Fast, Accurate and Resource scarce Paragraph Object Localization in Document Centric data

A report submitted to

RAMAIAH INSTITUTE OF TECHNOLOGY

Bengaluru

ISIN INTERNSHIP

as partial fulfillment of the requirement for

Bachelor of Engineering (B.E) in Information Science and Engineering

by

B. Adithya Rao
(USN- 1MS16IS131)

under the guidance of

INDUSTRY GUIDE

Mr Titus Thomas

Senior R&D Engineer

Stride.AI

INTERNAL GUIDE

Mrs Rajeshwari S.B

Associate Professor

Dept of ISE, RIT

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

RAMAIAH INSTITUTE OF TECHNOLOGY
May 2020

Department of Information Science and Engineering

Ramaiah Institute of Technology

Bengaluru – 54

CERTIFICATE

This is to certify that dissertation work entitled "**Fast,Accurate and Resource scarce Paragraph Object Localization in Document Centric data"** is carried out by B. Adithya Rao **[USN- 1MS16IS131],** a bonafide student of Ramaiah Institute of Technology, Bangalore, in partial fulfillment for the award of Bachelor of Engineering in Information Science and Engineering of the Visvesvaraya Technological University, Belgaum, during the year 2019-2020. The thesis has been approved as it satisfies the academic requirements in respect to dissertation work prescribed for the said degree.

(Internal Guide)
Rajeshwari S.B
Associate Professor, Dept. of ISE, RIT

(Head of the Department)
Dr. Vijay Kumar B P
Professor & Head, Dept. of ISE, RIT

Acknowledgement

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends.

A number of personalities, in their own capacities have helped me in carrying out this internship work.

I would like to take this opportunity to thank them all. I am grateful for Stride.AI, for giving me an opportunity to work on cutting edge industry technology, while also training me to become a better developer.

I would like to thank my guide, Titus Thomas, for his guidance and support throughout my internship.

I would also like to thank Ms. Rajeshwari S B. , M.Tech, (Ph.D.),Department of Information Science, MSRIT for taking the time to review and grade this project.

I would like to thank Dr. N. V. R. Naidu, Principal, Ramaiah Institute of Technology, Bengaluru, for his moral support. I would also like to thank Dr. Vijaya Kumar B P, Head of Department, Information Science of Engineering, Ramaiah Institute of Technology, Bengaluru, for his valuable suggestions and expert advice. My thanks to all the faculty members of Department of Information Science & Engineering at Ramaiah Institute of technology for their constant encouragement and inspiration.

I am also thankful to my family and friends for their continuous support.

Abstract

With the RPA (Robotic Process Automation) space becoming popular over the last few years, there are a multitude of different tasks that are repetitive, and manual in nature. These tasks can be automated through the use of Machine Learning, and Intelligent Heuristics that need minimal data to start producing results.

A subspace within the RPA space deals with the automated analysis and querying of large documents, typically resembling that of a Financial Report, A tender specification document, A KYC form, and other such information rich documents. Named Entity Recognition, Natural Language Querying, and other such tasks are commonplace within this space as well.

Before Modelling any of the above tasks, it becomes imperative to convert a design rich document such as PDF or a Document file, into a text rich format that models can understand and learn from, such as a series of strings or numbers.

While OCR is primarily used to obtain text from a design rich document, its accuracy dwindles due to the presence of figures,logos, and complicated templates. It is required that a data driven approach is made use of to identify free flowing text regions, like paragraphs.

This report gives the detailed description of a Data Driven, Platform Agnostic, and Low Resource pipeline that accepts a design rich document as input, and returns a detailed list of accurately detected and localized paragraphs, along with the text contained within them as the output.

This Project has been overseen and implemented with the resources provided by Stride.AI.

Internship Completion Certificate

Stride AI R&D Pvt Ltd

India : No. 1, A Block, 1st floor, CIL Layout, Sanjaynagar Main Road, Bengaluru - 560094

Date : Sept 02 2019

Certificate of Internship

This is to certify that Adithya Narayan completed his internship at Stride AI, from June 10th, 2019 to August 10th, 2019.

We have found him to be a self starter who is motivated and hard-working. He worked sincerely on his assignment and his performance was excellent.

We wish him all the best for a bright future.

Warm Regards,

R. Sendhil Kr

Sendhil Kumar
COO, Stride AI

Table of Contents:

List of Figures

List of Tables

About the Industry(Internship Workplace):

Stride.AI is a company that operates in the RPA (Robotic Process Automation) space, primarily catering to the needs of financial industries like banks and credit rating agencies.

Robotic Process Automation helps companies from 2 fronts:

1. Makes processes simpler, even obsolete through AI/Heuristic driven solutions, and thereby boost productivity as and when possible.

2. Reduce costs in Employment by making manual, repetitive tasks a non human dependent one.

Most of Stride.AI's automation work flow revolves around the analysis of Unstructured Financial Documents, usually in the form of a PDF, with the aim of building SaaS and PaaS solutions for automated data extraction/localization and NLU(Natural Language Understanding) models/Pattern based heuristics for Natural Language Querying.

Stride.AI's primary clients include a multitude of International Banks headquartered in Brazil, Europe, Bangalore, Japan etc.

Solutions are usually deployed as full stack applications, with an inhouse SDK used for PDF and Document processing, and trained models for Entity Extraction, Document Clustering, and Information Extraction.

The company places emphasis on research, especially in the fields of Data Scarce Natural Language Processing and Document Data Analysis.

Introduction:

1.1 Problem Statement:

In order to build Machine Learning models for any sort of task, it is first nessecary that the data be in a format that the computer understands. These formats can include strings, numbers, images,documents, videos, music etc, all encoded in their respective file formats, like jpg, mpeg, mp4, pdf.

It is observed that for building text driven models, training data has to be in the form of a string datatype, perhaps with some other labels encoded together as a tuple. Therefore, Documents with no sort of preprocessing cannot be incorporated as training data, or testing data for that matter in the process of building a text driven ML model, as they are of a completely different file structure, and are usually collections of figures, tables, design templates and free flowing text with varying fonts and sizes.

It is thus imperative to detect, localize and then extract text out of any sort of document and pdf available before using to build a text driven model.

While heuristic based solutions, such as contour detection are available, they tend to produce too many false positives, and as a result, make text extraction substantially slower than average. An improvement was needed, both in terms of speed, and accuracy of letters being detected and transcripted respectively. This was the problem statement of the internship in question.

1.2 Scope and Objectives:

The objectives for this task include:

a. Building a data driven,platform agnostic, format agnostic, free flowing text detection API in python for commercial documents.

b. Ensuring the API built is scalable, and can handle batch requests with as little of a response time as possible.

c. Should Integrate with the inhouse SDK emphasizing on minimal coupling with other heuristic based document processing functions

d. Should be multilingual, and handle a variety of fonts and designs present on the document.

e. Should provide detection,localization, and extraction responses in a machine readable format like JSON for each instance of a detected paragraph or heading.

f. Should provide an increase in localization accuracy, and decrease in processing time per PDF.

The applications and scope of this API cater primarily to document parsing softwares like Adobe, CamScanner etc that may use this as a preprocessing tool for inter-document copy pasting.

It can also be used to train text driven models directly on document data with file extensions including .docx, .odt, and .pdf files, thus reducing redundancy during data collection and preprocessing phases of the Model deployment pipeline.

On the same lines, this API can be used as a preprocessing module for tasks dealing with document centric processing.

Due to the API being multilingual, one can use the API on even Dutch, French,and Hindi documents with their native script, as a transliteration module can be used to convert this script to roman.

1.3 Proposed Model:

The task of identifying regions of free flowing text from a PDF can be modelled as an Object Detection problem, given that certain preprocessing is applied.

After the Preprocessing is Applied, A desktop application called LabelImg was used to record bounding boxes for each free flowing paragraph in each page in a series of PDF's.

Once bounding boxes had been generated, a Command line tool was developed to convert LabelImg specific annotations, to a machine readable, parseable format like JSON.

PDF's were then converted to a series of Images, so that computer vision strategies can be made use of during preprocessing and modelling. Due to a difference in the coordinate ordering schema between PDF and Image, a coordinate transform function needed to be made use of convert PDF coordinates to image coordinates.

Once the samples were labelled and converted to machine readable format, they are fed to a Faster RCNN, a popular Deep Learning architecture for Object Detection.

This model is then frozen using Google protobuf and then integrated into the Inhouse SDK for further use.

2. System Design and Analysis:

2.1 Preprocessing:

English Literature has 26 different Alphabets, that look very different from each other. Therefore, it can be said that the concatenation of different letters can create situations, where in too many features are learnt due to the structure of each word in a paragraph. This can create 2 problems:

1) Unnessecary increase in processing time

2) A very high degree of Over fitting, where in a model learns too many features.

Therefore, reducing the number of features, and at the same time retaining the most discriminative ones would ideally train a good object detection model.

It was found that the use of dilation drastically reduced the number of features, but at the same time, retained the geography of the paragraph object within the page/image(By geography, we mean the area/region within an image that the object to be detected lies in).

Each image was subject to a total of 7 dilations.

The preprocessing method, called dilation, is popular in computer vision tasks, where its main aim is to connect closely residing components together through the use of a kernel multiplication. It is useful in situations where edges of objects in the foreground are broken, or incomplete in nature.

This task of reducing variances has 2 main responsibilities, one of making various objects in the background seem as a single entity, and second to reduce the number of features hat a model may have to learn to represent an object.

The Preprocessing Step is an image level operation, and thus requires the PDF to be split into pages, and each page to be converted to the .jpg format using the PyPDF tool.

The preprocessing functions involved in order are:

Size Normalization => Gray scaling => Thresholding => Dilation => Erosion

The below diagram shows an example of how a single page is preprocessed.

Fig 1. Preprocessing Operation on PDF page

It is observed that the overall geography and shape structure of the paragraph is maintained, and the total number of features representing the image, has also reduced.

2.2. Training the Paragraph Detection Model:

Once the samples were labelled and converted to machine readable format, they are fed to a Faster RCNN, an object detection model that prioritizes accuracy over speed.

Its Architecture resembles that of the figure below:

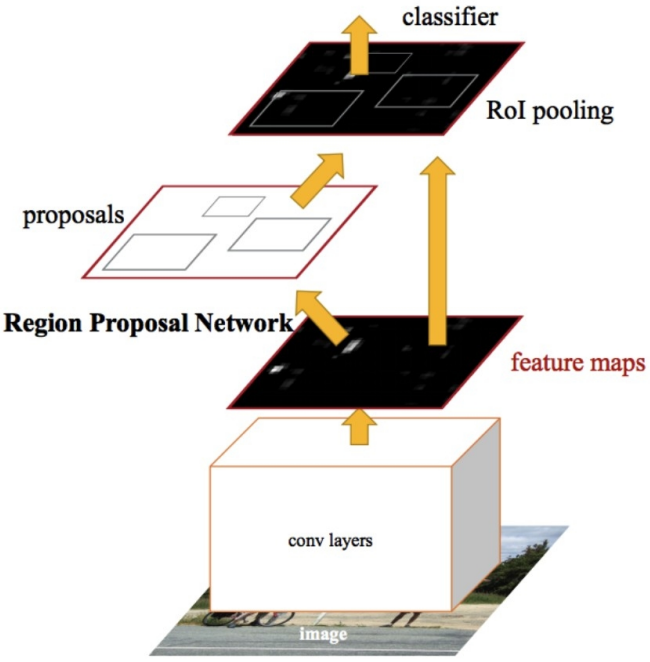


Fig 2.1 Faster RCNN Object Detection Model

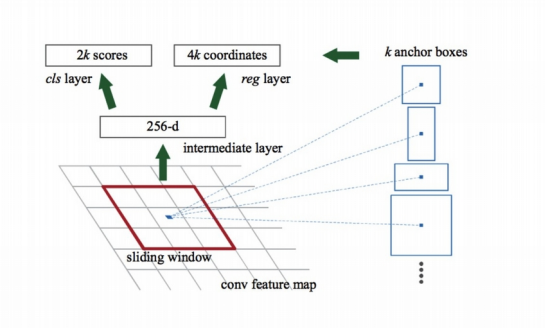


Fig 2.2 Region Proposal Network

There were 2 other architectures considered:

1. YOLO

2. SSD

The below table ranks Speed and Accuracy of each architecture

Architecture	Speed	Accuracy
Faster RCNN	3	1
SSD	1	3
YOLO	2	2

Table 1. Speed and Accuracy Ranking for each Architecture

Even though the Faster RCNN was the slowest among the 3, its speed was substantially greater than the Heuristic based method, and its Accuracy being the highest among the 3, also proved to be higher than the Heuristic based method.

Thus, The architecture chosen had both metrics of speed and accuracy improve with the use of a Faster RCNN.

Inorder to train a Faster RCNN, one would need to identify a framework that is:

1. Open Source

2. Supporting Model Deployment and Freezing services

3. Easily integrated into the Pipeline, and later the SDK

The Framework chosen was Google's Tensorflow.

The Tensorflow Object Detection API is an opensource sub-framework one can use to build Architecture specific Object detection models.

It is maintained by Google and is actively used by many global corporations, the most notable of them being AirBnB, a remote accomadation market place application, and Uber, A cab rental service deployed as an application.

Due to its fairly seamless process in integrating between various annotation types, choice in architecture, and choice in additional configurations for a particular architecture, this API was chosen for development of the Training and Deployment pipeline.

2.3 Deployment and Freezing of Detection Model:

Once a model had been trained on the Dataset,it is imperative to freeze the model, so it can be used in the future without unnecessary retraining.

While the sklearn framework for Statistical Machine Learning has joblib to save and load models, Tensorflow makes use of a combination of the graph, or the representation of the architecture, and a weights file.

This is saved in a file extension called protobuf, that is made especially for combining graph and weights into a single queryable variable loaded from the .pb file extension.

The .pb file is loaded on to a model variable, similar to that of loading a text file or a JSON file into memory.

This is then queried and the results are modified into an inhouse SDK specific format, that resembles the results returned by every other function of the SDK.

This response has the list of bounding boxes in order of time stamp of detection in a single pass of the object detector.

3. Modelling and Implementation:

The final pipeline is described as below:

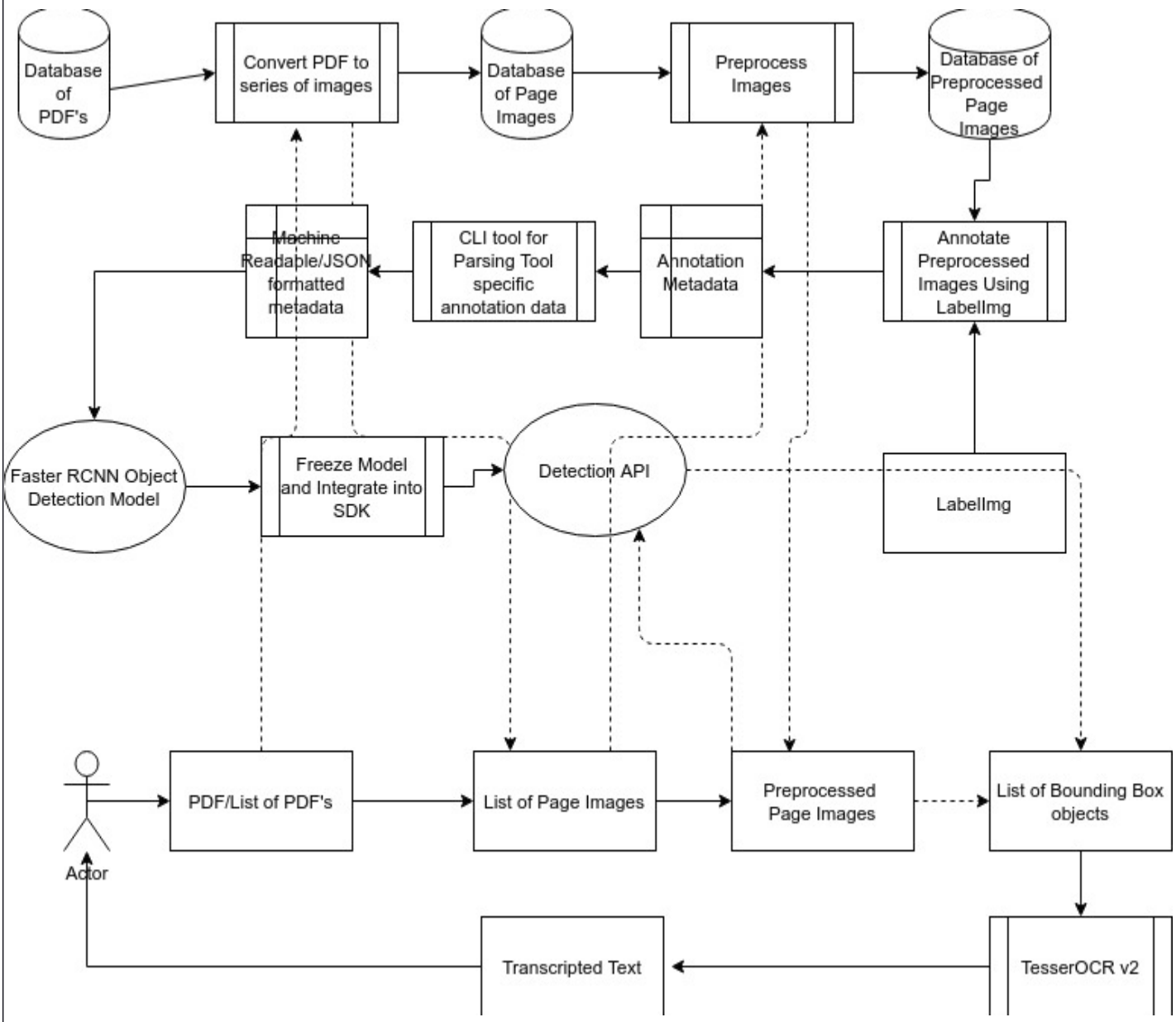


Fig 3. Architecture of Pipeline

The databases of the PDF's are locally stored and retrieved using the glob module in Python, used in collecting files that have a definitive structure in their filename.

Once these PDF's have been collected, only a small subset is chosen according to variety in Design templates and Fonts. This is to simulate a Resource Scarce environment.

In the same local space, each PDF is converted to a list Page objects using the in house SDK built in python.

Each page is converted to a .jpg file using the PyPDF module, that is a derivative of the popular Shell based preprocessing framework called Poppler.

Once these page images have been converted, they are sequentially processed by the Preprocessing module, again built in Python, using the Pillow and OpenCV image processing libraries.

Each of the preprocessed images are then stored in a separate local folder, where a Desktop based image annotation tool Labelmg is used to annotate paragraph specific regions on the Page. This Desktop based application is built using Tkinter, a popular Python framework for building GUI applications.

The annotations are converted to a Machine Readable format using Regex, ElementTree and JSON.

This machine readable annotation is then fed to the Faster RCNN model, that is pretrained on the VGG16 image dataset.

The model trained is then frozen using protobuf, and saved as a file in the disk.= with the extension as .pb.

This .pb file is loaded whenever the API is called, and the response after detection is a list of BoundingBox objects. Each of these objects are extracted from the response, and a new, inhouse SDK specific response is built for easy parsing and manipulation. These Bbox objects are coordinates of the smallest rectangle detected by the model that covers a paragraph on a page.

4. Testing,Results and Discussion:

4.1 Testing:

Testing was divided into 2 phases.

The Unit-testing phase individually tested the components of the pipeline, based on both the format, and the value of the result obtained.

Some of the Unit testing pipelines implemented were:

1. The Preprocessing pipeline

2. Model Querying and Response parsing pipeline

This allowed us to tune and modify both the approach and the data structure in which we stored the results in.

Therefore, this structure of iterative deployment and modification helped us deal with edge cases as and when they pop up.

The next phase involved the testing of the entire pipeline. Here, the most common bugs identified were the parsing and conversion of corrupted PDF's and documents. It is also required to test if the given PDF is Virus free or not. This is accomplished through the use of a 3rd party API, that makes use of checksums and other heuristic based functions to determine the safety of the PDF. If the API accepts the PDF, it is passed on to the pipeline, else it is discarded from the pipeline.

It was also required to validate the results of the pipeline, given a series of differently formatted and templated PDF's , in order to ensure the pipeline works for a variety of inputs.

4.2 Results:

Comparison Metrics Available:

These are the results for a sample validation PDF given to both the heuristic based detector and the Detection API developed.

Metric	Heuristic Based Detector	Detection API developed
Total Number of Paragraphs on Page(Ground Truth)	6	6
Number of Bounding Boxes detected	10	6
Ratio Between Paragraph Size and BoundingBox	0.8	1.0512
Speed per page	4 seconds	~0.5 seconds

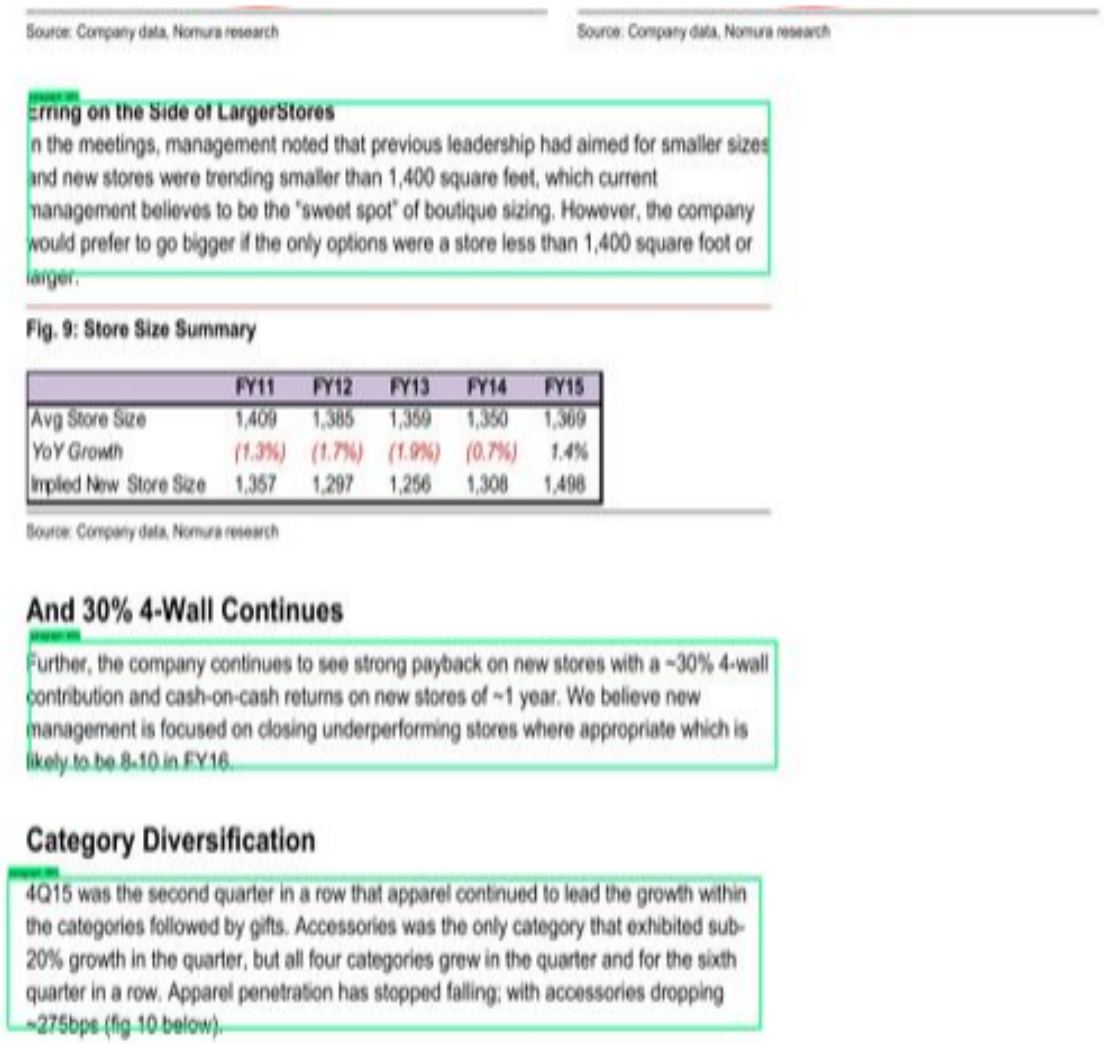
Table 2. Results at PDF level for given metrics

The ground truth establishes a value that can be used to calculate detection performance for both detectors.

The number of bounding boxes picked up will give us an insight as to how many bboxes were false positives, or false negatives.

The ratio gives us an approximate value of how well a bbox fits a paragraph. This is important to track as the accuracy of the bbox assigned to a paragraph is dependent on this value.

Detection and Extraction Sample result:



Source: Company data, Nomura research

Source: Company data, Nomura research

Erring on the Side of LargerStores

In the meetings, management noted that previous leadership had aimed for smaller sizes and new stores were trending smaller than 1,400 square feet, which current management believes to be the "sweet spot" of boutique sizing. However, the company would prefer to go bigger if the only options were a store less than 1,400 square foot or larger.

Fig. 9: Store Size Summary

	FY11	FY12	FY13	FY14	FY15
Avg Store Size	1,409	1,385	1,359	1,350	1,369
YoY Growth	(1.3%)	(1.7%)	(1.9%)	(0.7%)	1.4%
Implied New Store Size	1,357	1,297	1,256	1,308	1,498

Source: Company data, Nomura research

And 30% 4-Wall Continues

Further, the company continues to see strong payback on new stores with a ~30% 4-wall contribution and cash-on-cash returns on new stores of ~1 year. We believe new management is focused on closing underperforming stores where appropriate which is likely to be 8-10 in FY16.

Category Diversification

4Q15 was the second quarter in a row that apparel continued to lead the growth within the categories followed by gifts. Accessories was the only category that exhibited sub-20% growth in the quarter, but all four categories grew in the quarter and for the sixth quarter in a row. Apparel penetration has stopped falling; with accessories dropping ~275bps (fig 10 below).

Fig 4. Detection API result overlayed on input page

5. Conclusion:

A novel freeflowing paragraph detection algorithm was developed, with the use of minimal data, and an increase in both speed and accuracy of detection and localization was observed when compared to the heuristic based detector previously made use of in the inhouse SDK.

The Pipeline was incorporated into the companies inhouse SDK, where it continues to be used for the tasks of PDF metadata generation, and ETL pipeline development for Downstream text classification and tagging tasks.

A few improvements can be incorporated in the future, one of them being the improvement of preprocessing. Font size, color and spacing are useful features that can be used to discriminate PDF objects from one and other.

Increasing the Variety of PDF's and the size of the dataset in general can also lead to better results.

Text transcription can be improved for Indic languages, by retraining Tesseract on Indian Non Roman characters.

Conclusively, this internship taught me the basics of modifying configurations for an object detection model, and also the basics of deploying the same model as an API for different production environments. I learnt how to structure and pipeline a data intensive project from start to end.

Last but not least, I learnt how to make connections, and work in a professional environment. I learnt a lot during my internship, owing to the fact that Stride.AI was a Startup. After this experience, I'd recommend anyone planning to apply for an internship, to apply to a place that does both tasks of assigning you important tasks, and at the same provide a stable ground for training and mentorship.

This concludes my internship report, facilatated by Stride.AI and Ramaiah Institute of Technology, Bangalore.