

**DEEP LEARNING ON DOCUMENT LAYOUT ANALYSIS**

Liang Xu Chao

U1920092B

Supervisor: Loke Yuan Ren

School of Computer Science and Engineering

2023

Table of Content

**[Abstract 2](#_Toc31017)**

**[Acknowledgements 3](#_Toc22375)**

**List of Figure**

**[1. Introduction 5](#_Toc17788)**

**[2. Literature Review 6](#_Toc13757)**

2.1 [DLA Concepts 6](#_Toc30383)

2.2 Existing Solution 6

2.2.1 Detection Models

2.2.2 Training Tools

2.3 Document dataset

**[3. Implementation 14](#_Toc15272)**

[3.1 Design concept 14](#_Toc4161)

[3.2 Resource planning 15](#_Toc9114)

3.2.1 Hardware

3.2.2 Software Tools

3.2.3 Choice of model architecture

3.2.4 Choice of Dataset

3.3 Training implementation

3.3.1 Data Pre-processing

3.3.2 Model Training

**[4. Result summary 23](#_Toc26212)**

4.1 Evaluation Metric

4.2 Evaluation Result

4.3 Prediction Experiment

4.3.1 Prediction on Digital document

4.3.2 Prediction on Document in Photo Taking Images

**[5. Conclusion 34](#_Toc24961)**

5.1 [Project Summary 34](#_Toc32562)

5.2 [Project Limitations 34](#_Toc2184)

5.3 [Future Enhancements 34](#_Toc9911)

**[Reference 35](#_Toc14657)**

**Abstract**

Document layout analysis (DLA) involves understanding the physical structure of a document and breaking it down into multiple components. There are many different layout formats for documents, and a universal model to understand all of them is not yet available. DLA usually consists of pre-processing, model training, post-processing, and performance evaluation. This project will follow these four steps and experiment with building a model that can detect 11 different components of the layout of digital, scanned, or photographed documents.

Deep learning techniques will be used for model training and prediction. This project will use the Mask R-CNN approach, which is a Convolutional Neural Network for image segmentation and instance segmentation. The experimental results will be discussed, and it will be shown that the final model achieved an average Intersection over Union (IoU) of 86% on digital documents while also providing insightful detection on photos of documents.

To further enhance prediction accuracy, this project will introduce a second model that will be trained on manually labeled images to first locate the entire document in the image and then apply the prediction on the cropped area.

**Acknowledgements**

I would like to express my deepest appreciation to my supervisor, professor Loke Yuan Ren for guiding and keeping me stay on the right track in this final year project. I have pushed myself to achieve better result based on his suggestions. Without his help, this project would not be possible.

Liang Xu Chao

List of Figure

1. **Introduction**

Document Layout Analysis (DLA) can be considered as a form of object detection for images. However, it presents a unique challenge due to the varied and complex layouts found in different types of documents. DLA involves categorizing and detecting different regions in a document, and then applying different methods for detecting the content based on the type of region, such as text, table or figure.

**1.1 Motivation**

Existing solutions for DLA often perform well on one type of digital document, but struggle when presented with a photo of a document or a layout that is completely different. This motivated the need to explore existing techniques and train a model that can handle as many layouts as possible, regardless of whether they are digital or captured in a photo.

**1.2 Objectives and Scope**

The objective of this project is to build a generic model for document layout detection, which will require improvements in three key areas:

**1.2.1 Data Preparation**

To handle a wide range of document layouts, a substantial dataset of labeled images will be required. This dataset should include all major document components.

**1.2.2 Training Technique**

This project will explore different training tools and techniques to identify the most effective approach.

**1.2.3 Prediction Technique**

To improve the accuracy of predictions, this project will train a model to predict only on the document itself. Once the document has been predicted, the figure can be extracted and the generic layout prediction model can be applied to predict the layout.

1. **Literature Review**

The Literature Review will cover the relevant DLA concepts, review existing solutions and analyse available dataset for model training.

**2.1 DLA concepts**

DLA can be classified as a task of object detection for images. It is a changeling topic due to the complex layouts from different types of documents. Solutions for document analysis strategy often follow the Bottom-Up or Top-Down method. Bottom-Up strategy first locate local information and connected component to get the words then merge into line then merge into paragraph then into column. Top-Down strategy will start with analyse the global information of the document like the black and white stripes. It uses the information to break the document into different region and splitting them into paragraph then line then words.

To perform DLA usually will require following steps:

**Pre-processing**

Pre-processing is a crucial step in document layout detection, as it prepares the input image for further analysis. Pre-processing techniques involve image enhancement, noise reduction, and image binarization. Image enhancement techniques aim to improve the quality of the input image by adjusting its contrast, brightness, and sharpness. Noise reduction techniques are used to remove noise and artifacts from the input image, while image binarization techniques aim to convert the input image into a binary format, which makes it easier to detect and segment different components of the document.

**Model training**

A machine learning model is trained to recognize and classify different components of the document. One of the most popular machine learning models used for document layout detection is the convolutional neural network (CNN), which has shown excellent performance in detecting and classifying different components of the document. CNN-based models use a sliding window approach to scan the input image and classify each patch of the image as one of the document's components.

**Post-processing**

Post-processing techniques are used to refine the output of the machine learning model by removing false positives and improving the accuracy of the detected components. Post-processing techniques involve filtering, clustering, and merging of detected components. Filtering techniques are used to remove false positives by applying various rules and heuristics to the detected components. Clustering techniques group similar components together, while merging techniques combine adjacent components that belong to the same document component.

**2.2 Existing solutions**

**2.2.1 Detection models**

**Rule-Based Model**

One of the commonly used methods for DLA is the rule-based approach. In this approach, the rules are defined based on the characteristics of each document type to identify and extract the layout components. Xu et al. (2019) proposed a rule-based method for the detection of text regions in historical documents. The proposed method achieved an F1 score of 0.93 on the ICDAR 2017 dataset, which demonstrates the effectiveness of the rule-based approach in specific types of documents.

However, the rule-based approach has limitations as it heavily relies on the predefined rules and may not be suitable for complex layout structures. Therefore, researchers have proposed machine learning-based approaches that can automatically learn the layout features and predict the components of the documents.

**Conditional Random Fields (CRF)**

Another machine learning-based approach for DLA is the Conditional Random Fields (CRF). CRF is a probabilistic graphical model that can model the dependencies among the layout components. Chen et al. (2016) proposed a CRF-based approach for DLA that can detect text, image, and table regions in business documents. The proposed method achieved an F1 score of 0.83 on the ICDAR 2013 dataset, which demonstrates the effectiveness of the CRF-based approach in different types of documents.

**Convolutional Neural Network (CNN)**

CNN is a deep learning technique that has demonstrated outstanding performance in various computer vision tasks, including DLA. Gupta et al. (2016) proposed a CNN-based approach for DLA that can detect text, image, and table regions in scientific articles. The proposed method achieved an F1 score of 0.89 on the PubLayNet dataset, which consists of diverse layout structures in scientific articles.

**Fully Convolutional Network (FCN)**

FCN is a deep learning technique that can perform pixel-level prediction of the layout components. Hu et al. (2017) proposed an FCN-based approach for DLA that can detect text, image, and table regions in scientific articles. The proposed method achieved an F1 score of 0.91 on the PubLayNet dataset, which demonstrates the effectiveness of the FCN-based approach in complex document layouts.

**U-Net**

1. Net is another deep learning technique that can perform pixel-level prediction of the layout components. Ronneberger et al. (2015) proposed a U-Net-based approach for DLA that can detect text regions in forms. The proposed method achieved an F1 score of 0.93 on the ICDAR 2013 dataset, which demonstrates the effectiveness of the U-Net-based approach in specific types of documents.

**Faster R-CNN**

Faster R-CNN is an object detection architecture consisting of a Region Proposal Network (RPN) and a Fast R-CNN network. The RPN generates object proposals by sliding a window over a convolutional feature map and predicting objectness scores and bounding box coordinates. These proposals are then fed into the Fast R-CNN network, which classifies the proposals and refines their bounding boxes. Faster R-CNN has achieved state-of-the-art performance on benchmark datasets and is known for its ability to generate high-quality object proposals efficiently.

**Mask R-CNN**

Mask R-CNN is an advanced architecture of convolutional neural network (CNN) primarily used for performing image segmentation tasks. It is built upon Faster R-CNN, a well-known object detection model. Compared to Faster R-CNN, Mask R-CNN has an additional branch that predicts segmentation masks for each region of interest (RoI) in the image, in addition to the existing branches for classification and bounding box regression.

Mask R-CNN follows a similar two-stage detection process as Faster R-CNN. In the first stage, a set of candidate RoIs is generated using a Region Proposal Network (RPN). In the second stage, these RoIs are fed into a series of fully connected layers to perform classification and bounding box regression.

Apart from the classification and regression branches, Mask R-CNN also features a third branch dedicated to segmentation masks. This mask branch takes each RoI and produces a binary mask that outlines the object's shape within the RoI. The mask branch is a fully convolutional network that works on a coarse feature map of the RoI, and generates a mask with the same resolution as the input image.

Mask R-CNN has achieved remarkable performance on various benchmark datasets for instance segmentation tasks such as COCO, Cityscapes, and Pascal VOC. Its success is attributed to its ability to perform both object detection and instance segmentation simultaneously in a single framework, which simplifies the training and inference processes, and facilitates end-to-end learning of both tasks.

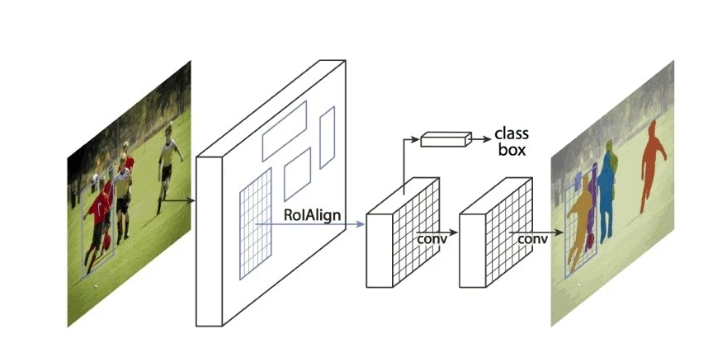


Figure 1: Mask R-CNN architecture

**YOLO**

YOLO (You Only Look Once) is a popular real-time object detection system that uses a single convolutional neural network (CNN) to simultaneously predict object classes and bounding boxes. Unlike other object detection systems that apply a sliding window approach to detect objects at multiple locations and scales, YOLO divides the input image into a grid of cells and predicts the object class probabilities and bounding boxes for each cell.

**2.2.2 Training tools**

**TensorFlow Object Detection API**

This is an open-source framework for building, training, and deploying object detection models. It provides pre-trained models for a wide range of object detection tasks and supports several popular object detection architectures such as Faster R-CNN, SSD, and YOLO.

**PyTorch**

This is another open-source deep learning framework that supports building custom object detection models. PyTorch offers a flexible and easy-to-use interface for designing and training deep neural networks and has become increasingly popular in recent years.

**Detectron2**

This is a modular object detection library built on PyTorch that provides a comprehensive set of pre-trained models and tools for training custom models. Detectron2 supports several state-of-the-art object detection architectures such as Faster R-CNN, Mask R-CNN, and RetinaNet.

**Keras**

This is a high-level neural network API that can run on top of TensorFlow, Theano, and other popular deep learning libraries. Keras provides an easy-to-use interface for building and training deep neural networks and includes pre-trained models for several popular object detection tasks.

**Caffe**

This is a deep learning framework developed by the Berkeley Vision and Learning Center. Caffe is particularly well-suited for image and video recognition tasks and supports several popular object detection architectures such as Faster R-CNN and SSD.

**Darknet**

This is an open-source neural network framework written in C and CUDA that supports a wide range of deep learning tasks, including object detection. Darknet includes an implementation of YOLO, a popular object detection architecture known for its speed and accuracy.

**Detectron2**

Detectron2 is considered as a next-generation object detection tool which introduce by Facebook’ AI research group. It is open-source in github based on the caffe2 framework which now is part of Pytorch. Detectron2 supports a number of computer vision research projects and production applications in Facebook.

**2.3 Document Dataset**

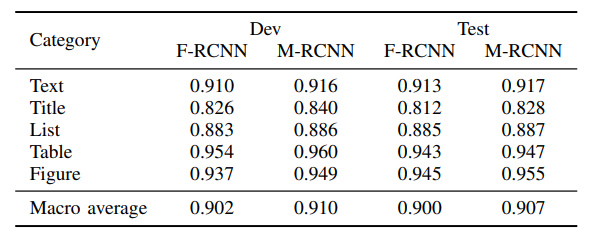
There are three focus on finding dataset for this project:

1. Should include majority of layout categories
2. Should include decent amount of training images
3. Must include accurate label data

**2.3.1 PubLayNet**

The PubLayNet dataset is designed for scientific papers and articles and includes 5 layout categories: text, list, figure, table, and title. It contains 335,703 training images, 11,245 validation images, and 11,405 test images. The dataset has become a popular choice for pre-training models in document layout detection, such as LayoutLMv3 and Layout Parser.

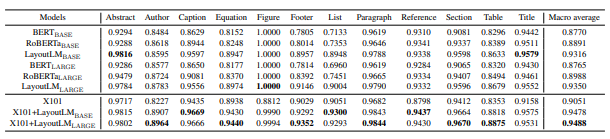
TABLE I: MAP @ IOU [0.50:0.95] of the F-RCNN and M-RCNN models



**2.3.2 DocBank**

The DocBank dataset contains 500,000 images that are categorized into 12 layout categories, including Abstract, Author, Caption, Equation, Figure, Footer, List, Paragraph, Reference, Section, Table, and Title. The dataset is focused on scientific papers and supports both text-based and image-based models. It is also expanding with automatic annotation methods. The availability of a large number of images and categories in DocBank makes it a valuable resource for research in document layout detection and related fields.

TABLE II: The performance of BERT, RoBERTa, LayoutLM and Faster R-CNN on the DocBank test sets



After reviewing the labeled data in DocBank, it was found that the dataset does not provide segmentation coordinates, indicating that the Mask R-CNN approach is not applicable to this dataset. This means that the instance segmentation task, which requires identifying the precise location and boundaries of objects within an image, cannot be performed with Mask R-CNN on this dataset. However, other methods such as object detection or classification may still be applicable depending on the specific task and goals of the project.

**2.3.3 DocLayNet**

DocLayNet is a dataset used for document layout analysis, containing 80,863 human-annotated images from diverse data sources, providing a wide range of layout formats. The dataset includes 11 layout categories, which are caption, footnote, formula, list-item, page-footer, page-header, picture, section-header, table, text, and title. The dataset is intended to be used for training and evaluating document layout analysis algorithms and models.

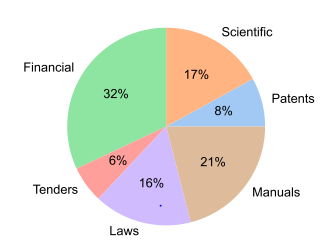


Figure 2: Distribution of DocLayNet pages across document

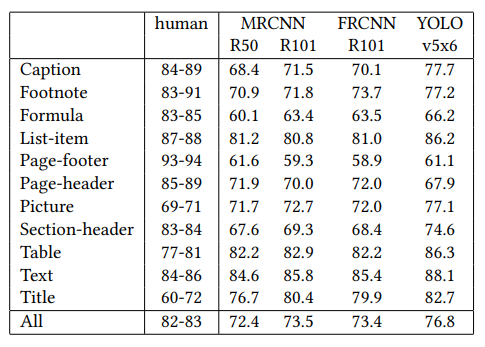
categories.

DocLayNet dataset has specific guidelines for annotation to ensure consistency and meaningfulness in the dataset. These guidelines include:

1. List items are considered as individual object instances, unlike PubLayNet and DocBank where grouped list paragraphs are considered as one list element.
2. A list-item is a paragraph with hanging indentation, and even single-line elements can qualify as list-items if the neighbor elements expose hanging indentation. Bullet or enumeration symbols are not required.
3. All captions are mapped to exactly one table or figure.
4. Figures that are connected will be considered as one whole figure.
5. Numbers are included in the formula object.
6. Italic and bold text alone in a new line are considered as section headers, but not at the beginning of the paragraph.

TABLE III: Prediction performance (mAP@0.5-0.95) of object

detection networks on DocLayNet test set.



2.3.4 WarpDoc

The purpose of the WarpDoc dataset is to train and evaluate machine learning algorithms for document image analysis tasks, including text detection and recognition. The dataset includes photo-taking document images captured from different angles, but it does not come with annotations provided by the source.

**3. Implementation**

This chapter will cover the design concept, resource planning and training implementation.

* 1. **Design concept**

To improve the accuracy of document layout prediction for complex and varied document images, this project proposes a novel two-step approach. The first step involves detecting the bounding box of the whole document, which is then cropped from the original image. The second step applies the prediction using a generic document layout model on the cropped bounding box.

In cases where no document is detected, the original image is passed through for further processing. This approach provides a more efficient way to predict document layouts for images with complex backgrounds or other non-document elements. By breaking down the prediction process into two steps and incorporating the use of the generic document layout model, this approach is expected to result in improved accuracy and performance for document layout detection.

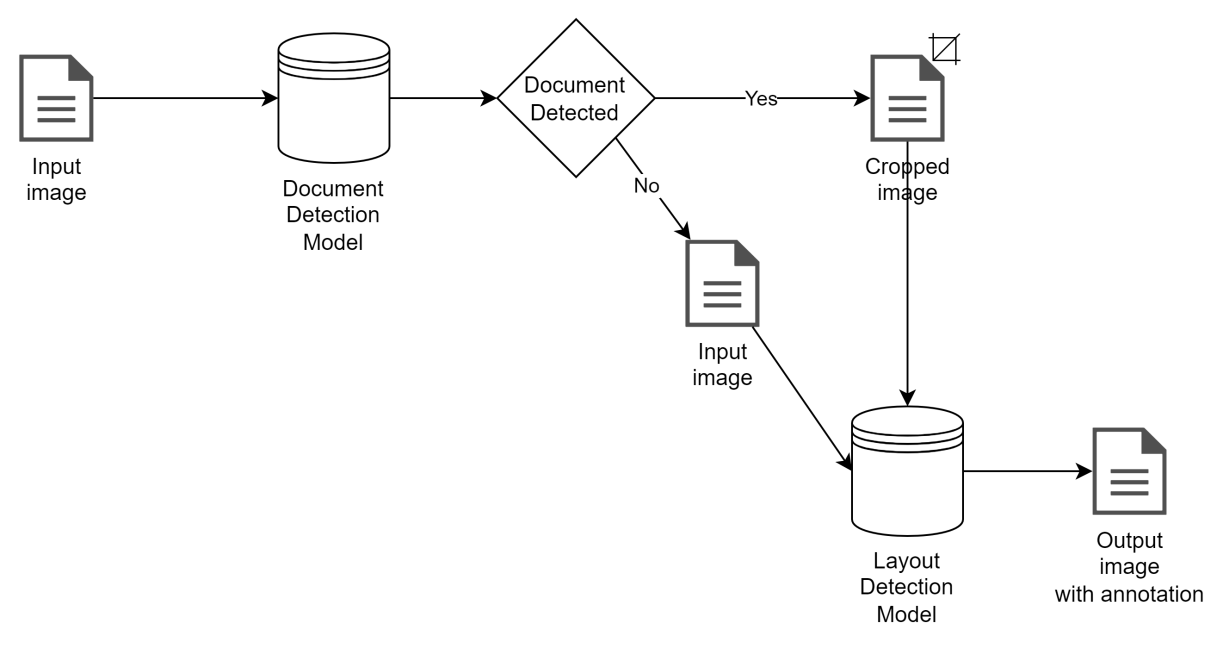


Figure 3: Document layout prediction work flow

* 1. **Resource planning**
     1. **Hardware**

|  |  |
| --- | --- |
| Device | Windows PC |
| RAM | 16GB |
| CPU | Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz |
| GPU | RTX 2060 @ 6GB RAM |

* + 1. **Software tools**

Based on the research finding, Detectron2 library will be used in the model training as it is considered a relative new library and many base-line models are provided by the Facebook official. Installation prerequisite as follow:

Python 3.7 or higher

PyTorch 1.8 or higher and torchvision match the PyTorch version

CUDA toolkit match PyTorch version

* + 1. **Choice of model architecture**

To achieve precise detection of document layouts, this project will utilize the Mask R-CNN method for model training. This approach is capable of handling both digital and scanned or photographed images, making it more effective than simply detecting the bounding box when the input images may be distorted or rotated.

To facilitate the training process, the Detectron2 library will be used, which offers various baseline models for Mask R-CNN architecture. Specifically, the R101-FPN baseline model will be utilized as it provides a good balance between training time and precision. This model has been pre-trained on a large-scale dataset and can detect objects at different scales, which is crucial for accurately detecting the various components of a document layout. Additionally, the use of this baseline model will reduce the need for extensive training and tuning, enabling the project to focus more on optimizing the model for document layout detection. Overall, the combination of the Mask R-CNN approach and the R101-FPN baseline model is expected to result in a highly accurate and efficient model for document layout detection.

TABLE III: Base Lines model provided by Detectron2 with Mask R-CNN

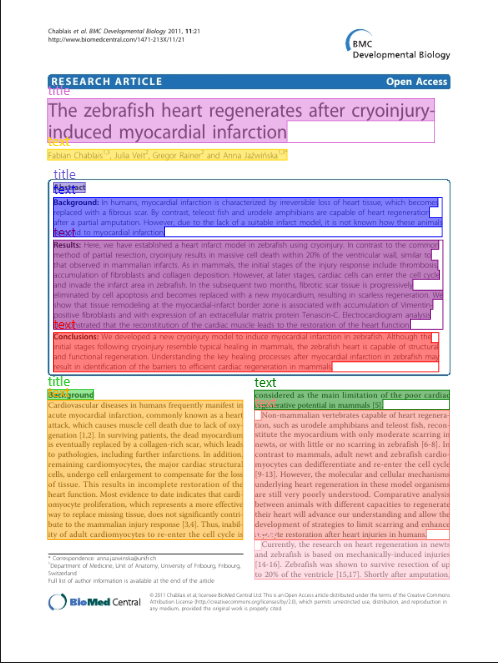


* + 1. **Choice of dataset**

**Layout detection Model**

The project has opted to use the DocLayNet dataset for training and testing data due to its reliable annotations, diverse document layouts, and reasonable number of images. This dataset's variety will enable the model to learn and detect various document layouts accurately. In contrast, the PubLayNet and DocBank models have limited effectiveness in detecting non-standard formats of research papers or articles.

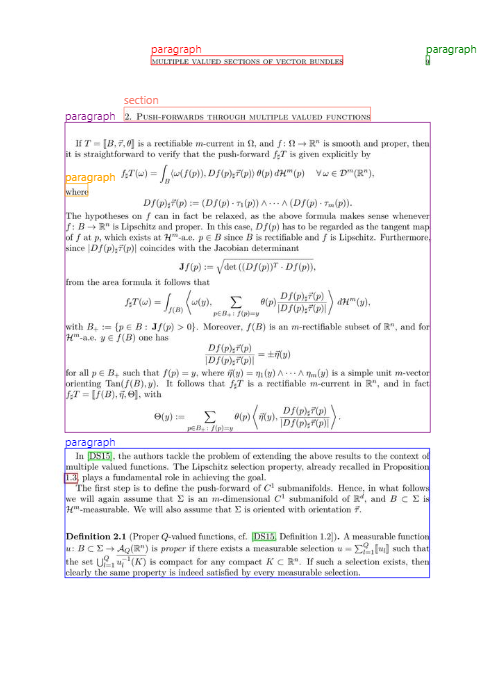
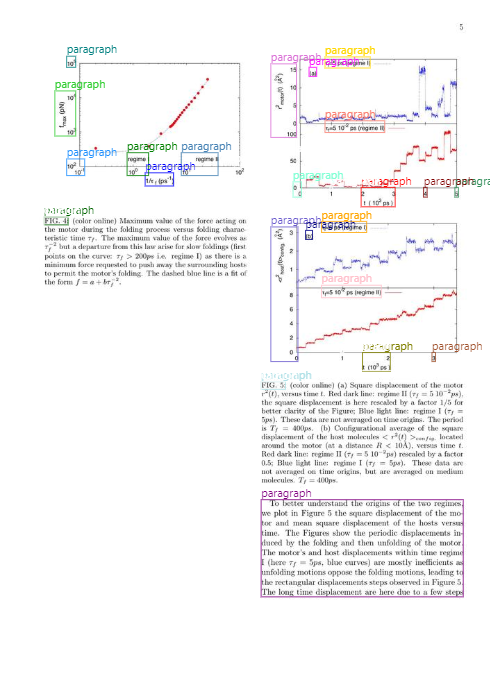
Furthermore, the dataset annotation in PubLayNet may not encompass all of the details presented in an image. An example of this is presented below.



Missing annotations

The absence of labels on the text and logo situated at the top and bottom of the document could result in inaccurate predictions.

The annotations in the DocBank dataset may not be entirely trustworthy, as some of the labels do not correspond accurately to the actual content. Samples of this are presented below.



The annotations are significantly inaccurate, rendering them unsuitable for training purposes. Upon reviewing a selection of samples at random, it became apparent that incorrect annotations are a frequent occurrence. Therefore, the DocBank dataset will not be used as a training dataset.

As a result, the DocLayNet dataset was deemed preferable due to its inclusion of a wide variety of document types, including manuals, financial reports, legal documents, and more. This diverse range of document layouts can aid in training the model to recognize a broader spectrum of layouts accurately. Utilizing the DocLayNet dataset will enable the model to perform more effectively in real-world scenarios, where document layouts and formats can vary considerably.

**Document detection Model**

Manual labeling photo taking images from warpdoc dataset.

**3.3 Training implementation**

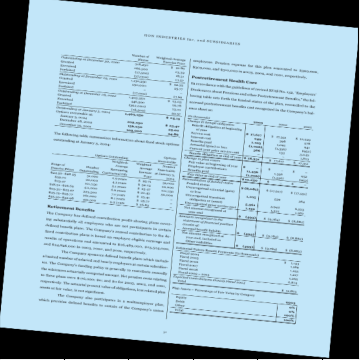
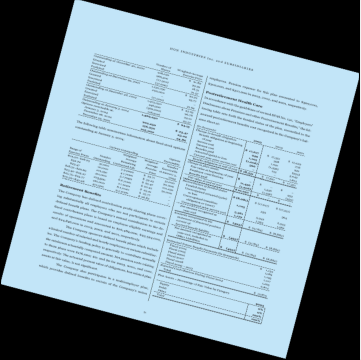
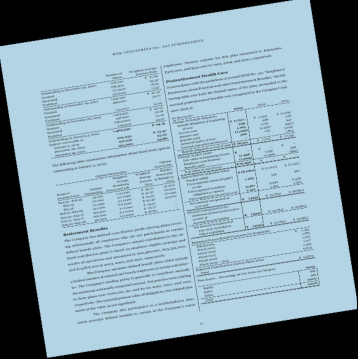
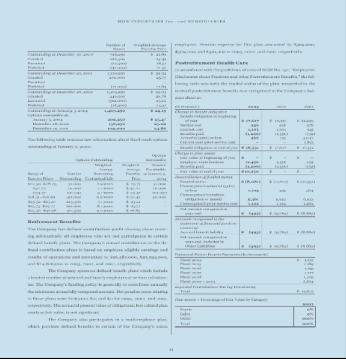
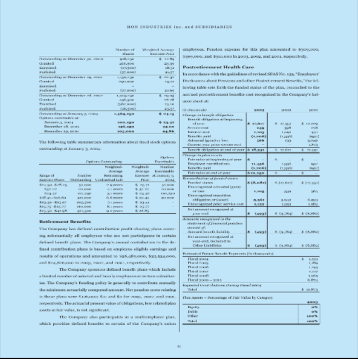
**3.3.1 Data pre-processing**

**Layout detection Model**

To enhance the model's ability to detect document layouts in photo-taking form, the original dataset, which consists only of digital documents, may not be sufficient. However, manually labeling and photographing at least 10,000 different documents is impractical. Instead, the training data can be augmented by transforming various samples randomly in each training iteration to simulate different textures and layouts.

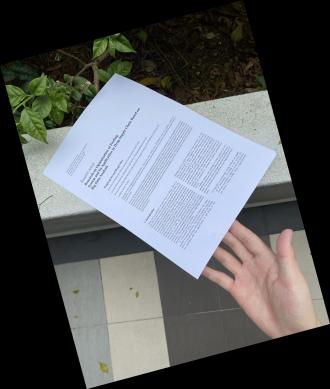
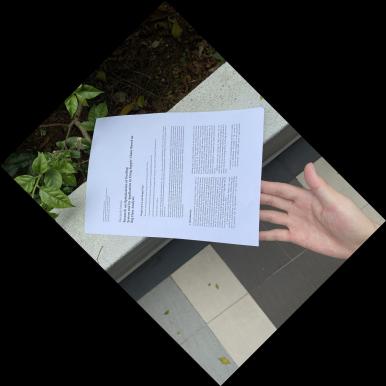
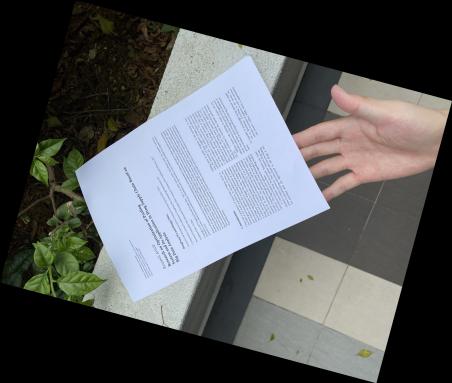
To achieve this, images can be randomly transformed by flipping 90 degrees, rotating between -15 and 15 degrees, and applying saturation, brightness, contrast, or lighting adjustments. Each transformation will have an equal probability of 0.2. This approach will help to increase the diversity and variability of the training dataset, allowing the model to learn and generalize better to new and unseen document layouts in photo-taking form. Further experimentation can be done to determine the optimal transformation combinations and probabilities for the dataset augmentation.

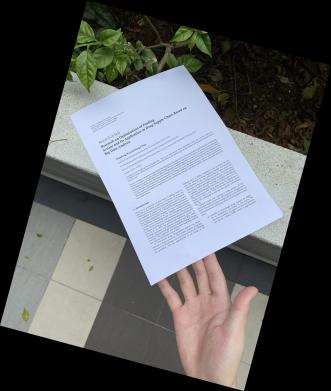
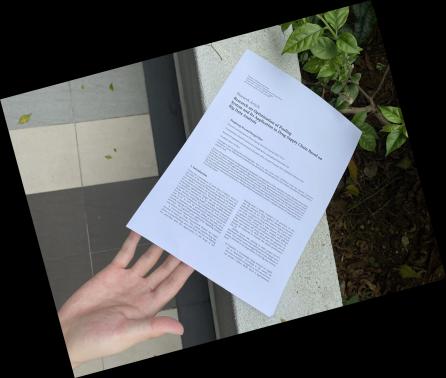
Below will be the illustration on the image transformation:



**Document detection Model**

To overcome hardware limitations, image transformations during training will cause out-of-memory errors. Therefore, the WarpDoc dataset is transformed beforehand. This dataset consists of photo-taking images, so only rotation transformations are applied. The following is an illustration of the image transformation:

**3.3.2 Model Training**

**Layout detection Model**

Due to the limited hardware resource, the model was break down to batches and each batch was evaluate using the test set and sample of photo taking images.

The training set contains 69375 images and below is total number of item in 11 categories.

|  |  |
| --- | --- |
| Category | AP (over 10 threshold)  bbox |
| Caption | 19218 |
| List-item | 161818 |
| Picture | 39667 |
| Text | 431251 |
| Footnote | 5619 |
| Page-footer | 61313 |
| Section-header | 118590 |
| Title | 4437 |
| Formula | 21167 |
| Page-header | 47973 |
| Table | 30070 |
| Total | 941123 |

All the training configuration is set based on guideline with adjustment on hardware constraint and custom implementation.

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Remark |
| Base line model | mask\_rcnn\_R\_101\_FPN\_3x |  |
| Number of Class | 11 | The total number categories will able to detect in the model |
| Image per Iteration | 1 | Due to hardware constraint, each iteration will only take 1 input image |
| Iteration | Per training >= 60000  Final iteration = 1460000 | Due to hardware constraint, the training process will be break down to many batches. In order to fit as many training data in each batch, 60000 iteration is set as only 1 image is able to fit in 1 iteration.  Based on the evaluation data, the model will not improve significantly on detecting all categories after 140000. Thus, it is stopped in iteration 1460000 |
| Learning Rate | 0.00025 |  |

**Document detection Model**

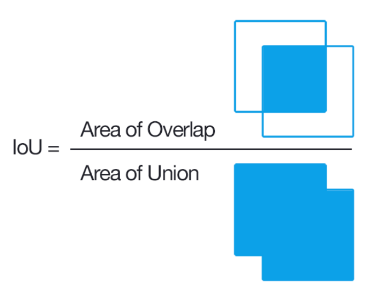
|  |  |  |
| --- | --- | --- |
| Parameter | Value | Remark |
| Base line model | mask\_rcnn\_R\_101\_FPN\_3x |  |
| Number of Class | 1 | Only 1 category which is ‘document’. |
| Image per Iteration | 1 | Due to hardware constraint, each iteration will only take 1 input image |
| Iteration | 80000 | First batch training is set to 80000 and the model will not improve significant over 50000. Thus, stop in first batch. |
| Learning Rate | 0.00025 |  |

1. **Result summary**

This chapter will discuss on the performance of the model and how the prediction process mentioned in Chapter 3.1 help to improve the accuracy on detecting layout from photo taking images.

**4.1 Evaluation Metric**

Intersection over Union (IoU) is measure the overlap of the predicted area and the ground truth area.



The evaluation result metric will use the IoU value to calculate the Average Precision (AP) for each category. The AP will use the 10 threshold IoU value: 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95 and average the output value.

**4.2 Evaluation Result**

**Layout detection Model**

The test set contains 6489 images and below is total number of item in 11 categories.

|  |  |
| --- | --- |
| Category | AP (over 10 threshold)  bbox |
| Caption | 1763 |
| List-item | 13320 |
| Picture | 2775 |
| Text | 49186 |
| Footnote | 312 |
| Page-footer | 5571 |
| Section-header | 15744 |
| Title | 299 |
| Formula | 1894 |
| Page-header | 6683 |
| Table | 2269 |
| Total | 99816 |

The Mask R-CNN model generates two types of prediction outputs: bounding boxes (bbox) and mask segmentation.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| AP (over 10 threshold)  bbox | AP50  bbox | AP75  bbox | AP (over 10 threshold)  segmentation | AP50  segmentation | AP75  segmentation |
| 63.598 | 86.445 | 71.208 | 64.655 | 86.420 | 72.469 |

TABLE IV: Overall Average Precision

|  |  |  |
| --- | --- | --- |
| Category | AP (over 10 threshold)  bbox | AP (over 10 threshold)  segmentation |
| Caption | 71.970 | 72.100 |
| List-item | 64.659 | 67.131 |
| Picture | 77.225 | 77.699 |
| Text | 77.603 | 79.288 |
| Footnote | 55.656 | 55.761 |
| Page-footer | 48.761 | 50.034 |
| Section-header | 57.685 | 58.545 |
| Title | 53.664 | 55.513 |
| Formula | 52.739 | 53.937 |
| Page-header | 62.205 | 62.780 |
| Table | 77.415 | 78.413 |

TABLE III: Average Precision on the 11 categories.

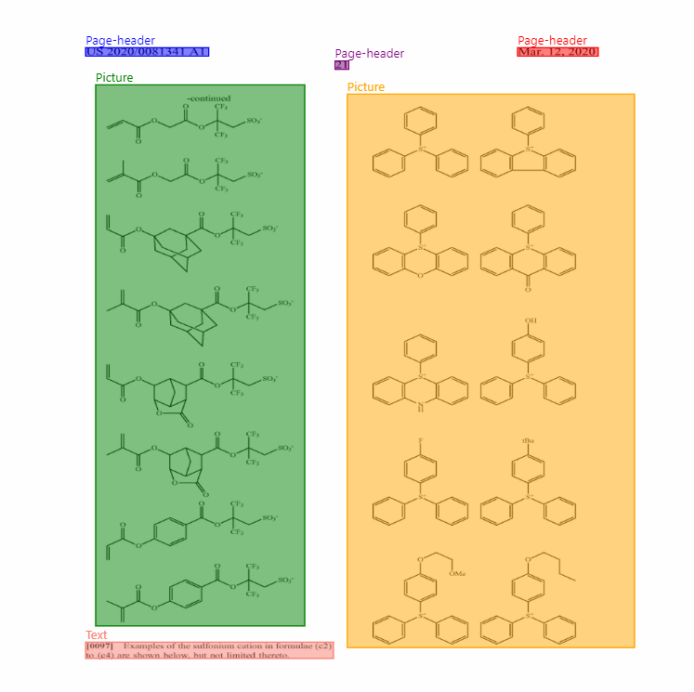
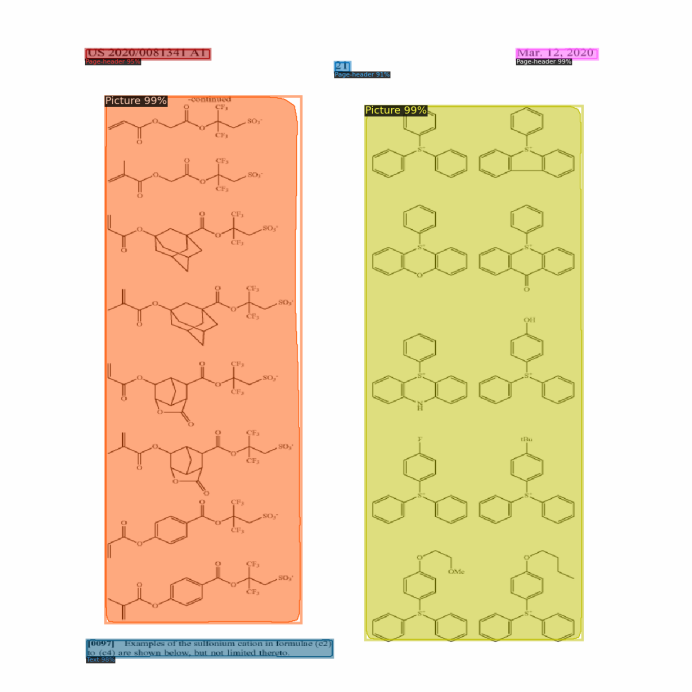
**4.3 Prediction experiment**

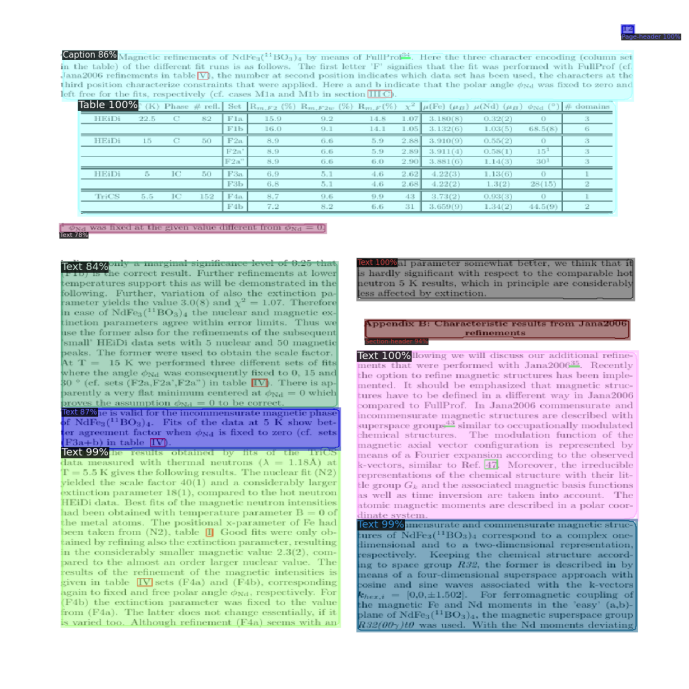
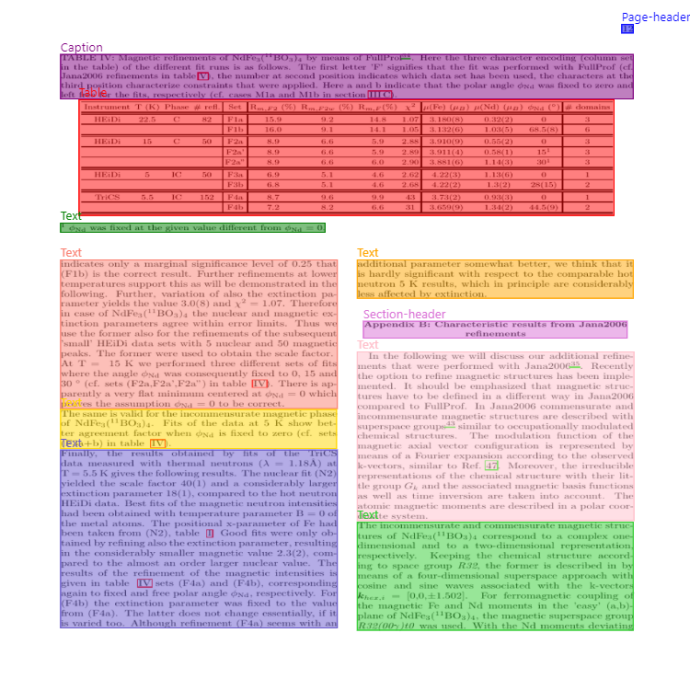
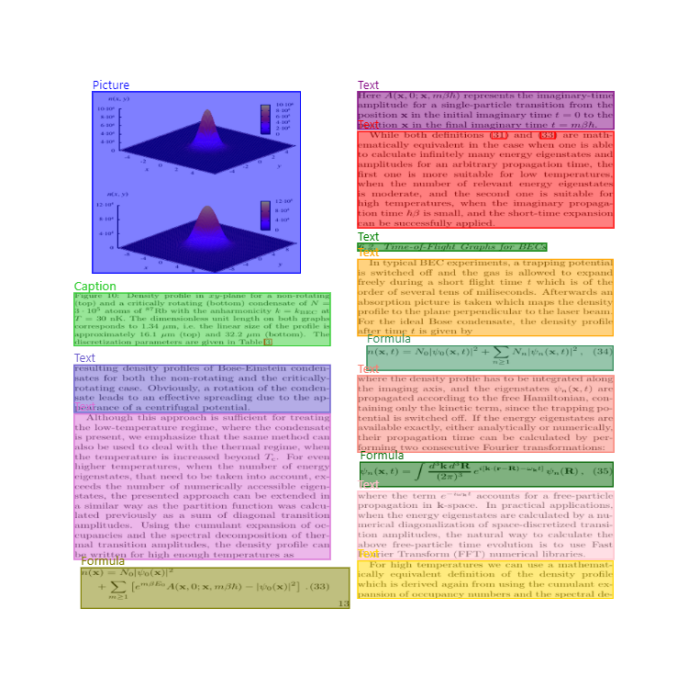
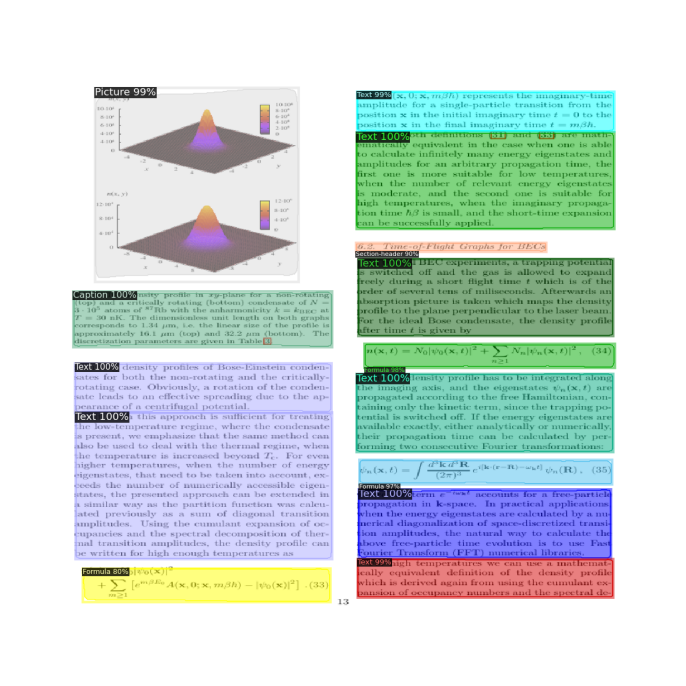
**4.3.1 Prediction on Digital document**

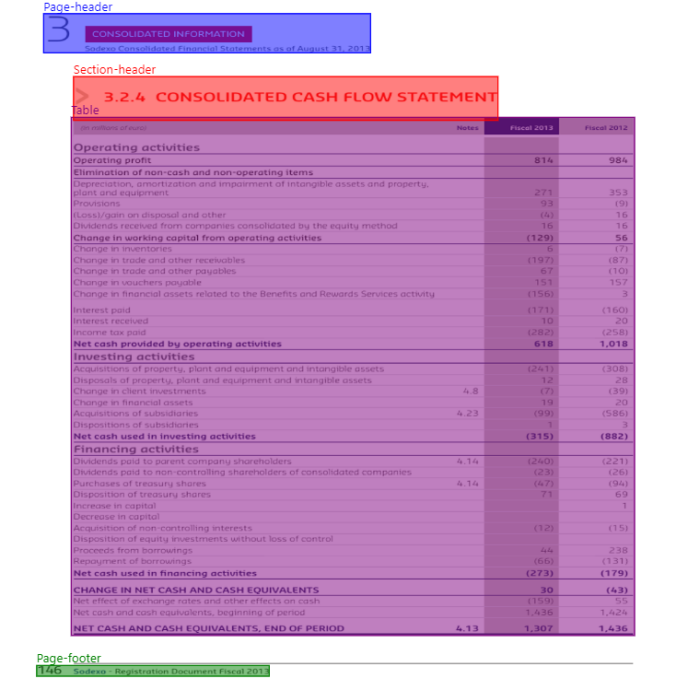
**DocLayNet**

Below are some examples of the model's predictions on the digital document test set, which includes all 11 categories.

Predict Actual



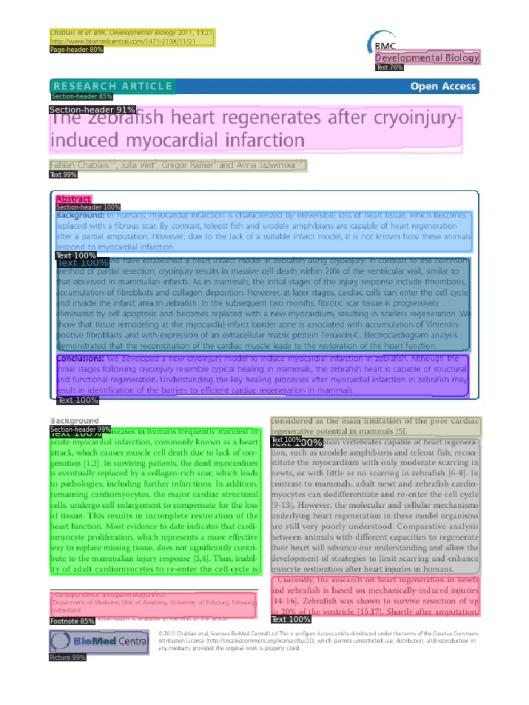
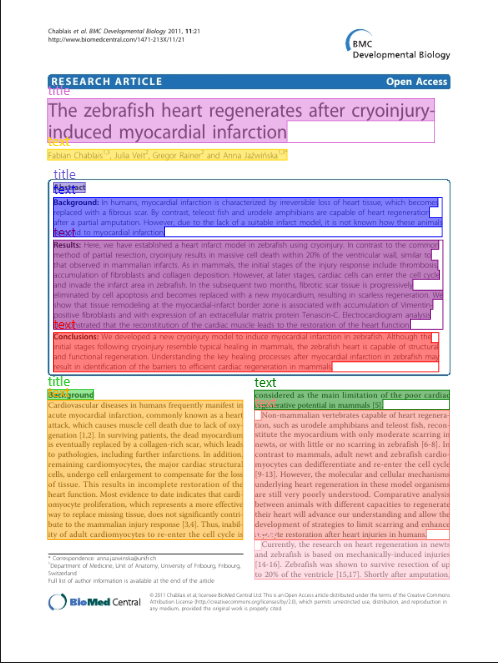




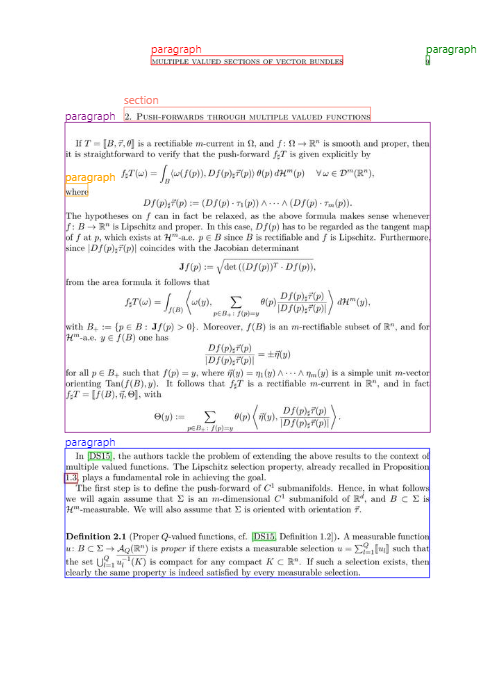
The model demonstrates strong performance in predicting layout from digital documents within the DocLayNet test set. In order to assess the model's ability to detect layout in a variety of digital documents, it was also tested on the DocBank and PubLayNet datasets. However, due to differences in the number of categories and annotation styles among these three datasets, it is not possible to calculate mAP. Instead, the results must be analyzed through observation. Sample predictions and actual annotations comparison are provided below.

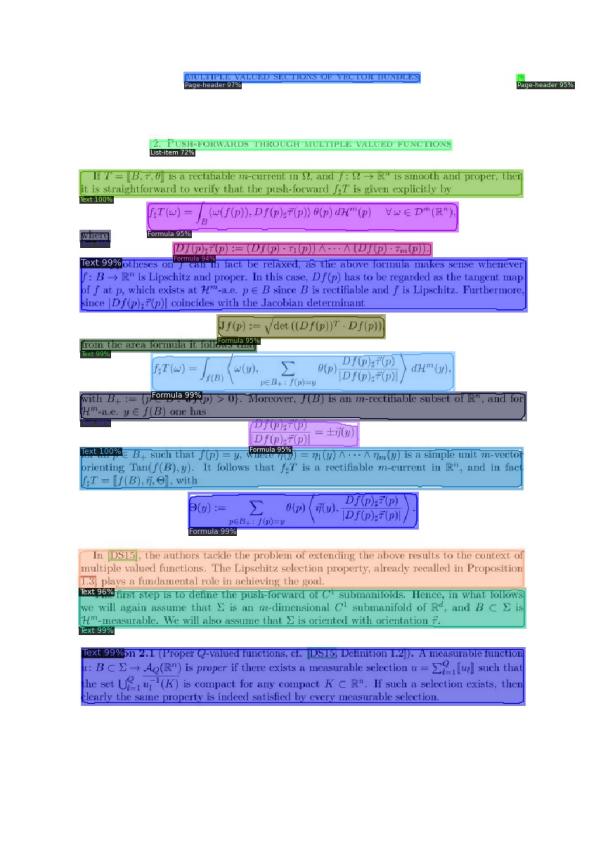
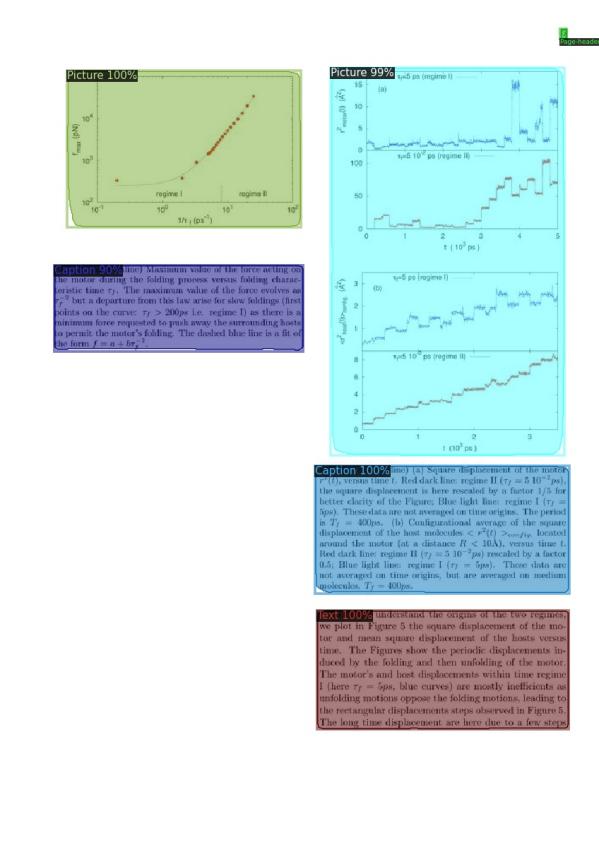
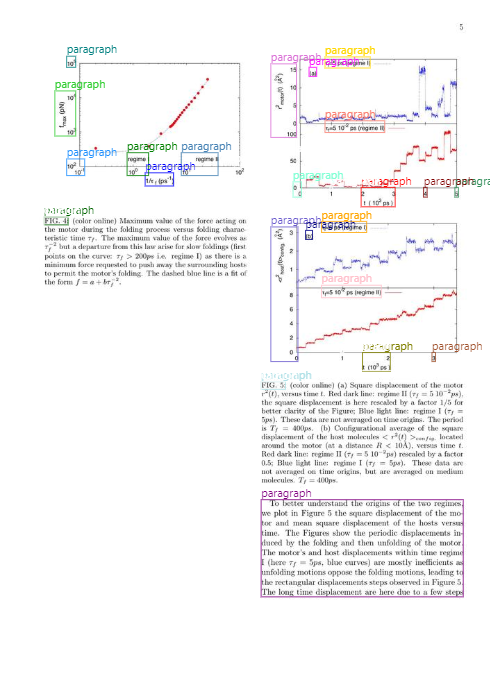
**PubLayNet**

Predict Actual



**DocBank**



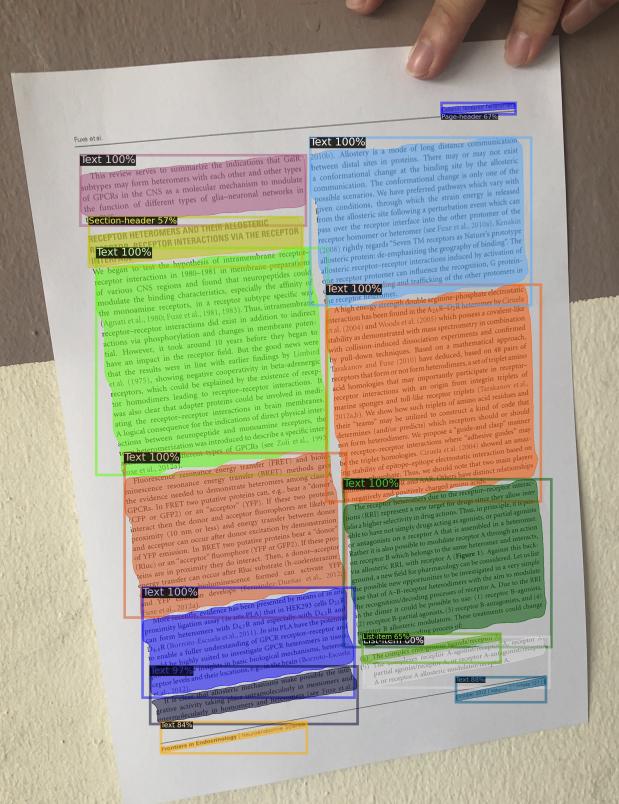
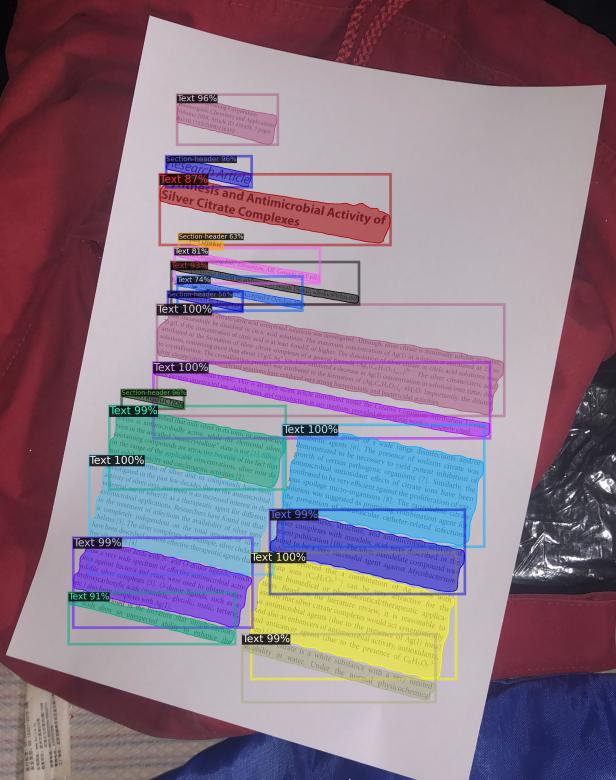
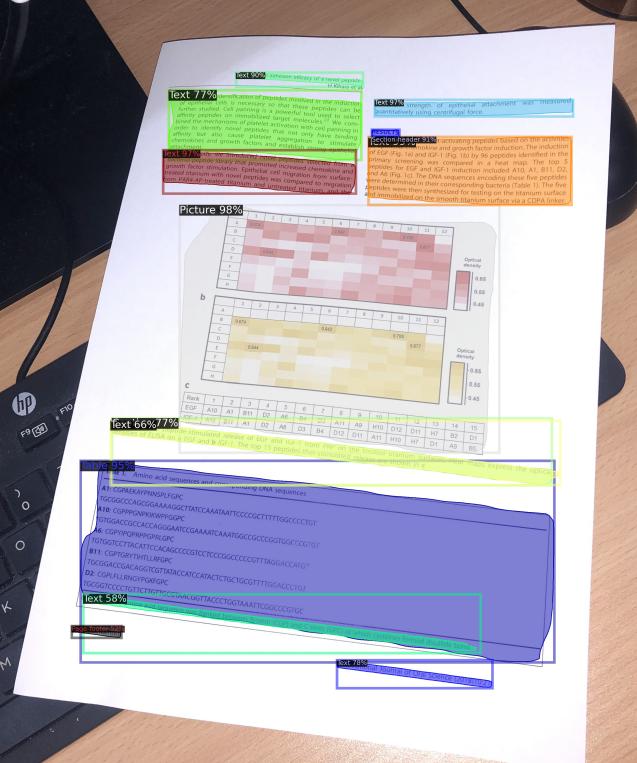
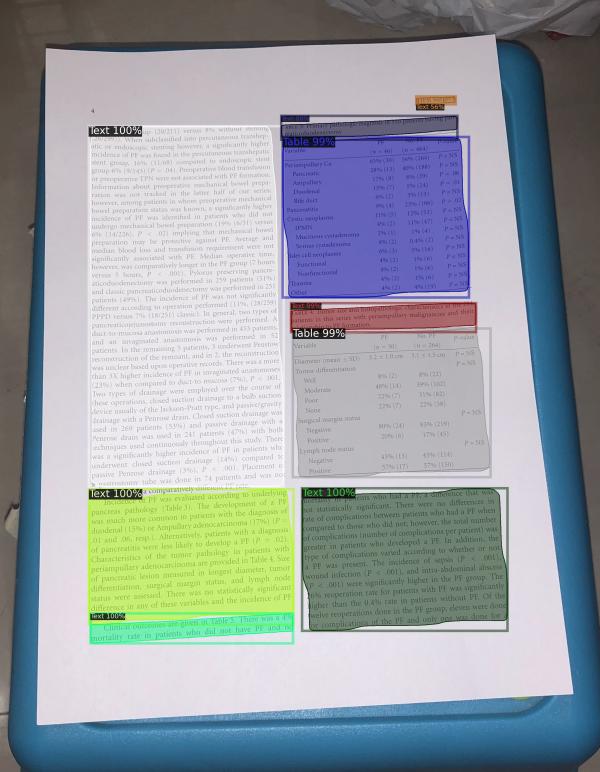


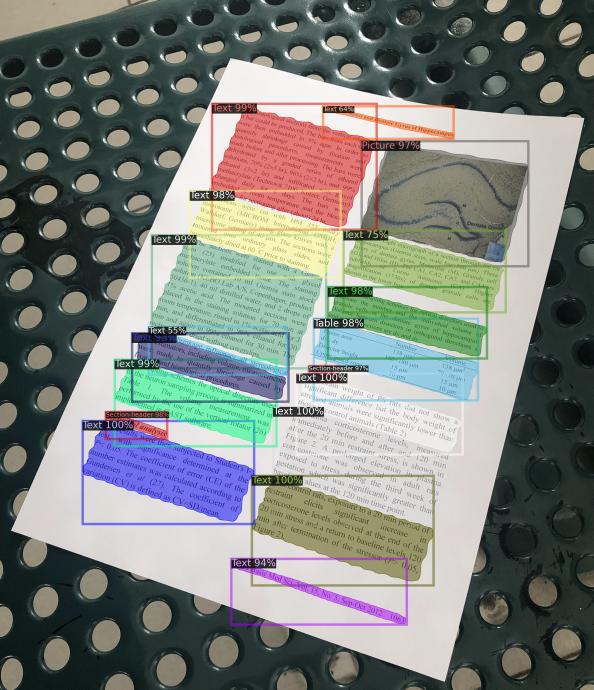
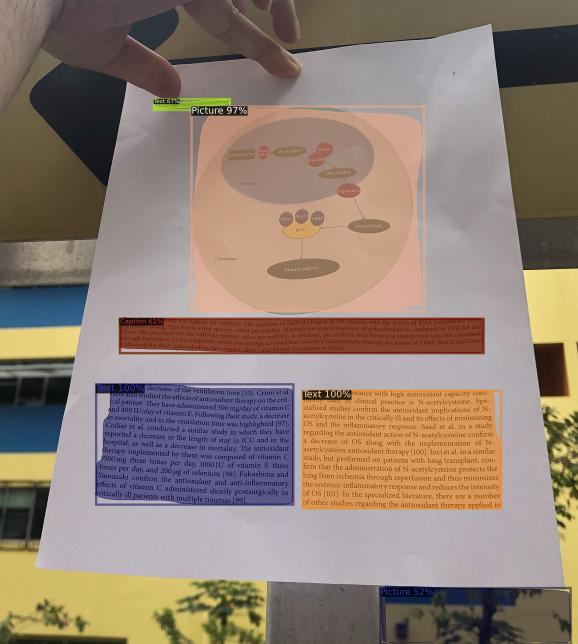
The prediction results accurately identify the layout contents and contain more information than the annotations from PubLayNet. Furthermore, the predictions for the DocBank dataset offer more precise and detailed information compared to the actual annotations.

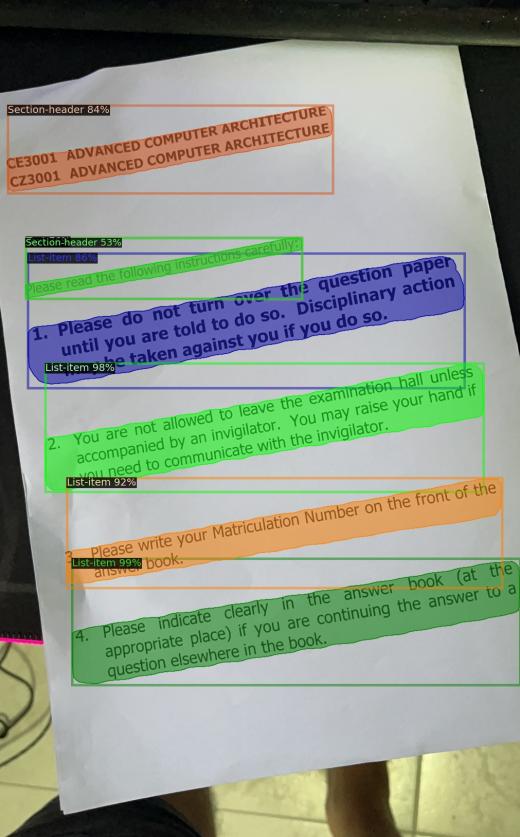
**4.3.2 Prediction on Document in Photo Taking Images**

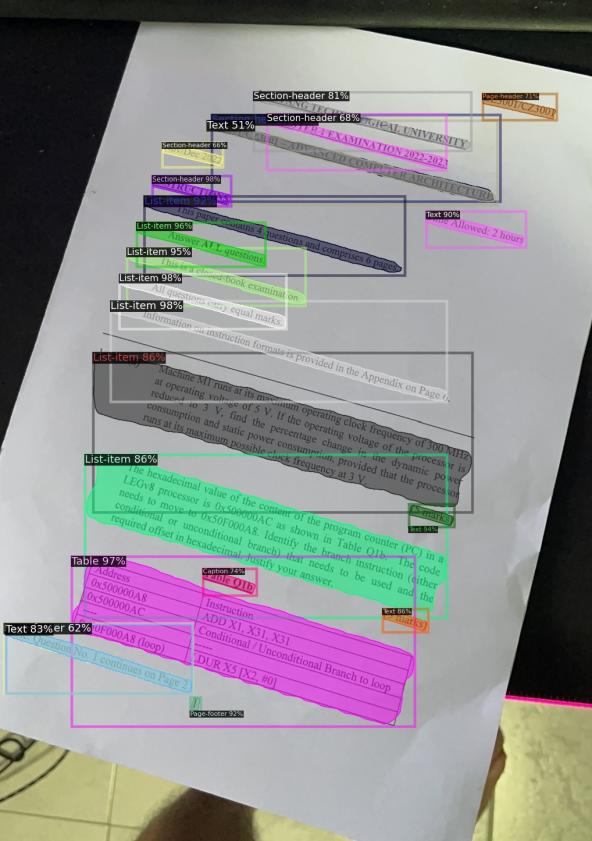
As mentioned in Chapter 3.1, this project included a prediction model for detecting the entire document and then perform the layout detection.

Layout Detection Samples

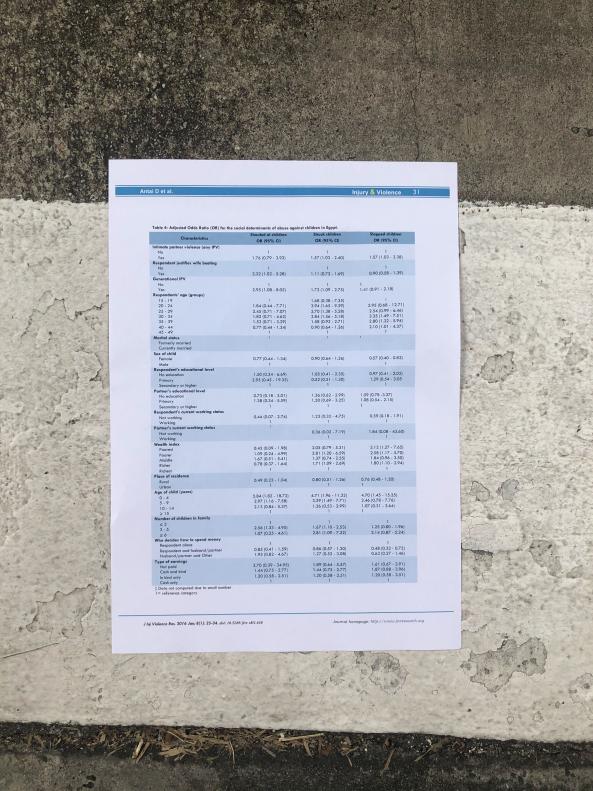


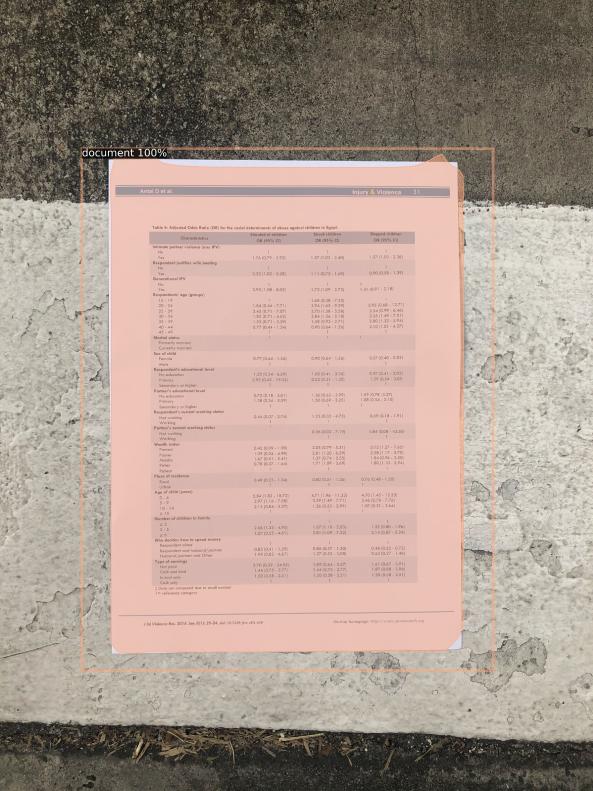




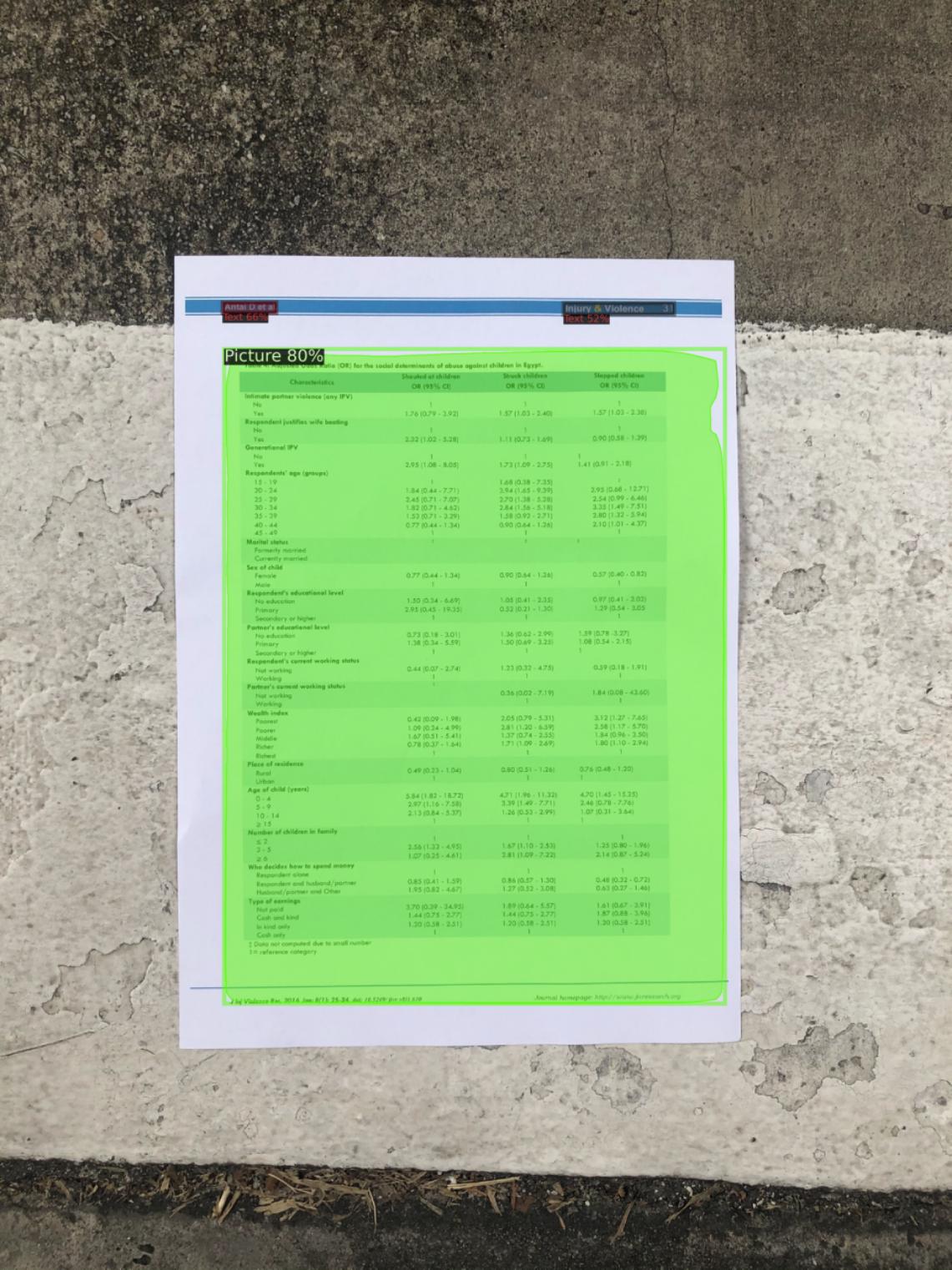


The below is an example show the detection of the document and the improvements when using the layout model to predict on the cropped document image.

Original Image Detection for Document

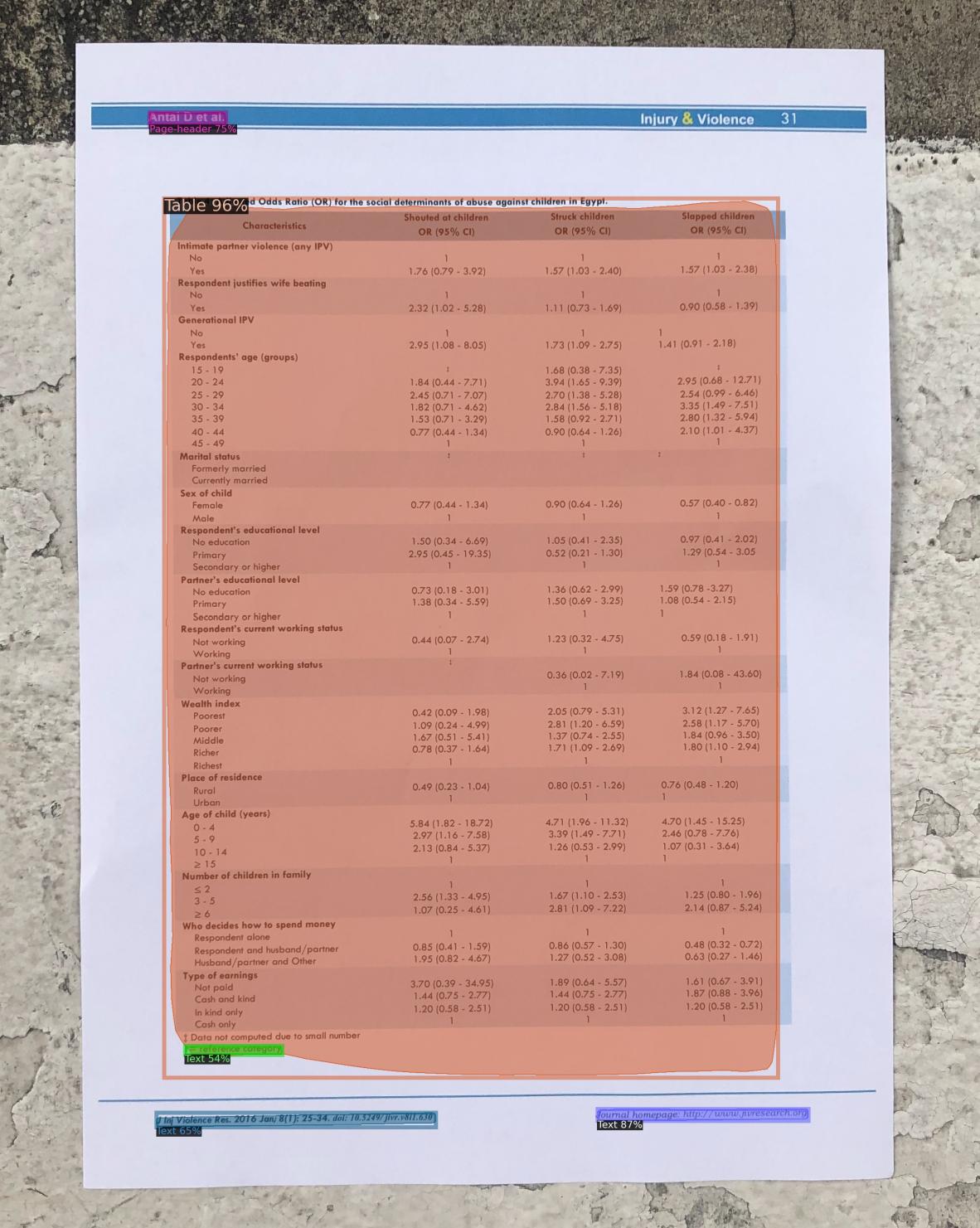


Prediction on original image using the Layout Model



The Layout Model is capable of generating predictions for the image, but its accuracy is limited as it classifies table content under the Picture category and includes the text at the bottom as part of the Picture.

Prediction on Cropped Document



The model's performance is improved when predicting on the cropped document, as it is able to correctly detect the table and label the text in the footer.

To sum up, the Layout Model attains a mean average precision of 64% across the 10 IoU thresholds ranging from 0.5 to 0.95, and at 0.5 IoU, the model yields an average precision of 86% using the test dataset. Additionally, with the aid of the Document Model, the Layout Model can also make reliable predictions for the 11 categories components in the layout of photo-taking document images.

1. **Conclusion**

**Project Summary**

The primary objective of this project was to develop an efficient and accurate document layout detection model using deep learning techniques, specifically the Mask R-CNN approach. The final model achieved an overall average precision of 86%, with each category having a minimum average precision of 45%. The project aimed to build a model capable of detecting layouts in various forms of documents, including digital, scanned, or photo form documents. The selection of dataset, training, and prediction techniques was made to achieve this goal. Results of the experiments are shared and discussed for future improvements and research in the field of document layout detection.

**Project Limitations**

One of the primary limitations of the project was the difficulty in adding photo-taking images to the dataset due to a lack of human resources available for labeling the samples for training. Although the model is designed to detect any type of document, it is more suitable for research papers or articles. The model may require additional fine-tuning for other types of documents, which can be further explored in future studies.

**Future Enhancements**

To enhance the accuracy and efficiency of document layout detection, future studies may consider introducing new models, each targeting a different document type such as magazines, exam papers, or comic books. The application of model assembly methods can be utilized to provide the most accurate result by utilizing the strengths of each model. Moreover, exploring the use of transfer learning techniques and the inclusion of more diverse datasets can also improve the model's performance and extend its applicability.

**Reference**

[1] Viso.ai. "Mask R-CNN." [Online]. Available: https://viso.ai/deep-learning/mask-r-cnn/#:~:text=%2DR%2DCNN.-What%20is%20Mask%20R%2DCNN%3F,Region%2DBased%20Convolutional%20Neural%20Network. [Accessed: Feb. 27, 2023].

[2] IBM Developer. "PubLayNet." [Online]. Available: https://developer.ibm.com/exchanges/data/all/publaynet/. [Accessed: Feb. 27, 2023].

[3] C. Van Beusekom and D. Keysers, "A Survey of Document Layout Analysis," ACM Journal on Computing and Cultural Heritage, vol. 14, no. 4, pp. 1-31, 2021. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3534678.3539043. [Accessed: Feb. 27, 2023].

[4] DocBank, "DocBank," [Online]. Available: https://doc-analysis.github.io/docbank-page/. [Accessed: Feb. 27, 2023].

[5] SG-VILab. "WarpDoc." [Online]. Available: https://sg-vilab.github.io/event/warpdoc/. [Accessed: Feb. 27, 2023].

[6] V7Labs. "Mean Average Precision (mAP) in Object Detection." [Online]. Available: https://www.v7labs.com/blog/mean-average-precision#:~:text=Average%20Precision%20is%20calculated%20as,mAP%20varies%20in%20different%20contexts.. [Accessed: Feb. 27, 2023].

[7] COCO. "Detection Evaluation." [Online]. Available: https://cocodataset.org/#detection-eval. [Accessed: Feb. 27, 2023].

[8] Appen. "PubLayNet Data Preview." [Online]. Available: https://dax-cdn.cdn.appdomain.cloud/dax-publaynet/1.0.0/data-preview/index.html. [Accessed: Feb. 27, 2023].

1. A. Littlepain, "Simple Understanding of Mask RCNN," Medium, 20-May-2020. [Online]. Available: https://alittlepain833.medium.com/simple-understanding-of-mask-rcnn-134b5b330e95. [Accessed: Feb. 27, 2023].