# Table Detection and Structure Recognition

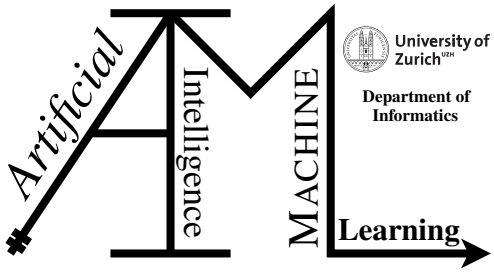
A pragmatic approach

Master Thesis

**Pratyush Singh**
19-762-988

**Submitted on**
April 15 2022

**Thesis Supervisor**
Dr. Moritz Platscher and Prof. Dr. Manuel Günther
Acodis AG and University of Zurich



**Master Thesis**

**Author:** Pratyush Singh, pratyush.singh@uzh.ch

**Project period:** October 18 2021 - April 18 2022

Artificial Intelligence and Machine Learning Group
Department of Informatics, University of Zurich

# Acknowledgements

I would like to thank Dr. Moritz Platscher, Senior Machine Learning Engineer at Acodis AG, and Prof. Dr. Manuel Günther, Assistant Professor at Artificial Intelligence and Machine Learning (AIML) Group, Department of Informatics, the University of Zurich for providing me the opportunity to work on this topic and supporting me throughout the thesis with constant guidance and feedback. I am also grateful to them for ensuring that the thesis goes smoothly despite COVID-19 infection and restriction. I would also like to thank my brother, Mr. Pranjal Singh for his constant support and input and everyone at Acodis AG for keeping a friendly gesture toward me. Lastly, I would like to thank all my family members, Professors, and members of the administration at the University of Zurich for their kind support throughout my Master's studies.

# Abstract

Tables are one of the most convenient ways to present complex, correlated, and structured information. While rule and heuristic based approaches have long dominated the table detection and structure recognition, their usefulness has been confined to a subset of tables that follow these rules. Significant research has been conducted to localize table structure, the majority of which focuses on using heuristics and rules with the assistance of optical character recognition (OCR) to manually select layout characteristics of the tables. With the rise of Deep Learning, new models have shown to be applicable across multiple unseen domains by incorporating transfer learning. This thesis presents an end-to-end object detection approach to detect tables and recognize their structures in a document and thus, help in table data extraction with the use of a deep learning model namely, *Faster R-CNN*. This work will also introduce a new metric based on Intersection over Union (IoU) for the task of table detection which does not penalize large bounding box predictions up to a defined extent and reduces the dependency of the F1 score on the chosen IoU threshold. A significant amount of experiments will be discussed on many popular publicly available datasets like ICDAR 2013, ICDAR 2019, ISRI-OCR, Marmot, TableBank, and PubTables-1M to carefully adapt and design the parameters of the Faster R-CNN model and demonstrate the robustness of the model across unseen datasets. The model present in this thesis outperforms other models including a transformer-based model to establish the state-of-the-art results on these datasets proving once again the superiority of the Faster R-CNN architecture.

# Zusammenfassung

Tabellen sind eine der bequemsten Möglichkeiten zur Darstellung komplexer, korrelierter und strukturierter Informationen. Während regelbasierte und heuristische Ansätze lange Zeit die Erkennung von Tabellen und deren Struktur dominiert haben, ist ihr Nutzen auf eine Teilmenge von Tabellen beschränkt, die diesen Regeln folgen. Es wurden umfangreiche Forschungsarbeiten zur Lokalisierung von Tabellenstrukturen durchgeführt, die sich größtenteils auf die Verwendung von Heuristiken und Regeln mit Hilfe von optischer Zeichenerkennung (OCR) zur manuellen Auswahl von Layoutmerkmalen der Tabellen konzentrieren. Mit dem Aufkommen von Deep Learning haben neue Modelle gezeigt, dass sie durch die Einbeziehung von Transferlernen auf mehrere ungesehene Domänen anwendbar sind. In dieser Arbeit wird ein durchgängiger Ansatz zur Objekterkennung vorgestellt, um Tabellen zu erkennen und ihre Strukturen in einem Dokument zu erkennen und somit bei der Extraktion von Tabellendaten mit Hilfe eines Deep-Learning-Modells, nämlich *Faster R-CNN*, zu helfen. In dieser Arbeit wird auch eine neue Metrik auf der Grundlage von Intersection over Union (IoU) für die Aufgabe der Tabellenerkennung eingeführt, die große Bounding-Box-Vorhersagen bis zu einem bestimmten Ausmaß nicht benachteiligt und die Abhängigkeit des F1-Scores vom gewählten IoU-Schwellenwert reduziert. Eine Vielzahl von Experimenten mit vielen beliebten öffentlich zugänglichen Datensätzen wie ICDAR 2013, ICDAR 2019, ISRI-OCR, Marmot, TableBank und PubTables-1M werden diskutiert, um die Parameter des Faster R-CNN-Modells sorgfältig anzupassen und zu entwerfen und die Robustheit des Modells bei unbekannten Datensätzen zu demonstrieren. Das in dieser Arbeit vorgestellte Modell übertrifft andere Modelle, einschließlich eines auf Transformer basierenden Modells, um die neuesten Ergebnisse dieser Datensätze zu ermitteln, was erneut die Überlegenheit der Faster R-CNN-Architektur beweist.

# Contents

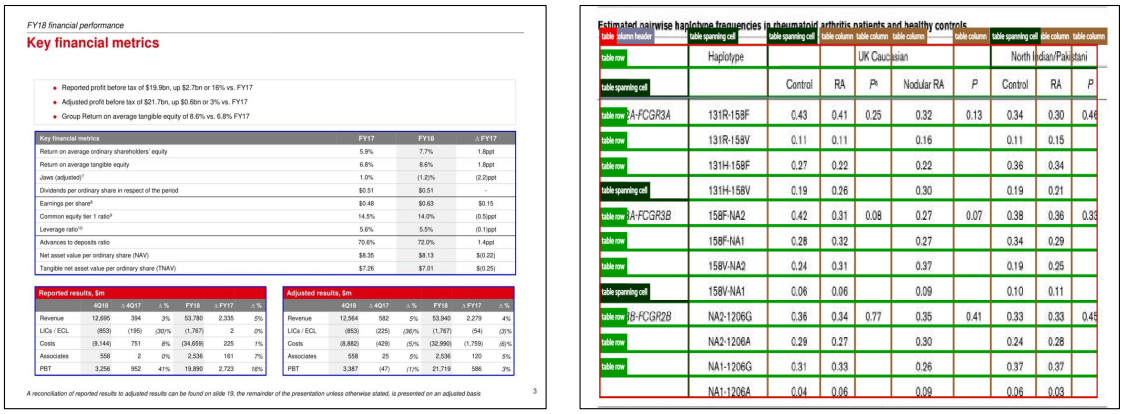
# Chapter 1

# Introduction

Two of the most significant parts of intelligent document processing for digitization and information extraction are document structure analysis and parsing of information and tables are one of the most essential parts of any document for structural analysis and extraction. In recent times, table analysis has garnered the interest of researchers to extract the information out of the table in a structured manner. Tables are a systematic means of portraying data that allows for the visual and logical arrangement of information in an easily understandable manner. A table is an organized arrangement of rows and columns that is commonly used to convey a collection of information in a concise manner (Khan et al., 2019). They are commonly used to convey vital information in research articles, newspapers, invoices, and financial documents. With the advent of mobile technology like scanning documents using mobile phones, the number of documents being processed on a daily basis have increased by a significant number. Thus, there is also the need to extract the information from the tables contained in these documents.

The terms table detection and structure recognition are often used interchangeably but there is a fundamental difference between them. Table detection entails detecting the image pixel coordinates comprising the tabular sub-image as in Figure 1.1(a), and table structure recognition includes identification of the individual rows, columns, cells *etc.* in the detected table for layout understanding as shown in Figure 1.1(b). Table data extraction is a difficult task as usually the data is available as scanned document images that contain no structural information or metadata unlike in the case of native PDF documents. Tables contain a variety of layouts, making it difficult for traditional feature engineering methodologies to decode table structures generally. These systems often rely on visual cues such as ruling lines, column spacing, the kind of data in table cells, their connections with overlapping neighbors, or color encoded cell blocks (Khan et al., 2019). They function quite well on a certain layout of tables but fail to generalize over numerous domains. Furthermore, tables spanning across multiple pages without page breaks or the cells spanning multiple columns and rows with varying fonts, text alignment poses additional challenges. One of the biggest problem with table structure recognition, especially with rows, are the large number of objects in a relatively small space as well as the radical aspect ratios of the structure elements. The inconsistent use of ruling lines for table or structure demarcation, as well as scanning or digitization errors, complicates the identification process (Coüasnon and Lemaitre, 2014). In general, there are no handwritten rules to draw or create a table. The focus of this work is to tackle these challenges and develop a pragmatic solution to table processing which generalizes well to unseen documents also. It is worth noting that the variety of datasets available for both the tasks is scarce and very limited in terms of annotations and the quality of annotations and gives rise to additional problems for supervised learning approaches.

The existing approaches are not able to perform well on the unseen datasets because of the *domain gap*. In this work, we try to bridge this gap by identifying the peculiarities of tables like elongated rows, whitespace between columns, different aspect ratios of rows/columns *etc.* and




(a) Table Detection

(b) Table Structure Recognition

Figure 1.1: TABLE DETECTION VERSUS STRUCTURE RECOGNITION. This figure highlights the fundamental difference between the task (a) and task (b). In Figure (a), blue bounding boxes highlights the tables in the image and in Figure (b), different annotated rows, columns, spanning cells *etc.* are detected in the task of structure recognition.

this leads to modifications in standard convolutional kernels, anchors, region proposals *etc.* The carefully chosen design parameters for the model helps in producing robust results across several domains. Most often, researchers use a common objection metric like F1 score based on Intersection over Union (IoU) for all the tasks. This metric does not produce good quantitative results even if the qualitative results are well qualified. It can happen due to the differing sizes of the ground truth and predicted bounding box. We show that it is not a good idea to directly use this metric out-of-the-box rather a suitable modification is done to the standard IoU to reduce the dependency of F1 scores on the chosen IoU threshold.

Deep learning approaches have substantially alleviated the limitations of computer vision in recent years. Convolutional Neural Network (CNN) is a useful deep learning technique for extracting meaningful features from visual data in practice. Because document images primarily provide visual information, CNNs are becoming more useful for table detection and table structure recognition. State-of-the-art CNNs are efficient at extracting visual features from data. For this reason, we rely on the CNN based model to achieve good results on this task.

In this thesis work, we present an end-to-end deep learning model for solving the problem of table detection and structure recognition. The method described in this thesis is data-driven and it does not rely on hand-crafted features or any rules and heuristics to detect the presence of the table and dissect its structure. The model generalizes well to the unseen datasets especially for the table detection task. The following are the contributions of this work:

- We introduce a deep learning based supervised solution for table detection and structure recognition, in which the general-purpose object detector is tailored to the quite distinct world of document, newspaper, article *etc.* images. Transfer learning is employed by fine-tuning the pre-trained model on COCO dataset (Lin et al., 2014) using *Faster R-CNN* (Ren et al., 2015) for detection and recognition. In comparison to the existing state-of-the-art models, this model is simpler to execute, adaptable to any dataset, faster to train and generates better and robust results.

- This work introduces a tailored quantitative metric based on Intersection over Union (IoU)

to not penalize the predicted bounding boxes bigger than the ground truth annotations by a certain factor inspired from Günther et al. (2017).

- Furthermore, this works contributes towards the unification of different data sources into a common annotation format. We also present an in-depth discussion with many different models, parameters and comparison to other works. We show the effectiveness of carefully fine-tuning the model on a rather small training dataset and adapt it to tables from unseen and completely different domains.

The results present in this thesis are state-of-the-art. The idea to use simple-to-work model for object detection once again prove its dominance.

## Document Organization

The rest of this thesis is outlined as follows: The chapter 2 "Related work" gives a detailed timeline of works done in the area of table detection and structure recognition from models trained on hand-crafted features to data-driven models. We also draw comparison to the existing models in this chapter. Chapter 3 "Datasets" describes all the available datasets and their different formats. We also highlight on how the documents are collected and what their compositions are. In chapter 4 "Background and Proposed methodology", We explain the process followed to train the deep neural network like data pre-processing, use of pre-trained models *etc.* and the different elements that constitute the Faster R-CNN network. This chapter also gives a detailed overview of model parameters and other implementation details. Chapter 5 "Experiments and Results" describes all the experiments performed. We also introduce the quantitative metrics used for training and evaluating and present the tailored Intersection over Union (IoU) metric for the task of table detection and structure recognition. This chapter also details the results obtained from different experiments and draw insights from them. We also compare our results with other state-of-the-art methods and show the superiority of our results. In chapter 6 "Discussion", We present the in-depth explanations of different model combinations and prove the choice for our model. We, also, explain the choice of tuning different parameters and their effects on the prediction samples and highlight the models limitations in this chapter. The last chapter 7 "Conclusion and Future work" provides the summary of the work and describe the readers with the take-aways and concludes with an outlook on potential future research directions.

# Chapter 2

# Related Work

We generate and handle documents almost every day and despite the progress made in Natural Language Processing and Computer Vision, the domain of table data extraction is attracting more attention from the research community recently. This is also confirmed by the limited availability of highly annotated datasets.

Figure 2.1 highlights the major milestones achieved in the field of table data extraction. One of the first contribution work was done by Kieninger and Dengel (1998), Kieninger and Dengel (1999), Kieninger and Dengel (2001) using the rule-based approach in 1998. First successful use of machine learning algorithm applied to table extraction was in 1999 by Ng et al. (1999) using decision tree and artificial neural network. Then, in 2013, there was a considerable progress made in this field with the advent of deep learning techniques like image transformations or gainful use of machine learning models like support vector machines (SVM). With the introduction of Faster R-CNN (Ren et al., 2015) in 2015, there was already a huge success in table detection as described in DeepDeSRT (Schreiber et al., 2017). In the later years, there are many modifications introduced in DeepDeSRT and there was a continuous improvement in this research. Researchers also introduced the use of graph neural networks for the first time in Qasim et al. (2019b) to reconstruct the inherent relation between different cells, rows and columns in a table. Researchers also built a decent performing model based on Gated Recurrent Units (GRUs) (Cho et al., 2014) in 2019 (Khan et al., 2019). What lacked throughout these researches though was the involvement of the highly annotated, diverse datasets for table structure recognition. In this work, We make use of the largest table extraction dataset, *PubTables-1M* (Smock et al., 2021), for table structure recognition and present the state-of-the-art results on this dataset.

In this chapter, We will discuss the important works done in this field. First, We will describe the methods which heavily rely on heuristics and rules and later we will focus the attention on the methods that employ deep learning techniques. It is crucial to highlight that certain deep learning methods rely on a few rules to correct predictions, and it is difficult to determine if the approach should be categorized as rule-based or deep learning-based.

## 2.1 Heuristic and Rule-based methods

Tabular data extraction is a well-known topic with solutions that have evolved over the last two decades. One of the first works in table detection in text files using heuristics was performed by Tupaj et al. (1996) followed by Pyreddy and Croft (1997). During the next few years, a novel heuristic-based approach of detecting tables in document images was demonstrated in Kieninger and Dengel (1998), Kieninger and Dengel (1999) and Kieninger and Dengel (2001) forming a combined table detection and recognition system named T-Recs. The input to the T-Recs is the bounding boxes around the word. Depending on the vertical and horizontal overlaps, these boxes are

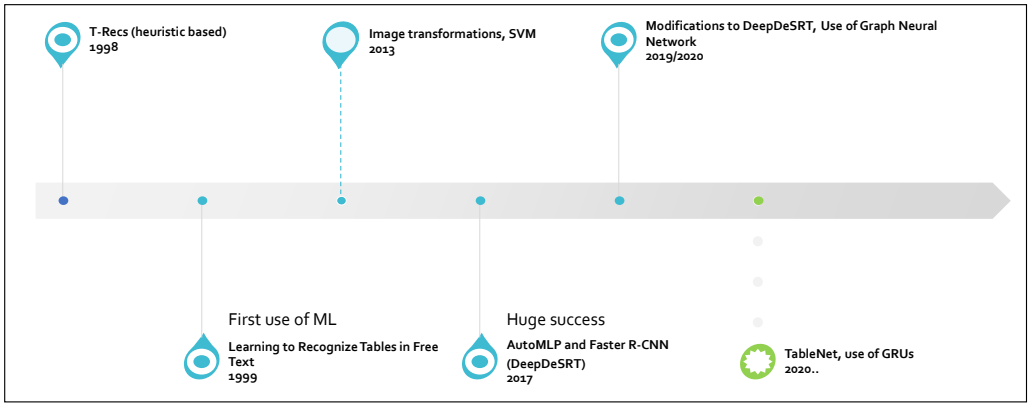



Figure 2.1: TIMELINE OF TABLE RESEARCH. The figure represents a timeline of major milestones achieved in table detection and structure recognition.


Figure 2.2: T-RECS BOTTOM UP CLUSTERING APPROACH. A sample block and the area that contains potential overlapping words (gray stripe over the initial word "consists"). The words (or bounding boxes) that are "touched" by this virtual stripe will be clustered top the same block as the initial words. *Source* Kieninger and Dengel (1998).

merged into rows and columns using a bottom-up method by forming a segmentation graph as shown in Figure 2.2. The main issue with this technique is that the outcome is dependent on a large number of rule-based defined parameters. Furthermore, the method fails if the OCR system fails to accurately detect bounding boxes of the words.

Hu et al. (1999) developed an algorithm that accepts $n$ lines as input and groups a number of lines to construct a table by taking a measure of confidence into account. Merit, scores, and line correlation are some of the indicators used to assess confidence. However, this method has the disadvantage of not working with multi-column table layouts.

Ng et al. (1999) presented a rule based method involving a machine learning model for table structure recognition in free text. Their method used decision trees and artificial neural network. The learning method uses purely surface features like the proportion of the kinds of characters and their relative locations in a line and across lines to recognize tables. They define a set of rules to identify rows *(hline)*, columns *(vline)* and *hline*, *vline*, special characters *etc.* are used to define what constitutes a column and a row. The models fails to generalizes because of the number of rules defined to create features for the learning model.

Wang et al. (2004) attempted to solve the table structure recognition problem using a data-driven method like the X-Y cut algorithm in Shafait et al. (2008a) which is a probabilistic method. The probabilities used in this statistical approach are generated from a huge training corpus. This approach, which takes into consideration the distances between neighboring words, is suitable to single column, double column, and mixed column layouts.

Kasar et al. (2013) trained a support vector machine (SVM) using the hand-crafted features. Despite the fact that no heuristic rules or user-defined parameters are required, the method's use remains limited since it is largely reliant on the occurrence of ruling lines. The technique uses an

algorithm to detect the presence of vertical and horizontal lines in the input image. A collection of 26 low-level features are yielded by the set of intersecting vertical and horizontal lines. Their systematic approach can be seen in the block diagram in Figure 2.3.

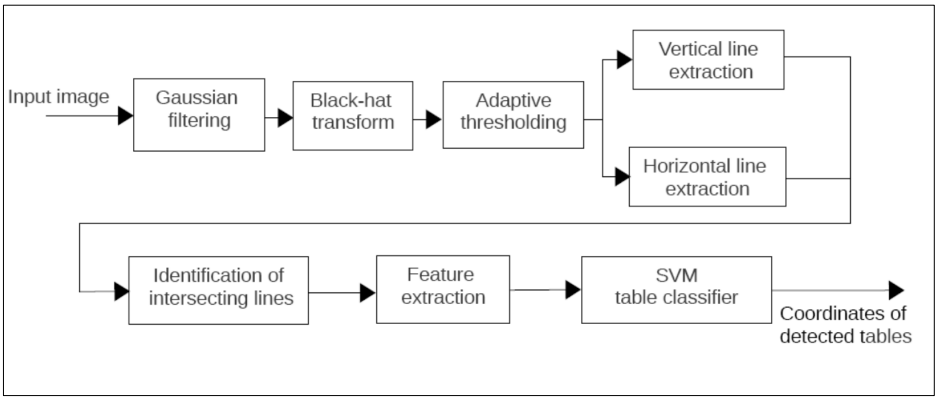



Figure 2.3: SCHEMATIC BLOCK DIAGRAM OF THE PROPOSED APPROACH IN (KASAR ET AL., 2013). During pre-processing, horizontal and vertical lines are identified in the image, and an SVM classifier is trained to detect tables based on the properties of the detected lines.

For table detection, Rashid et al. (2017) use a machine learning method. They used a bottom-up method, extracting characteristics from each word as feature vectors, which are then used to train an AutoMLP classifier. OCR is used to retrieve the word list and geometrical position of each word for the identification of "table" or "non-table" words. The feature vectors include characteristics such as the distance between each word and its neighbor, the width and height of the word, white space between words *etc*. During post processing, nearby class labels of a word are evaluated and the current word's class label is updated to the majority count in the neighborhood. If both, the right and the left neighbors are recognized as "table" then the label of the word under consideration is also changed to "table".

## 2.2 Deep Learning methods

From 2015 onward, Deep Learning models have been extensively used for tackling the table detection and structure recognition problem. Gilani et al. (2017), Schreiber et al. (2017), Arif and Shafait (2018) and Siddiqui et al. (2018) made use of Faster R-CNN (Ren et al., 2015) (Section 4.2.1) each with their own approach to extract hand-crafted features and employ pre-processing methodologies.

Schreiber et al. (2017) proposed a deep network and data-driven based approach, DeepDeSRT, for table detection and structure recognition. DeepDeSRT detects rows and columns using a semantic segmentation model which has skip pooling characteristics and FCN-8 architecture. Additionally, during pre-processing, all the tables are stretched vertically to aid for the row separation and also expanded horizontally to emphasize the boundaries between columns. Gilani et al. (2017) converted images of the documents to natural images by using distance transformation techniques and then passed them to the Faster R-CNN network. This strategy does not take into account the foreground and the background features of the tables. Arif and Shafait (2018) further enhanced this work by pre-processing the document images in 2 stages. They color coded both

types of data to aid the deep learning system after seeing that the tables include more numeric data than textual data. In the second stage, they applied image modification technique from Gilani et al. (2017) on document images and passed them to the Faster R-CNN network. On the UNLV data-set, this technique yielded even better results.

For table detection, Shahzad et al. (2019) employed hand-crafted features and a deep neural network (see Figure 2.4 and Figure 2.5). They utilize both background and foreground features. The classic feature engineering technique (T-Recs) is used to encode foreground features, which are then supplied as input to the deep learning module. As in Gilani et al. (2017), the background information is encoded and added to all three channels. They employ the distance transform to distinguish the background text from the foreground text. They improved the accuracy of Faster R-CNN by adding external features, effectively combining hand engineering with deep learning approaches.

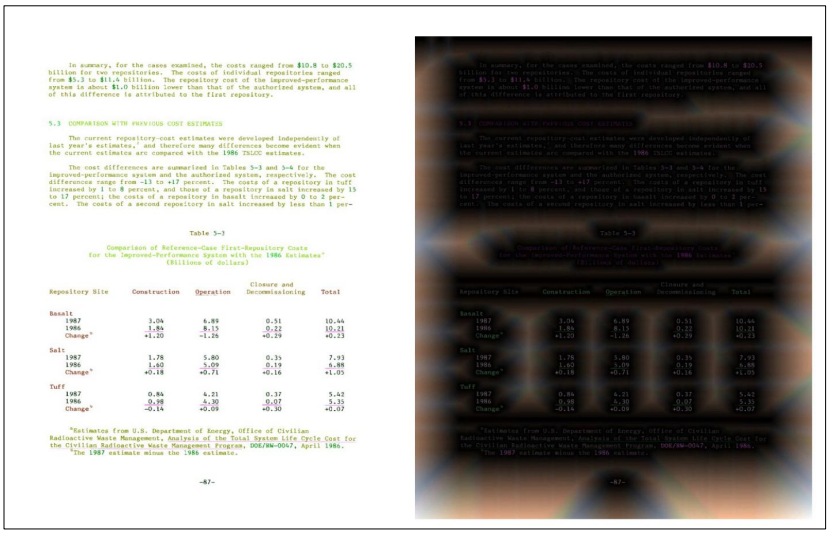


Figure 2.4: FEATURE ENGINEERED IMAGES. Left: Feature Engineered Images without background. Right: Feature Engineered Images with Background Transformation. *Source* Shahzad et al. (2019)

Another important work in table data extraction called TableNet was done by Paliwal et al. (2019). The researchers presented an end-to-end deep learning model that has been trained to extract table data from scanned document images. The model employs a pre-trained VGG-19 base network (Simonyan and Zisserman, 2015) which is used as encoding layers. Then there are two decoder branches for: 1) Table region segmentation, and 2) Column segmentation inside the table. Masks for table and column regions are generated to filter out these regions. All the word coordinates are inferred using Tesseract OCR. A row is defined using few rules utilizing these filtered words. Following that, a rule-based row extraction is used during post-processing to extract data from table cells. They produce comparable results to DeepDeSRT. Their approach can be seen in Figure 2.6.

One of the contrasting approaches in table data extraction is employed by Khan et al. (2019) where the researchers make use of the bi-directional GRUs, see Figure 2.7, rather than the Faster R-CNN as we saw above. In this work, researchers try to tackle the limitations of DeepDeSRT. In DeepDeSRT, parts of input image is processed by an FCN (Fully Convolutional Network) filter to convert it into the output information. As described by Khan et al. (2019), "this fails to capture

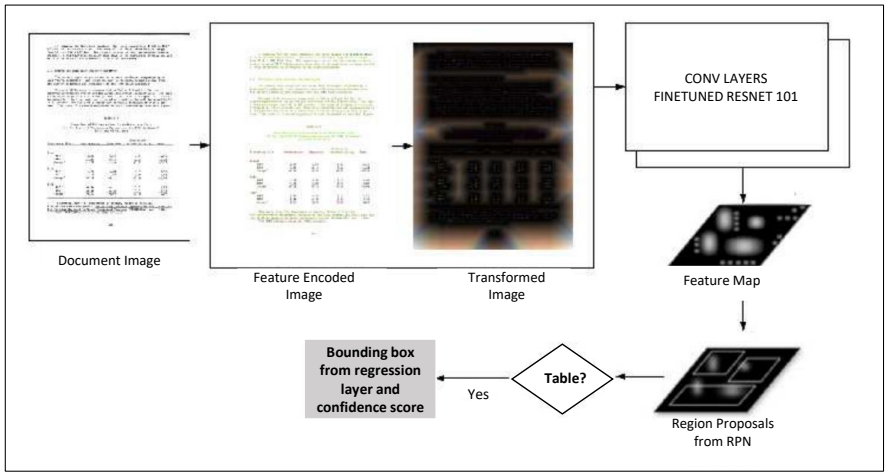



Figure 2.5: PROPOSED APPROACH BY SHAHZAD ET AL. (2019). The original image is subjected to a distance transform. Both images are added, and the final image is passed into the feature extractor. The feature map created by the feature extractor is then sent to the region proposal network (RPN), which suggests regions where tables may exist. As input, the detection network analyses the suggested areas and categorizes them as table or non-table regions. *Source* Shahzad et al. (2019)

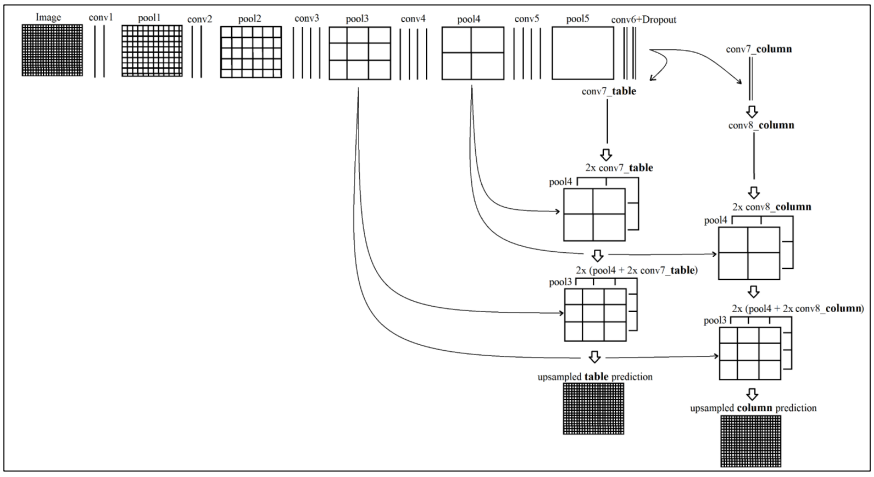



Figure 2.6: TABLENET ARCHITECTURE. Pre-trained VGG-19 is used as base network. Common encoding layers are from `conv1` to `pool5` for both table and column network. `conv7` column and `conv7` are the two decoding networks to produce separate table predictions and column predictions. *Source* Paliwal et al. (2019).

the fact that the rows and columns in a table follow a unique repetitive sequence of in-between spacing and data length as the information of the next and the previous row-column elements is not taken into account". The essence of the strategy described in this paper is to use recurrent neural networks to detect segmentation space between rows and columns. To tackle this challenge, they employ three modules: 1) *Image processing:* To begin, the images are cleaned up by

eliminating the ruling lines and other non-text foreground items so that the structure of the table is more evident. The images are then adaptively binarized (Shafait et al., 2008b) to make the pixel intensities uniform and later images are run through a dilation kernel. The dilation assists the model in learning the pattern of dividers of row and column. 2) *Row-Column classifier:* The bi-directional GRU models rows and columns as timesteps and predicts future row/column components using information from prior row/column components. The model is able to learn the pattern of inter-row and inter-column gaps, as well as the sequence of recurrence of row-column components. 3) *Post processing:* Finally, during the classification, the segmentation space is parsed and the classifier predicts the rows and columns.

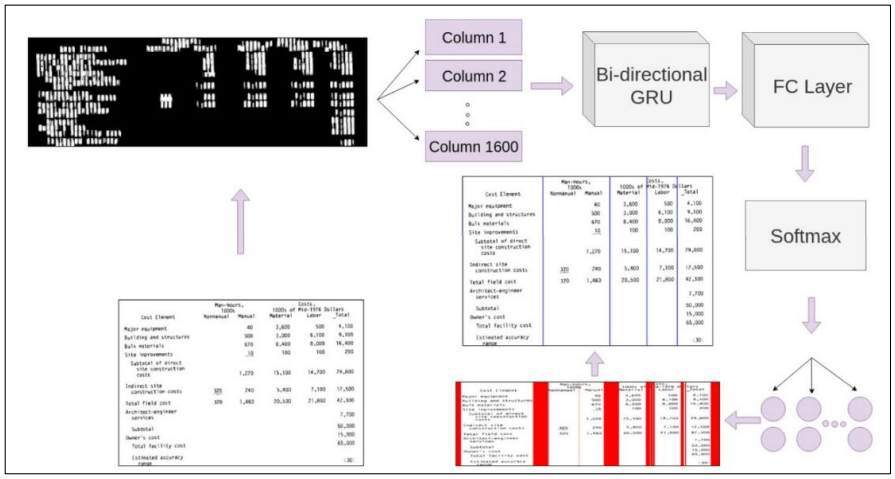



Figure 2.7: BI-GRU ARCHITECTURE FOR COLUMN CLASSIFICATION. The output vector of the network is post-processed to get column segmentation boundaries. A similar network is used for row classifier. *Source* Khan et al. (2019).

Qasim et al. (2019b) make use graph neural network for the first time to solve the task of table structure recognition as shown in Figure 2.8. They mostly use synthetic data to work with. They extract a feature map for an image using the CNN model and words' positions are extracted using an OCR engine. Then image features, corresponding to the words' positions, are gathered and concatenated with positional features to form the input features to an interaction network to generate representative features. The interaction network used is the modified versions of DGCNN and GravNet presented by Wang et al. (2019) and Qasim et al. (2019a) respectively. In the interaction network, sampling for cells, rows, and columns is done independently for each vertex (i.e. word). The representative characteristics for each sample pair are concatenated again and fed into the three dense networks for classification (cell, row, and column network). In addition, they use an unique Monte Carlo-based approach to improve the training routine.

The current researches also highlight the importance of data augmentation for enhanced recognition. Khan et al. (2021) describe the data augmentation approach, which results in structural changes in table images by replicating and deleting rows and columns. They also present a data-driven probabilistic approach for producing parameters that influence the augmented data's layout. The leaders of the ICDAR 2019 table detection and recognition competition, TableRadar (Gao et al., 2019) also use post-processing methods over the original output of the network. TableRadar is a Faster R-CNN based model that merges regions with overlapping areas greater than the set

threshold and detects lines in prospective table areas and if the detected lines continues beyond the table border, the table bounding box is expanded accordingly. The runner-up, NLPR PAL (Gao et al., 2019), employed some heuristics with a fully convolutional network to divide image pixels into two categories: table and background, and subsequently table areas are recovered using Connected Component Analysis (CCA)[1].

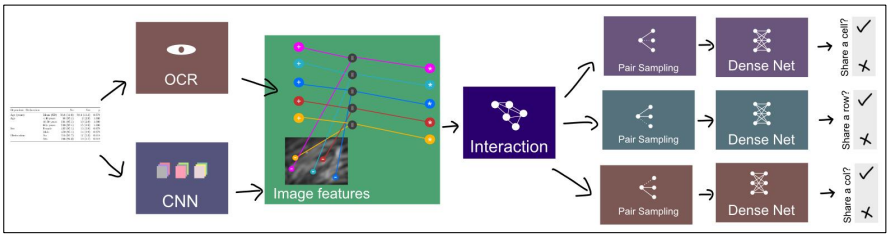



Figure 2.8: PICTORIAL REPRESENTATION OF ARCHITECTURE FROM QASIM ET AL. (2019B). For an image, a feature map is extracted using the CNN and the corresponding words' positions are extracted using an OCR and concatenated before passing to the interaction network. In the interaction network, sampling for cells, rows, and columns is done independently for each vertex (i.e. word). The representative characteristics for each sample pair are concatenated again and fed into the three dense networks for classification (cell, row, and column network).

In this thesis, we draw inspiration from some of these works and compare the results obtained for both the tasks.

[1] https://en.wikipedia.org/wiki/Connected-component_labeling

# Chapter 3

# Datasets

The works mentioned in the last chapter make use of only few of the publicly available datasets published by the research community. The size of these datasets is very small to capture the variety of tables existing in the real world. Thus, these small datasets fail to capture the variation of tables and the models do not perform well on when tested on the unseen dataset. It risks overfitting in deep neural networks and hence, poor generalization. Researchers have tried the transfer learning approach to tackle this issue but this does not offset the requirement for a large-scale diverse dataset.

In this work we make use of the small datasets - ICDAR 2013 (Göbel et al., 2013), ICDAR 2019 (Gao et al., 2019), ISRI-OCR (part of UNLV) (Shahab, 2010), Marmot (Fang et al., 2012) and large datasets - TableBank (Li et al., 2020), PubTables-1M (Smock et al., 2021) for the table detection and structure recognition.

## ICDAR 2013 Dataset

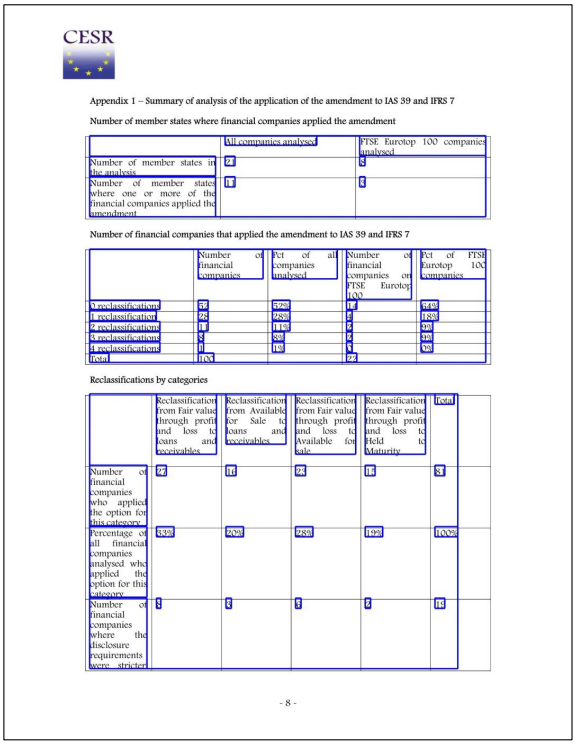
ICDAR 2013 table dataset is a widely used small dataset for the problem of table detection and structure recognition. The dataset (see sample in Figure 3.1) is made up of PDF files that have been transformed into images for the purpose of performing an image-based table identification approach. The dataset consists of 238 pages overall with 67 documents, with word-level annotations provided. We use this dataset for table detection only but evaluation is performed on some images for structure recognition.

## ICDAR 2019 Dataset

ICDAR 2019 dataset contains two types of document images - historical and modern (see Figure 3.2 and Figure 3.3). Modern document images are mostly derived from scientific papers and commercial publications, whereas historical document images are primarily derived from handwritten historical documents. 600 images are set aside for training and 240 images are set aside for testing when it comes to modern document images. For the historical portion, 600 images are set aside for training and 199 images are set aside for testing. It should be highlighted that there is no annotation for table structure detection in the modern images, therefore the historical dataset is obsolete for current-day application. As a result, we solely utilize this dataset for the table detection part.

## ISRI-OCR Dataset

ISRI-OCR is a part of the UNLV dataset. This dataset has served its purpose to benchmark OCR algorithms in the past and recently a part of it has also been used in the table detection task. The



CESR

Appendix 1 – Summary of analysis of the application of the amendment to IAS 39 and IFRS 7

Number of member states where financial companies applied the amendment

| | All companies analysed | FTSE Eurotop 100 companies analysed |
|---|---|---|
| Number of member states in the analysis | 27 | 8 |
| Number of member states where one or more of the financial companies applied the amendment | 15 | 6 |

Number of financial companies that applied the amendment to IAS 39 and IFRS 7

| | Number of financial companies | Pct of all companies analysed | Number of financial companies on FTSE Eurotop 100 | Pct of FTSE Eurotop 100 companies |
|---|---|---|---|---|
| 0 reclassifications | 54 | 52% | 1 | 14% |
| 1 reclassification | 25 | 28% | 3 | 43% |
| 2 reclassifications | 11 | 11% | 0 | 0% |
| 3 reclassifications | 3 | 3% | 2 | 29% |
| 4 reclassifications | 0 | 0% | 0 | 0% |
| Total | 103 | | 7 | |

Reclassifications by categories

| | Reclassification from Fair value through profit and loss to loans and receivables | Reclassification from Available for Sale to loans and receivables | Reclassification from Fair value through profit and loss to Available for sale | Reclassification from Fair value through profit and loss to Held to Maturity | Total |
|---|---|---|---|---|---|
| Number of financial companies who applied the option for this category | 27 | 11 | 23 | 11 | 37 |
| Percentage of all financial companies analysed who applied the option for this category | 73% | 30% | 28% | 11% | 100% |
| Number of financial companies where the disclosure requirements were stricter | 8 | 3 | 8 | 2 | 11 |

- 8 -

Figure 3.1: AN ANNOTATED IMAGE SAMPLE FROM ICDAR 2013. The annotation for the table can be seen which is not very precise. ICDAR 2013 provides text bounding box annotations rather than row/column annotations.

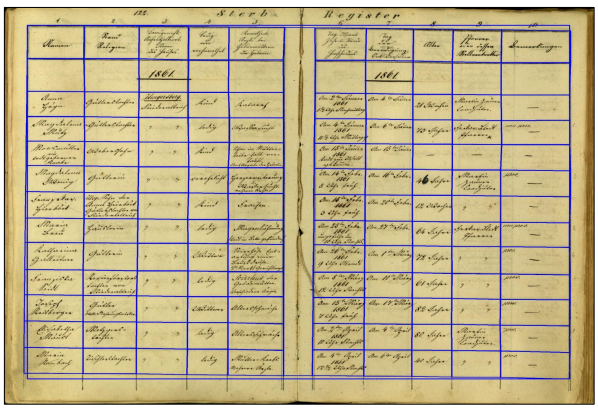
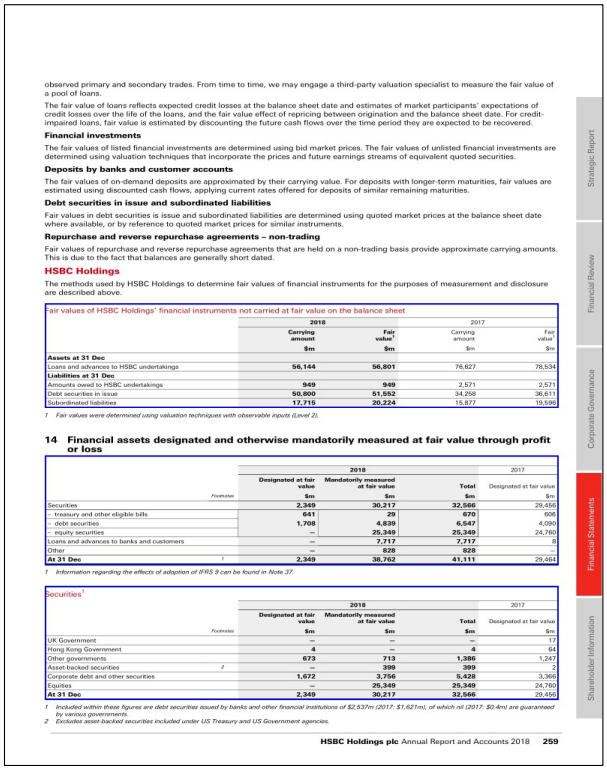


Figure 3.2: AN ANNOTATED IMAGE SAMPLE FROM ICDAR 2019 HISTORICAL DATASET. The figure shows annotations for the table itself and the cell annotations rather than text annotations like in ICDAR 2013. It does not include row/column annotations.

dataset contains documents of varying layouts and domains including research articles, magazines, technical reports *etc.* There are total of 2889 images in this dataset and 427 of them contains



observed primary and secondary trades. From time to time, we may engage a third-party valuation specialist to measure the fair value of a pool of loans.

The fair value of loans reflects expected credit losses at the balance sheet date and estimates of market participants' expectations of credit losses over the life of the loans, and the fair value effect of repricing between origination and the balance sheet date. For credit-impaired loans, fair value is estimated by discounting the future cash flows over the time period they are expected to be recovered.

**Financial investments**

The fair values of listed financial investments are determined using bid market prices. The fair values of unlisted financial investments are determined using valuation techniques that incorporate the prices and future earnings streams of equivalent quoted securities.

**Deposits by banks and customer accounts**

The fair values of on-demand deposits are approximated by their carrying value. For deposits with longer-term maturities, fair values are estimated using discounted cash flows, applying current rates offered for deposits of similar remaining maturities.

**Debt securities in issue and subordinated liabilities**

Fair values in debt securities is issue and subordinated liabilities are determined using quoted market prices at the balance sheet date where available, or by reference to quoted market prices for similar instruments.

**Repurchase and reverse repurchase agreements – non-trading**

Fair values of repurchase and reverse repurchase agreements that are held on a non-trading basis provide approximate carrying amounts. This is due to the fact that balances are generally short dated.

**HSBC Holdings**

The methods used by HSBC Holdings to determine fair values of financial instruments for the purposes of measurement and disclosure are described above.

Fair values of HSBC Holdings' financial instruments not carried at fair value on the balance sheet

| | 2018 | | 2017 | |
|---|---|---|---|---|
| | Carrying amount | Fair value[1] | Carrying amount | Fair value |
| | $m | $m | $m | $m |
| **Assets at 31 Dec** | | | | |
| Loans and advances to HSBC undertakings | 56,144 | 56,801 | 76,627 | 78,534 |
| **Liabilities at 31 Dec** | | | | |
| Amounts owed to HSBC undertakings | 949 | 949 | 2,571 | 2,571 |
| Debt securities in issue | 50,800 | 51,552 | 34,258 | 36,611 |
| Subordinated liabilities | 17,715 | 20,224 | 15,877 | 19,596 |

1 *Fair values were determined using valuation techniques with observable inputs (Level 2).*

**14 Financial assets designated and otherwise mandatorily measured at fair value through profit or loss**

| | 2018 | | | 2017 |
|---|---|---|---|---|
| | Designated at fair value | Mandatorily measured at fair value | Total | Designated at fair value |
| Footnotes | $m | $m | $m | $m |
| Securities | 2,349 | 30,217 | 32,566 | 29,466 |
| – treasury and other eligible bills | 641 | 29 | 670 | 606 |
| – debt securities | 1,708 | 4,839 | 6,547 | 4,090 |
| – equity securities | – | 25,349 | 25,349 | 24,760 |
| Loans and advances to banks and customers | – | 7,717 | 7,717 | 8 |
| Other | – | 828 | 828 | – |
| **At 31 Dec** | 2,349 | 38,762 | 41,111 | 29,464 |

1 *Information regarding the effects of adoption of IFRS 9 can be found in Note 37.*

Securities[1]

| | 2018 | | | 2017 |
|---|---|---|---|---|
| | Designated at fair value | Mandatorily measured at fair value | Total | Designated at fair value |
| Footnotes | $m | $m | $m | $m |
| UK Government | – | – | – | 17 |
| Hong Kong Government | 4 | – | 4 | 64 |
| Other governments | 673 | 713 | 1,386 | 1,247 |
| Asset-backed securities 2 | – | 399 | 399 | 2 |
| Corporate debt and other securities | 1,672 | 3,756 | 5,428 | 3,366 |
| Equities | – | 25,349 | 25,349 | 24,760 |
| **At 31 Dec** | 2,349 | 30,217 | 32,566 | 29,466 |

1 *Included within these figures are debt securities issued by banks and other financial institutions of $2,537m (2017: $1,621m), of which nil (2017: $0.4m) are guaranteed by various governments.*
2 *Excludes asset-backed securities included under US Treasury and US Government agencies.*

**HSBC Holdings plc** Annual Report and Accounts 2018 **259**

Strategic Report | Financial Review | Corporate Governance | Financial Statements | Shareholder Information

Figure 3.3: AN ANNOTATED IMAGE SAMPLE FROM ICDAR 2019 MODERN DATASET. The blue bounding box represents the tables in the image. The modern dataset does not include annotations for table structure recognition.

at least one table. Sometimes it's questionable if the bounding box covers a table or some other document objects like figures, list *etc*. See the sample of an image in Figure 3.4.

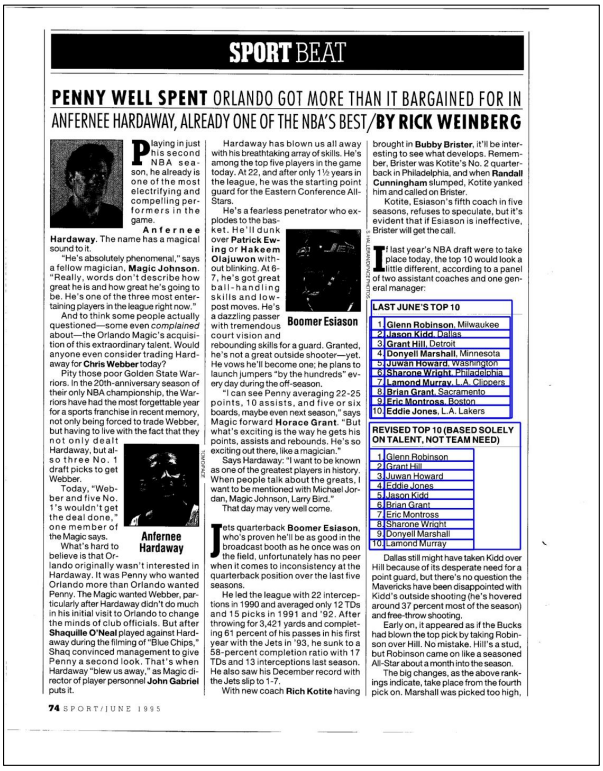
## Marmot Dataset

Marmot (see sample in Figure 3.5) is one of the most used publicly available datasets for the task of table detection and structure recognition. There are 2000 pages in PDF format in Chinese and English language. Over 1500 conference and journal articles are crawled, spanning numerous topics from 1970 to the most recent 2011 publications. The Chinese pages are extracted from over 120 e-books with a variety of subjects and the English pages are crawled from the Citeseer website[1]. The dataset shows a great variety in terms of layout and table styles. This dataset is used for the table detection part only as it is impossible to expand the text bounding boxes to row/column bounding boxes with no information on how text bounding boxes were created unlike in ICDAR 2013 as described in Section 4.1.2.

## TableBank Dataset

TableBank (see sample in Figure 3.6) is one of the largest datasets available with over 417,000 tables and 278,000 images. It has been created from business documents, official filings, research

[1] https://en.wikipedia.org/wiki/CiteSeerX



SPORT BEAT

**PENNY WELL SPENT** ORLANDO GOT MORE THAN IT BARGAINED FOR IN ANFERNEE HARDAWAY, ALREADY ONE OF THE NBA'S BEST/**BY RICK WEINBERG**

Playing in just his second NBA season, he already is one of the most electrifying and compelling performers in the game.

**Anfernee Hardaway.** The name has a magical sound to it.

"He's absolutely phenomenal," says a fellow magician, **Magic Johnson**. "Really, words don't describe how great he is and how great he's going to be. He's one of the three most entertaining players in the league right now."

And to think some people actually questioned—some even *complained* about—the Orlando Magic's acquisition of this extraordinary talent. Would anyone even consider trading Hardaway for **Chris Webber** today?

Pity those poor Golden State Warriors. In the 20th-anniversary season of their only NBA championship, the Warriors have had the most forgettable year for a sports franchise in recent memory, not only being forced to trade Webber, but having to live with the fact that they not only dealt Hardaway, but also three No. 1 draft picks to get Webber.

Today, "Webber and five No. 1's wouldn't get the deal done," one member of the Magic says.

What's hard to believe is that Orlando originally wasn't interested in Hardaway. It was Penny who wanted Orlando more than Orlando wanted Penny. The Magic wanted Webber, particularly after Hardaway didn't do much in his initial visit to Orlando to change the minds of club officials. But after **Shaquille O'Neal** played against Hardaway during the filming of "Blue Chips," Shaq convinced management to give Penny a second look. That's when Hardaway "blew us away," as Magic director of player personnel **John Gabriel** puts it.

Anfernee Hardaway

Hardaway has blown us all away with his breathtaking array of skills. He's among the top five players in the game today. At 22, and after only 1½ years in the league, he was the starting point guard for the Eastern Conference All-Stars.

He's a fearless penetrator who explodes to the basket. He'll dunk over **Patrick Ewing** or **Hakeem Olajuwon** without blinking. At 6-7, he's got great ball-handling skills and low-post moves. He's a dazzling passer with tremendous court vision and rebounding skills for a guard. Granted, he's not a great outside shooter—yet. He vows he'll become one; he plans to launch jumpers "by the hundreds" every day during the off-season.

"I can see Penny averaging 22-25 points, 10 assists, and five or six boards, maybe even next season," says Magic forward **Horace Grant**. "But what's exciting is the way he gets his points, assists and rebounds. He's so exciting out there, like a magician."

Says Hardaway: "I want to be known as one of the greatest players in history. When people talk about the greats, I want to be mentioned with Michael Jordan, Magic Johnson, Larry Bird."

That day may very well come.

Boomer Esiason

Jets quarterback **Boomer Esiason**, who's proven he'll be as good in the broadcast booth as he once was on the field, unfortunately has no peer when it comes to inconsistency at the quarterback position over the last five seasons.

He led the league with 22 interceptions in 1990 and averaged only 12 TDs and 15 picks in 1991 and '92. After throwing for 3,421 yards and completing 61 percent of his passes in his first year with the Jets in '93, he sunk to a 58-percent completion ratio with 17 TDs and 13 interceptions last season. He also saw his December record with the Jets slip to 1-7.

With new coach **Rich Kotite** having brought in **Bubby Brister**, it'll be interesting to see what develops. Remember, Brister was Kotite's No. 2 quarterback in Philadelphia, and when **Randall Cunningham** slumped, Kotite yanked him and called on Brister.

Kotite, Esiason's fifth coach in five seasons, refuses to speculate, but it's evident that if Esiason is ineffective, Brister will get the call.

If last year's NBA draft were to take place today, the top 10 would look a little different, according to a panel of two assistant coaches and one general manager:

| LAST JUNE'S TOP 10 |
|---|
| 1 Glenn Robinson, Milwaukee |
| 2 Jason Kidd, Dallas |
| 3 Grant Hill, Detroit |
| 4 Donyell Marshall, Minnesota |
| 5 Juwan Howard, Washington |
| 6 Sharone Wright, Philadelphia |
| 7 Lamond Murray, L.A. Clippers |
| 8 Brian Grant, Sacramento |
| 9 Eric Montross, Boston |
| 10 Eddie Jones, L.A. Lakers |

| REVISED TOP 10 (BASED SOLELY ON TALENT, NOT TEAM NEED) | |
|---|---|
| 1 | Glenn Robinson |
| 2 | Grant Hill |
| 3 | Juwan Howard |
| 4 | Eddie Jones |
| 5 | Jason Kidd |
| 6 | Brian Grant |
| 7 | Eric Montross |
| 8 | Sharone Wright |
| 9 | Donyell Marshall |
| 10 | Lamond Murray |

Dallas still might have taken Kidd over Hill because of its desperate need for a point guard, but there's no question the Mavericks have been disappointed with Kidd's outside shooting (he's hovered around 37 percent most of the season) and free-throw shooting.

Early on, it appeared as if the Bucks had blown the top pick by taking Robinson over Hill. No mistake. Hill's a stud, but Robinson came on like a seasoned All-Star about a month into the season.

The big changes, as the above rankings indicate, take place from the fourth pick on. Marshall was picked too high,

74 SPORT/JUNE 1995

Figure 3.4: AN ANNOTATED IMAGE SAMPLE FROM ISRI-OCR. The bounding boxes represent the ground truth tables and the cell annotations.

papers *etc.* We use this dataset for table detection due to the variety of tables it offers in a document. However, the dataset format for table structure recognition is quite convoluted and gives the HTML tag only without any bounding box coordinates like all the other datasets. It is difficult to integrate this dataset format with the current row/column bounding box annotations and for this purpose, we leave this dataset out of the structure recognition part and leave it for the future to expand and improve the structure recognition dataset.

## PubTables-1M Dataset

It is the largest dataset available for table detection and structure recognition with very high-quality annotations. It has over 460,000 document pages for table detection and over 947,000 document pages for table structure recognition. We use this dataset for table structure recognition because of the high-quality annotations it offers for identifying *table, table row, table column, table projected row header, table column header, table spanning cell* in a table. The table class contains the annotation for the bounding box around the table. The row/column class corresponds to the rows/columns in the table. Table projected row header is also known as section headers (Pinto et al., 2003) and these are in-between rows which contain header(s). The column header is the top-most row in the table. The table spanning cell is the class of cells which span across multiple rows or multiple columns. These classes can be observed in Figure 3.7. Each intersection of a table column and a table row is considered to form a seventh implicit class, table grid cell (Smock et al., 2021).

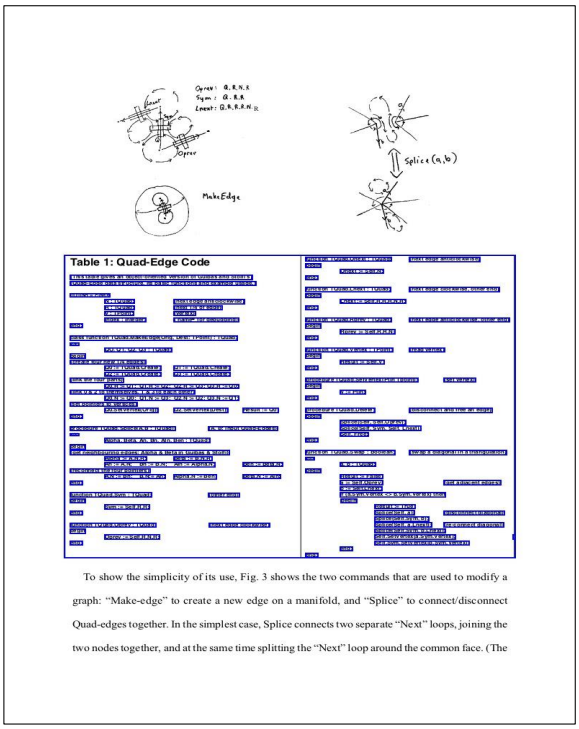



Figure 3.5: AN ANNOTATED IMAGE SAMPLE FROM MARMOT. The figure represents the annotations for a table and the text bounding box annotations. Moreover it can be questioned if this ground truth image is actually a table or not. Such images are used during the training and testing for table detection.

The dataset also contains the content of the cells - words, numbers *etc.* with its bounding box coordinates to facilitate further modeling using natural language processing techniques. Additionally, cells in the headers are *canonicalized* as described in Appendix A.2 and the authors implement multiple quality control steps to ensure the annotations are as free of noise as possible.

*Toyama International Workshop on Higgs as a Probe of New Physics 2015, 11–15, February, 2015* 1

**The classically conformal B−L extended Ma model**

Y. Orikasa
*School of Physics, KIAS, Seoul 130-722, Korea and*
*Department of Physics and Astronomy, Seoul National University, Seoul 151-742, Korea*

We discuss a classically conformal radiative seesaw model with gauged B−L symmetry, in which the B−L symmetry breaking can occur through the Coleman-Weinberg mechanism. As a result, all mass terms are generated and EW symmetry breaking also occurs. We show some allowed parameters to satisfy several theoretical constraints.

**I. INTRODUCTION**

The standard model (SM) has to be still extended so as to include massive neutrinos and dark matter (DM), even though the SM Higgs has been discovered. One of the elegant solutions to resolve this issue is known as radiative seesaw models [1–6], in which active neutrino masses are generated at multi-loop level and exotic fields are naturally introduced in order to get such radiative masses. Such an exotic field can be often identified as a DM candidate. Especially, Ma model [3] is known as a minimal radiative seesaw model including fermionic and bosonic DM candidates.

On the other hand the hierarchy problem arises in the SM. One solution of the hierarchy problem is supersymmetry. This is a beautiful theory however there are no signals in the LHC experiments. In this paper, we take another approach to the hierarchy problem following Bardeen's argument[7]. Bardeen has argued that once the classically conformal symmetry and its minimal violation by quantum anomalies are imposed on the SM, it may be free from quadratic divergences. The models based on this idea are called classically conformal models[8–14]. The classical Lagrangian for these models has no mass terms and all dimensional parameters are dynamically generated. The models need an absence of intermediate scales between the TeV scale and Planck scale. Then the Planck scale physics is directly connected to the electroweak (EW) physics.

We consider the classically conformal Ma model which is combined Ma model and the classically conformal model. This model connects tiny neutrino mass scale and Planck scale. However the minimal classically conformal Ma model doesn't realize for following two reasons. First the EW symmetry doesn't occur for large top Yukawa coupling. Second the classically conformal symmetry forbids Majorana mass term. The Majorana mass plays an important role in Ma model. We need the extended model. The minimal extension is gauged B−L model. In this model, EW symmetry breaking is triggered by B−L symmetry breaking and Majorana mass term is generated by B−L symmetry breaking.

This paper is based on our work[15].

**II. THE CLASSICALLY CONFORMAL B−L EXTENDED MA MODEL**

| Fermion | $L_L$ | $e_R$ | $N_R$ |
|---|---|---|---|
| $(SU(2)_L, U(1)_Y)$ | $(\mathbf{l}, -1/2)$ | $(\mathbf{1}, -1)$ | $(\mathbf{1}, 0)$ |
| $U(1)_{B-L}$ | $-1$ | $-1$ | $-1$ |
| $Z_2$ | $+$ | $+$ | $-$ |

TABLE I: $L_L$, $e_R$, and $N_R$ have three generations, which is abbreviated.

| Boson | $\Phi$ | $\eta$ | $\varphi$ |
|---|---|---|---|
| $(SU(2)_L, U(1)_Y)$ | $(\mathbf{2}, 1/2)$ | $(\mathbf{2}, 1/2)$ | $(\mathbf{1}, 0)$ |
| $U(1)_{B-L}$ | 0 | 0 | 2 |
| $Z_2$ | $+$ | $-$ | $+$ |

TABLE II: The particle contents for bosons.

Figure 3.6: AN ANNOTATED IMAGE SAMPLE FROM TABLEBANK. The blue bounding boxes indicates the presence of tables in the ground truth image.

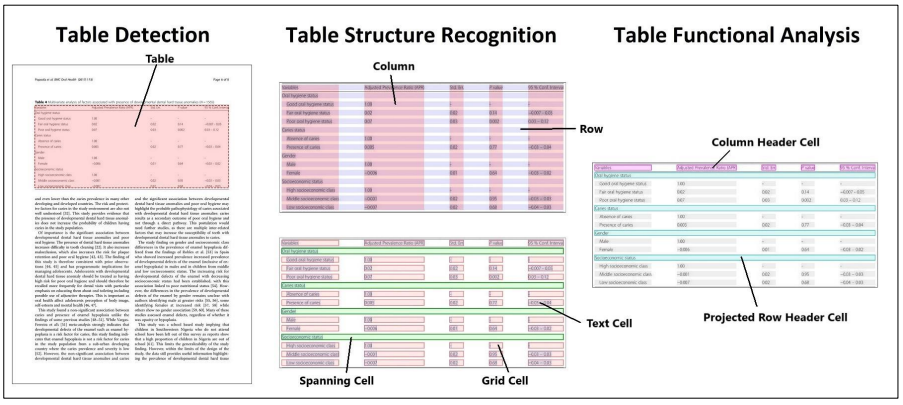



Figure 3.7: AN ANNOTATED IMAGE SAMPLE FROM PUBTABLES-1M. The annotations are added to show the different classes in this dataset. *Source* Smock et al. (2021).

# Chapter 4

# Background and Proposed Methodology

This chapter informs the readers about the data pre-processing steps, model architecture used in this work, use of pre-trained models, and other crucial experimental set-up details. The readers will get acquainted with the important advancements made in object detection and also learn about the important sub-networks and parameters to consider when training a Faster R-CNN network.

## 4.1 Data pre-processing

### 4.1.1 COCO Dataset format

The datasets, ICDAR 2013, ICDAR 2019, ISRI-OCR, TableBank, Marmot, PubTables-1M, come in varying XML (Extensible Markup Language) formats according to the annotator's working comfort. Since our model involves working with the mixture of datasets as input, there is a need to convert all the formats to a common annotation format. For this purpose, we convert all the existing datasets to the MS COCO (Common Objects in Context) format (Lin et al., 2014). The COCO format is a JSON (JavaScript Object Notation) structure describing how labels, coordinates of bounding boxes, metadata (like height, width, sources *etc.*) for each image samples are saved. The XML annotations for above mentioned datasets are usually available for individual page along with the image. The XML annotations are not uniform i.e. some datasets have information for words like coordinates and content and some of the times there is an individual XML annotation file for each table in a document. While processing all datasets requires different approaches, a general process to merge all the XML files into a single COCO file for table detection is to iterate through each file and extract bounding box coordinates for tables and calculate its area and add it to the COCO JSON file along with other available information like page width, page height, path to the image *etc.* Each table in a page and each page in a document is given a unique ID. Tables with incomplete information are left out.

This is also the contribution of this work to provide all the datasets in the common COCO format and encourage the research community to make use of a single format for this task.

### 4.1.2 Bounding Box Expansion

For the task of table detection, we manipulate the bounding box coordinates to have the better predictions. We increase the bounding box coordinates in the horizontal and vertical direction

for tables by 5% relative to the page width and page height. This is done so that the predicted bounding boxes fully contain the tables with some extra margin and, in most cases, it also solves the issue where table captions are not included in the ground truth annotation as in Figure 3.6. The incorrect ground truth annotations which have either no annotations or negative coordinates (implying bounding box outside document) for left/right boundaries or top/bottom boundaries, we set the bounding box boundary as the page width or the page height. Thus, creating more correct data for training. A sample of such an incorrect ground truth file can be seen in Figure 4.1.

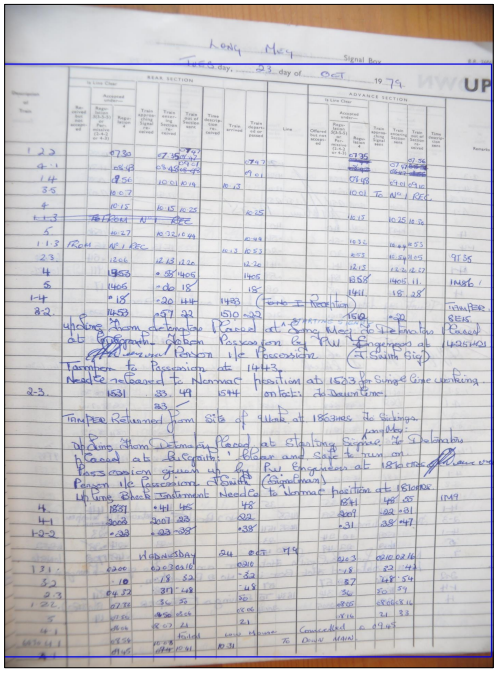


Figure 4.1: AN INCORRECT GROUND TRUTH SAMPLE. This ground truth sample from ICDAR 2019 has incorrect/no table bounding box on the left boundary. Such examples are corrected before training.

For the task of table structure recognition, ICDAR 2013 provides word-level annotations (annotations around the text/text bounding box) for the tables but we predict rows, columns *etc.* for the structure recognition task. Therefore, we use the information provided in the XML annotation files to expand the text bounding box annotations to row, column annotations systematically according to the given `start/end - row/col` tag. These tags have integer values and help in identifying the row and column where the text bounding box belongs. Then the text bounding boxes with the same row tag are merged into one row annotation and the same process is followed for the columns. The intersection of a row and a column generates the cell annotation. The reason to do this is that we use the data-driven approach to model the structure of the table rather than relying on any post-processing methods to relate different text bounding boxes after prediction with each other or to identify in which row/column they belong.

### 4.1.3 Dataset splits

The datasets are split in the ratio of 80-10-10 for the purpose of training, validation, and testing respectively. For some datasets, we do not use all the samples available but rather a subset of them

because of limited training resources and, also, the big dataset like TableBank fails to capture variety of tables alone. This can also lead the model to over-fit on one large dataset i.e. model learns the features dominant in the big dataset and fails to generalize well across different domains.

### Table Detection

For the task of task of table detection, we use the following datasets with their mentioned number of samples for the purpose of training: ICDAR 2013 - 114 samples; ICDAR 2019 - 1200 samples; Marmot - 876 samples; ISRI-OCR - 308 samples; TableBank - 1000 samples.

As noted earlier, the TableBank dataset has more than 278K images. However, we use only 1000 images for training because of the aforementioned reasons. Training on the complete TableBank dataset with current resources will take multiple weeks.

### Table Structure Recognition

To tackle the problem of table structure recognition, we use the ICDAR 2013 and PubTables-1M datasets. While 156K images of PubTables-1M are used in training with 6 annotations: *table, row, column, projected row header, column header, spanning cell*, only 105 images of ICDAR 2013 are used with 2 annotations: *row, column*. Initial experiments reveal that the results are dominated by the PubTables-1M dataset. Hence, for the sake of brevity, we only use PubTables-1M for training. Training on 156K images takes around 5 days. More on training time can be inferenced from Section 4.3.

## 4.2 Model

This section discusses the details of the proposed approach with some background information about different models as described by Girshick et al. (2013), Girshick (2015), Ren et al. (2015) and Gad (2021). We refer to the aforementioned sources to present the model in a concise and understandable manner. Because of how Convolutional Neural Network (CNNs) has impacted the research in the computer vision domain, we use Faster R-CNN network to model the task of table data extraction as an object detection problem. The detection of graphical objects like figures, tables, equations *etc.* is essentially the localization of these objects in a document. This is also called layout segmentation of a document (Zhong et al., 2019). It is easy to draw this analogy to the detection of objects present in the natural environment or a scene. Like natural objects have identifiable characteristics, the tables in a document also have certain identifiable traits. The CNN can harness these characteristics of the table(s) in a document and can extract those features.

The extraordinary performance of Faster R-CNN (Ren et al., 2015) on the PASCAL VOC (Everingham et al., 2010) and COCO (Lin et al., 2014) datasets motivates us to adapt this framework for the task of table data extraction. Since the training of deep neural networks requires vast amounts of labeled data which is a recurring issue in table data extraction, we use domain adaptation and transfer learning in this thesis. The original Faster R-CNN framework is initialized with ImageNet (Russakovsky et al., 2015) pre-trained model and then trained further on the COCO dataset by OpenMMLab (Chen et al., 2019) before making it available to the public.

It is also important to understand the main building blocks of a Faster R-CNN network since it helps in tuning the performance of the model and affects the output significantly. R-CNN (Region-based CNN) (Girshick et al., 2013) is one of the first works done towards object detection by extracting features using a pre-trained CNN. In comparison to the general pipeline of the object identification approach as shown in Figure 4.2 where proposal[1] for prospective objects

[1]Proposal is the area of interest in the image which potentially contains the foreground object.

are generated by the proposal generator and fed to the feature extractor for further extraction of classifying features, R-CNN's contribution is simply extracting features using a convolutional neural network (CNN). Aside from that, everything is the same as in the general object detection pipeline. The R-CNN model is depicted in Figure 4.3.

There are some drawbacks of using R-CNN like the large number of proposals i.e. 2000 generated takes an exceptional amount of time to train as the network needs to classify 2000 proposals per image. Also, it cannot be implemented in real-time as it takes ~47s per image during evaluation since each region proposal is input separately to the CNN for feature extraction. R-CNN relies on the selective search algorithm (Uijlings et al., 2013) to generate region proposals which is a fixed algorithm. Hence, as a general purpose object detector, it leads to the generation of bad proposals.

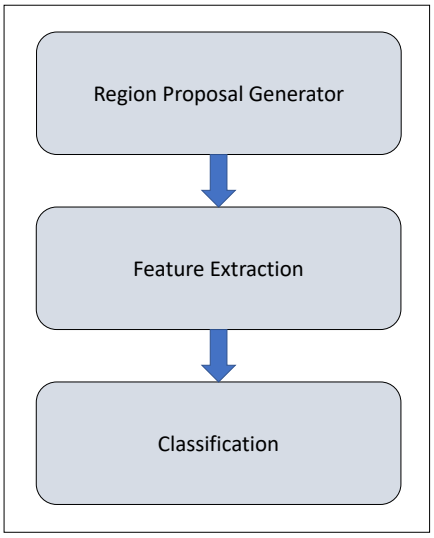



Figure 4.2: GENERAL OBJECT DETECTION PIPELINE. The diagram represents the general constituents of the most object detection models. Proposal for prospective objects are generated by the proposal generator and fed to the feature extractor for further extraction of classifying features.

The same authors address some of these drawbacks of R-CNN and introduce Fast R-CNN (Girshick, 2015). As the name implies, Fast R-CNN is faster in speed than R-CNN. The method is comparable to R-CNN. However, instead of passing the region proposals to the CNN as input, the whole image is given to the CNN to build a convolutional feature map. The region of proposals is determined using the convolutional feature map and by using the proposed RoI Pooling (Region of Interest Pooling) layer (Section 4.2.1), the features are reshaped into equal-length feature vectors before being fed to the fully connected layer.

The reason why Fast R-CNN is faster than R-CNN is that it does not require all the 2000 region proposals to be fed to the CNN independently rather a convolution operation is performed only one time per image to generate the features. This also attributes to using the RoI Pooling layer. The Fast R-CNN's framework can be seen in Figure 4.4. The model is a single stage architecture compared to the three stage architecture of R-CNN. It inputs an image and returns the class probabilities and coordinates of bounding boxes of the detected objects as described in Girshick (2015). Despite being faster, this model also has drawbacks. The generation of region proposals becomes

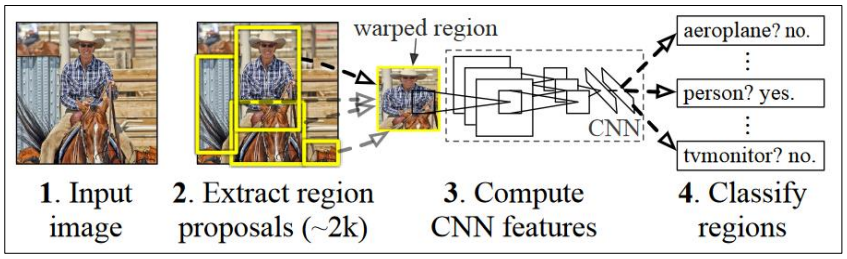



Figure 4.3: OVERVIEW OF R-CNN FRAMEWORK. An image is fed to the system and 2000 region proposals are extracted. For each proposal, features are generated by the CNN and then classified by the class-specific linear SVMs. *Source* Girshick et al. (2013).

the bottleneck as this model also relies on the selective search algorithm to generate proposals. Hence, it is also not able to detect all the target objects in some cases because of bad proposals.

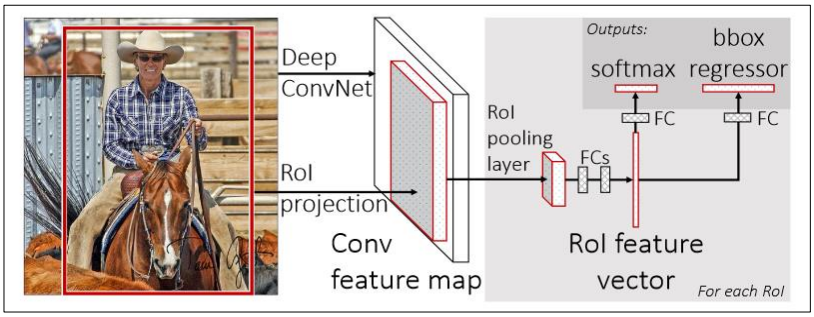


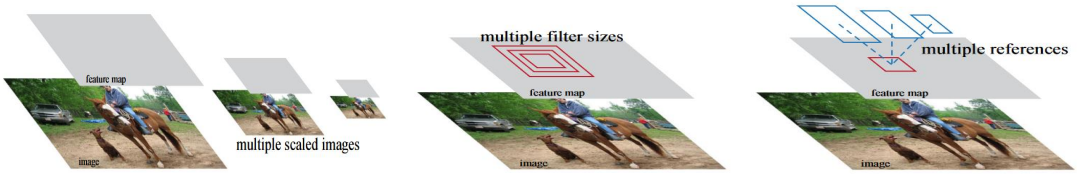

Figure 4.4: OVERVIEW OF FAST R-CNN FRAMEWORK. A FCN is fed an input image and several regions of interest (RoIs). Each RoI is pooled into a equal-length feature map, which is then mapped to a feature vector using fully connected layers. Per RoI, the network produces two output vectors: softmax probabilities for classification and per-class bounding box coordinates. The architecture is trained end-to-end using a multi-task loss. *Source* Girshick (2015).

### 4.2.1 Faster R-CNN

The R-CNN and Fast R-CNN depends on the selective search algorithms to generate region proposals. It is a costly and slow process. Hence, Ren et al. (2015) introduced Faster R-CNN where the model does not require the selective search algorithm but rather the network learns and predicts the region proposals. It is able to do so because of another network called Region Proposal Network (RPN). Some of the main building blocks of Faster R-CNN, as described in Gad (2021), are explained in the sections below. The Faster R-CNN framework consists of two networks: 1) RPN: To generate region proposals. 2) Fast R-CNN: To detect objects in the proposed region.

## Base Network

The base network or the backbone network in the original Faster R-CNN model is either the Zeiler and Fergus model (ZF) (Zeiler and Fergus, 2014) or the VGG-16 model (Simonyan and Zisserman, 2015). The base network is a fully convolutional network. The ZF model has 5 shareable convolutional layers and the VGG-16 has 13 shareable convolutional layers. The base networks help in extracting the high level features of the object for further processing by the other networks. It is the very first network to which the input image is sent. Other variations of backbone network include using ResNet (He et al., 2016) as a backbone with a Feature Pyramid Network (FPN) (Lin et al., 2017) before passing the features to the RPN network. As described in Ren et al. (2015), in contrast to other methods like Fast R-CNN that use pyramids of images (see Figure 4.5(a)) or pyramids of filters (see Figure 4.5(b)), Faster R-CNN introduce novel "anchor" boxes that serves as references at multiple scales and aspect ratios. Their framework can be thought of as a pyramid of reference boxes or pyramid of anchors (see Figure 4.5(c)), which avoids enumerating images or filters of multiple scales or aspect ratios.




(a) Pyramids of images and feature maps (b) Pyramids of filters (c) Pyramids of reference boxes

Figure 4.5: DIFFERENT SCHEMES FOR ADDRESSING MULTIPLE SCALES AND SIZES. In (a), pyramids of images and feature maps are functioned at different scales. In (b), pyramids of filters with multiple scales/sizes are passed on the feature map. Pyramids of reference boxes are used in (c). *Source* Ren et al. (2015).

ResNet (He et al., 2016), Appendix A.1, is one of the most studied architectures in deep learning. It has gained a lot of popularity in the computer vision research community. Consequently, a lot of ResNet variants have been proposed in the last few years and one of which called ResNeXt (Xie et al., 2017) has garnered a lot of attention. The name, ResNeXt, contains Next. It means the next dimension, on top of the ResNet. This next dimension is called the "cardinality" dimension. ResNeXt became the 1st Runner Up of ILSVRC classification task and showed improvements over ResNet. A building block of ResNet and ResNeXt is shown in Figure 4.6.

To improve accuracy, more layers can be stacked or wide layers can be used but the problem is more parameters are introduced as well. Eventually, more computational power is required. ResNeXt deals with this problem. The principle of ResNeXt is stacking the same topology blocks. The hyperparameters like width and filter sizes are shared within the residual blocks (Ma, 2020). In summary: 1) ResNeXt has much more parallel stacking rather than sequential stacking. 2) Cardinality is the number of independent paths, and it provides a new way of adjusting the model capacity. Accuracy can be increased by increasing cardinality rather than going wide or deep.

In this work, we make use of ResNet and ResNeXt architecture and describe their choices by showing experiments in chapter 6.

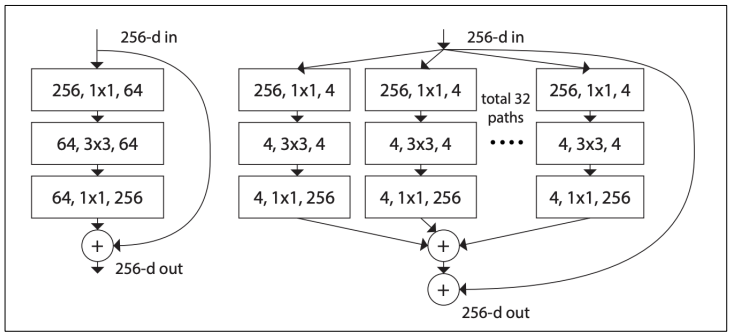



Figure 4.6: RESNET AND RESNEXT BLOCK. A building block of ResNet (left) and ResNeXt (right). ResNeXt has cardinality = 32 with approximately the same complexity. *Source* Xie et al. (2017).

## Region Proposal Network (RPN)

The RPN is a fully convolutional network that generates proposals of potential objects with different sizes and aspect ratios. The RPN uses the mechanism of *attention*[2] to guide the detection network (Fast R-NN) where to look. The visualization of Faster R-CNN's framework can be seen in Figure 4.7. The features generated by the base network and its computations are shared across RPN and Fast R-CNN detection networks.

The main benefits of using the RPN is the generation of learnable region proposals which can be adapted according to the detection task. To do so, it depends on assigning positive/negative values to the anchors (see next Section 4.2.1). This generates better proposals compared to algorithms like selective search. The RPN module is also translation-invariant and uses the same convolutional layers as the detection (Fast R-CNN) network to process the image. As a result, when compared to algorithms like selective search, RPN takes less time to generate proposals and training is done only once.

The output feature map passed to the RPN from the convolutional layers (backbone network) is operated on by a sliding window of size `n x n` followed by two sibling (or similar) `1 × 1` convolutional layers (for regression (*reg*) and classification (*cls*), respectively). Several candidate region proposals are produced for each window and then filtered based on their *objectness scores* depending if the region contains an object or not (Ren et al., 2015). This can be seen in Figure 4.8.

## Anchor

As illustrated in Figure 4.8, several region proposals (maximum $k$) are predicted concurrently at each sliding-window position. Each proposal is parameterized based on a reference box known as an anchor box (or anchors). At default settings, the model employs 3 scales and 3 aspect ratios for anchors which, in turn, generates $k$=9 anchors at each sliding position (Ren et al., 2015). As the anchors are of different scales and ratios, this circumvents the need of using multiple images or filters of different scales. The features are shared between the RPN and further detection network due to the presence of multi-scale anchors without adding any extra cost related to varying image scales.

For each `n x n` region proposal, a fixed size feature vector of lower dimension is extracted depending on the base network by the RoI pooling layer as described later. Thus, the fixed-size feature vector length is 256 for the ZF model and 512 for VGG-16 net. It is then fed to the 2 sibling fully-connected layers: 1) *Box-classification layer:* It is a binary classifier which provides an

[2] https://en.wikipedia.org/wiki/Attention_(machine_learning)

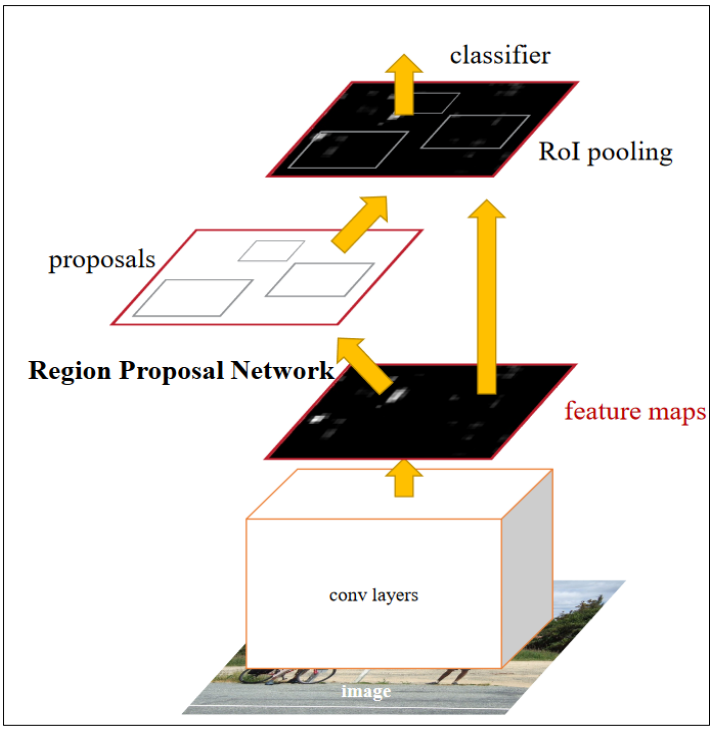



Figure 4.7: OVERVIEW OF FASTER R-CNN FRAMEWORK. The base convolutional network generates feature maps on which RPN operates using the *attention* mechanism. The generated proposals and the corresponding feature maps are then passed through RoI pooling to convert it into a fixed-length vector and for further classification. *Source* Ren et al. (2015).

objectness score for each region proposal. 2) *Box-regression layer:* This layer returns the 4D vector which defines the bounding box coordinates of the object of interest. These 4 coordinates are relative to the anchor box, and not in absolute image coordinates as defined in Equation (4.1) and Ren et al. (2015).

$$
\begin{aligned}
t_x &= \frac{(x - x_a)}{w_a}, t_x^* = \frac{x^* - x_a}{w_a} \\
t_y &= \frac{(y - y_a)}{h_a}, t_y^* = \frac{y^* - y_a}{h_a} \\
t_w &= log(\frac{w}{w_a}), t_w^* = log(\frac{w^*}{w_a}) \\
t_h &= log(\frac{h}{h_a}), t_h^* = log(\frac{h^*}{h_a})
\end{aligned}
\tag{4.1}
$$

here $x$, $y$, $w$, $h$ correspond to the $(x, y)$ coordinates of the center of the box and the height, $h$ and width, $w$ of the box. $x_a$, $x^*$ denote the coordinates of the anchor box and its corresponding ground truth bounding box respectively.

As described in Ren et al. (2015), based on Intersection over Union (IoU) with the ground truth bounding box, each anchor is assigned a positive or negative objectness score (see Equation (5.4)) for training the RPN. The anchor with the greatest IoU overlap with the ground truth box or an anchor with IoU greater than 0.7, gets a positive label. Both of these conditions must be

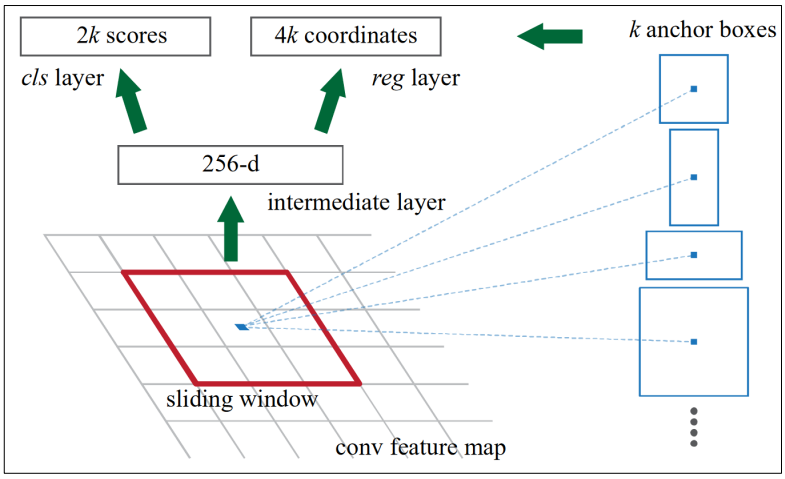



Figure 4.8: REGION PROPOSAL NETWORK (RPN). Multiple region proposals of different sizes are predicted at each sliding window location which is then mapped to a lower-dimensional feature (256-d for ZF model) and fed to two fully sibling connected layers: *reg* and *cls* for coordinates prediction and classification score respectively. *Source* Ren et al. (2015).

satisfied in order to ensure that positive anchors are present in all situations. For all ground-truth boxes, negative labels are assigned to a non-positive anchor with an IoU less than 0.3. Unlabeled anchors are not considered in the training process. The objective function to train the RPN is the multi-task loss for an image as described in Fast R-CNN (Girshick, 2015) and Faster R-CNN (Ren et al., 2015). The loss function is defined as:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*) \tag{4.2}$$

Where, index of an anchor is denoted by $i$ in a mini-batch. The output of the classification, *cls* layer is $p_i$ and that of the regression, *reg* layer is $t_i$. $p_i$ is the anchor's predicted probability of being an object. If the anchor $i$ is an object, $p_i^*$ is 1 otherwise 0. $t_i$and $t_i^*$ are the four parametrized coordinates and the ground truth coordinates respectively. The $L_{cls}$, classification loss, is a log loss (or cross-entropy loss) over two classes (i.e. if it is an object or not an object) as defined in Equation (4.3).

The equation $L_{reg}(t_i - t_i^*) = R(t_i - t_i^*)$ is the regression loss where $R$ is the smooth L1 loss function. Smooth L1 loss in Equation (4.4) can be interpreted as a combination of L1 loss and L2 loss. It acts as L1 loss when the absolute value of the term is high i.e. it produces steady gradients for high values, and it behaves like L2 loss i.e. low fluctuations for small values when the absolute value of the term is close to zero. Thus, it offers the advantages of both loss functions together. The term $p_i^* L_{reg}$ represents the regression loss is available only for positive anchors $(p_i^* = 1)$ and is inactive otherwise $(p_i^* = 0)$.

$$L_{cls}(p_i, p_i^*) = -p_i^* log(p_i) - (1 - p_i^*) log(1 - p_i) \tag{4.3}$$

$$L_{1;smooth} = \begin{cases} |x|, & \text{if} |x| \geq \alpha \\ \frac{1}{|\alpha|}x^2, & \text{otherwise} \end{cases} \tag{4.4}$$

where $\alpha$ is a hyperparameter and is generally set to 1.

The coefficient $\lambda = 10$ is a balancing constant and the normalizing constants, $N_{reg}$ and $N_{cls}$ are the number of mini-batch and the anchor boxes respectively. The default normalizing values for $N_{cls}$ is 256 and that for the $N_{reg}$ is ~2400. It is important to note that the authors mention that it is not necessary to follow this protocol for normalization and also performed experiments to demonstrate that outcomes are insensitive to values of $\lambda$ in a broad range.

### Region of Interest Pooling (RoI) Layer

This type of layer is first introduced in the Fast R-CNN network (Girshick, 2015). There is a need to convert all proposals from RPN to a fixed length vector before passing it to the fully connected layers. The RoI pooling layer performs this function.

RoI pooling is used for utilizing a single feature map for all the proposals generated by the RPN in a single pass. It generates equal-length feature maps from non-uniform inputs by doing max-pooling on the inputs. RoI pooling layer takes two inputs as illustrated in Figure 4.7: 1) Feature maps generated by the convolutional neural network in the backbone. 2) *k* proposals (or regions of interest) from region proposal network.

RoI pooling takes every region of interest from the input, and selects a part of the input feature map that corresponds to that RoI, and turns that feature map segment into a fixed dimension map. The fixed dimension output of the RoI pooling for each RoI is independent of the input feature map or the proposal sizes; it is purely determined by the layer parameters like spatial scale, pooling height and width. Pooling height and width decide how to split the region proposal into the grid of cells and it also indicates the output dimension of the layer. Spatial scale, as the name suggests, is a scaling parameter to resize the proposals as per the feature map dimensions. An interesting visualization of the forward pass in RoI pooling layer can be found at this website[3].

## 4.2.2 Deformable Convolutions

Convolutional Neural Networks (CNNs) are excellent for visual recognition tasks but they are inherently invariant to large and unknown transformations and relies on data augmentation techniques to learn geometric transformations. Few simple geometric transformations are scaling, rotation, translation *etc.* CNN's capability to represent geometric modifications is limited by the fixed structure of the kernel used to sample from the feature map. Using a fixed rectangular window, a CNN kernel samples from the input feature map at a specified position. To overcome this limitation and increase the capability of CNN, Microsoft researchers introduced Deformable Convolutional Networks (Dai et al., 2017).

The deformable convolutions present a simple, efficient, and end-to-end technique for modeling dense spatial variations, assisting CNN in learning numerous geometric transformations based on the given data. In the standard convolution process, 2D offsets are added to the regular grid sampling locations to account for the size of various objects and to create adaptable receptive fields depending on the size of the object (as seen in Figure 4.9). This deforms the previous activation unit's constant receptive field. The offsets introduced are learned from the prior feature maps using further convolutional layers. As a result, the deformation performed depends on the input data in a local, dense, and adaptable manner.

The deformable convolution has dynamic and learnable receptive field. This can be observed in Figure 4.10 from Dai et al. (2017) where the illustration is shown using two layers. On the topmost feature map, two activation units on two objects of different scales and shapes are shown. The activation is from a 3 X 3 filter. In the middle feature map, two activation units are highlighted again and the sampling location of the 3 X 3 filter is shown. The bottommost feature map represents the sampling locations on the preceding feature maps for a 3 X 3 filter. Due to the

[3] https://deepsense.ai/region-of-interest-pooling-explained/

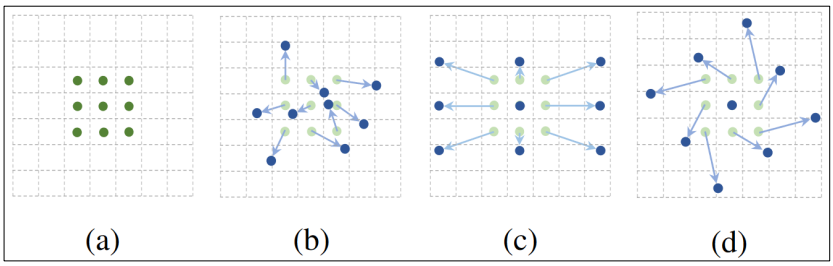



Figure 4.9: SAMPLING LOCATIONS IN 3 X 3 STANDARD AND DEFORMABLE CONVOLUTIONS. Regular grid sampling of standard convolution is shown in (a) with green dots and sampling with augmented 2D offsets of deformable convolutions is shown in (b),(c),(d) through blue arrows. *Source* Dai et al. (2017).

adaptability of the deformable convolutions, we use it in this work. We perform experiments to compare the performance of different convolutions for the task of table data extraction and the results of which are presented in chapter 6.

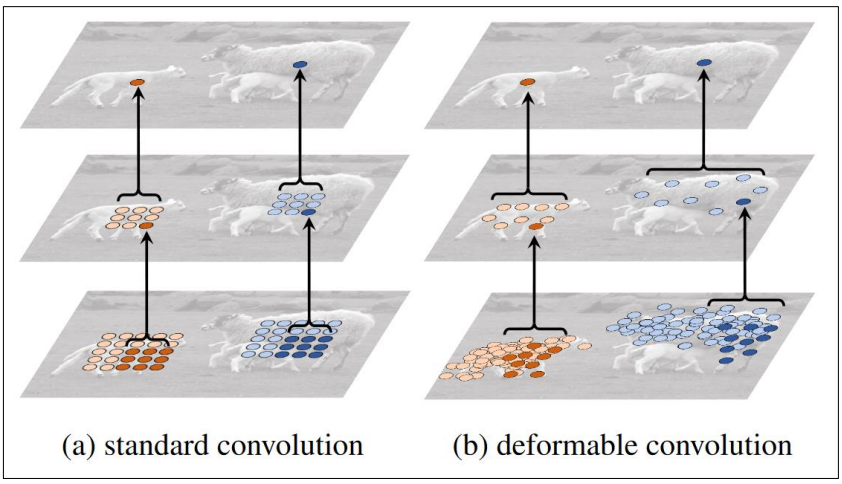



Figure 4.10: STANDARD AND DEFORMABLE CONVOLUTION. Left: Illustration of fixed receptive field in standard convolution. Right: Illustration of adaptive receptive field in deformable convolution operation using two layers. *Source* Dai et al. (2017).

### 4.2.3 Type of Regression Loss

It is always a discussion to select or modify the regression loss wisely for a perfect bounding box prediction. In this section, we describe the three different loss functions: Smooth L1 (Equation 4.4), Generalized Intersection over Union (GIoU) (Equation (4.5), Rezatofighi et al. (2019)), and Distance Intersection over Union loss (DIoU) (Equation (4.6), Zheng et al. (2020)) functions.

$$GIoULoss = \frac{|G \cap D|}{|G \cup D|} - \frac{|C \setminus (G \cup D)|}{|C|} = IoU - \frac{|C \setminus (G \cup D)|}{|C|} \tag{4.5}$$

where G and D are ground truth and predicted bounding box respectively. C is the smallest convex hull that encloses both G and D.

$$DIoULoss = 1 - IoU + \frac{||d - g||}{c^2} \tag{4.6}$$

where $c$ is the diagonal length of the smallest enclosing box covering the two boxes and $g$ and $d$ are the center points of the ground truth and predicted bounding boxes respectively.

Authors of GIoU Loss (Rezatofighi et al., 2019) claim that it is not a good idea to train a network by optimizing a loss function such as $l_1$-norm or $l_2$-norm and then evaluate the performance on IoU function since $l_n$-norm based losses are not scale invariant. Therefore, bounding boxes with the same level of overlap, but different scales will give different values. In a case, where ground truth and predicted bounding box does not overlap, the IoU will be 0 but GIoU will not necessarily be 0 and GIoU value will increase as the predicted bounding box moves closer to the ground truth bounding box indicating improvement in the performance while IoU will still remain 0. This is the advantage of GIoU loss i.e. it is always differentiable. The readers are encouraged to visualize the GIoU loss on the authors' website[4].

DIoU loss inherits some of the properties of IoU and GIoU and also alleviates the limitations of both of them. It is a scale invariant function and converges much faster than the GIoU loss especially in the non-overlapping case as it directly minimizes the normalized distance between ground truth and predicted bounding box.

The results of the experiments involving these different loss functions are reported in chapter 6.

## 4.3 Implementation Details

The model is trained and tested on various datasets with fixed image sizes. The dataset splits for both the tasks are described in Section 4.1. All the models are trained and tested using the MMDetection framework developed by OpenMMLab (Chen et al., 2019). It is an open-source object detection toolbox based on PyTorch. Here, we describe the implementation details for table detection and table structure recognition separately to avoid any confusion.

### 4.3.1 Table Detection

For the purpose of training, we use a single Nvidia RTX 3090 GPU with 24GB memory with a batch size of 1. The image is resized to a size of `1024 X 1536` while maintaining the aspect ratio as expected by the convolutional neural network in the backbone. We are able to do so by resizing the shorter side and adapting the longer side accordingly. The choice of this large size is discerned from the related works and smaller image size with increased batch size can also be tried. The backbone of the Faster R-CNN is a ResNeXt-101 which is 101 layers deep with a Feature Pyramid Network (FPN). We use a model originally pre-trained on ImageNet (Russakovsky et al., 2015) and later trained on COCO dataset (Lin et al., 2014) by OpenMMLab. ResNeXt architecture is a 4 stage architecture like ResNet (Appendix A.1) and since we are using the pre-trained model, we freeze the first two stages of the ResNeXt to retain the high level features learnt by the pre-trained model. This also assists in faster training. We use the deformable convolutions (Dai et al., 2017) in

[4] https://giou.stanford.edu/

the backbone network. The benefit of utilizing deformable convolutions is that the receptive field adapts to the size of the object, allowing the CNN to model multiple spatial transformations.

The number of anchors generated is $k$ = 3 at each feature point by using the anchor scale of size 8 and aspect ratios of 0.5, 1 and 2. This lower number of anchors is suitable for the task of table detection and is able to detect tables of any sizes, even very large or very small tables. In the RPN, anchor with the greatest IoU overlap with the ground truth box, or an anchor with IoU > 0.7, gets a positive label. Both of these conditions must be satisfied in order to ensure that positive anchors are present in all situations. For all ground-truth boxes, negative labels are assigned to a non-positive anchor with an IoU less than 0.3.

The ratio of positive to negative samples is usually much smaller than 1. This often has the consequence that in the case of unbalanced datasets, *for example*, where the number of negative samples is much larger than the number of positive samples, the whole training process is often dominated by negative samples, and the loss function is also influenced by negative samples. To counter this issue, a `RandomSampler` strategy is followed to deal with the unbalanced number of positive and negative samples. Many of the RPN proposals overlap with each other and to reduce this redundancy, non-maximum suppression (NMS) is used on the proposal regions. Thus, the network is trained with 2000 region proposals but at test time only 1000 region proposals are evaluated.

After resizing the images while maintaining the aspect ratio, the images are randomly flipped, normalized and padded as part of image augmentation in the training pipeline. During testing, the images are not flipped. We use the stochastic gradient descent (SGD) as an optimizer with learning rate = 0.00125, momentum = 0.9 and weight decay = 0.0001. A learning rate scheduler and momentum policy are also defined. The network is trained for 30 epochs however, it is not necessary to train it for 30 epochs to get the best accuracy. A total of 3500 samples from mixed datasets are used to train the model as described in Section 4.1.3.

### 4.3.2 Table Structure Recognition

Most of the implementation details remain the same as in the table detection and only the changes are discussed here. For this task, we use the ResNet-50 backbone model solely because of its speed and due to a larger number of training samples compared to table detection task. The backbone is initialized with a pre-trained model first trained on ImageNet and then trained and fine-tuned on COCO dataset. The first stage (out of three) of the ResNet-50 is frozen while training. The convolutions in the backbone are deformable convolutions.

The number of anchors generated in this task is $k$ = 18 for each feature point. The model generates anchors in six different ratios $[1/20, 1/10, 1/5, 1/2, 1, 2]$ which captures the different sizes and scales of bounding boxes. The anchor scales are defined as $[4, 8, 16]$. These different sizes of anchors are necessary to capture shapes of predicted rows, columns, spanning cells, projected row header and column header as described in Section 6.5.3. In the RPN, the IoU threshold is set as 0.5 (IoU > 0.5) to define the anchor as a positive sample and the threshold to define a negative sample is set as 0.3 (IoU < 0.3). The network uses 2000 proposals during training and 1000 proposal while testing. The R-CNN non-maximal suppression (NMS) IoU threshold is 0.2 from the default value of 0.5 in the testing phase. It is the last stage NMS threshold for the R-CNN detection network before the predictions. The threshold is a key value for the performance of the model. A higher value results in many rows or cells not being detected which are very tightly spaced or closely located in a confined region.

The images are resized to `1100 X 800` while maintaining the aspect ratio as explained above. The images are randomly flipped, randomly cropped, normalized and padded before passing to the model for training. The model is trained with SGD optimizer with learning rate = 0.01, momentum = 0.9 and weight decay = 0.0001. It is trained for 25 epochs with a batch size of 8. The

initial learning rate of 0.01 is divided by 10 every 5 epochs. However, the model converges earlier than 25 epochs. 156K samples from PubTables-1M are used for training, 18K for validation and testing. During testing, samples from ICDAR 2013 are also tested to check the robustness of the model on an unseen dataset.

# Chapter 5

# Experiments and Results

In this chapter, we will go through the numerous experiments performed for table detection and structure recognition. First, we define the different metrics used for training and evaluating the experiments.

## 5.1 Evaluation Metrics

Algorithms for table data extraction are evaluated through different measures. We have evaluated our approach by precision, recall, F1 scores over different Intersection over Union. The model is also evaluated using the wide-spread mean Average Precision (mAP) (Everingham et al., 2010) in MS COCO style (Lin et al., 2014) which is an apt metric for object detection. An overview of all metrics can be seen in Figure 5.1.

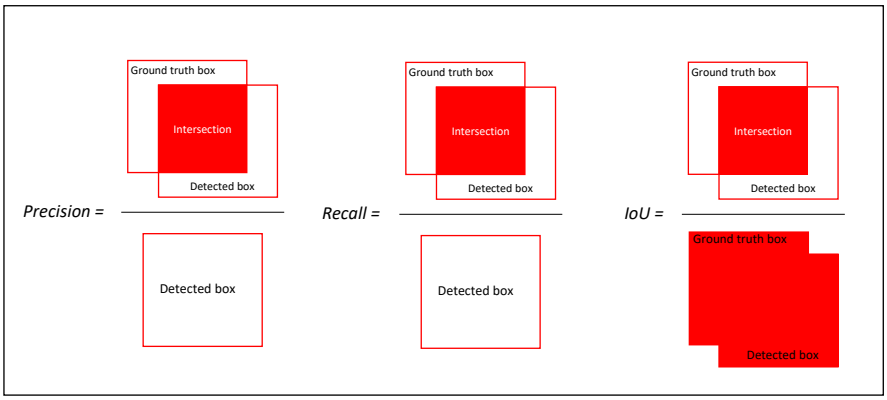



Figure 5.1: OVERVIEW OF OBJECT DETECTION METRICS. The more intuitive way to understand precision, recall and intersection-over-union (IoU) is shownn in the figure for table detection.

### Precision

Precision measures how accurate are the predictions i.e. the percentage of the predictions that are correct. For example, it computes the proportion of identified tables that belong to table regions

of the ground truth table in the case of table detection. This is done by defining an IoU threshold (explained later) for any bounding box to be counted as correctly detected. Secondly, a confidence threshold is defined to decide whether the detection counts as correctly classified. Equation (5.1) states the most commonly known definition of precision and the definition of precision used in this work taken from Shahzad et al. (2019). True positive (TP) refers to a detected bounding box which matches a ground truth box. It is determined based on the IoU of the two boxes. False positive (FP) refers to a detected box which does not match any ground truth boxes. False negative (FN) refers to a ground truth bounding box which is not matched by any detected boxes. It can also be visualized in Figure 5.1 for better understanding of the terms in numerator and denominator. For example, *Area of ground truth regions in detected regions* means the intersection area of the ground truth and the predicted box.

$$\begin{aligned} Precision &= \frac{True\ Positive}{True\ Positive + False\ Positive} \\ Precision &= \frac{Area\ of\ ground\ truth\ regions\ in\ detected\ regions}{Area\ of\ all\ detected\ table\ regions} \end{aligned} \tag{5.1}$$

## Recall

Recall measures how accurate are all the positives. It quantifies the number of correct positive predictions made out of all positive predictions. For example, in table detection it is calculated by identifying the correct table areas in the detected table areas. IoU threshold and confidence threshold are defined to calculate the correctly detected and classified bounding boxes as in Precision. Recall is defined in Equation (5.2) in its most commonly used form and the more intuitive form (Shahzad et al., 2019). It can also be visually understood from Figure 5.1.

$$\begin{aligned} Recall &= \frac{True\ Positive}{True\ Positive + False\ Negative} \\ Recall &= \frac{Area\ of\ ground\ truth\ regions\ in\ detected\ regions}{Area\ of\ all\ ground\ truth\ table\ regions} \end{aligned} \tag{5.2}$$

## F1 Score

F1 score represents both Precision and Recall, making it an excellent aggregated indication; yet, it obscures the source of the problem. It is usually reported at the confidence threshold that maximizes F1 on a particular test set. This might lead to poor quantitative results on an unseen dataset due to the same confidence threshold. Our adapted IoU metric, as explained later, better aligns the qualitative findings and the quantitative outcome of table detection and reduces this dependency of the F1 scores on the chosen IoU threshold. F1 score is the harmonic mean of precision and recall defined as in Equation (5.3).

$$F1score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5.3}$$

These metrics complement each other effectively when assessing a model at different confidence levels, offering useful insight into how the model is doing and what values improve model performance based on the design parameters. Typically, as the confidence threshold is increased, the precision rises but the recall falls.

### Intersection over Union (IoU)

Intersection over Union (IoU) measures the overlap between the predicted area and the ground truth area. It is basically the Jaccard index. In the most simplest form, IoU is formulated as seen in Equation (5.4):

$$IoU = \frac{G \cap D}{G \cup D} = \frac{G \cap D}{|G| + |D| - |G \cap D|} \tag{5.4}$$

where G is the ground truth area and D is the area of the detected bounding box.

In our case, we adapt and modify the IoU metric inspired from Günther et al. (2017). The reason to do so, as explained in detail in Section 4.1.2, is that we predict larger bounding box for the table detection than the ground truth because of the stretching of bounding boxes in ground truth during data pre-processing. The predicted bounding box area is approximately 15% bigger than the ground truth bounding box area as visualized in Figure 5.2. Hence, we modify the union term in Equation (5.4) to not penalize the 15% bigger detections. The new IoU metric (also referred as aIoU) used for evaluation is defined in Equation (5.5). Please note that we did not test the metric with a constant other than 1.15, and there might be a better constant.

$$aIoU = \frac{|G \cap D|}{(|G| + max\{\frac{|D|}{1.15}, |G \cap D|\} - |G \cap D|)} \tag{5.5}$$

Therefore, when the detected bounding box covers at most 15% more of the ground truth bounding box area and the ground truth bounding box is entirely contained inside the detected bounding box, then the IoU value is 1. For evaluation, we use IoU at different thresholds like 0.6, 0.7, 0.8 and 0.9 indicated by IoU@0.6, IoU@0.7 and so on. The further analysis on aIoU is shown later in the Section 5.2 in Figure 5.5 and 5.6.

### Mean Average Precision (mAP)

The general definition of Average Precision (AP) is the area under the precision-recall curve for one class (Equation (5.6)). A perfect model will have the AP score = 1. Precision and recall are always in the interval [0,1]. Therefore, AP is also within 0 and 1.

$$Average\ Precision\ (AP) = \int_0^1 p(x)\, dx \tag{5.6}$$

where *dx* is the difference between the current and next recall and multiplied by the current precision *p(x)*.

AP[1] can also be understood as weighted sum of precisions at each threshold, with the weight corresponding to the increase in recall. After defining AP, it is simple to measure mean average precision (mAP). The mAP is the mean of the AP over all classes. In the case of table detection, we have only one class and therefore, mAP is essentially AP. Sometimes mAP is reported as mAP@0.5 or mAP@0.9. This notation indicates the IoU threshold used to calculate the confusion matrix.

For the COCO dataset (Lin et al., 2014), mAP is calculated by averaging it over IoUs from [0.5,0.95] with a step size of 0.05. As stated by COCO:

> "AP is averaged over all categories. Traditionally, this is called "mean average precision" (mAP). We make no distinction between AP and mAP (and likewise AR and mAR)[2] and assume the difference is clear from context."

[1] https://blog.paperspace.com/mean-average-precision/
[2] https://cocodataset.org/#detection-eval

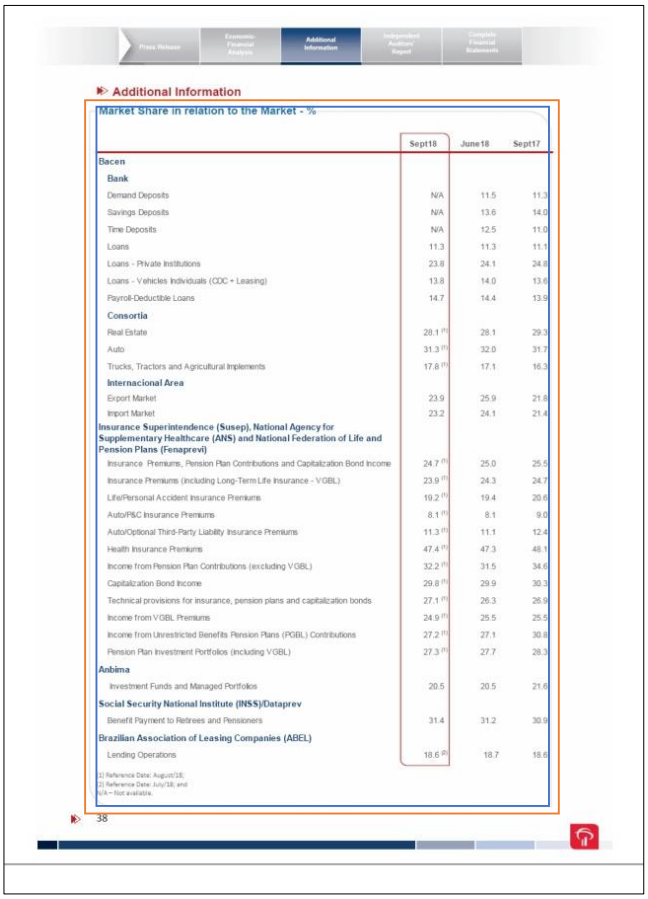



Figure 5.2: GROUND TRUTH BOX AND THE PREDICTED BOX. The image from ICDAR 2019 test set represents the ground truth bounding box in blue and the predicted bounding box in orange which is approximately 15% bigger in area than the ground truth box. Please note that the predicted bounding box contains the table much better than the ground truth which is too tight many times.

There are several factors to consider when providing unified mAP benchmarks that would fit any object detection scenario, such as the number of classes, trade-off between precision and recall, IoU threshold, and so on. Depending on the task, the same metric might be excellent or poor. Hence, it is required to determine the desired value of threshold for the task at hand. The current state-of-the-art mAP for the object detection task on the COCO dataset leaderboard is 0.588 (by Noah CV Lab (Huawei)) followed by MMDetection (Chen et al., 2019) with mAP = 0.578. The developers of Faster R-CNN (Ren et al., 2015) report mAP = 21.9 on COCO standard test dataset. This additionally promotes the reason for choosing MMDetection for the model development which provides customizable framework.

## 5.2 Experimental Setups and Results

Several experiments are performed to check for the generalization and robustness of the model. It is important to note that it is difficult to compare the results quantitatively with other works because of the metric modification and different combinations of the subset of the datasets as mentioned in chapter 3 used for training by different models. However, visual differences are drawn to prove the superiority of the results obtained in this thesis.

### 5.2.1 Table Detection

The results of the best performing model are shown in Table 5.1 with F1 scores calculated at different IoUs using the aIoU metric defined in this work. The model is trained for 30 epochs on the image samples from ICDAR 2013, ICDAR 2019, TableBank, ISRI-OCR and Marmot. The model is tested on the complete standard 240 test-set images from ICDAR 2019 dataset because ICDAR 2019 contains the most visually different and complex tables in terms of colors, the number of tables in a page, different styles *etc.* and a mixture of randomly selected 1000 images from all of the above mentioned datasets. Table 5.1 highlights that with increasing IoU threshold the F1 scores decrease but stay consistent i.e. does not decrease rapidly. This happens due to the fact 90% or above overlapping is always difficult to achieve. The F1 score (at IoU = 0.9) of 0.945 on the ICDAR 2019 test set reveals the advantage of the approach on high precision table localization. The training mAP score achieved is 0.68. Figure 5.3 and Figure 5.4 displays the prediction result obtained by this model on all the datasets. These predictions cover the tables perfectly. It is compelling to note the predictions of the model on the ISRI-OCR dataset which are not the highest resolution and best scanned images. The model is able to detect multiple tables on a single page, and is also able to differentiate between the tables, figures and graphs. It can also predict correctly on the tables with complex styles and varying color backgrounds.

| **F1 scores** | **ICDAR 2019** | **Mix** |
|---|---|---|
| *IoU@0.6* | 0.969 | 0.970 |
| *IoU@0.7* | 0.960 | 0.960 |
| *IoU@0.8* | 0.954 | 0.950 |
| *IoU@0.9* | 0.945 | 0.918 |

Table 5.1: TABLE DETECTION RESULTS OF THE BEST PERFORMING MODEL. F1 scores are calculated for different IoU thresholds using the aIoU metric.

| **Work** | **IoU** | | | |
|---|---|---|---|---|
| | 0.6 | 0.7 | 0.8 | 0.9 |
| *TableRadar* | 0.969 | 0.957 | 0.951 | 0.897 |
| *NLPR-PAL* | 0.979 | 0.966 | 0.939 | 0.850 |
| *CascadeTabNet* | 0.943 | 0.966 | 0.939 | 0.850 |
| *This work (Iou)* | **0.962** | **0.955** | **0.919** | **0.750** |
| *This work (aIoU)* | **0.969** | **0.960** | **0.954** | **0.945** |

Table 5.2: COMPARISON WITH THE STATE-OF-THE-ART MODELS ON ICDAR 2019 TEST SET. F1 scores are calculated for different IoU thresholds with standard IoU and the aIoU described in this work.

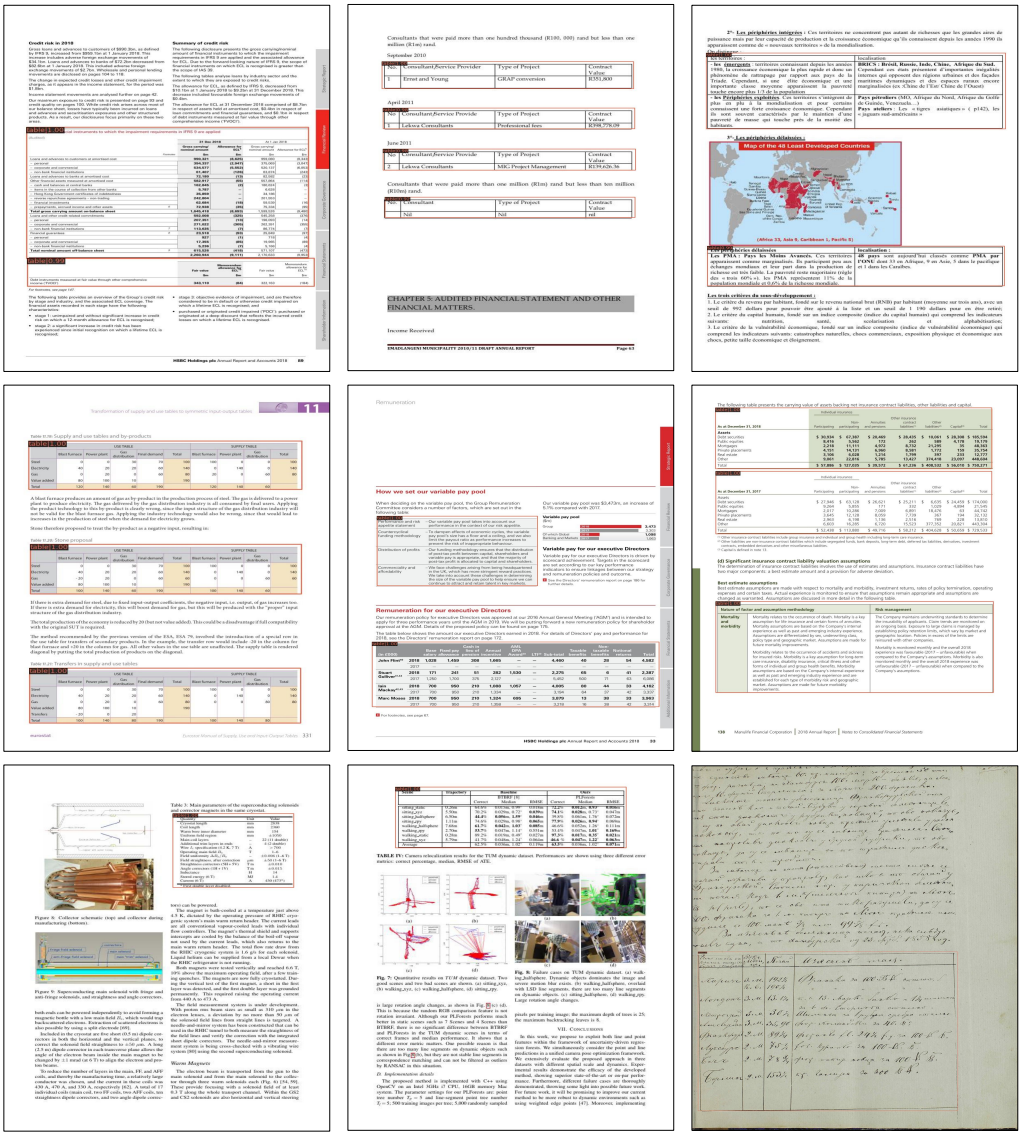


Figure 5.3: TABLE DETECTION RESULTS ON ICDAR 2019 AND TABLEBANK DATASET. The results shown are obtained by the model represented in Table 5.1. The orange box represents the detected table. A hand written table is also identified correctly as can be seen in bottom-rightmost image.

Table 5.2 presents the comparison of our results with other state-of-the-art methods on the ICDAR 2019 test dataset. The results are compared to the leaders of the ICDAR 2019 table competition TableRadar, NLPR-PAL (Gao et al., 2019) and CascadeTabNet (Prasad et al., 2020). While the first two works use post-processing techniques as described in the chapter 2 to further refine the results, CascadeTabNet and our work directly output the table regions in the images without relying on any post-processing methods. CascadeTabNet is a cascade mask region-based CNN High-Resolution Network (Cascade mask R-CNN HRNet) that identifies the table in a document. Unlike the work performed in this thesis, the researchers of CascadeTabNet uses several image

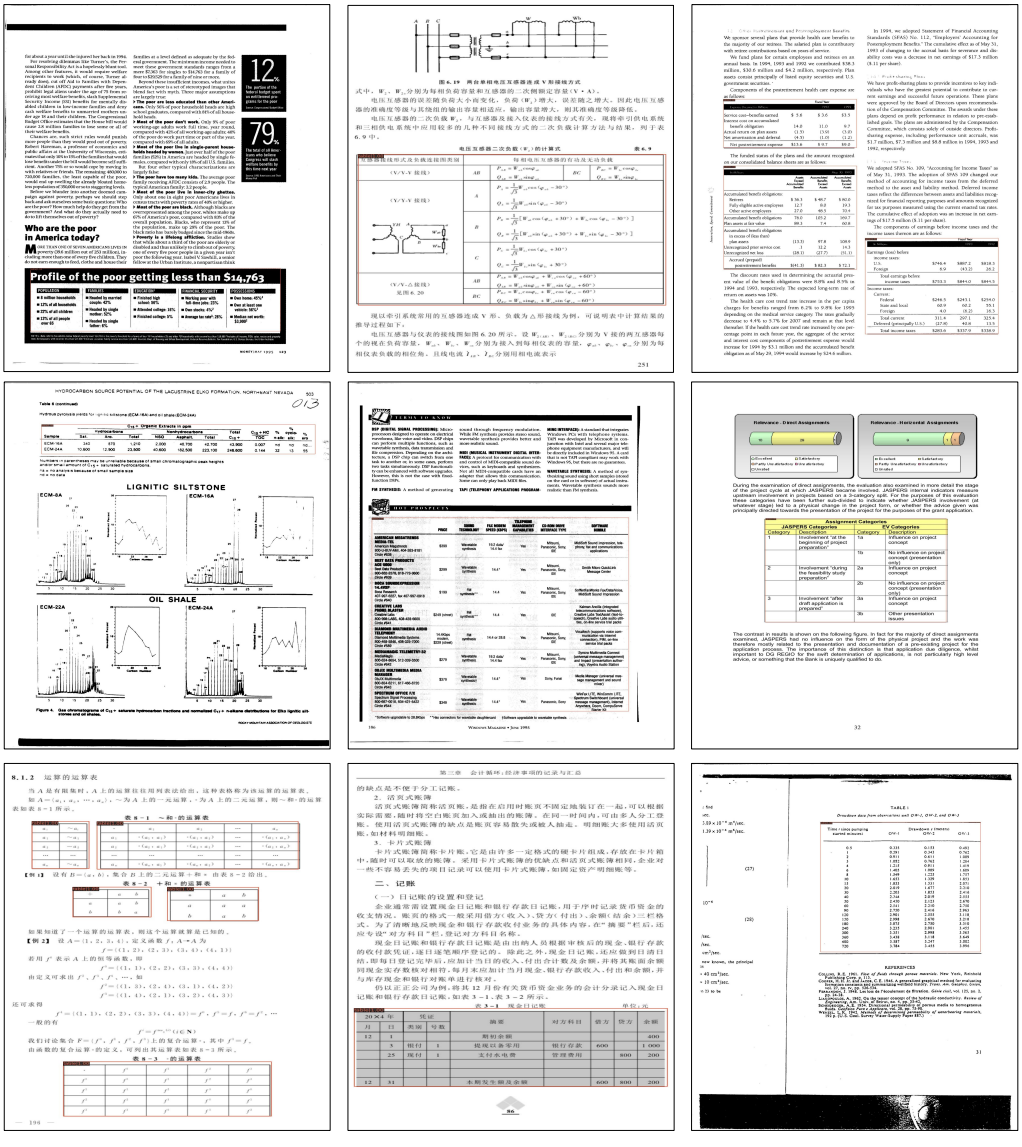


Figure 5.4: TABLE DETECTION RESULTS ON ICDAR 2013, ISRI-OCR AND MARMOT DATASET. The results shown are obtained by the model represented in Table 5.1. The orange box represents the detected table.

augmentation techniques like dilation and smudging of images before passing them to the network and manually correct the annotations in the Marmot dataset. The direct comparison of the results in this thesis is greater than these methods. It is to be noted here that we use aIoU as the metric for the results of our model and report the results of other researchers as mentioned in their respective papers which is calculated using standard IoU. It is also important to know that aIoU and standard IoU generates almost similar results for lower IoU thresholds like 0.6 or 0.7, as shown in the Table 5.2 and Figure 5.5 and 5.6, because of how aIoU is defined to only penalize bigger predictions than a certain threshold. Hence, it is viable to compare results in Table 5.2 for, at least, IoU = 0.6 and 0.7 with other works. Hereon, we report the results using the aIoU metric only.


Figure 5.5: IOUS VISUALIZATION AT THRESHOLD = 0.6. (a) The blue box represents the ground truth and various predictions in dashed lines. For an IoU threshold of 0.6, only the dark green predictions will count in case of standard IoU metric, while all green and light green predictions will be included in the case of aIoU. (b) The figure represents the sliding (off-set) predictions over the ground truth. While aIoU allows more 'bigger' predictions that contain the ground truth entirely, it does penalize predictions that are offset compared to the ground truth; however, not as much as the standard IoU.

We also compare our results on ICDAR 2013 with the best performing models like TableNet (Paliwal et al., 2019) and DeepDeSRT (Schreiber et al., 2017) in Table 5.3. We calculate Precision, Recall and F1 scores for IoU = 0.5 based on completeness and purity as stated in Paliwal et al. (2019):

> "A region is complete if it includes all sub-objects present in the ground-truth. A region is pure if it does not include any sub-objects which are not in the ground-truth. Sub-objects are created by dividing the given region into meaningful parts like heading of a table, body of a table etc. But these measures do not discriminate between minor and major errors. So, individual characters in each region are treated as sub-objects. Precision and recall measures are calculated on these sub-objects in each region and the average is taken across all the regions in a given document."

Moreover, it is important to note that we use 124 images from ICDAR 2013 for testing and 114 images for training while DeepDeSRT and TableNet use 34 images for testing and the rest of the

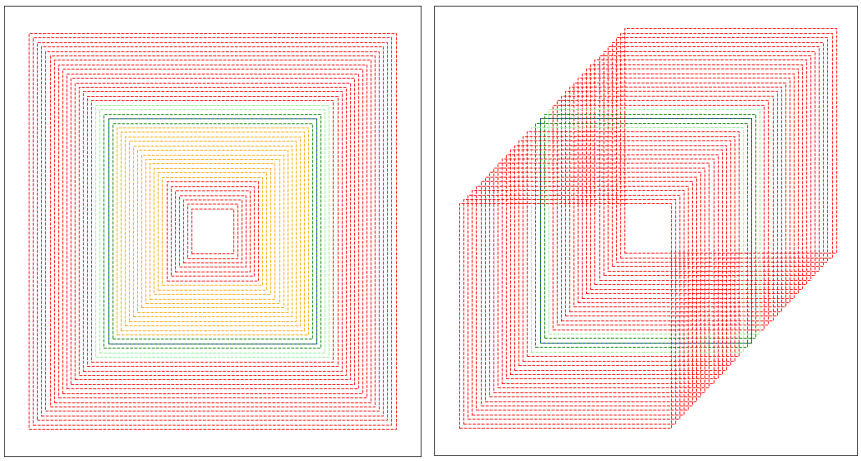



Figure 5.6: IoUs VISUALIZATION AT THRESHOLD = 0.9. (a) The blue box represents the ground truth and various predictions in dashed lines. For an IoU threshold of 0.9, only the dark green predictions will count in case of standard IoU metric, while all green and light green predictions will be included in the case of aIoU. (b) The figure represents the sliding (off-set) predictions over the ground truth. While aIoU allows more 'bigger' predictions that contain the ground truth entirely, it does penalize predictions that are offset compared to the ground truth; however, not as much as the standard IoU.

images for training the model. All the models mentioned in Table 5.3 use different combinations of datasets for training for which results are less comparable. Our model achieves Precision, Recall, and F1 score equal to 1 at IoU = 0.5. This further proves that our model is more robust and scales well.

| Work | Precision | Recall | F1 |
|---|---|---|---|
| *TableNet* | 0.969 | 0.962 | 0.966 |
| *DeepDeSRT* | 0.961 | 0.974 | 0.967 |
| *This work* | **1.0** | **1.0** | **1.0** |

Table 5.3: COMPARISON WITH THE STATE-OF-THE-ART MODELS ON ICDAR 2013 DATASET. Scores are calculated on completeness and purity as in (Paliwal et al., 2019) for IoU = 0.5.

The results are also benchmarked on the TableBank dataset (Li et al., 2020). While the authors of TableBank use 415,234 samples for training, we only use 1000 samples for training the model. The results are evaluated on 1000 test samples picked randomly from the dataset as can be seen in Table 5.4. CascadeTabNet uses 3000 samples for training the model. To compare the results, we calculate the Precision, Recall, F1 as described in TableBank and in Equation (5.1), (5.2), and (5.3) for IoU = 0.5. Researchers in TableBank also use the Faster R-CNN model with ResNeXt-101 and ResNeXt-152 backbone. The good quality of results is attributed to the careful design choices like deformable convolutions, anchor sizes and other parameters for this task as described before compared to the default parameters used in other works. We are able to achieve competitive results despite training on very few samples from the TableBank dataset.

| Model | Precision | Recall | F1 |
|---|---|---|---|
| *ResNeXt-101* | 0.959 | 0.904 | 0.931 |
| *ResNeXt-152* | 0.967 | 0.8895 | 0.926 |
| *CascadeTabNet* | 0.929 | 0.957 | 0.943 |
| *This work* | **0.966** | **0.952** | **0.958** |

Table 5.4: COMPARISON WITH THE STATE-OF-THE-ART MODELS ON TABLEBANK DATASET. Scores are calculated for IoU = 0.5.

## Cross-testing Experiments

To prove the robustness of the table detection model described in this work, we also perform cross-testing experiments on different datasets by training different models i.e. the model is trained by leaving out one dataset and tested on the unseen (left out) dataset with complete out-of-distribution samples. While other models do not perform at all (gives no prediction) or perform on a few samples (predicts few tables out of many) from the unseen dataset, the model in this thesis produces high quality results in quantitative and qualitative terms.

In the first experiment, we test our model on the 125 unseen samples from the TableBank dataset. This means that the model is trained with TableBank dataset left out. The results are described in the Table 5.5 and few of the predictions can be seen in Figure 5.7. The cross-testing results for ICDAR 2019 on 240 images (Figure 5.8), Marmot (1900 images) (Figure 5.9), ISRI-OCR (427 images) (Figure 5.10) are also presented in Table 5.5. The training mAP scores for all the 4 experiments are in the range 65-70%. As can be seen from the table, the F1 scores for IoU = 0.6 are greater than 0.9 in each case.

| | Dataset | | | |
|---|---|---|---|---|
| **F1 scores** | **TableBank** | **ICDAR 2019** | **Marmot** | **ISRI-OCR** |
| *IoU@0.6* | 0.949 | 0.927 | 0.966 | 0.92 |
| *IoU@0.7* | 0.94 | 0.92 | 0.956 | 0.913 |
| *IoU@0.8* | 0.928 | 0.913 | 0.934 | 0.902 |
| *IoU@0.9* | 0.918 | 0.901 | 0.91 | 0.89 |

Table 5.5: CROSS TESTING RESULTS. F1 scores are calculated for different IoU thresholds and scores are shown for the four different datasets.

It is interesting to note that the results achieved in the Table 5.5 for Marmot dataset at IoU = 0.6 is higher than the results achieved by researchers of DeCNT (Siddiqui et al., 2018) which is a Faster R-CNN model for IoU = 0.5 in Table 5.6. They report the results for training on a combination of datasets (like in this work) and testing on the unseen Marmot dataset. We also compare the results to CDeC-Net (Agarwal et al., 2021) which is Mask R-CNN based model where they train and test the model on the Marmot dataset (unlike in our case). Compared to both the works, where in one work the dataset has been seen by the model and in other work, not; it can be concluded that our model generalizes well to the Marmot dataset and always gives better results. We achieved a significantly higher F1 score of 0.97 than that of DeCNT model.

To compare the results on unseen ISRI-OCR (UNLV) dataset, we choose the state-of-the-art models: GOD (Saha et al., 2019), CDeC-Net (Agarwal et al., 2021), and Tesseract (Smith, 2007). None of the models see any samples from the ISRI-OCR dataset during training. They are trained on different subsets of datasets mentioned in chapter 3. The models are tested on 427 images

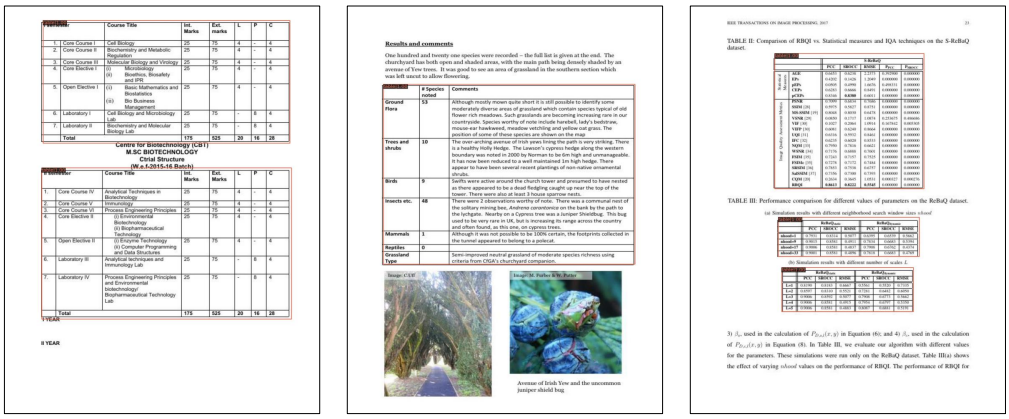


Figure 5.7: TABLE DETECTION RESULTS ON UNSEEN TABLEBANK DATASET. The orange box depicts the predicted tables correctly during cross-testing.

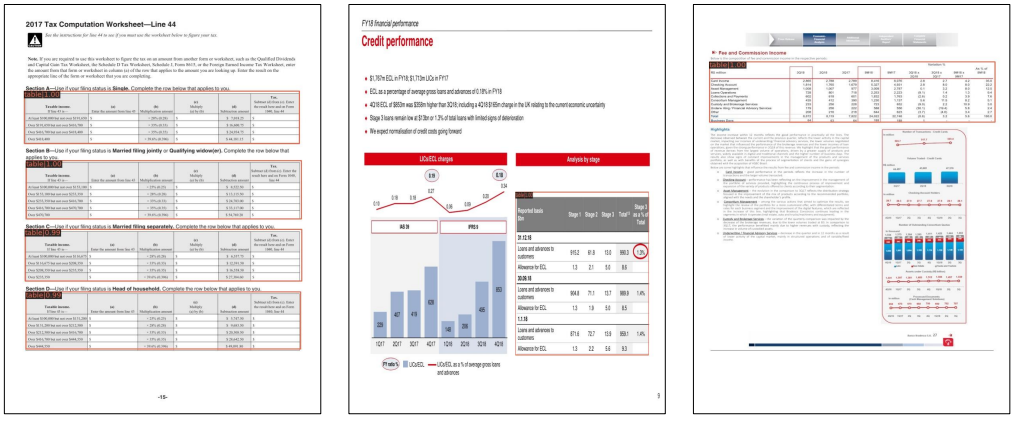


Figure 5.8: TABLE DETECTION RESULTS ON UNSEEN ICDAR 2019 DATASET. The orange box depicts the predicted tables correctly during cross-testing.

| Work | F1 |
|---|---|
| *DeCNT* | 0.895 |
| *CDeC-Net* | 0.952 |
| *This work* | 0.97 |

Table 5.6: COMPARISON ON MARMOT DATASET. F1 scores are calculated for IoU = 0.5.

containing tables. To make our results more comparable to that of GOD, we train the model only on Marmot dataset like in GOD and report the results. The results are calculated for IoU = 0.5 and are described in Table 5.7. GOD (Saha et al., 2019) is a graphical object detection model similar to the work presented in this thesis but it uses Mask R-CNN (He et al., 2017) for table detection. In this experiment, CDeC-Net performs marginally better by an absolute margin of 0.005 on the F1

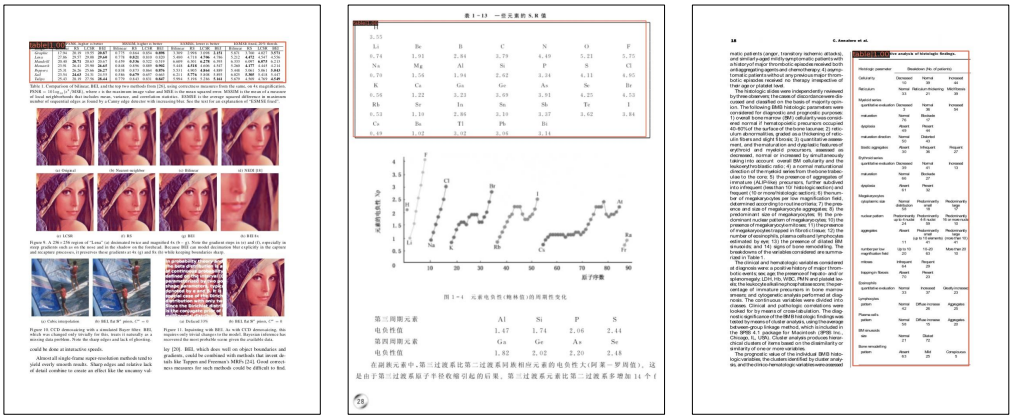


Figure 5.9: TABLE DETECTION RESULTS ON UNSEEN MARMOT DATASET. The orange box depicts the predicted tables correctly during cross-testing.

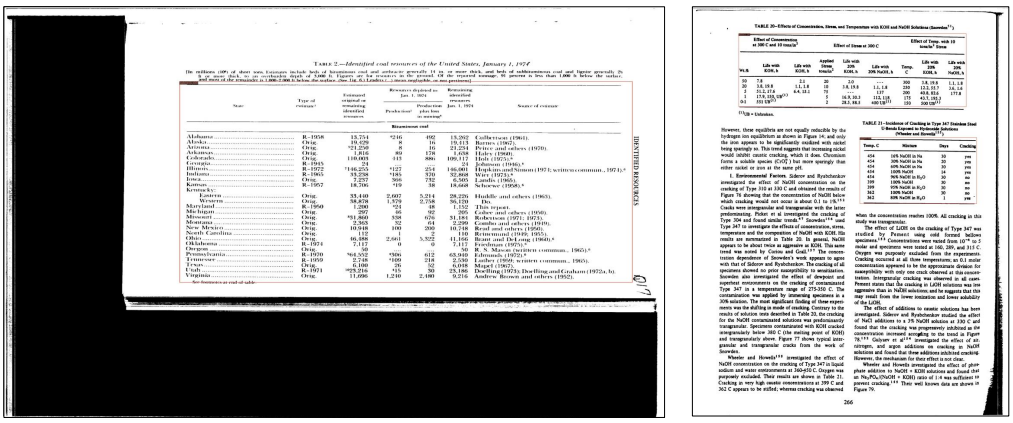


Figure 5.10: TABLE DETECTION RESULTS ON UNSEEN ISRI-OCR DATASET. The orange box depicts the predicted tables correctly during cross-testing.

score. CDeC-Net is trained on the complete IIIT-AR-13K dataset (Mondal et al., 2020) in this case.

| Work | F1 |
|---|---|
| *Tesseract* | 0.761 |
| *GOD (Mask R-CNN)* | 0.928 |
| *CDeC-Net* | 0.938 |
| *This work* | 0.933 |

Table 5.7: COMPARISON ON ISRI-OCR DATASET. F1 scores are calculated for IoU = 0.5.

Another example to prove the robustness of the model is shown in Figure 5.11 where the model correctly predicts two extra tables despite the incorrect ground truth annotation. From the Figures 5.7, 5.8, 5.9, 5.10 where different kinds of documents, articles containing tables are present,

we can say that our model generalizes very well on the unseen dataset and correctly identifies the tables in the images. This is attributed to the carefully defined parameters of the Faster R-CNN network (see chapter 6), which the competing works fail to address despite using similar object detection models like Faster R-CNN (TableRadar, DeCNT), Mask R-CNN (GOD, CDeC-Net) *etc.* in a cascaded (CascadeTabNet) or non-cascaded fashion.

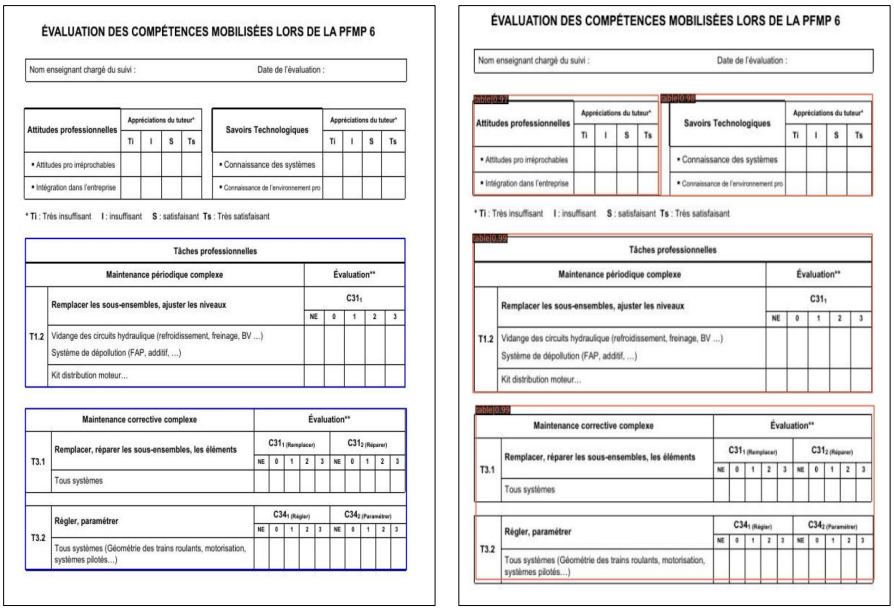



Figure 5.11: TABLE DETECTION ON UNSEEN DATASET. The left image with blue box shows the incorrect ground truth annotation while the image on the right with orange box shows the correctly detected tables.

## 5.2.2 Table Structure Recognition

Unlike the case in Table Detection, there is not much prominent research work done in Table Structure Recognition. Few of the different approaches like CascadeTabNet(Prasad et al., 2020) or LGPMA (Qiao et al., 2021) predicts the bounding box around the text rather than rows and columns. Therefore, they rely on post-processing techniques to assign text bounding boxes/cells to rows, columns. Hence, this work is not directly comparable to them. Few of the other works like RobusTabNet (Ma et al., 2022) and DeepDeSRT (Schreiber et al., 2017) use different datasets to predict only the rows and columns and one of the most basic problems here is when the table has multiple headers and some of it is between the rows.

A dataset, PubTables-1M (Smock et al., 2021) was recently released with high quality annotations of *table, row, column, projected row header, column header, spanning cell* (Section 3). We use this dataset for training and testing purpose and with the results in this thesis, we will also like to promote the usage of this dataset for this task to unify the research direction by having a common large benchmark dataset.

The results for this task are compared to the benchmark models presented in PubTables-1M

one of which is based on Faster R-CNN and the other model is an attention-based model called DETR (Detection Transformer) introduced in Carion et al. (2020). Both of their models use a ResNet-18 backbone pre-trained on ImageNet with the first few layers frozen.

The original DETR model is trained for 500 epochs with a batch size of 1. It achieves the COCO style mean Average Precision (mAP) score of 0.42 on the COCO 2017 validation set. As stated by the authors:

> "The main ingredients of the new framework, called DEtection TRansformer or DETR, are a set-based global loss that forces unique predictions via bipartite matching, and a transformer encoder-decoder architecture."

The authors in PubTables-1M use 758,849 tables for training, 94,959 for validating and 93,834 for testing the Faster R-CNN and DETR model whereas we only use 150,000 randomly selected samples out of original 758,849 samples for training, 18,000 (out of 94,959) for validating, and the exactly same 93,834 images for testing the model. The COCO style mAP (or AP) scores are presented in Table 5.8 to compare with the existing models. mAP is calculated by averaging it over IoUs from [0.5,0.95] with a step size of 0.05. It is to be noted that we neither increase the bounding box area in the training set nor use the new IoU metric for evaluation described in this work in Equation (5.5) rather we use the original IoU metric defined in Equation 5.4.

| Work | AP | $AP_{50}$ | $AP_{75}$ | AR |
|---|---|---|---|---|
| PubTables-1M (Faster R-CNN) | 0.722 | 0.815 | 0.785 | 0.762 |
| PubTables-1M (DETR) | 0.912 | 0.971 | 0.948 | 0.942 |
| This work | **0.933** | **0.97** | **0.963** | **0.953** |

Table 5.8: COMPARISON ON PUBTABLES-1M DATASET FOR TABLE STRUCTURE RECOGNITION. COCO style AP scores are reported for comparison which is calculated by averaging it over IoUs from [0.5,0.95] with a step size of 0.05.

As can be inferred from Table 5.8, our model outperforms the DETR model establishing once again the dominance of the Faster R-CNN model. Though the authors in PubTables-1M want to have a data driven approach and because of which they mostly use default parameters for Faster R-CNN and DETR, it is not a good approach to train the model for longer hours without giving any attention to the design specifications and parameters of the model which can drastically reduce the need of much larger training set; one of the core problems in Table Structure Recognition task. It is important to pay close attention while designing the parameters (see chapter 6) for the model to perform robustly across domains. The models presented in PubTables-1M research paper, Faster R-CNN and DETR, fails to produce meaningful predictions on unseen domains as we will see in the next section.

While it is not pragmatic to have the training and testing set ratio to be 8:5, as in our case, deviating from the usual 8:1 ratio, it is only done to draw the comparison to the DETR model. Due to the availability of limited resources, we only train on a single GPU with a learning rate of 0.01 compared to that of DETR which is trained with a much smaller learning rate of $5 * 10^{-5}$. Also, training on the complete 758,000 images will alone take more than 20 days as in the case of DETR model. The class-wise AP scores are presented in Table 5.9. The score for each class except *spanning cell* is greater than 0.9. This is attributed to the fact that there are very few tables with *spanning cell* annotations in the 150,000 training set compared to annotations for other classes. This score is observed to increase with more annotations of *spanning cell* in the tables during training. Confusion matrix for all the classes is shown in Figure 5.12. It can be seen that *column header* is

confused with *row* sometimes as both are defined in the horizontal direction with *column header* being the topmost row annotation in the table. The background is identified as *spanning cell* many times leading to wrong predictions.

Figure 5.13 depicts the qualitative results of the Table Structure Recognition model presented in this work. The predicted bounding boxes for all the classes match the ground truth exactly. Another example with plain image, ground truth annotation and prediction result is shown in Figure 5.14 which is a very complex table to identify the structure. But the model is able to localize the table structures correctly.

The quantitative and qualitative results both agree to prove the robustness of the model. The results obtained in the Table Structure Recognition task sets a new benchmark on the PubTables-1M dataset.

| Class-wise AP scores | | | | | |
|---|---|---|---|---|---|
| **Table** | **Row** | **Column** | **Project Row Header** | **Column Header** | **Spanning Cell** |
| 0.99 | 0.9 | 0.976 | 0.943 | 0.944 | 0.848 |

Table 5.9: TABLE STRUCTURE RECOGNITION SCORES FOR EACH CLASS IN THE TEST SET. COCO style AP scores are reported for each class which is calculated by averaging it over IoUs from [0.5,0.95] with a step size of 0.05.

## Cross-testing Experiments

As in the case of the table detection model to check its generalization power over an unseen dataset sample, we also perform inference with the table structure recognition model and compare the performance with the DETR model qualitatively due to the unavailability of the annotated out-of-distribution sample to calculate quantitative metric. Figure 5.15 displays the prediction results on a sample from ICDAR 2013. The prediction of our model is shown in the middle and the bottom image shows the prediction from the DETR model. As can be seen from the figure, the model presented in this work successfully differentiates all rows and all columns and also identifies the column header. However, the DETR model fails to deliver any kind of conclusive result. A similar kind of inference can be drawn from Figure 5.16 which represents prediction on a sample from the Marmot dataset. It is interesting to see from this example how our model is able to detect spanning cell and column header both in an unseen sample. TableNet (Paliwal et al., 2019) model is also used to perform inference on the unseen dataset and it failed to deliver any results. It can be inferenced here that the DETR model overfits the single large domain on which it is trained with 758,849 images and does not perform robustly across domains. The thoughtfully chosen parameters of DETR might help the model to produce meaningful results.

More results from the Table Structure Recognition model on a random unseen sample collected from the internet can be observed in Figure 5.17 and 5.18. It is particularly evident in the Figure 5.18 how well the model performs in localizing all the three spanning cells.

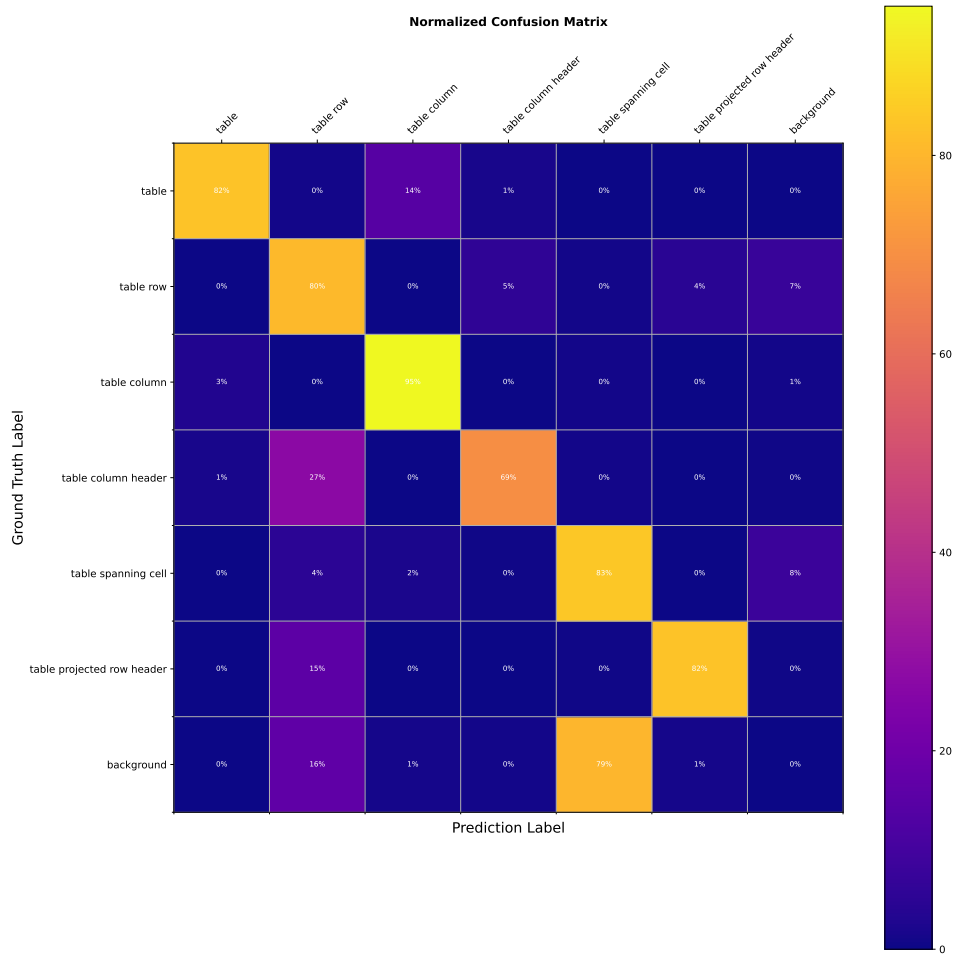



Figure 5.12: CONFUSION MATRIX FOR TABLE STRUCTURE RECOGNITION. The matrix shows all the classes in the ground truth plus an additional background class to differentiate foreground objects (classes) from the background.

## 5.3 Failure Cases

The table detection and structure recognition models perform very well but it is always important to study the examples where models fail to overcome the limitations of the model. In this section, we present such failure cases of both the models.

In Figure 5.19, the ground truth annotation is presented in the left image and the right image contains the prediction of the detection model. From this example, it can be seen that the model fails to output a single table as in the ground truth because of the presence of bold ruling lines separating the two tables. However, some may reason that it is actually two tables. Such cases leads to lower scores where it is a ground truth mislabeling rather than a failed prediction.

In the second example in Figure 5.20, we observe that one of the predictions is incorrect. In these cases, where multi-line sentences are present abruptly without any ruling lines, it is hard

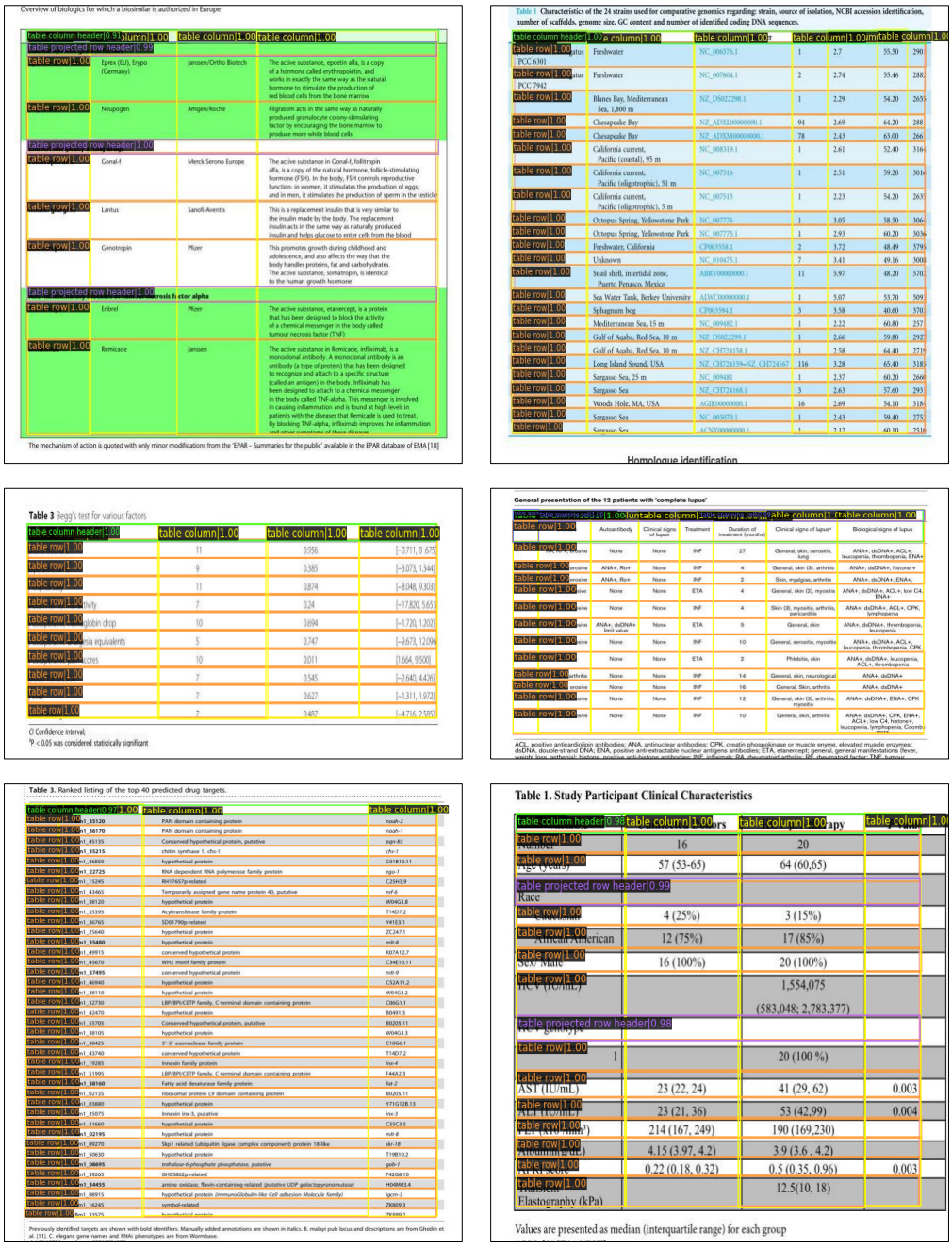
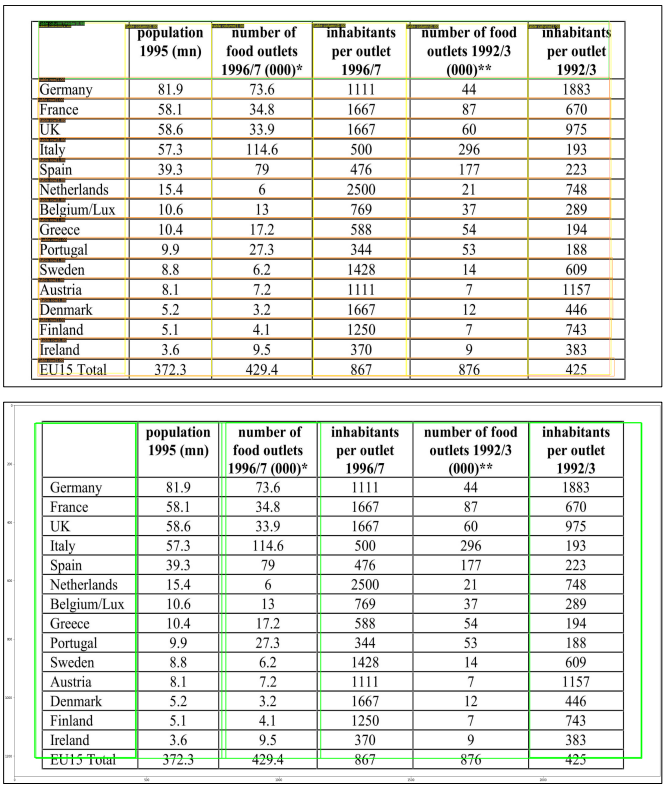


Figure 5.13: TABLE STRUCTURE RECOGNITION RESULTS ON PUBTABLES-1M DATASET. The model correctly identifies the different rows, columns, spanning cells, column header and the project row headers drawn with different colors.

for the model to localize the table perfectly. More annotated samples of such cases will help this issue.

Another example of a failure case on ISRI-OCR is shown in Figure 5.21. It is also debatable if

Figure 5.14: TABLE STRUCTURE RECOGNITION RESULT ON PUBTABLES-1M SAMPLE. Left is the plain image, middle is the plain image with ground truth annotations and on the right is the output of the model shown with different colors.



| | population 1995 (mn) | number of food outlets 1996/7 (000)* | inhabitants per outlet 1996/7 | number of food outlets 1992/3 (000)** | inhabitants per outlet 1992/3 |
|---|---|---|---|---|---|
| Germany | 81.9 | 73.6 | 1111 | 44 | 1883 |
| France | 58.1 | 34.8 | 1667 | 87 | 670 |
| UK | 58.6 | 33.9 | 1667 | 60 | 975 |
| Italy | 57.3 | 114.6 | 500 | 296 | 193 |
| Spain | 39.3 | 79 | 476 | 177 | 223 |
| Netherlands | 15.4 | 6 | 2500 | 21 | 748 |
| Belgium/Lux | 10.6 | 13 | 769 | 37 | 289 |
| Greece | 10.4 | 17.2 | 588 | 54 | 194 |
| Portugal | 9.9 | 27.3 | 344 | 53 | 188 |
| Sweden | 8.8 | 6.2 | 1428 | 14 | 609 |
| Austria | 8.1 | 7.2 | 1111 | 7 | 1157 |
| Denmark | 5.2 | 3.2 | 1667 | 12 | 446 |
| Finland | 5.1 | 4.1 | 1250 | 7 | 743 |
| Ireland | 3.6 | 9.5 | 370 | 9 | 383 |
| EU15 Total | 372.3 | 429.4 | 867 | 876 | 425 |



| | population 1995 (mn) | number of food outlets 1996/7 (000)* | inhabitants per outlet 1996/7 | number of food outlets 1992/3 (000)** | inhabitants per outlet 1992/3 |
|---|---|---|---|---|---|
| Germany | 81.9 | 73.6 | 1111 | 44 | 1883 |
| France | 58.1 | 34.8 | 1667 | 87 | 670 |
| UK | 58.6 | 33.9 | 1667 | 60 | 975 |
| Italy | 57.3 | 114.6 | 500 | 296 | 193 |
| Spain | 39.3 | 79 | 476 | 177 | 223 |
| Netherlands | 15.4 | 6 | 2500 | 21 | 748 |
| Belgium/Lux | 10.6 | 13 | 769 | 37 | 289 |
| Greece | 10.4 | 17.2 | 588 | 54 | 194 |
| Portugal | 9.9 | 27.3 | 344 | 53 | 188 |
| Sweden | 8.8 | 6.2 | 1428 | 14 | 609 |
| Austria | 8.1 | 7.2 | 1111 | 7 | 1157 |
| Denmark | 5.2 | 3.2 | 1667 | 12 | 446 |
| Finland | 5.1 | 4.1 | 1250 | 7 | 743 |
| Ireland | 3.6 | 9.5 | 370 | 9 | 383 |
| EU15 Total | 372.3 | 429.4 | 867 | 876 | 425 |

Figure 5.15: COMPARISON WITH DETR ON ICDAR 2013 DATASET SAMPLE. The top image and the bottom image correspond to our model's and the DETR model's prediction respectively.

it is a table or not or if these kinds of mislabeled examples hamper the feature set. This is also a case where more of such training samples can help to locate the tables in the images correctly.

Figure 5.22 shows an example where two extra tables are detected. While one of those is not a table at all, the other one has a table like characteristics. The model is seen to get confused with

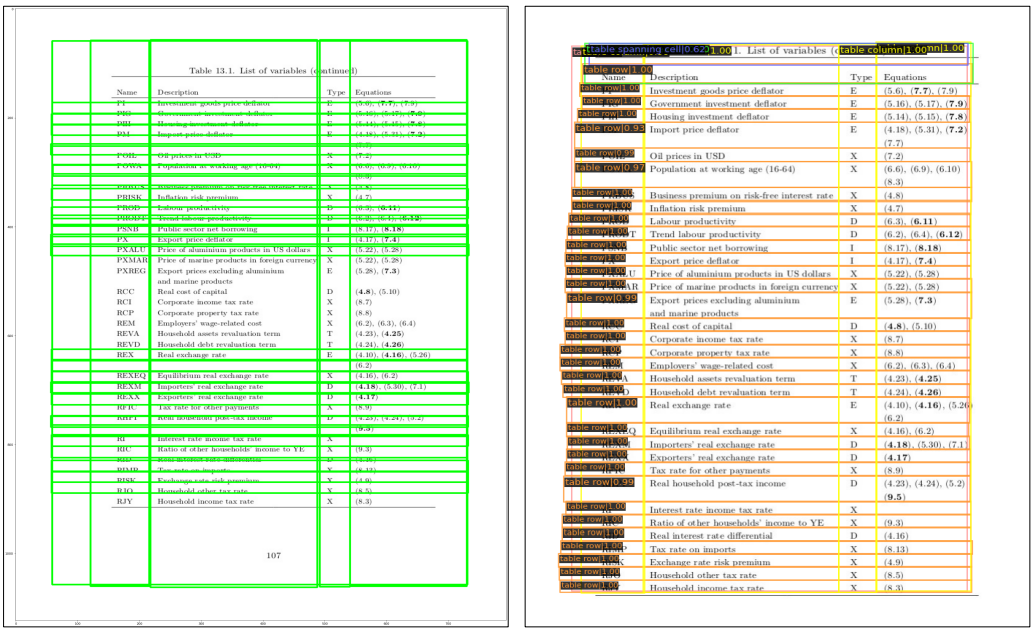


Figure 5.16: COMPARISON WITH DETR ON MARMOT DATASET SAMPLE. The left image and the right image correspond to the DETR model's and our model's prediction respectively.

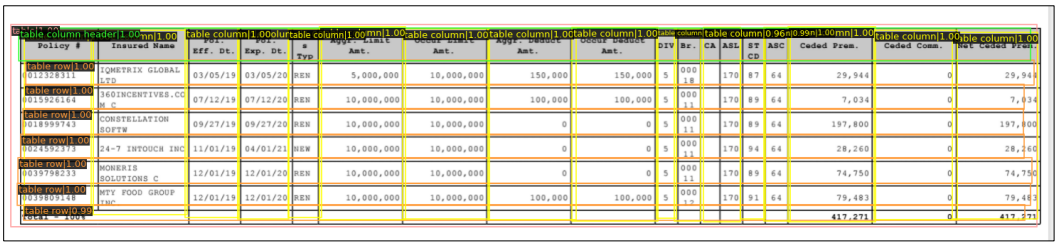
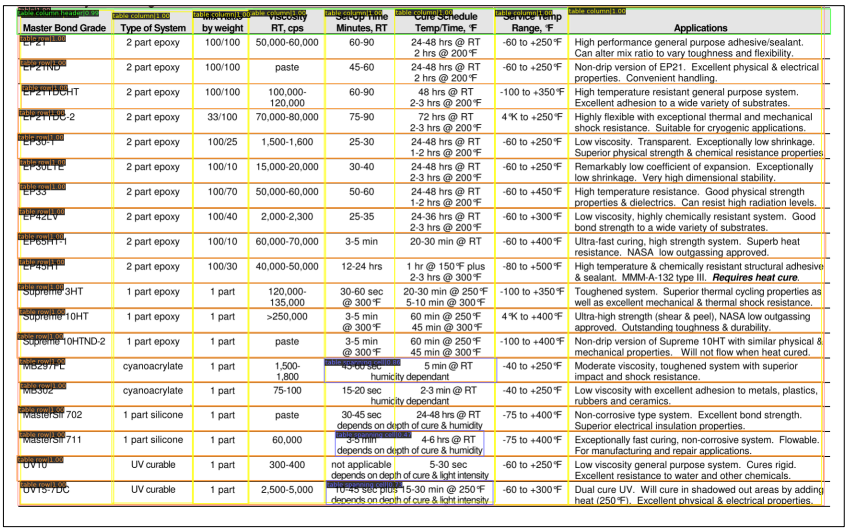


Figure 5.17: TABLE STRUCTURE RECOGNITION ON AN UNSEEN SAMPLE. The model is able to dissect the table into its elements by classifying the table elements correctly.

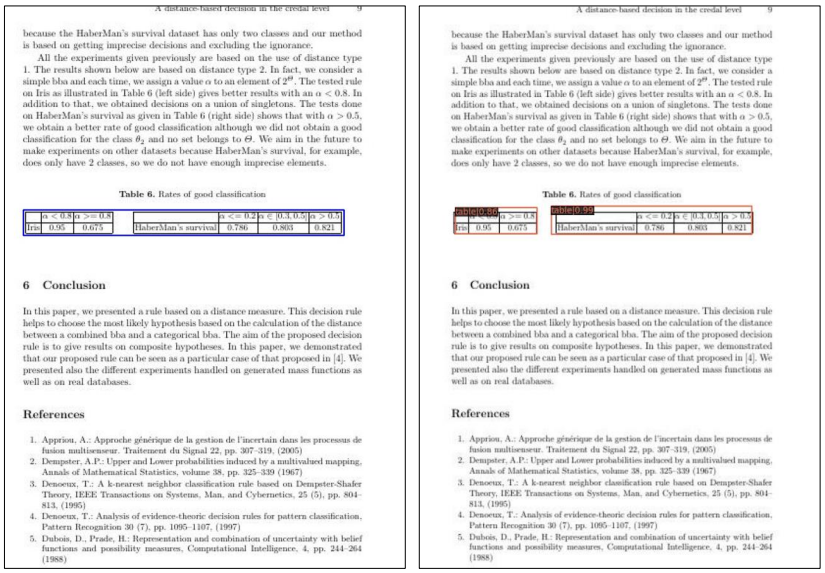
such examples where it is difficult to differentiate a figure from a table.

Table Structure Recognition model also produces some erroneous outputs one of which is shown in Figure 5.23. Here, the table contains the figures of organic compounds as well which is a difficult sample for the model to analyze and predict. It is observed that the spanning cells class had the highest false-positive cases amongst all the classes as also indicated by Table 5.9. Scores are found to increase by incorporating more samples from the dataset during training. An example where spanning cells localization fails is shown in Figure 5.24 where the image on the left only shows spanning cells ground truth annotation to have a better visual comparison. It can be noticed how the model misidentifies spanning cells with the projected row headers in some occurrences. There is no conclusive evidence as to why spanning cells localization fails in such cases.



| Master Bond Grade | Type of System | Mix Ratio by weight | Viscosity RT, cps | Set-Up Time Minutes, RT | Cure Schedule Temp/Time, °F | Service Temp Range, °F | Applications |
|---|---|---|---|---|---|---|---|
| EP21 | 2 part epoxy | 100/100 | 50,000-60,000 | 60-90 | 24-48 hrs @ RT<br>2 hrs @ 200°F | -60 to +250°F | High performance general purpose adhesive/sealant. Can alter mix ratio to vary toughness and flexibility. |
| EP21ND | 2 part epoxy | 100/100 | paste | 45-60 | 24-48 hrs @ RT<br>2 hrs @ 200°F | -60 to +250°F | Non-drip version of EP21. Excellent physical & electrical properties. Convenient handling. |
| EP21TDCHT | 2 part epoxy | 100/100 | 100,000-120,000 | 60-90 | 48 hrs @ RT<br>2-3 hrs @ 200°F | -100 to +350°F | High temperature resistant general purpose system. Excellent adhesion to a wide variety of substrates. |
| EP21TDC-2 | 2 part epoxy | 33/100 | 70,000-80,000 | 75-90 | 72 hrs @ RT<br>2-3 hrs @ 200°F | 4°K to +250°F | Highly flexible with exceptional thermal and mechanical shock resistance. Suitable for cryogenic applications. |
| EP30T | 2 part epoxy | 100/25 | 1,500-1,600 | 25-30 | 24-48 hrs @ RT<br>1-2 hrs @ 200°F | -60 to +250°F | Low viscosity. Transparent. Exceptionally low shrinkage. Superior physical strength & chemical resistance properties. |
| EP30LTE | 2 part epoxy | 100/10 | 15,000-20,000 | 30-40 | 24-48 hrs @ RT<br>2-3 hrs @ 200°F | -60 to +250°F | Remarkably low coefficient of expansion. Exceptionally low shrinkage. Very high dimensional stability. |
| EP33 | 2 part epoxy | 100/70 | 50,000-60,000 | 50-60 | 24-48 hrs @ RT<br>1-2 hrs @ 200°F | -60 to +450°F | High temperature resistance. Good physical strength properties & dielectrics. Can resist high radiation levels. |
| EP42LV | 2 part epoxy | 100/40 | 2,000-2,300 | 25-35 | 24-36 hrs @ RT<br>2-3 hrs @ 200°F | -60 to +300°F | Low viscosity, highly chemically resistant system. Good bond strength to a wide variety of substrates. |
| EP65HT-1 | 2 part epoxy | 100/10 | 60,000-70,000 | 3-5 min | 20-30 min @ RT | -60 to +400°F | Ultra-fast curing, high strength system. Superb heat resistance. NASA low outgassing approved. |
| EP45HT | 2 part epoxy | 100/30 | 40,000-50,000 | 12-24 hrs | 1 hr @ 150°F plus<br>2-3 hrs @ 300°F | -80 to +500°F | High temperature & chemically resistant structural adhesive & sealant. MMM-A-132 type III. ***Requires heat cure***. |
| Supreme 3HT | 1 part epoxy | 1 part | 120,000-135,000 | 30-60 sec<br>@ 300°F | 20-30 min @ 250°F<br>5-10 min @ 300°F | -100 to +350°F | Toughened system. Superior thermal cycling properties as well as excellent mechanical & thermal shock resistance. |
| Supreme 10HT | 1 part epoxy | 1 part | >250,000 | 3-5 min<br>@ 300°F | 60 min @ 250°F<br>45 min @ 300°F | 4°K to +400°F | Ultra-high strength (shear & peel), NASA low outgassing approved. Outstanding toughness & durability. |
| Supreme 10HTND-2 | 1 part epoxy | 1 part | paste | 3-5 min<br>@ 300°F | 60 min @ 250°F<br>45 min @ 300°F | -100 to +400°F | Non-drip version of Supreme 10HT with similar physical & mechanical properties. Will not flow when heat cured. |
| MB297FL | cyanoacrylate | 1 part | 1,500-1,800 | 45-60 sec<br>humicity dependant | 5 min @ RT | -40 to +250°F | Moderate viscosity, toughened system with superior impact and shock resistance. |
| MB302 | cyanoacrylate | 1 part | 75-100 | 15-20 sec<br>humicity dependant | 2-3 min @ RT | -40 to +250°F | Low viscosity with excellent adhesion to metals, plastics, rubbers and ceramics. |
| MasterSil 702 | 1 part silicone | 1 part | paste | 30-45 sec<br>depends on depth of cure & humidity | 24-48 hrs @ RT | -75 to +400°F | Non-corrosive type system. Excellent bond strength. Superior electrical insulation properties. |
| MasterSil 711 | 1 part silicone | 1 part | 60,000 | 3-5 min<br>depends on depth of cure & humidity | 4-6 hrs @ RT | -75 to +400°F | Exceptionally fast curing, non-corrosive system. Flowable. For manufacturing and repair applications. |
| UV10 | UV curable | 1 part | 300-400 | not applicable | 5-30 sec<br>depends on depth of cure & light intensity | -60 to +250°F | Low viscosity general purpose system. Cures rigid. Excellent resistance to water and other chemicals. |
| UV15-7DC | UV curable | 1 part | 2,500-5,000 | 10-45 sec plus<br>depends on depth of cure & light intensity | 15-30 min @ 250°F | -60 to +300°F | Dual cure UV. Will cure in shadowed out areas by adding heat (250°F). Excellent physical & electrical properties. |

Figure 5.18: TABLE STRUCTURE RECOGNITION ON AN UNSEEN SAMPLE. The model localizes all three spanning cells correctly along with other classes.



A distance-based decision in the credal level 9

because the HaberMan's survival dataset has only two classes and our method is based on getting imprecise decisions and excluding the ignorance.

All the experiments given previously are based on the use of distance type 1. The results shown below are based on distance type 2. In fact, we consider a simple bba and each time, we assign a value $\alpha$ to an element of $2^{\Theta}$. The tested rule on Iris as illustrated in Table 6 (left side) gives better results with an $\alpha < 0.8$. In addition to that, we obtained decisions on a union of singletons. The tests done on HaberMan's survival as given in Table 6 (right side) shows that with $\alpha > 0.5$, we obtain a better rate of good classification although we did not obtain a good classification for the class $\theta_2$ and no set belongs to $\Theta$. We aim in the future to make experiments on other datasets because HaberMan's survival, for example, does only have 2 classes, so we do not have enough imprecise elements.

**Table 6.** Rates of good classification

| | $\alpha < 0.8$ | $\alpha >= 0.8$ |
|---|---|---|
| Iris | 0.95 | 0.675 |

| | $\alpha <= 0.2$ | $\alpha \in ]0.3, 0.5]$ | $\alpha > 0.5$ |
|---|---|---|---|
| HaberMan's survival | 0.786 | 0.803 | 0.821 |

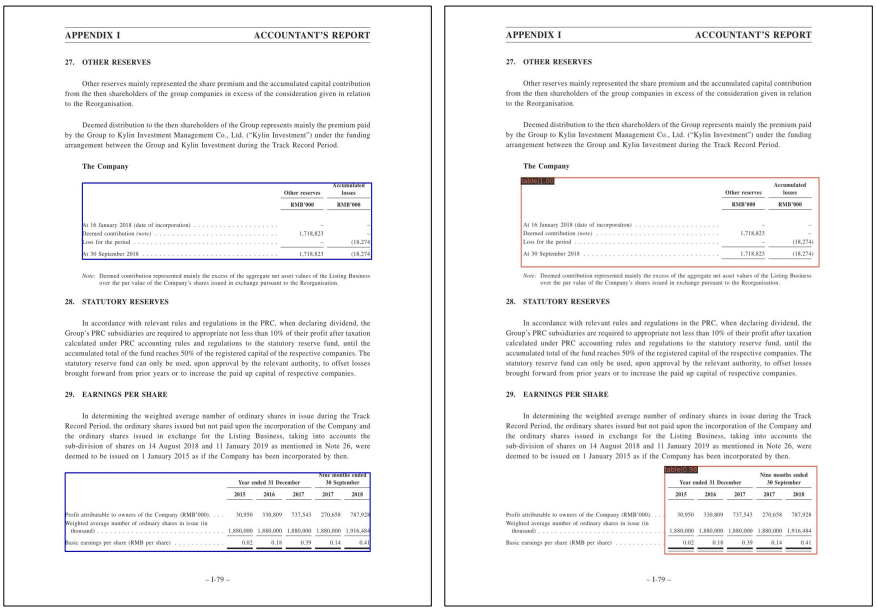
## 6 Conclusion

In this paper, we presented a rule based on a distance measure. This decision rule helps to choose the most likely hypothesis based on the calculation of the distance between a combined bba and a categorical bba. The aim of the proposed decision rule is to give results on composite hypotheses. In this paper, we demonstrated that our proposed rule can be seen as a particular case of that proposed in [4]. We presented also the different experiments handled on generated mass functions as well as on real databases.

## References


1. Appriou, A.: Approche générique de la gestion de l'incertain dans les processus de fusion multisenseur. Traitement du Signal 22, pp. 307-319, (2005)
2. Dempster, A.P.: Upper and Lower probabilities induced by a multivalued mapping, Annals of Mathematical Statistics, volume 38, pp. 325-339 (1967)
3. Denoeux, T.: A k-nearest neighbor classification rule based on Dempster-Shafer Theory, IEEE Transactions on Systems, Man, and Cybernetics, 25 (5), pp. 804-813, (1995)
4. Denoeux, T.: Analysis of evidence-theoric decision rules for pattern classification, Pattern Recognition 30 (7), pp. 1095-1107, (1997)
5. Dubois, D., Prade, H.: Representation and combination of uncertainty with belief functions and possibility measures, Computational Intelligence, 4, pp. 244-264 (1988)


Figure 5.19: INCORRECT DETECTION OF TABLE. The left image corresponds to ground truth while right image corresponds to the prediction.



APPENDIX I ACCOUNTANT'S REPORT

27. OTHER RESERVES

Other reserves mainly represented the share premium and the accumulated capital contribution from the then shareholders of the group companies in excess of the consideration given in relation to the Reorganisation.

Deemed distribution to the then shareholders of the Group represents mainly the premium paid by the Group to Kylin Investment Management Co., Ltd. ("Kylin Investment") under the funding arrangement between the Group and Kylin Investment during the Track Record Period.

The Company

| | Other reserves | Accumulated losses |
|---|---|---|
| | RMB'000 | RMB'000 |
| At 16 January 2018 (date of incorporation) | – | – |
| Deemed contribution (note) | 1,718,823 | – |
| Loss for the period | – | (18,274) |
| At 30 September 2018 | 1,718,823 | (18,274) |

*Note:* Deemed contribution represented mainly the excess of the aggregate net asset values of the Listing Business over the par value of the Company's shares issued in exchange pursuant to the Reorganisation.

28. STATUTORY RESERVES

In accordance with relevant rules and regulations in the PRC, when declaring dividend, the Group's PRC subsidiaries are required to appropriate not less than 10% of their profit after taxation calculated under PRC accounting rules and regulations to the statutory reserve fund, until the accumulated total of the fund reaches 50% of the registered capital of the respective companies. The statutory reserve fund can only be used, upon approval by the relevant authority, to offset losses brought forward from prior years or to increase the paid up capital of respective companies.

29. EARNINGS PER SHARE

In determining the weighted average number of ordinary shares in issue during the Track Record Period, the ordinary shares issued but not paid upon the incorporation of the Company and the ordinary shares issued in exchange for the Listing Business, taking into accounts the sub-division of shares on 14 August 2018 and 11 January 2019 as mentioned in Note 26, were deemed to be issued on 1 January 2015 as if the Company has been incorporated by then.

| | Year ended 31 December | | | Nine months ended 30 September | |
|---|---|---|---|---|---|
| | 2015 | 2016 | 2017 | 2017 | 2018 |
| Profit attributable to owners of the Company (RMB'000) | 30,950 | 330,809 | 737,543 | 270,658 | 787,928 |
| Weighted average number of ordinary shares in issue (in thousand) | 1,880,000 | 1,880,000 | 1,880,000 | 1,880,000 | 1,916,484 |
| Basic earnings per share (RMB per share) | 0.02 | 0.18 | 0.39 | 0.14 | 0.41 |

– I-79 –

Figure 5.20: INCORRECT DETECTION OF TABLE. The left image corresponds to ground truth while right image corresponds to the prediction.

SPORT BEAT

PENNY WELL SPENT ORLANDO GOT MORE THAN IT BARGAINED FOR IN ANFERNEE HARDAWAY, ALREADY ONE OF THE NBA'S BEST/BY RICK WEINBERG

Playing in just his second NBA season, he already is one of the most electrifying and compelling performers in the game. Anfernee Hardaway. The name has a magical sound to it.

"He's absolutely phenomenal," says a fellow magician, Magic Johnson. "Really, words don't describe how great he is and how great he's going to be. He's one of the three most entertaining players in the league right now."

And to think some people actually questioned—some even complained about—the Orlando Magic's acquisition of this extraordinary talent. Would anyone even consider trading Hardaway for Chris Webber today?

Pity those poor Golden State Warriors. In the 20th-anniversary season of their only NBA championship, the Warriors have had the most forgettable year for a sports franchise in recent memory, not only being forced to trade Webber, but having to live with the fact that they not only dealt Hardaway, but also three No. 1 draft picks to get Webber.

Today, "Webber and five No. 1's wouldn't get the deal done," one member of the Magic says.

What's hard to believe is that Orlando originally wasn't interested in Hardaway. It was Penny who wanted Orlando more than Orlando wanted Penny. The Magic wanted Webber, particularly after Hardaway didn't do much in his initial visit to Orlando to change the minds of club officials. But after Shaquille O'Neal played against Hardaway during the filming of "Blue Chips," Shaq convinced management to give Penny a second look. That's when Hardaway "blew us away," as Magic director of player personnel John Gabriel puts it.

Hardaway has blown us all away with his breathtaking array of skills. He's among the top five players in the game today. At 22, and after only 1½ years in the league, he was the starting point guard for the Eastern Conference All-Stars.

He's a fearless penetrator who explodes to the basket. He'll dunk over Patrick Ewing or Hakeem Olajuwon without blinking. At 6-7, he's got great ball-handling skills and low-post moves. He's a dazzling passer with tremendous court vision and rebounding skills for a guard. Granted, he's not a great outside shooter—yet. He vows he'll become one; he plans to launch jumpers "by the hundreds" every day during the off-season.

"I can see Penny averaging 22-25 points, 10 assists, and five or six boards, maybe even next season," says Magic forward Horace Grant. "But what's exciting is the way he gets his points, assists and rebounds. He's so exciting out there, like a magician."

Says Hardaway: "I want to be known as one of the greatest players in history. When people talk about the greats, I want to be mentioned with Michael Jordan, Magic Johnson, Larry Bird."

That day may very well come.

Boomer Esiason

Anfernee Hardaway

Jets quarterback Boomer Esiason, who's proven he'll be as good in the broadcast booth as he once was on the field, unfortunately has no peer when it comes to inconsistency at the quarterback position over the last five seasons.

He led the league with 22 interceptions in 1990 and averaged only 12 TDs and 15 picks in 1991 and '92. After throwing for 3,421 yards and completing 61 percent of his passes in his first year with the Jets in '93, he sunk to a 58-percent completion ratio with 17 TDs and 13 interceptions last season. He also saw his December record with the Jets slip to 1-7.

With new coach Rich Kotite having brought in Bubby Brister, it'll be interesting to see what develops. Remember, Brister was Kotite's No. 2 quarterback in Philadelphia, and when Randall Cunningham slumped, Kotite yanked him and called on Brister.

Kotite, Esiason's fifth coach in five seasons, refuses to speculate, but it's evident that if Esiason is ineffective, Brister will get the call.

If last year's NBA draft were to take place today, the top 10 would look a little different, according to a panel of two assistant coaches and one general manager:

LAST JUNE'S TOP 10

1. Glenn Robinson, Milwaukee
2. Jason Kidd, Dallas
3. Grant Hill, Detroit
4. Donyell Marshall, Minnesota
5. Juwan Howard, Washington
6. Sharone Wright, Philadelphia
7. Lamond Murray, L.A. Clippers
8. Brian Grant, Sacramento
9. Eric Montross, Boston
10. Eddie Jones, L.A. Lakers

REVISED TOP 10 (BASED SOLELY ON TALENT, NOT TEAM NEED)

1. Glenn Robinson
2. Grant Hill
3. Juwan Howard
4. Eddie Jones
5. Jason Kidd
6. Brian Grant
7. Eric Montross
8. Sharone Wright
9. Donyell Marshall
10. Lamond Murray

Dallas still might have taken Kidd over Hill because of its desperate need for a point guard, but there's no question the Mavericks have been disappointed with Kidd's outside shooting (he's hovered around 37 percent most of the season) and free-throw shooting.

Early on, it appeared as if the Bucks had blown the top pick by taking Robinson over Hill. No mistake. Hill's a stud, but Robinson came on like a seasoned All-Star about a month into the season.

The big changes, as the above rankings indicate, take place from the fourth pick on. Marshall was picked too high,

74 SPORT/JUNE 1995

Figure 5.21: INCORRECT DETECTION OF TABLE. The left image corresponds to ground truth while right image corresponds to the prediction.

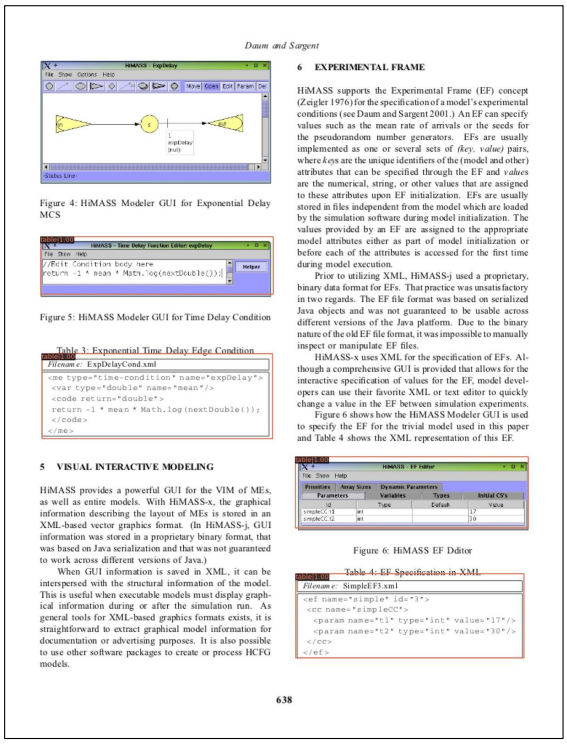



Figure 5.22: INCORRECT DETECTION OF TABLE. The ground truth contains two tables, Table 3 and Table 4, in the image but the model predicts two extra tables with one of them having table like characteristics. It is one of the exceptional cases where a figure has table like features.

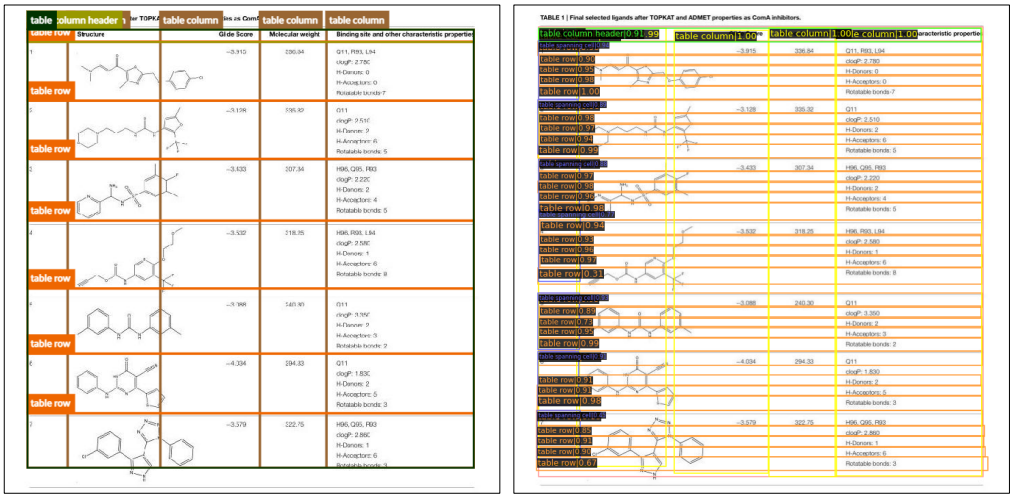



Figure 5.23: INCORRECT DETECTION OF TABLE STRUCTURE. A sample from PubTables-1M where ground truth is on the left and the predictions on the right. The model predicts more rows than the ground truth and gets confused with the figures inside the table.


Figure 5.24: INCORRECT DETECTION OF TABLE STRUCTURE. A sample for PubTables-1M where ground truth of spanning cells is on the left and the predictions are on the right. The detection of spanning cells is quite inaccurate.

# Chapter 6

# Discussion

We conduct several experiments in table detection to define the choices and parameters for the model development. It is important to understand the importance of the building blocks of the Faster R-CNN model to compensate for the labeled data or, in general, resource requirements. The default parameters are used for Faster R-CNN where ever possible keeping the rest of the runtime settings like learning rate, momentum *etc.* same for each model to study the experiments. The F1 scores are calculated using the aIoU metric.

## 6.1 Depth of the Backbone Network

In the first experiment, we use different depths of ResNet backbone with normal convolutions (not deformable convolutions) to check for the detection quality. The quantitative results are shown in Table 6.1 for IoU = 0.6. In this case, the predictions improve with the depth of the ResNet backbone network. Also, the training time increases due to more number of layers in ResNet-101 and ResNet-152. This experiment motivates us to use the backbone with 101 layers which gives good accuracy without much expense of the training time.

| Model | F1 score |
|---|---|
| ResNet-50 | 0.89 |
| ResNet-101 | 0.912 |
| ResNet-152 | 0.931 |

Table 6.1: COMPARISON ON DIFFERENT DEPTHS OF THE RESNET BACKBONE. F1 scores are calculated at IoU = 0.6.

## 6.2 Variant of the ResNet Backbone Network

After deciding the depth of the backbone network, we experiment with the variant of the ResNet backbone network, as explained in chapter 4, with normal convolutions. The results are described in Table 6.2 and it demonstrate the expected higher scores for the ResNeXt model as compared to the ResNet model which warrants the need to use ResNeXt-101 architecture.

| Model | F1 score |
|---|---|
| ResNet-101 | 0.912 |
| ResNeXt-101 | 0.935 |

Table 6.2: COMPARISON ON DIFFERENT VARIANTS OF THE RESNET BACKBONE. F1 scores are calculated at IoU = 0.6.

## 6.3 Type of Convolutions

We perform experiments to compare the performance of the model with different type of convolutions for the task of table data extraction and the results of which are shown in Table 6.3. The best results are obtained when using ResNeXt-101 with deformable convolutions yielding a F1 score = 0.946 at IoU = 0.6. The gradual improvement in F1 scores is visible in Table 6.3 due to the addition of ResNeXt and deformable convolution module. An example of a visual improvement is shown in Figure 6.1 and 6.2 to distinguish between the performance of ResNet-101 (with normal convolution) and ResNeXt-101 (with deformable convolution).

| Model | Convolution type | F1 score |
|---|---|---|
| ResNet-50 | Normal | 0.89 |
| ResNet-101 | Normal | 0.912 |
| ResNet-101 | Deformable | 0.923 |
| ResNeXt-101 | Normal | 0.935 |
| ResNeXt-101 | Deformable | 0.946 |

Table 6.3: COMPARISON ON DIFFERENT TYPES OF CONVOLUTIONS FOR THE BACKBONE. F1 scores are calculated at IoU = 0.6.

## 6.4 Type of Regression Loss

In this experiment, we test the Faster R-CNN with ResNeXt-101 backbone by using three different loss functions defined in Section 4.2.3: Smooth L1 (Equation 4.4), Generalized Intersection over Union (GIoU) (Equation (4.5)), and Distance Intersection over Union loss (DIoU) (Equation (4.6)) function.

| Model | Convolution type | Regression Loss | F1 score |
|---|---|---|---|
| ResNeXt-101 | Deformable | GIoU | 0.883 |
| ResNext-101 | Deformable | DIoU | 0.897 |
| ResNeXt-101 | Deformable | Smooth L1 | 0.946 |

Table 6.4: COMPARISON ON DIFFERENT TYPES OF LOSSES FOR THE REGRESSION BRANCH. F1 scores are calculated at IoU = 0.6.

The results of the experiments are shown in Table 6.4. While the DIoU loss is the fastest to converge, it still did not yield better results than the smooth L1 loss function. The model trained

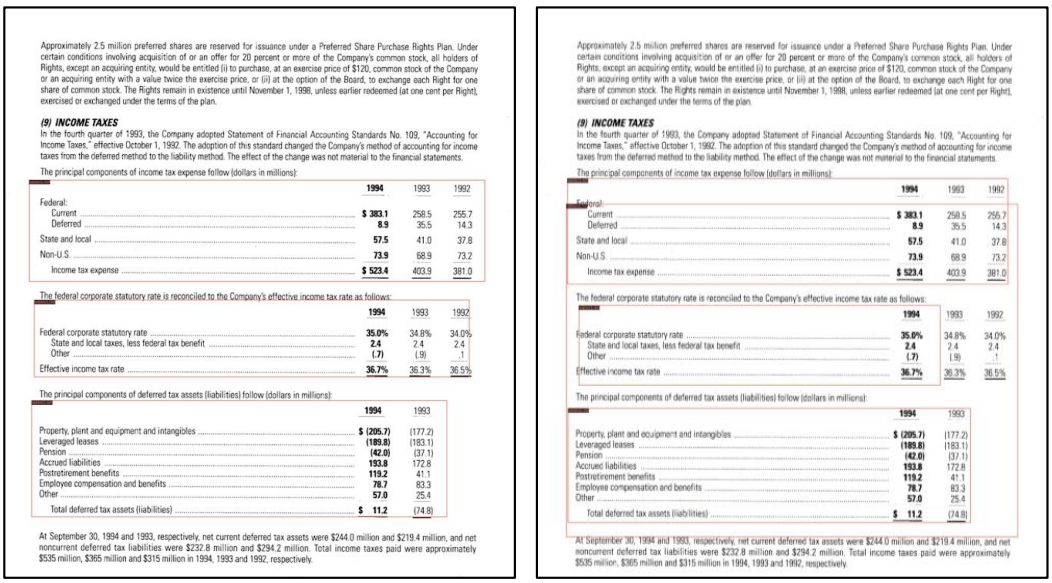


Figure 6.1: PREDICTIONS FROM RESNET-101 MODEL (LEFT) AND RESNEXT-101 (RIGHT). ResNet-101 is trained with normal convolution operation while ResNeXt-101 is trained with deformable convolutions.

with GIoU yields poor quantitative and qualitative results. It fails to localize even a simple table in a few of the cases. An example of the prediction results yielded by the model trained with GIoU loss is shown in Figure 6.3. Model with GIoU loss need many additional epochs to converge better. Because of these results, we use the smooth L1 loss as the regression loss for training the Faster R-CNN.

## 6.5 Training Speed

Training the table detection model takes up quite a lot of time and it is important to optimize the training time where ever possible. On default settings by Chen et al. (2019) and training from scratch without using a pre-trained model for Faster R-CNN, it can take up to 20 hours for 30 epochs on 3500 samples. In the following series of experiments, we evaluate the effect of some prominent parameters on training time.

### 6.5.1 Number of Frozen Stages of Backbone

ResNeXt is a 4 stage architecture. If all the 4 stages are frozen during the training, the model trains the fastest as compared to when ResNeXt is partly frozen or not frozen at all. The results are described in Table 6.5. It highlights the fact that it is not a good idea to completely freeze the backbone network as the backbone network is not able to learn any new features and does not pass relevant features to the Region Proposal Network (RPN). The difference in F1 scores is not significant when 2 or none of the stages of the backbone are frozen but the difference in training time is large. A partially frozen backbone is able to provide high-level features like the boundary of tabular sub-regions to the RPN for further processing which helps in good detection. Hence,

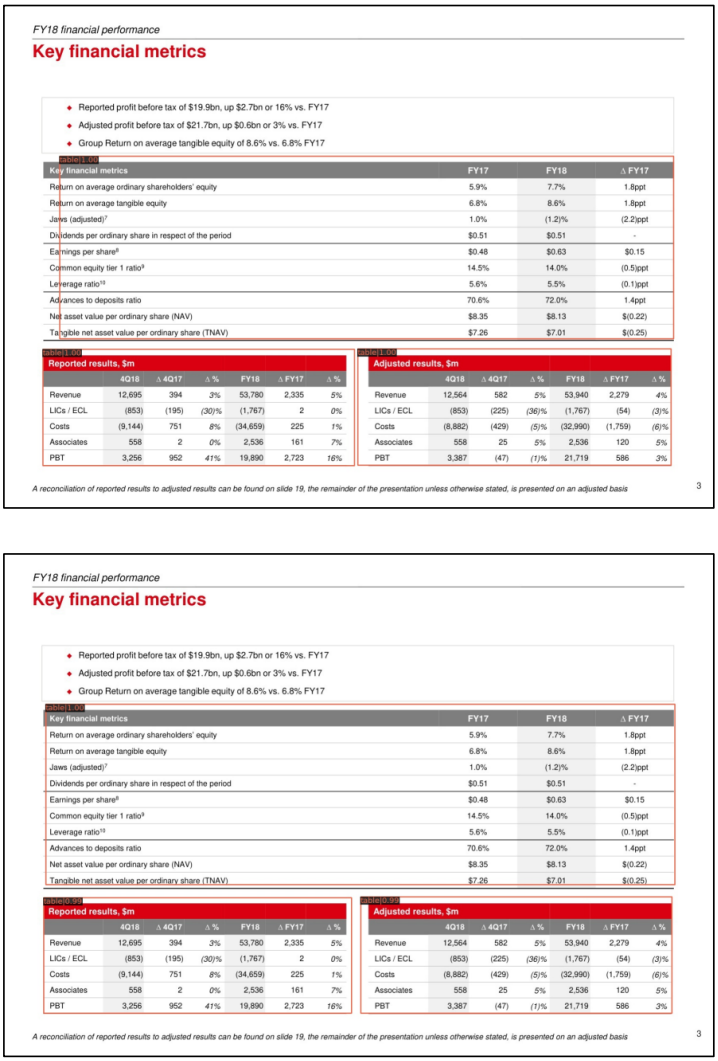






Figure 6.2: PREDICTIONS FROM RESNET-101 MODEL (TOP) AND RESNEXT-101 (BOTTOM). ResNet-101 is trained with normal convolution operation while ResNeXt-101 is trained with deformable convolutions.

this calls the need to freeze the first 2 stages of the ResNeXt backbone model. The complete frozen backbone model produces similar results as shown in Figure 6.3. This happens because the base network is not able to learn any new high-level features at all.

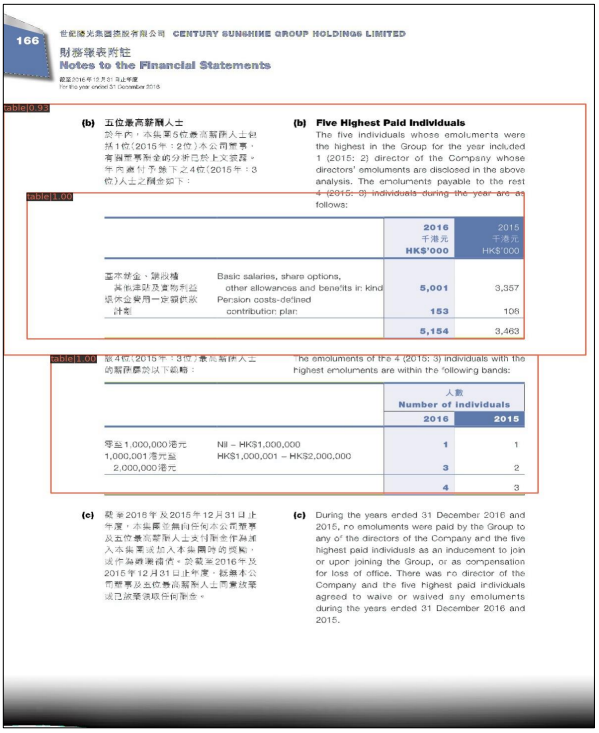



Figure 6.3: PREDICTIONS FROM RESNEXT-101 MODEL TRAINED WITH GIOU LOSS.

| Model | Frozen stages | Training time (hours) | F1 score |
|---|---|---|---|
| ResNeXt-101 | 4 (complete backbone frozen) | ~4.5 | 0.799 |
| ResNext-101 | 2 (partially frozen) | ~8 | 0.935 |
| ResNeXt-101 | 0 (not frozen at all) | ~20 | 0.94 |

Table 6.5: COMPARISON ON THE NUMBER OF FROZEN STAGES OF RESNEXT. F1 scores are calculated at IoU = 0.6.

### 6.5.2 Number of Proposals

We have seen the usage of generating proposals in Faster R-CNN during prediction in the Section 4.2.1. The number of proposals can be defined according to the task at hand. More proposals will be required if there are many objects to be predicted in an image like cars in a traffic jam. While more number of proposals does no unnecessary harm, it is important to study how the number of proposals can influence the training time. The F1 scores with the different number of proposals are shown in Table 6.6. It is observed that increasing the number of proposals to 5000 leads to a minor decrease in the F1 score. This is observed by the model predicting extraneous tables in some images. A lower number of proposals misses the potential table elements in the document. In the case of table structure recognition, where there are many rows, columns *etc.*, the lower number of proposals leads to model not being able to detect these elements. Thus, we use

2000 proposals in the RPN.

| Model | Frozen stages | Training time (hours) | Proposals | F1 score |
|---|---|---|---|---|
| ResNeXt-101 | 2 (partially) | ~6.5 | 500 | 0.911 |
| ResNext-101 | 2 (partially) | ~8 | 2000 | 0.935 |
| ResNeXt-101 | 2 (partially) | ~10.5 | 5000 | 0.924 |

Table 6.6: COMPARISON ON THE NUMBER OF PROPOSALS OF RPN. F1 scores are calculated at IoU = 0.6.

### 6.5.3 Scale of Anchors

Each region proposal is parametrized according to an anchor box (see Section 4.2.1). In this experiment, we analyze the effect of varying the anchor scale on training time while keeping the aspect ratios the same, $[0.5, 1.0, 2.0]$. This means that for every feature point 3 anchors are generated. The results are summarized in the Table 6.7. A too small anchor size of 4 reduces the computation time and hence, the training time. However, it is not able to detect wide and large tables. An average scale of 8 for the anchors is found to be reasonable for table detection.

In the case of table structure recognition where rows, columns, cells are of varying scales, multiple anchors scales $[4, 8, 16]$ and ratios $[0.05, 0.1, 0.2, 0.5, 1.0, 2.0]$ are used. This creates 18 anchor boxes at each feature point in the feature map generated by the backbone network. These boxes are defined to capture the scale and aspect ratio of specific object classes and are usually chosen based on object sizes in the training dataset. These different sizes of anchors are necessary to detect shapes of different rows, columns, spanning cells, projected row headers and column header. By using the same values as in table detection in which only 3 anchors are generated at each feature point, it is observed that model is unable to capture rows, spanning cells in the horizontal direction very well as shown in Figure 6.4. The number of anchors for each feature point can be increased without adding much training time to capture much more varying sizes of objects in the images.

| Model | Frozen stages | Anchor scale | Training time (hours) | Proposals | F1 score |
|---|---|---|---|---|---|
| ResNeXt-101 | 2 (partially) | 4 | ~6 | 2000 | 0.89 |
| ResNeXt-101 | 2 (partially) | 8 | ~8 | 2000 | 0.935 |
| ResNeXt-101 | 2 (partially) | 16 | ~9 | 2000 | 0.93 |

Table 6.7: COMPARISON ON THE SCALE OF ANCHORS. F1 scores are calculated at IoU = 0.6.

These experiments helps in modeling the Faster R-CNN architecture for table detection and, also, table structure recognition model.

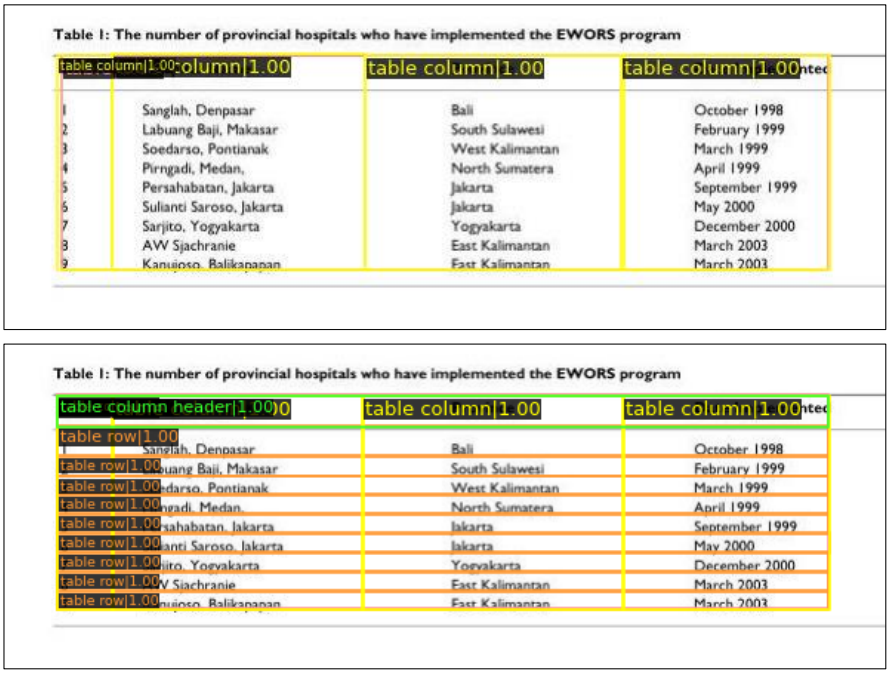



Figure 6.4: EFFECT OF ANCHOR SCALES. The left image shows predictions of the model with 3 anchors only which is unable to capture rows *etc.* and the right image depicts the prediction of the model with 18 anchors of different shapes.

# Chapter 7

# Conclusion and Future Work

Table Detection and Table Structure Recognition is concerned with extracting data from a table in an orderly manner whilst retaining the relations between different rows, columns, cells *etc.*. In this thesis, we present an end-to-end Faster R-CNN model with ResNeXt backbone to perform the task of table data extraction with limited access to resources. This model is simpler to execute, adaptable to any dataset, fast to train and generates comparable or even better results than state-of-the-art models. The Faster R-CNN model is extensively studied and carefully designed to eliminate the need for the huge annotated datasets by making use of pre-trained models on ImageNet and MS COCO dataset. The model does not rely on any significant data pre-processing or any post-processing methods to refine the output. This thesis work establishes the superiority of Faster R-CNN once again in the task of object detection and yields state-of-the-art results for table detection on ICDAR 2013, ICDAR 2019, TableBank, ISRI-OCR and Marmot datasets. The cross-testing results on these datasets also yield great results as compared to other popular models. From the experiments, it is also concluded that the model is able to localize multiple tables in a document.

In this work, we also demonstrate the state-of-the-art performance on PubTables-1M dataset for the task of table structure recognition and encourage the usage of this dataset to further unify the research direction. The results are explained in detail with the insights drawn from the prediction results. Qualitative detection samples are provided for both table understanding tasks, highlighting the method's good performance. This work also presents a new quantitative metric, aIoU, based on Intersection over Union (IoU) which is tailored for the task of table detection. All the datasets in varying formats are converted into a common COCO format and with this, we also promote the usage of a common annotation file for all the datasets and hence, saving extra time. An in-depth discussion is provided in the chapter 6 on how the model parameters and design choices are made.

## Future Work

While a solid foundation for table detection and table structure recognition is laid, there are few gaps that can be filled in future work or new directions which can be pursued. Table Detection and Structure Recognition model produces high quality results. Even though our model generalizes well to the unseen domains, there can always be tables with a completely different domain because of the style and variety it offers and in such a scenario the model might fail to produce good results on a sample. In this case, an unsupervised compound domain adaptation strategy can be employed to overcome this limitation as in Panwar et al. (2021) and Liu et al. (2020) for facial recognition problem. More in-depth analyses and experiments can be performed to understand failure cases better and tackle them. It will be advantageous to study why spanning cells localization fails in certain cases. A probabilistic data-driven augmentation technique (Khan

et al., 2021) can also be employed to expand the dataset by producing structural changes and check if this method can offer samples from different data distributions. PubTables-1M offers more information about the tables like text for all words appearing in each table. Maybe this information along with other semantic knowledge could be used to boost the performance of the table structure recognition model. TableBank dataset also offers a variety of tables for the structure recognition task. However, it provides only HTML tags for individual files which is difficult to relate and mix up with existing datasets in COCO format. This can be solved in the future and the TableBank dataset can be mixed with PubTables-1M to increase the generalization power of the dataset. New metrics for table structure recognition have been proposed like Grid Table Similarity (GriTS) (Smock et al., 2022) and Tree-Edit-Distance-based Similarity (TEDS) (Zhong et al., 2020) where the authors claim that it offers a more meaningful comparison between the ground truth and the predicted structure of the table. It will be interesting to use these metrics to gain more insights into the structure of the model and also compare the performance to other existing models. Finally, Graph Neural Networks have also gained popularity in recent times. It can reconstruct the inherent relation between different cells, rows and columns in a table. It might be insightful to use this completely different strategy on the PubTables-1M dataset to compare the performance with our existing approach or to examine if graph neural network can be used as post-processing of the detections from our network.

# Appendix A

# Attachments

## A.1 ResNet Architecture Overview

ResNet, short for Residual Network (He et al., 2016) is a specific type of neural network which utilizes skip connections to jump over some layers. The addition of more layers help the neural network to learn more complex features but it can also increase the parameters of the models significantly leading to increased training time. Increasing the depth of the network can also lead to poor quantitative results beyond certain threshold. ResNet mitigates the training issues by allowing to add more layers with skip connections i.e. there is a direct connection between two layers separated by some layers. There are two major reasons to add skip connections: 1) To prevent the problem of vanishing gradients 2) To alleviate the accuracy saturation problem, which occurs when adding additional layers to a sufficiently deep model results in increased training error. The architecture of ResNet-50 is shown in the Figure A.1. It has 48 convolution layers along with 1 max pooling and 1 average pooling layer. The architecture consists of 4 stages and these stages can be frozen during training when using a pre-trained model. In this work, we freeze the first two stages in the case of table detection and first stage in the case of table structure recognition.

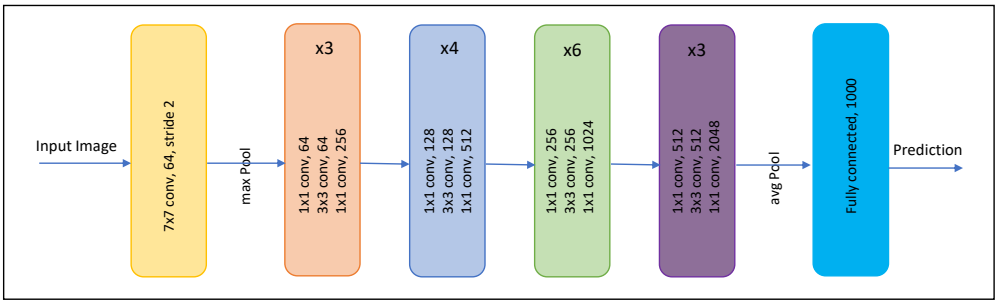
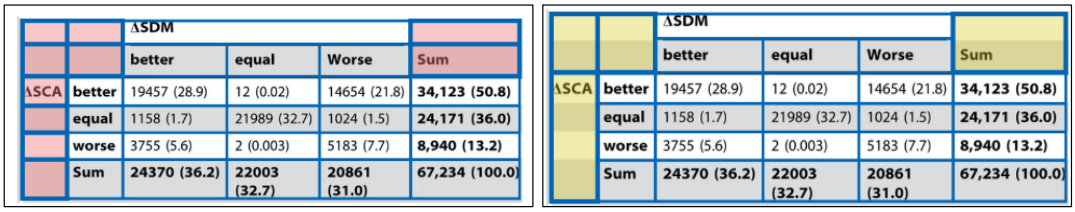



Figure A.1: RESNET-50 ARCHITECTURE. The architecture of ResNet consists of four stages which can be frozen depending on the task and pre-trained model.

## A.2 PubTables-1M Dataset Canonicalization

The canonicalization algorithm described in Smock et al. (2021) rectifies the oversegmentation in a table's structural annotations. To do so, the authors make assumptions about the intended structure of the table i.e. canonicalization is the merging of neighboring cells under particular conditions. An example of such canonicalization can be seen in Figure A.2.



| | | ΔSDM | | | |
|---|---|---|---|---|---|
| | | better | equal | Worse | Sum |
| ΔSCA | better | 19457 (28.9) | 12 (0.02) | 14654 (21.8) | **34,123 (50.8)** |
| | equal | 1158 (1.7) | 21989 (32.7) | 1024 (1.5) | **24,171 (36.0)** |
| | worse | 3755 (5.6) | 2 (0.003) | 5183 (7.7) | **8,940 (13.2)** |
| | Sum | **24370 (36.2)** | **22003 (32.7)** | **20861 (31.0)** | **67,234 (100.0)** |

| | | ΔSDM | | | |
|---|---|---|---|---|---|
| | | better | equal | Worse | Sum |
| ΔSCA | better | 19457 (28.9) | 12 (0.02) | 14654 (21.8) | **34,123 (50.8)** |
| | equal | 1158 (1.7) | 21989 (32.7) | 1024 (1.5) | **24,171 (36.0)** |
| | worse | 3755 (5.6) | 2 (0.003) | 5183 (7.7) | **8,940 (13.2)** |
| | Sum | **24370 (36.2)** | **22003 (32.7)** | **20861 (31.0)** | **67,234 (100.0)** |

Figure A.2: AN ANNOTATED IMAGE SAMPLE FROM PUBTABLES-1M. Left: *over-segmented* structural annotation, resulting in unnecessary blank cells in the headers. Right: The canonical structure annotation combines these cells and represents their true logical structure. *Source* Smock et al. (2021).

The algorithm for canonicalization of the tables, as stated in Smock et al. (2021), is given in Algorithm 1.

**Algorithm 1** PubTables-1M Canonicalization

ADD CELLS TO THE COLUMN AND ROW HEADERS
Split every blank spanning cell into blank grid cells
**if** the first row starts with a blank cell **then** add the first row to the column header
**if** there is at least one row labeled as part of the column header **then**
**while** every column in the column header does not have at least one complete cell that only spans that column **do**: add the next row to the column header
**end if**
**for each** row **do**: **if** the row is not in the column header and has exactly one non-blank cell that occupies the first column **then** label it a projected row header
if any cell in the first column below the column header is a spanning cell or blank **then** add the column (below the column header) to the row header
MERGE CELLS
**for each** cell in the column header **do** recursively merge the cell with any adjacent cells above and below in the column header that span the exact same columns
**for each** cell in the column header **do** recursively merge the cell with any adjacent blank cells below it if every adjacent cell below it is blank and in the column header
**for each** cell in the column header **do** recursively merge the cell with any adjacent blank cells above it if every adjacent cell above it is blank
**for each** projected row header **do** merge all of the cells in the row into a single cell
**for each** cell in the row header **do** recursively merge the cell with any adjacent blank cells below it

# List of Figures

# List of Tables

# Bibliography


Agarwal, M., Mondal, A., and Jawahar, C. V. (2021). CDeC-Net: Composite Deformable Cascade Network for Table Detection in Document Images. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 9491–9498, Los Alamitos, CA, USA. IEEE Computer Society.

Arif, S. and Shafait, F. (2018). Table Detection in Document Images using Foreground and Background Features. In *2018 Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–8. IEEE.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-End Object Detection with Transformers. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J.-M., editors, *Computer Vision – ECCV 2020*, pages 213–229, Cham. Springer International Publishing.

Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C. C., and Lin, D. (2019). MMDetection: Open MMLab Detection Toolbox and Benchmark. *arXiv preprint arXiv:1906.07155*.

Cho, K., van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014). On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Coüasnon, B. and Lemaitre, A. (2014). Recognition of Tables and Forms. In *Handbook of Document Image Processing and Recognition*, pages 647–677, London. Springer London.

Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., and Wei, Y. (2017). Deformable Convolutional Networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 764–773.

Everingham, M., Gool, L. V., Williams, C. K. I., Winn, J. M., and Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338.

Fang, J., Tao, X., Tang, Z., Qiu, R., and Liu, Y. (2012). Dataset, Ground-Truth and Performance Metrics for Table Detection Evaluation. In *2012 10th IAPR International Workshop on Document Analysis Systems*, pages 445–449.

Gad, A. F. (2021). Faster R-CNN explained for Object Detection Tasks. *Paperspace Blog*.

Gao, L., Huang, Y., Déjean, H., Meunier, J.-L., Yan, Q., Fang, Y., Kleber, F., and Lang, E. (2019). ICDAR 2019 Competition on Table Detection and Recognition (cTDaR). In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1510–1515.

Gilani, A., Qasim, S. R., Malik, I., and Shafait, F. (2017). Table Detection Using Deep Learning. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 771–776.

Girshick, R. (2015). Fast R-CNN. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2013). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Günther, M., Hu, P., Herrmann, C., Chan, C. H., Jiang, M., Yang, S., Dhamija, A. R., Ramanan, D., Beyerer, J., Kittler, J., Jazaery, M. A., Nouyed, M. I., Guo, G., Stankiewicz, C., and Boult, T. E. (2017). Unconstrained Face Detection and Open-Set Face Recognition Challenge. *2017 IEEE International Joint Conference on Biometrics (IJCB)*.

Göbel, M., Hassan, T., Oro, E., and Orsi, G. (2013). ICDAR 2013 Table Competition. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1449–1453.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask R-CNN. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Hu, J., Kashi, R. S., Lopresti, D. P., and Wilfong, G. (1999). Document Recognition and Retrieval - Medium-Independent Table Detection. In *SPIE Proceedings*, volume 3967, pages 291–302. SPIE.

Kasar, T., Barlas, P., Adam, S., Chatelain, C., and Paquet, T. (2013). Learning to Detect Tables in Scanned Document Images Using Line Information. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1185–1189.

Khan, S. A., Khalid, S. M. D., Shahzad, M. A., and Shafait, F. (2019). Table Structure Extraction with Bi-Directional Gated Recurrent Unit Networks. *2019 International Conference on Document Analysis and Recognition (ICDAR)*.

Khan, U. Y., Zahid, S., Ali, M. A., Ul-Hasan, A., and Shafait, F. (2021). TabAug: Data Driven Augmentation for Enhanced Table Structure Recognition. In *ICDAR*.

Kieninger, T. and Dengel, A. (1998). A Paper-to-HTML Table Converting System. In *Proceedings DAS98. IAPR International Workshop on Document Analysis Systems (DAS)*, pages 356–365. Int'l Association for Pattern Recognition Workshop on Document Analysis Systems. Nagano, Japan.

Kieninger, T. and Dengel, A. (1999). Table Recognition and Labeling Using Intrinsic Layout Features. In *International Conference on Advances in Pattern Recognition*, pages 307–316, London. Springer London.

Kieninger, T. and Dengel, A. (2001). Applying the T-Recs table recognition system to the business letter domain. In *Proceedings of Sixth International Conference on Document Analysis and Recognition*, pages 518–522.

Li, M., Cui, L., Huang, S., Wei, F., Zhou, M., and Li, Z. (2020). TableBank: Table Benchmark for Image-based Table Detection and Recognition. In *LREC*.

Lin, T., Dollar, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017). Feature Pyramid Networks for Object Detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944, Los Alamitos, CA, USA. IEEE Computer Society.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing.

Liu, Z., Miao, Z., Pan, X., Zhan, X., Lin, D., Yu, S. X., and Gong, B. (2020). Open Compound Domain Adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ma, C., Lin, W., Sun, L., and Huo, Q. (2022). Robust Table Detection and Structure Recognition from Heterogeneous Document Images. *CoRR*, abs/2203.09056.

Ma, E. (2020). Enhancing resnet to resnext for image classification. *Medium*.

Mondal, A., Lipps, P., and Jawahar, C. V. (2020). IIIT-AR-13K: A New Dataset for Graphical Object Detection in Documents. *CoRR*, abs/2008.02569.

Ng, H. T., Lim, C. Y., and Koo, J. L. T. (1999). Learning to recognize tables in free text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL '99, page 443–450, USA. Association for Computational Linguistics.

Paliwal, S., D, V., Rahul, R., Sharma, M., and Vig, L. (2019). TableNet: Deep Learning Model for End-to-end Table Detection and Tabular Data Extraction from Scanned Document Images. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 128–133, Los Alamitos, CA, USA. IEEE Computer Society.

Panwar, A., Singh, P., Saha, S., Paudel, D. P., and Gool, L. V. (2021). Unsupervised Compound Domain Adaptation for Face Anti-Spoofing. *2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021)*, pages 1–8.

Pinto, D., Mccallum, A., Wei, X., and Croft, W. (2003). Table Extraction Using Conditional Random Fields. *ACM SIGIR' 03*.

Prasad, D., Gadpal, A., Kapadni, K., Visave, M., and Sultanpure, K. (2020). CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2439–2447.

Pyreddy, P. and Croft, W. B. (1997). TINTI: A System for Retrieval in Text Tables TITLE2:. In *University of Massachusetts*, USA.

Qasim, S. R., Kieseler, J., Iiyama, Y., and Pierini, M. (2019a). Learning representations of irregular particle-detector geometry with distance-weighted graph networks. *The European Physical Journal C*, 79:1–11.

Qasim, S. R., Mahmood, H., and Shafait, F. (2019b). Rethinking Table Recognition using Graph Neural Networks. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 142–147.

Qiao, L., Li, Z., Cheng, Z., Zhang, P., Pu, S., Niu, Y., Ren, W., Tan, W., and Wu, F. (2021). LGPMA: Complicated Table Structure Recognition with Local and Global Pyramid Mask Alignment. In *Document Analysis and Recognition – ICDAR 2021*, pages 99–114. Springer International Publishing.

Rashid, S. F., Akmal, A., Adnan, M., Aslam, A. A., and Dengel, A. (2017). Table Recognition in Heterogeneous Documents Using Machine Learning. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 777–782.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., and Savarese, S. (2019). Generalized Intersection over Union. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

Saha, R., Mondal, A., and Jawahar, C. V. (2019). Graphical Object Detection in Document Images. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 51–58, Los Alamitos, CA, USA. IEEE Computer Society.

Schreiber, S., Agne, S., Wolf, I., Dengel, A., and Ahmed, S. (2017). DeepDeSRT: Deep Learning for Detection and Structure Recognition of Tables in Document Images. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 1162–1167.

Shafait, F., Keysers, D., and Breuel, T. (2008a). Performance Evaluation and Benchmarking of Six-Page Segmentation Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6):941–954.

Shafait, F., Keysers, D., and Breuel, T. M. (2008b). Efficient implementation of local adaptive thresholding techniques using integral images. In Yanikoglu, B. A. and Berkner, K., editors, *Document Recognition and Retrieval XV*, volume 6815, pages 317 – 322. International Society for Optics and Photonics, SPIE.

Shahab, A. (2010). Table Ground Truth for the UW3 and UNLV datasets (DFKI-TGT-2010).

Shahzad, M. A., Noor, R., Ahmad, S., Mian, A., and Shafait, F. (2019). Feature Engineering Meets Deep Learning: A Case Study on Table Detection in Documents. In *2019 Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–6.

Siddiqui, S., Malik, M. I., Agne, S., Dengel, A., and Ahmed, S. (2018). DeCNT: Deep Deformable CNN for Table Detection. *IEEE Access*, PP:1–1.

Simonyan, K. and Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Smith, R. (2007). An Overview of the Tesseract OCR Engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 629–633.

Smock, B., Pesala, R., and Abraham, R. (2021). PubTables-1M: Towards Comprehensive Table Extraction from Unstructured Documents. *arXiv preprint arXiv:2110.00061*.

Smock, B., Pesala, R., and Abraham, R. (2022). GriTS: Grid table similarity metric for table structure recognition. *arXiv.2203.12555*.

Tupaj, S., Shi, Z., Chang, C. H., and Chang, D. C. H. (1996). Extracting Tabular Information From Text Files. In *EECS Department, Tufts University*.

Uijlings, J., Sande, K., Gevers, T., and Smeulders, A. (2013). Selective Search for Object Recognition. *International Journal of Computer Vision*, 104:154–171.

Wang, Y., Phillips, I., and Haralick, R. (2004). Table Structure understanding and its performance evaluation. *Pattern Recognition*, 37:1479–1497.

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., and Solomon, J. M. (2019). Dynamic Graph CNN for Learning on Point Clouds. *ACM Trans. Graph.*, 38(5).

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. (2017). Aggregated Residual Transformations for Deep Neural Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. In *ECCV 2014*. Springer International Publishing.

Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., and Ren, D. (2020). Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12993–13000.

Zhong, X., ShafieiBavani, E., and Jimeno Yepes, A. (2020). Image-Based Table Recognition: Data, Model, and Evaluation. In *ECCV 2020*, pages 564–580, Cham. Springer International Publishing.

Zhong, X., Tang, J., and Yepes, A. J. (2019). Publaynet: largest dataset ever for document layout analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1015–1022. IEEE.