Fig. 3 Highlighted are the area of intersection and area of union

Instead of ICIoU, RetinaNet used SmoothL1 Loss which works well, but Intersection over Union seems like a more intuitive choice as IoU is defined as the amount of overlap between two boxes. Research carried out by Zheng and his teammates showed that Distance IoU is a more efficient loss function than IoU or any other loss function [19]. ICIoU is a variation of Distance Intersection over Union.

The other loss function that the system uses is Focal Loss [5]. Focal loss is nothing but an improved version of the cross-entropy function. Here focal loss is used because the model gets overwhelmed as many bounding in ground truth would be labeled as background, making the model biased. The equation for focal loss is mentioned below.

Diagram

Description automatically generated (2)

The net loss function is defined as the ICIoU and Focal Loss summation.

* 1. *Elevation Angle And Classification*.

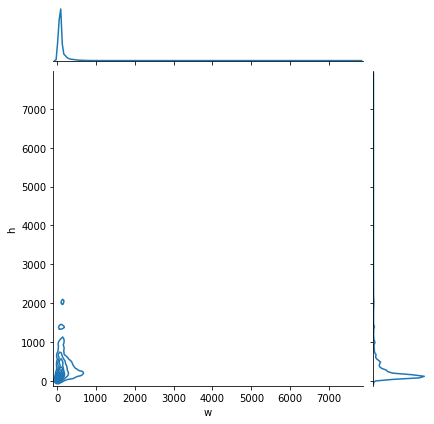
A fully connected layer calculates the elevation angle after CNN outputs features. Here ReLU is for regression of Elevation angle. The range of the elevation is between 0 and 360. Each anchor box could be a background, cell, or table for the classification problem. Table I and Fig 4 show that the orientation of cells and table majority lies in 0.5, 1.0, 2.0, 3.5, and area between 75,174,300,450,550. These parameters are used to generate the anchor boxes.

TABLE I

STATISTICAL ANALYSIS OF THE DATASET

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | X | Y | H | W | W/H | AREA |
| count | 214229.000000 | 214229.000000 | 214229.000000 | 214229.000000 | 214229.000000 | 2.142290e+05 |
| mean | 2279.128563 | 2251.315139 | 253.465367 | 127.470987 | 2.780221 | 7.043496e+04 |
| std | 1776.890588 | 1242.035709 | 334.785056 | 246.217757 | 6.909794 | 8.139667e+05 |
| min | 0.000000 | 0.000000 | 0.000000 | 1.000000 | 0.000000 | 0.000000e+00 |
| 25% | 935.000000 | 1278.000000 | 96.000000 | 58.000000 | 1.315217 | 6.682000e+03 |
| 50% | 1745.000000 | 2066.000000 | 147.000000 | 89.000000 | 1.804598 | 1.354200e+04 |
| 75% | 3294.000000 | 3075.000000 | 282.000000 | 111.000000 | 3.662617 | 3.682200e+04 |
| max | 7724.000000 | 7883.000000 | 7838.000000 | 7773.000000 | 2213.000000 | 4.250426e+07 |

Chart, scatter chart

Description automatically generated Chart, scatter chart

Description automatically generatedA picture containing whiteboard

Description automatically generated

Fig 4. Joint Plot of the dataset and KDE of the dataset.

*E*. *System's Architecture.*

After preprocessing of the image is done, they are normalized using ResNet50 weights as here CNN uses transfer learning. More specifically, these images are normalized to zero-centered. After this, the Anchor Boxes are generated where each area is targeted, and bounding of varying dimensions are generated. In Fig 4., we can see several anchor boxes are generated and placed on an image. These anchor boxes are distributed uniformly all over the image, and the model uses them to generate final bounding boxes and classify them. The preprocessed image is given to FPN to extract features on different dimension levels. Then CNN using kernel, tries to find the patterns of the table's element. Once we have features from CNN, we then pass them to a fully connected network. The model tries to predict the difference between ground truth and anchor box. Ground truth and Anchor Box are matched based on IoU, a threshold used to label if the anchor box contains background or table elements. Matched boxes can be seen in Fig 5, and the threshold of IoU was 0.4. Here the encoder-decoder structure is used to regress the bounding box. After the regression step, Non-Maximum Suppression [20] is applied to get non-overlapping bounding boxes. The other parameter inclination angle is predicted along with the bounding box.

Equation for encoding is,

*encoded\_boxes = (matched\_boxes [:2] – anchor\_boxes [:2])/anchor\_boxes [2:], log2(matched\_boxes[2:]/anchor\_boxes[2:]).*

*encoded\_boxes = encoded\_boxes/box\_variance (3)*

*Equation for decoding is,*

*decoded\_boxes = (prediction\_diff + anchor\_boxes) \* box\_variance*

*decoded\_boxes = (decoded\_boxes [:2]\* anchor\_boxes [2:]) + anchor\_boxes[:2], exp(decoded\_boxes[2:]\*anchor\_boxes[2:]) (4)*

* In equations (3) and (4), matched\_boxes are anchor boxes closest to ground truth.
* The box variance used in encoding process of anchor box is [0.1, 0.1, 0.2, 0.2]

A picture containing diagram

Description automatically generated Graphical user interface

Description automatically generated

Fig 5. Generated anchor boxes and matched anchor boxes.

In this research, instead of directly predicting the bounding box, we try to predict the offset. Consider the two boxes which overlap. Now the offset would be the distance between them. The model here tries to find the offset between the ground truth and anchor box, which is why Intersection over Union seems like an intuitive function. In Fig 6, it is illustrated what the model is trying to predict, i.e., X offset and Y offset.

Y offset

X offset

Fig 6. Anchor Box (Orange Box) and Ground Truth (Yellow Box)

To summarize, the system's flow is as follows, the given document is firstly preprocessed, and three images are obtained from it. These three images are passed to Feature Pyramid Networks, which generate features passed to Convolution Neural Network, which extract patterns. A fully connected layer uses these patterns to predict the margin, inclination angle and classify which element of the table is the anchor box. The margin predicted here is offset by which the anchor box is away from the given object (element of the table). The loss functions used are Focal Loss and Improved Complete Intersection over Union.

1. EXPERIMENT VALIDATIONS

This section will evaluate the proposed model's performance on three different datasets. The section will also provide details regarding the environment setup and evaluation metrics used to evaluate the model performance. Further, the section will also highlight the limitation of the proposed system.

* 1. *Environment Setup And Datasets*

The system is built using TensorFlow Framework with version 2.8.0. The system was trained using Graphics Processing Unit (GPU), and the GPU used was NVIDIA 1050. The additional libraries that the system needs are NumPy, Keras, and cv2. The cv2 library was used for the preprocessing task. Here Resnet was used for transfer learning, and the system used Resnet-50 as its backbone. Lastly, the batch size for training is 4.

The system is trained on three datasets: TableBank, DocBank, and TNCR. These modern datasets comprise tables of varying structures and orientations. In the table bank, there are around 145,000 documents containing tables. Out of which, around 90,000 are in a LaTeX format and the remaining in word format. DocBank contains around 500,000 documents, and TNCR, based on the marmot dataset, gives around 5000 documents with differing orientations. Moreover, as many tables do not have differing orientations, we added noise and differed the orientation randomly for images giving the model a variety.

* 1. *Evaluation Metrics*

The metrics that are used for evaluations here are Average Precision (AP), Average Recall (AR), and mean Average Precision (mAP) [21] on different datasets. Here mAP is the average of 'Average Precision (AP)' for N samples. To classify an output as TP or FN, an IoU threshold is used. The calculation for the above is as follows:

*Average Precision (AP) = True Positive (TP) / (True Positive (TP) + False Positive (FP))* (5)

*Average Recall (AR) = True Positive (TP) / (True Positive (TP) + False Negative(FN)).* (6)

* 1. *Results*

Table II shows the evaluation metrics of the model with different backbones. Furthermore, the system performed well with ResNet50. It achieved the highest accuracy with ResNet50 of 87%. The model achieves the highest accuracy for the IoU threshold of 50%, with Resnet-50 as its backbone.

TABLE II

MODEL'S SCORE

|  |  |  |  |
| --- | --- | --- | --- |
| Backbone |  | IoU | |
|  |  | 50% | 75% |
| Resnet-50 | Precision | 0.833 | 0.770 |
| Recall | 0.851 | 0.670 |
| mAP | 0.873 | 0.811 |
| Resnet-101 | Precision | 0.756 | 0.775 |
| Recall | 0.650 | 0.719 |
| mAP | 0.701 | 0.682 |

.

Fig 7 illustrates that the model can easily detect cells of tables with complex structures. The model was trained for three epochs to achieve this result. The system can easily recognize the table structure, but the time to train the model is comparatively higher as the number of parameters increases. In figure 7, it can be seen that the system misses a few cells in the table, which are highlighted in the black portion. Moreover, the portion in the green is where cells are predicted, but they are not cells.

A screenshot of a computer

Description automatically generated with low confidence A white board with writing on it

Description automatically generated with medium confidence

A picture containing diagram

Description automatically generated Table

Description automatically generated

Fig. 7 Predictions done by the system on Bordered, Borderless, and slanted tables.

As the model is trying to predict offset for each cell, the number of the parameter will increase drastically. Instead of trying to predict the rows and columns here, the system individually predicts the offset for each cell.

Earlier the number of parameters would be,

*NP = (NR + NC) \* NAnchorBox (7)*

And now,

*NP = NR \* NC \* NAnchorBox* (8)

Here, NP is the number of parameters, NR is the number of rows, NC is the number of columns, and NAnchorBox is the number of anchor boxes. The equation (8) shows that the parameter number would increase if cells were targeted instead of rows and columns.

1. CONCLUSION AND FUTURE WORKS

To conclude, the study proposes an efficient way of recognizing the table's structure using a combination of Feature Pyramid Network and Convolution Neural Network. The proposed architecture can detect table structures in one go and does not require any post-processing. The final output that the system produces is a bounding box, elevation angle with respect to the x-axis, and, lastly, it classifies the anchor box. Using the proposed architecture, the system achieved an accuracy of 87% within three training iterations.

While the model can detect tables with complex structures and varying alignment, the model takes time to train. This is because the model tries to predict all the cell's offsets, and due to it, the number of anchor boxes increases, which increases the model's parameters. Moreover, due to it, the time taken while training increases.

Here the model tries to determine the offset between cells and the anchor box, and the number of parameters increases. To avoid increasing the model parameter, a heuristic algorithm can be employed, which first generates the proposal for elements of the table, and then these proposals are used for predictions. However, it will make the model a two-stage system, and the time taken to generate this proposal will be comparatively higher, which will directly increase the time taken to predict. So, another approach is to drop some of the anchor boxes intuitively, or they could be dropped randomly during training. The latter approach is cost-effective.

REFERENCES

[1]. A. Gilani, S. R. Qasim, I. Malik and F,. Shafait, "Table Detection Using Deep Learning," 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), 2017, pp. 771-776. [Online] Available: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8270062&isnumber=8269930>.

[2]. S. Mandal, S. P. Chowdhury, A. K. Das, and B. Chanda, "A simple and effective table detection system from document images," *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 8, no. 2–3, pp. 172–182, Mar. 2006, doi: 10.1007/s10032-005-0006-5.

[3]. D.-D. Nguyen, "TableSegNet: a fully convolutional network for table detection and segmentation in document images," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 25, no. 1, pp. 1–14, Nov. 2021, doi: 10.1007/s10032-021-00390-4.

[4]. T. -Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and, S. Belongie, "Feature Pyramid Networks for Object Detection," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 936-944, 2017, doi: 10.1109/CVPR.2017.106.

[5]. T. -Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "Focal Loss for Dense Object Detection," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 2, pp. 318-327, 1 Feb. 2020, doi: 10.1109/TPAMI.2018.2858826.

[6]. X. Wang and J. Song, "ICIoU: Improved Loss Based on Complete Intersection Over Union for Bounding Box Regression," in IEEE Access, vol. 9, pp. 105686-105695, 2021, doi: 10.1109/ACCESS.2021.3100414.

[7]. N. Nayef et al., "ICDAR2017 Robust Reading Challenge on Multi-Lingual Scene Text Detection and Script Identification - RRC-MLT," 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), 2017, pp. 1454-1459, doi: 10.1109/ICDAR.2017.237.

[8]. M. Li *et al.*, "DocBank: A Benchmark Dataset for Document Layout Analysis," *arXiv:2006.01038 [cs]*, Nov. 2020, Accessed: May 16, 2022. [Online]. Available: <https://arxiv.org/abs/2006.01038>

[9]. A. Abdallah, A. Berendeyev, I. Nuradin, and D. Nurseitov, "TNCR: Table Net Detection and Classification Dataset," *Neurocomputing*, vol. 473, pp. 79–97, Feb. 2022, doi: 10.1016/j.neucom.2021.11.101.

[10]. P. Riba, L. Goldmann, O. R. Terrades, D. Rusticus, A. Fornés, and J. Lladós, "Table detection in business document images by message passing networks," *Pattern Recognition*, vol. 127, p. 108641, Jul. 2022, doi: 10.1016/j.patcog.2022.108641.

[11]. L. Gao et al., "ICDAR 2019 Competition on Table Detection and Recognition (cTDaR)," 2019 International Conference on Document Analysis and Recognition (ICDAR), 2019, pp. 1510-1515, doi: 10.1109/ICDAR.2019.00243.

[12]. T. Kieninger and A. Dengel, "Table Recognition and Labeling Using Intrinsic Layout Features," *International Conference on Advances in Pattern Recognition*, pp. 307–316, 1999, doi: 10.1007/978-1-4471-0833-7\_31.

[13]. Shafait, Faisal, and Ray Smith. "Table Detection in Heterogeneous Documents." Proceedings of the 9th IAPR International Workshop on Document Analysis Systems, ACM, pp. 65–72, 2010, <https://doi.org/10.1145/1815330.1815339> .

[14]. M. Li, L. Cui, S. Huang, F. Wei, M. Zhou, and Z. Li, "TableBank: A Benchmark Dataset for Table Detection and Recognition," *arXiv:1903.01949 [cs]*, Jul. 2020, Accessed: May 16, 2022. [Online]. Available: <https://arxiv.org/abs/1903.01949>

[15]. J. Yepez and S.-B. Ko, "Improved license plate localisation algorithm based on morphological operations," *IET Intelligent Transport Systems*, vol. 12, no. 6, pp. 542–549, Aug. 2018, doi: 10.1049/iet-its.2017.0224.

[16]. G. Yao et al., "An Empirical Study of the Convolution Neural Networks Based Detection on Object With Ambiguous Boundary in Remote Sensing Imagery—A Case of Potential Loess Landslide," in IEEE Journal of Selected Topics in the Applied Earth Observations and Remote Sensing, vol. 15, pp. 323-338, 2022, doi: 10.1109/JSTARS.2021.3132416.

[17]. Kaiming He, et al. "Deep Residual Learning for Image Recognition." 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2016, pp. 770–78, <https://doi.org/10.1109/CVPR.2016.90>.

[18]. J. Yan, H. Wang, M. Yan, W. Diao, X. Sun, and H. Li, "IoU-Adaptive Deformable R-CNN: Make Full Use of IoU for Multi-Class Object Detection in Remote Sensing Imagery," *Remote Sensing*, vol. 11, no. 3, p. 286, Feb. 2019, doi: 10.3390/rs11030286.

[19]. Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression," *arXiv:1911.08287 [cs]*, Nov. 2019, Accessed: May 16, 2022. [Online]. Available: <https://arxiv.org/abs/1911.08287>.

[20]. A. Neubeck and L. Van Gool, "Efficient Non-Maximum Suppression," 18th International Conference on Pattern Recognition (ICPR'06), 2006, pp. 850-855, doi: 10.1109/ICPR.2006.479.

[21]. P. Henderson and V. Ferrari, "End-to-end training of object class detectors for mean average precision," *arXiv:1607.03476 [cs]*, Mar. 2017, Accessed: May 16, 2022. [Online]. Available: <https://arxiv.org/abs/1607.03476>.

‌