AUTOMATING FILED EXTRACTION IN ITRA FORMS USING ML

A Graduate Project Report submitted to Manipal Academy of Higher Education in partial fulfilment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY
In

Electronics and Communication Engineering

Submitted by

Abhimanyu Borthakur
Reg. No.:180907742

Under the guidance of

Mr Samir Dadia		**Dr Pramod Kumar**
VP- PS&IT	**&**	**Professor**
Searce Inc		**Dept of ECE, MIT, Manipal**

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

MANIPAL-576104, KARNATAKA, INDIA

AUGUST 2022

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

MANIPAL INSTITUTE OF TECHNOLOGY

(A Constituent Institution of Manipal Academy of Higher Education)

MANIPAL – 576 104 (KARNATAKA), INDIA

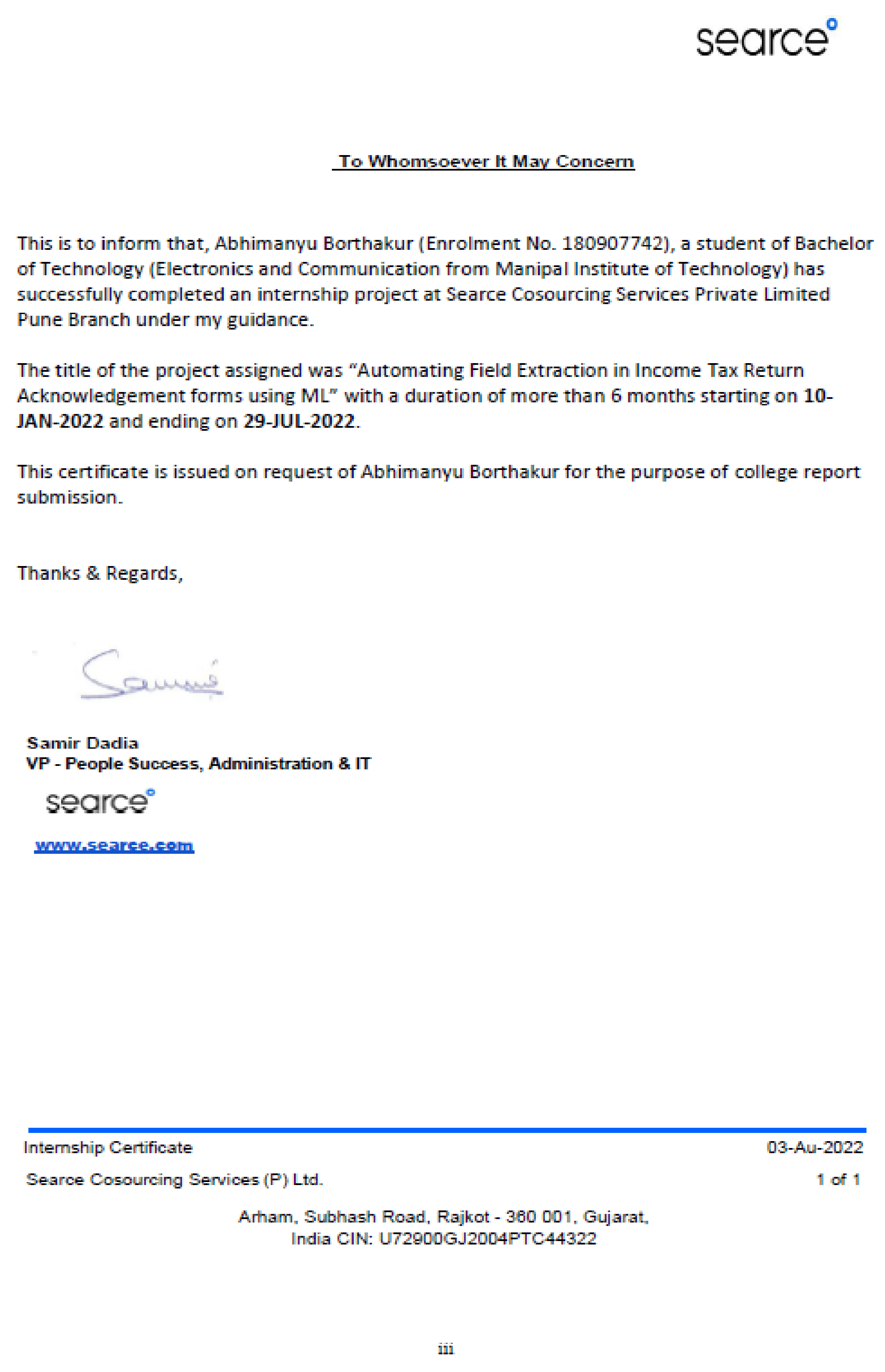
Manipal
09/08/2022

CERTIFICATE

This is to certify that the project titled **AUTOMATING FIELD EXTRACTION IN ITRA FORMS USING ML** is a record of the bonafide work done by **ABHIMANYU BORTHAKUR** (*Reg. No.180907742*) submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology (BTech) in **ELECTRONICS AND COMMUNICATION ENGINEERING** of Manipal Institute of Technology, Manipal, Karnataka, (A Constituent Institution of Manipal Academy of Higher Education), during the academic year 2021 - 2022.

Dr Pramod Kumar
PROFESSOR,
DEPT OF ECE, MIT,
MANIPAL

Dr. Subramanya Nayak G
HOD, ECE
M.I.T, MANIPAL



searce

To Whomsoever It May Concern

This is to inform that, Abhimanyu Borthakur (Enrolment No. 180907742), a student of Bachelor of Technology (Electronics and Communication from Manipal Institute of Technology) has successfully completed an internship project at Searce Cosourcing Services Private Limited Pune Branch under my guidance.

The title of the project assigned was “Automating Field Extraction in Income Tax Return Acknowledgement forms using ML” with a duration of more than 6 months starting on **10-JAN-2022** and ending on **29-JUL-2022**.

This certificate is issued on request of Abhimanyu Borthakur for the purpose of college report submission.

Thanks & Regards,

[illegible]

Samir Dadia
VP - People Success, Administration & IT
searce
www.searce.com

Internship Certificate 03-Au-2022
Searce Cosourcing Services (P) Ltd. 1 of 1

Arham, Subhash Road, Rajkot - 360 001, Gujarat,
India CIN: U72900GJ2004PTC44322

iii

ACKNOWLEDGMENTS

I would like to take this moment to acknowledge everyone who helped me with this industrial internship that will contribute to the final semester project at Manipal Institute of Technology. I consider myself fortunate to have achieved this opportunity to work in the capacity of an ML Engineering Intern at Searce Inc.

I humbly thank Mr. Samir Dadia, VP of People Success, Administration and IT, who helped me smoothly onboard on to the company. He not only helped me hone my skills, but also addressed my concerns and provided me with valuable insights. I am grateful he spent the time to equip me with the necessary facilities despite his other engagements.

I want to thank Dr.Pramod Kumar, who helped me with his guidance and close communication throughout my internship. I also want to extend gratitude to my college, Manipal Institute of Technology and the Director and Head of Department of ECE, whose efforts have provided me with the professionalism and knowledge that have helped me complete my internship at Searce.

I would also like to mention the support of my parents and friends, who helped me stay motivated throughout the duration of my internship.

I will take this moment as a stepping stone for the commencement of my professional career, and carry forward the newly learned skills that have augmented me personally and professionally, into future projects and endeavours.

Sincerely,

Abhimanyu Borthakur
180907742
ECE-D, Roll-53
Department of Electronics and Communication Engineering,
Manipal Institute of Technology

ABSTRACT

Document extraction identifies data relationships within a document as key-value pairs. For example, an invoice document contains several form fields and thus, the process of document extraction would serve to identify the field names and values that are paired together e.g., ‘Name of the customer’ as a field name and its corresponding value, ‘Abhimanyu Borthakur’.

It is humanly impossible for clients like big banks to extract, process and derive insights from unstructured data due to the sheer volume of documentation that happens on a daily basis. Around 90% of data within organisations is unstructured and locked in documents or images. This information - if extracted, structured and contextualized – and made available on demand – could provide valuable insights for better business decisions. Hence, arises the need for an end-to-end document extraction, processing and comprehension AI solution.

For the purpose of this task, we will be working primarily with images and in doing so, the pages comprising any pdf files will be converted into their corresponding images using appropriate libraries. The first objective is to make region proposals on the input image using an appropriate object detection neural network. The next objective would be to extract the detected regions and feed them into an OCR for the purpose of text extraction. The final step in the pipeline would be to clean the extracted text of any non-alphanumeric or non-ASCII characters and display the extracted text in the appropriate format on the prescribed platform.

On the work done so far, an object detection model has been trained and deployed for the purpose of region proposals and the results are promising – since fields and their values are small objects, an appropriate metric is mAP (mean average precision) at 0.5 IOU (intersection over union) and our model offers a mAP of 0.921 and 0.969 on the test and validation sets respectively, with 1 being the maximum achievable score.

In conclusion, our model is not only highly precise, but also fast at making inferences and occupies miniscule space which are essential components for a deployable and scalable architecture on the cloud. As a backbone for recognic.ai, our document AI tool, this model fits perfectly – recognic itself has helped clients like ICICI bank in reducing their document processing time from 10 minutes to 10 seconds which is a 60x improvement. In order to successfully complete the work leading up to this mid-term report, Google Cloud Platform was used for training, validating and testing the model and the technical stack includes languages like Python 3, deep learning frameworks like PyTorch, image processing libraries like OpenCV and front and backend tools like Flask and Streamlit.

LIST OF TABLES

LIST OF FIGURES

Contents

CHAPTER 1
INTRODUCTION

1.1 Introduction

In this chapter we shall discuss the current state of document extraction and our proposed approach leading to the evolution of object detection neural networks which is the main focus of this report. We shall then discuss the motivation for automating document extraction in order to drive business value and the objectives in order to accomplish the same. Our objectives though, must be modelled on key performance specifications which will then be presented. The end of chapter will then be in the form a project work schedule and project report organization.

1.2 Motivation, Objectives and Target Specifications

1.2.1 Motivation

In the aftermath of COVID-19, we will see accelerated digitization across the enterprise. Not leveraging insights contained in unstructured documents can impact your process efficiencies and put your business at a competitive disadvantage. Why waste 1000’s of person-hours for work that can be done more accurately and efficiently by an AI engine? Why not give your employees a reprieve from repetitive manual tasks, and empower them for better decision making? Document extraction, processing and comprehension done right can help generate revenue opportunities, save costs, reduce compliance risks, improve operational efficiencies, and yield faster RoI. AI is integral to business success in the new normal, and the faster you adapt it, the farther you will be in business value creation [1].
Around 90% of data within organisations is unstructured and locked in documents or images. This information - if extracted, structured and contextualized – and made available on demand – could provide valuable insights for better business decisions. Hence, arises the need for an end-to-end document extraction, processing and comprehension AI solution.
For the purpose of this task, we will be working primarily with images and in doing so, the pages comprising any pdf files will be converted into their corresponding images using appropriate libraries. The first objective is to make region proposals on the input image using an appropriate object detection neural network. To have gone with a machine learning approach like object detection neural networks is an attempt to make our final model pipeline extremely

robust to any kind of real world user data, which is a shortcoming present in traditional image processing approaches, and in particular, we will be harnessing our focus on small object detection as the field-value pairs in Income Tax Return Acknowledgment forms are quite small compared to actual image size and there exists no set framework for small object detection [2]. Our neural network will be used to detect these small objects by making region proposals, following which we will extract the detected regions and feed it into a state-of-the-art OCR [3] for the purpose of text extraction and then display the text in the appropriate format after a clean-up operation.
Data augmentation being an important part of object detection [4] will also be leveraged by us in order to simulate user data.

1.2.2 Objective

Our objective as a part of this project work will be to set up a groundwork for a scalable yet efficient architecture by leveraging a model that quick at extraction, yet highly precise at the same time, which means we have to select a model that doesn't compromise its precision for efficiency or vice-versa. The accuracy-efficiency trade-off is a common issue within the field of machine learning and object detection [5] and our model selection will be dictated on the basis of the same metrics.
In accomplishing this objective, we must have a few target specifications in order for our aspirations to be realized in the form of key numerical metrics. In doing so, it is helpful to note some conclusions about Searce's document AI tool recognic – It enables quicker processing of supporting documents on customer applications led to a higher onboarding volume per quarter, and for clients like ICICI Prudential, has helped achieve a 60x improvement in document processing time from 10 minutes to 10 seconds [6].
Such is the level of performance we will aim to leverage without compromising on accuracy and precision.

1.2.3 Target Specifications

In order to achieve the objectives listed in the above chapter, we must represent them in terms of key achievable numerical metrics which are listed as follows in the table 1.1 below:

Table 1.1: Target specifications

Metric type	Metric unit	Target value
Model size	Gigabytes (GB)	< 1 GB
Extraction time	Seconds (s)	< 10 s

Such performance specifications eliminate the initial waiting time which can result in customer drop-offs – this helps in customers knowing whether their documents are sufficient immediately or if they need more.

1.3 Schedule and Organization of Report

1.3.1 Schedule

The schedule of the project is manifested in two parts – The first part comprises the prerequisites for the project and is carried out during the months of January, February and March. The second part is the implementation and is distributed across April, May and June totalling a duration of six months.
The prerequisites form the backbone of this project as it helped the organization communicate the essential workplace skills and best practices that every employee must develop in order to innovate and add value. They are also essential in understanding the suite of cloud computing solutions available to meet the key performance metrics and perhaps in some cases, exceed them. In summary, it is to drive home the value of automation, ML/AI and cloud computing in this digital era and the best practices one can use to unlock that value.
The implementation serves to not only develop innovative and efficient solutions as per the defined problem statement but also to showcase growth across the internship period and demonstrate learning ability through the application of skill.
The entire schedule is detailed in the table 1.2 below: -

Table 1.2: Project Schedule

January 2022	o Onboarding and beginning of mentor led training
February 2022	o Mentor lead training and weekly assessment
March 2022	o Google Cloud Digital Leader Certification training
April 2022	o Business Unit Task – field extraction from ITR forms 1) Proposing framework and approach for pipeline 2) Choosing tool for annotating and labelling dataset 3) Model selection and environment setup 4) Model training and validation 5) Logging results and inference on test images and retraining if need be
May 2022	o Design, OCR and Extraction 1) Start work on front-end and back-end design for the purpose of a deployable web application 2) After inference on test images, extract regions of interest and feed into OCR 3) Observe OCR performance and make improvements if need be 4) Start text extraction and clean-up and observe performance
June 2022 *And* *July 2022*	o Deployment, Versioning and Documentation 1) Use GCP to integrate a deployable web app with the cloud or as an API endpoint 2) Use version control like Git for versioning and documentation

In the table above, the mentor led training was purposed for a period of 6 weeks, which was broken up as two weeks in January and for all 4 weeks in the month of February. This was in order to be trained in the important concepts of Machine Learning and Python. It was followed by a 4-week Google certification called 'Cloud Digital Leader' which was essential in learning about most of the products offered as a part of the Google Cloud Platform and how these products can be used to drive value for businesses and for the purpose of this project, provide an entire suite to train, validate and test your models and consequently, deploy them at scale. The implementation of said project followed and will continue until June.

1.3.2 Organization of report

This report is organised in the form of chapters with titles and content listed as in table 1.3:

Table 1.3: Organization of report

Chapter	Chapter Title	Chapter Content
1	Introduction	Brief introduction, motivation, objectives, target specifications, schedule and organization of the report
2	Background Theory	Evolution of Object Detection and model selection
3	Methodology	Top-level view of architecture, data management and breakdown of the object detection model and training said model
4	Result Analysis	Explaining the result metrics and analysis of result on validation set and inference(test) set
5	Conclusion and Future Scope	Conclusions of the work performed and future scope of the same including possible improvements and next steps

Each chapter will have two sub-sections, the first being a gentle introduction the topics covered in the chapter and the second being a collection of sub-chapters dedicated to specifying, detailing and explaining the relevant information pertaining to that chapter.

CHAPTER 2
BACKGROUND THEORY

2.1 Introduction

The first task in our document extraction pipeline is to make region proposals for detecting the appropriate fields using deep neural networks for object detection and since the bulk of the work in this mid-term report focuses on Object Detection through Deep Learning, the latter will naturally be the focus of the background theory and literature review. This chapter will start by introducing the field of object detection and outlining its evolution. It will then go onto to present theoretical discussions on the same and consequently offer justification for the selection of the model used in this project. It will then be concluded by summarising the information important before moving ahead.

2.2 Evolution of Object detection, Model selection and Theoretical and Mathematical aspects

2.2.1 Evolution of Object Detection

Object detection is a computer vision task that proposes detection instances of objects of one or more classes (pedestrian, vehicles, plants etc) in images. The purpose of this computer vision task is to ingest images, analyse them and provide two very crucial pieces of information that are central to problems in computer vision: the class of the object(s) and its location. Both of these comprise the label predicted by our models which are learned by means of minimising the error between the predicted and ground truth labels.
Since this is one of the most fundamental research problems in the field of computer vision, object detection is the baseline for many other problems in this area as well, such as instance segmentation [7], object tracking [8], object counting [9] and so on. This field of object detection can be broadly classified into two areas of development: “fundamental object detection” where researchers develop object detection aimed at precision and efficiency improvements and “applied object detection” where engineers and developers leverage the power of existing architectures and deploy them into production for the purpose of scalability and for driving business value and creating revenue.

In the years hence, the emergence of the phenomena of deep learning-based object detection has led to huge developments in the field of object detection with the ability to arrive at breakthroughs and push boundaries more often and frequently than traditional approaches. This body of work has now been deployed in both industry and academia in the form of multiple applications in the real world such as self-driven automobiles, automated surveillance, robotics and automated document extraction

Over the last 20-25 years, it is proposed that object detection as a field has undergone evolution through two separate tracks: “Traditional” and “deep learning”, as shown in Fig. 2.1

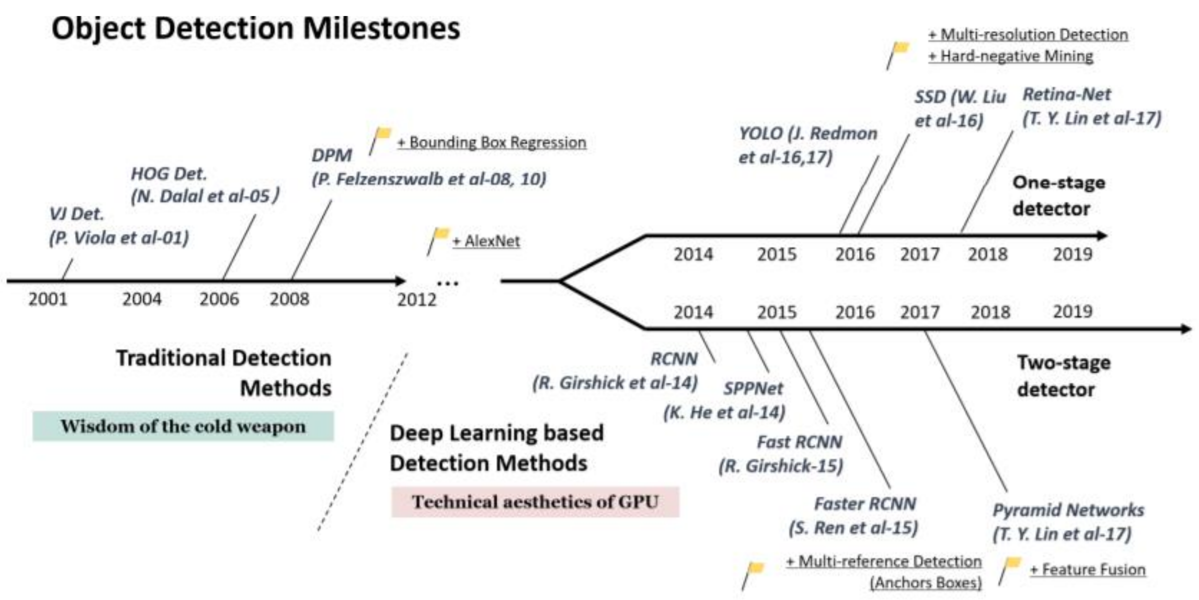
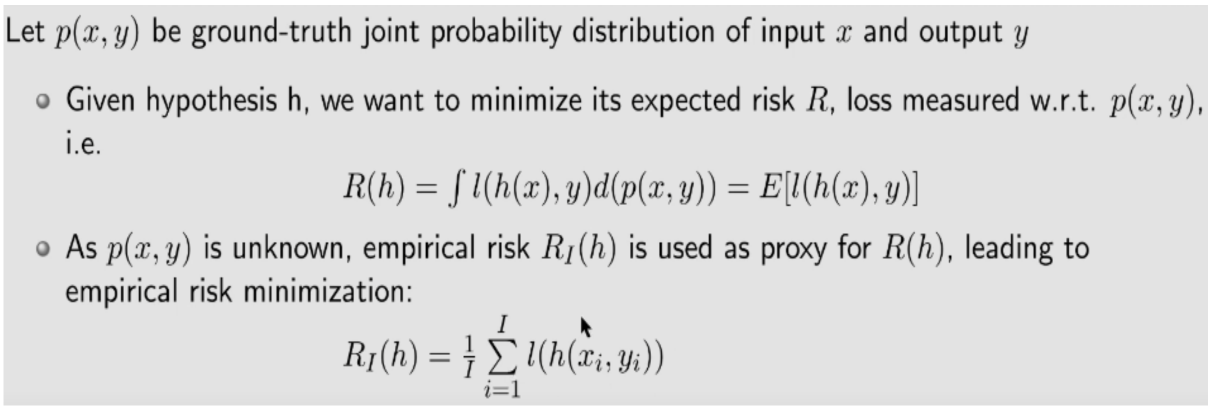

Object Detection Milestones
VJ Det. (P. Viola et al-01)
HOG Det. (N. Dalal et al-05)
DPM (P. Felzenszwalb et al-08, 10)
+ Bounding Box Regression
+ AlexNet
2001
2004
2006
2008
2012
Traditional Detection Methods
Wisdom of the cold weapon
Deep Learning based Detection Methods
Technical aesthetics of GPU
+ Multi-resolution Detection
+ Hard-negative Mining
YOLO (J. Redmon et al-16,17)
SSD (W. Liu et al-16)
Retina-Net (T. Y. Lin et al-17)
One-stage detector
2014
2015
2016
2017
2018
2019
Two-stage detector
RCNN (R. Girshick et al-14)
SPPNet (K. He et al-14)
Fast RCNN (R. Girshick-15)
Faster RCNN (S. Ren et al-15)
Pyramid Networks (T. Y. Lin et al-17)
+ Multi-reference Detection (Anchors Boxes)
+ Feature Fusion

Figure 2.1: A timeline for object detection: Viola-Jones Detector, Histogram of Oriented Gradients Detector, Deformable Part Model, Region-based CNN, Spatial Pyramid Pooling Net, Fast Region-based CNN, Faster Region-based CNN [10], You Only Look Once Model [11,12,13], Single Shot Detector, Feature Pyramid-based Networks, Retina-Net [14]

2.2.2 Theoretical and Mathematical aspects:



Let $p(x, y)$ be ground-truth joint probability distribution of input x and output y

- Given hypothesis h, we want to minimize its expected risk R, loss measured w.r.t. $p(x, y)$, i.e.

$$R(h) = \int l(h(x), y) d(p(x, y)) = E[l(h(x), y)]$$

- As $p(x, y)$ is unknown, empirical risk $R_I(h)$ is used as proxy for $R(h)$, leading to empirical risk minimization:

$$R_I(h) = \frac{1}{I} \sum_{i=1}^{I} l(h(x_i, y_i))$$

Fig 2.2: Empirical risk minimization for machine learning [15]

Let us consider the theorem of empirical risk minimization in Fig 2.2 above – here,

$$\mathbf{x = \{x_i\},} \qquad \mathbf{(1)}$$
$$\mathbf{y = \{y_i\}\ where,\ i = 1,2,3\ldots\ldots I} \qquad \mathbf{(2)}$$

Where,

- **x** and **y** denote the input images and target output detections (ground truth bounding box co-ordinates and class labels), **h** denotes the selected machine learning/deep learning model for the task and **I** denoted the number of input output pairs.

- **h(x)** denotes the prediction of the model on the input data tensor **x** and **l(h(x), y)** denotes the loss function or the error/cost between predictions $\mathbf{\{\hat{b}_x, \hat{b}_y, \hat{w}, \hat{h}, \hat{l}, \hat{c}\}_i}$ and $\mathbf{\{b_x, b_y, w, h, l, c\}_i}$.

- $\mathbf{(b_x, b_y)}$, $\mathbf{(\hat{b}_x, \hat{b}_y)}$ denote the ground truth and predicted bounding box centre co-ordinates respectively

- **(w, h)**, $\mathbf{(\hat{w}, \hat{h})}$ denote the ground truth and predicted bounding box width and height respectively,

- **l**, $\mathbf{\hat{l}}$ denote the ground truth and predicted class label respectively,

- **c**, $\mathbf{\hat{c}}$ denote the ground truth and predicted confidence respectively

The loss function **l(h(x), y)** is hence, constructed as sum of 3 terms: -

$$\mathbf{l(h(x), y) = \lambda_1 \sum L_b (c_i, \hat{c}_i) + \lambda_2 \sum [l_i \geq 1]\, L_r \{(\hat{b}_x, \hat{b}_y, \hat{w}, \hat{h})_i, (b_x, b_y, w, h)_i\} + \lambda_3 \sum L_m (l_i, \hat{l}_i)} \quad \mathbf{(3)}$$

Where,

- **$[l_i \geq 1]$** is the Iverson operator which outputs 1 if the condition within the operator is true (if an object exists) and 0 if there is no object within the proposed region

- **L_b** denotes the binary cross entropy which is applicable for cases for ground truth is 0 or 1
- **L_r** denotes mean squared error or regression loss which is applicable for continuous variables

- **L_m** denotes multiclass categorical cross-entropy function which is applicable for categorical variables

- **λ** terms are constants used to weight different parts of the loss function.

In order for us to estimate the best model for the task, **h_I,** we have: -

$$\mathbf{h_I = \underset{h \in H}{\arg\min}\, R_I(h)} \qquad \mathbf{(4)}$$

The model **h**, will also have a set of weights or learnable parameters that it seeks to modify in order to leverage the best performance. Those weight matrices **w**, are given by: -

$$\mathbf{w = \underset{w \in W}{\arg\min}\, R\, (h\, (x; w))} \qquad \mathbf{(5)}$$

There are multiple algorithms/optimizers available for the purpose of this optimization task, however the selection of the same will depend on our model selection which we will now transition into.

2.2.3 Model Selection

It is impossible to scan the entire search space for the best model **h_I,** however we can limit our search space by observing a few key experimental metrics.

Table 2.1: Popular models in torch vision [16]

Model name	Memory (GB)
Faster R-CNN ResNet-50 FPN	5.2
RetinaNet ResNet-50 FPN	4.1
SSD300 VGG16	1.5
Mask -RCNN ResNet-50 FPN	5.4

Observing the sizes of the above popular object detection models in Table 2.1, leads to the conclusion that neither of them are suitable for the task at hand as they exceed the 1 GB max size requirement.

This begs the need for a different family of object detection models and luckily, YOLOv5 [17] comes to our rescue.

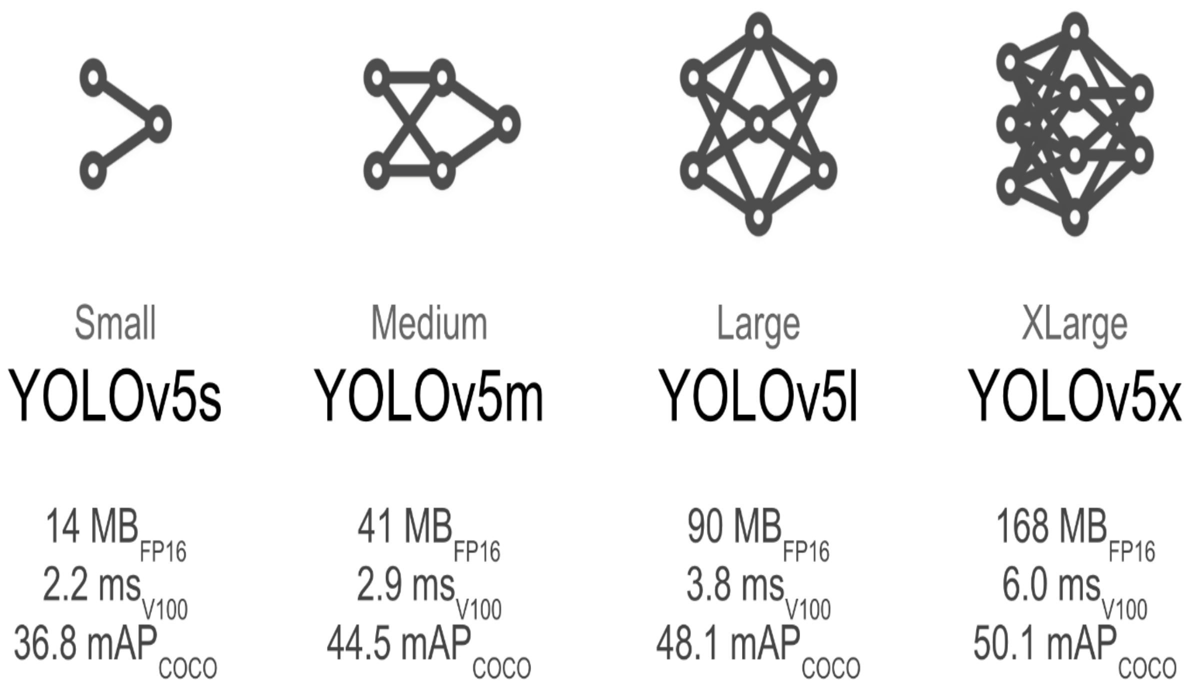


Figure 2.3: Family of YOLOv5 models [18]

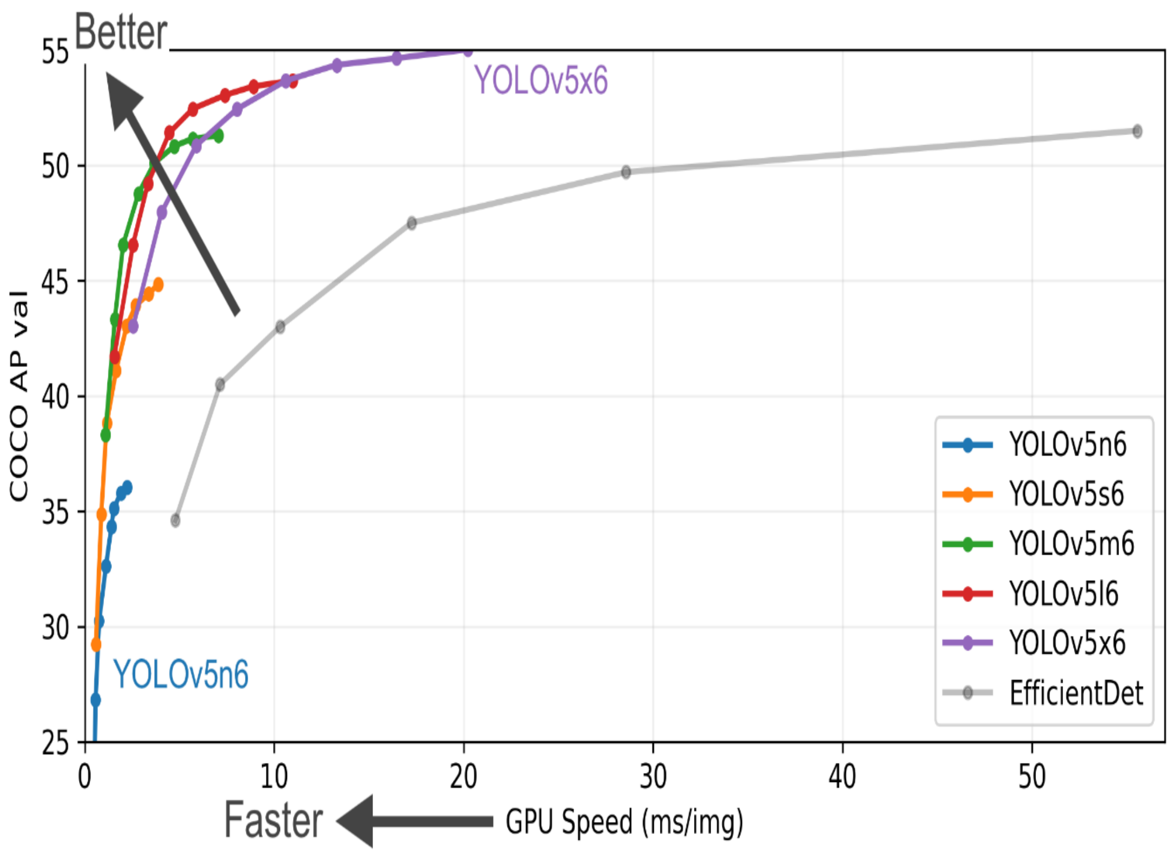


Figure 2.4: YOLOv5 performance: precision vs speed [19]

It is clear that YOLOv5s forms the sweet-spot and is thus selected as our model of choice.

This is because: -

1) YOLOv5x will have a large number of parameters that won't scale well

2) YOLOv5l will also have a large number of parameters that won't scale well

3) YOLOv5s provides a 10% increase in performance while maintaining similar speeds when compared to YOLOv5n

4) YOLOv5s is not too far behind YOLOv5m when it comes to peak performance

CHAPTER 3
METHODOLOGY

3.1 Introduction

In this chapter we will first introduce a top-level view of our architecture to illustrate the task, specifics and significance of each step as well us the complete flow from input to output. We will then go on to expand upon the details of our object detector as it is the prime focus of this report and the work done so far. We will also specify the technical details within the training procedure such as the hyperparameters used.

3.2 Pipeline and Object Detector training, OCR integration, deployment versioning and documentation

3.2.1 Complete extraction pipeline and training

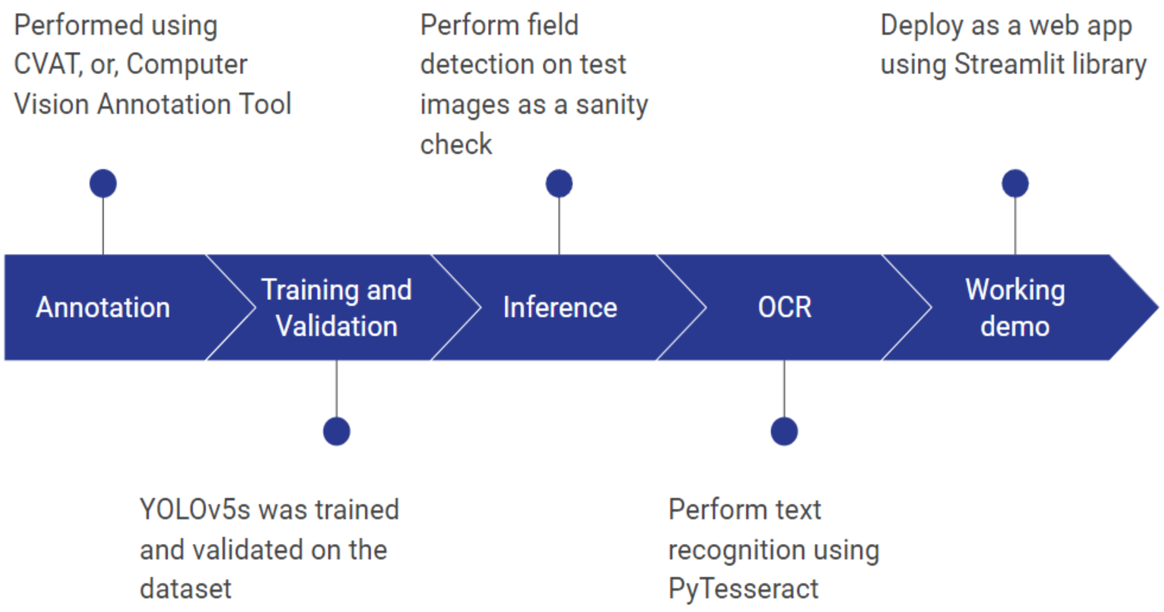

Performed using CVAT, or, Computer Vision Annotation Tool
Perform field detection on test images as a sanity check
Deploy as a web app using Streamlit library
Annotation
Training and Validation
Inference
OCR
Working demo
YOLOv5s was trained and validated on the dataset
Perform text recognition using PyTesseract

Figure 3.1: Complete extraction pipeline

The above pipeline begins with annotation and labelling of the given images using CVAT annotation tool [20] which is free and open-sourced by Intel. The advantage of using this tool is that it allows a panoply of formats to export the annotations in and especially, the YOLO format which exports annotations as one text per image containing the annotations for each class. So, it naturally begs of us to create training, validation sets which are then uploaded to a Google Cloud Storage bucket which can be linked to your training Jupyter notebook using the Vertex AI platform on G.C.P, a parser that would parse each of these files and create input, target output pairs as mandated by the model. This can be achieved with the help of PyTorch data loaders.
The model is then trained for 1500 epochs where one epoch is a full pass through the training data in the form of batches. The training details are as follows: -

1) Optimizer: Stochastic gradient descent

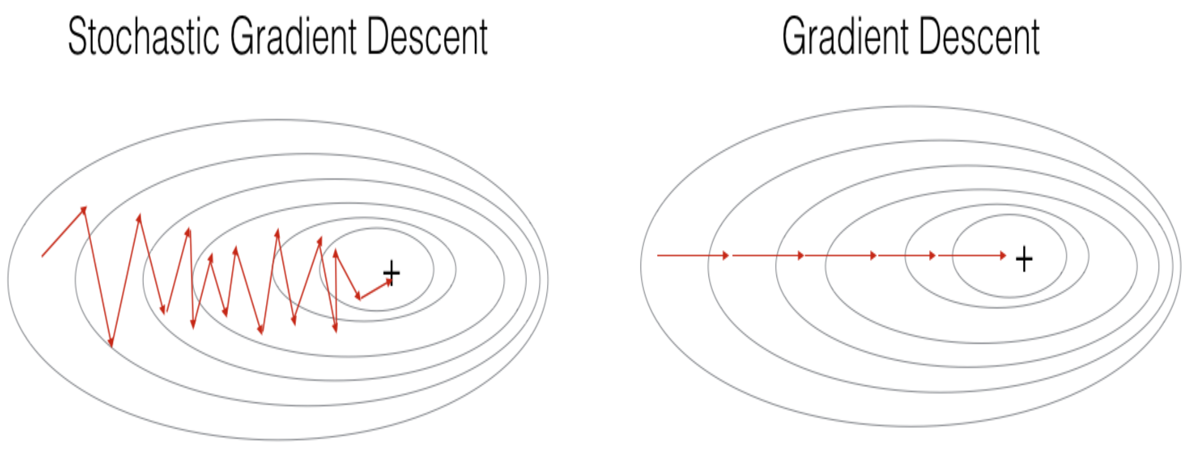


Fig 3.2: Stochastic gradient descent [21]

In order to learn trainable weights such that our loss function is minimized, stochastic gradient descent is used for YOLOv5s model – the difference between gradient descent and stochastic gradient descent is that in the former, looks at all training examples before updating the weights whereas the latter selects one mini batch of data at random and updates the weights immediately after that examples, this leads to oscillations in the learning curves because one batch may be more difficult/easy to learn than the other.

2) Learning rate

A learning rate of 0.01 is used for YOLOv5s - it is the rate at which the weights are updated.

Then, we proceed to make inference on test images and evaluate our model which serves as a good sanity check. It is then that we must decide whether the preliminary performance is enough or whether we need to retrain. Then we proceed to use the OCR to extract text from the detected regions and as a part of this step we will also perform a text clean-up mechanism before deployment as a web-app.

3.2.2 OCR Integration

Optical Character recognition or OCR, is the process of converting printed text present within digital media like images into editable and tangible text. The advent of OCR is really beneficial and can provide the following benefits to business that aim to leverage their power: -

1) A capable OCR engine facilitates higher productivity by enabling quicker data retrieval – the time and effort spent in querying data from records can be saved by an OCR automation mechanism and channelled elsewhere, thereby increasing employee efficiency.

2) Seamlessly integrating an OCR into an organization's workflow can help in cutting costs and reducing expenditure – a company would have to forego hiring data extraction and entry professionals and the money saved would be better spent elsewhere, such as development and maintenance of the OCR engine. The company would also no longer bear the cost of misplaced or lost documents as an OCR would aid digitization

3) A major hurdle in manual data extraction and entry is the risk of erroneous data entry - an OCR would enable higher accuracy and reduced errors and inaccuracies in our data wrangling step, as it is automated and trained to deliver a high degree of performance. Any problems like data loss can also be tackled with the help of an OCR engine

4) An automated data wrangling approach with the help of an OCR helps us assimilate a 'paperless' approach throughout an organization's hierarchy – it helps us on cutting down on storage space required for storing and maintaining huge paper files. The data can now be stored electronically on servers on-premise or on cloud storage buckets on cloud platforms that are available in a serverless manner

5) Choosing an OCR for our organization's needs helps in securing our data from threats and preventing its loss. Data recorded on paper can be easily misplaced or manhandled, our destroyed by the means of natural phenomena such as moisture and fires. Scanning, digitizing and storing such data can safeguard against such things and also help in security from malicious elements and threats with the help of access control policies available on cloud platforms.

6) Another advantage of using an OCR is the text-searching functionality that is afforded once it is used - scanning, digitizing and storing our data with the help of an OCR helps us make the resulting documents a hundred percent editable and searchable which allows us to quickly lookup important numbers, figures, addresses and other relevant

   information. Such raw data can then be structured into certain formats like comma separated value files to enable machine learning models to learn from the data and add further value to the organization by forecasting projections and making predictions.

7) An appropriate OCR helps us improve customer service beyond comprehension – clients giving us access to their data would have a large number of customers who would require timely and regular access to said data. If the data would be stored in a raw and unprocessed manner, we would have to set up a million call centres and employ and million more employees to facilitate timely retrieval of data, and that too manually. An OCR eliminates this by processing and retrieving said data at really high speeds and that is really advantageous in such situations. Thus, waiting times are drastically reduced for customers and so is customer drop off for our client

8) One of the biggest upsides to using an OCR to convert images into text is to have an editable end output that can be modified and leveraged for further processing an consequently enable further development in terms of analytics and machine learning models to serve predictions. We can readily produce text documents through the output of an OCR that can be immediately edited and updated which enables us to store volatile content.

9) Insuring against disasters is one of the best benefits of employing the services of an OCR as the processed data output would be stored electronically on cloud native distributed systems that would prevent losing access to data in cases of a natural disaster or calamity and/or outage.

10) Another huge benefit of an optical character recognition system is it’s ability to seamlessly integrate with both legacy on-premise hardware and software systems distributed around the globe through cloud native applications – this gives us a great deal of flexibility and versatility in our data wrangling pipeline.

OCRs can be leveraged by a wide variety of stakeholders ranging from big banks to insurance vendors. In essence, any customer interested in automating their document wrangling pipeline to drive business value and efficiency should seriously consider using OCRs for the purpose of their workflows and take advantage of the aforementioned merits that arrive with such automation.

Some kinds of documents that are extensively exposed to OCR automation include but are not limited to: -

1) Purchase invoices
2) Industry policy related articles
3) Income tax and property tax documents
4) Employee and company payroll information
5) Legal filings from lawyers
6) Stakeholder contact information – client and customer
7) Business cards and flyers with advertisements
8) Investment strategy documents or other documents financial in nature

For the purpose of this project, we aim to use our OCR to convert the text within our detected regions or regions of interest into actual and editable text for the purpose of automated extraction. We receive the bounding box co-ordinated for each region/field and used the area enclosed by those pixel co-ordinates to return the concerned subset of the images in the form of a region of pixel values which itself is like a smaller image. Our objective is to extract the text within these multiple small images from one large image of the income tax return acknowledgment form by using an OCR.
However, before we leverage the power of a readymade industrial OCR engine, we must first task ourselves to understand how an OCR works.
Every OCR will consist of the following basic components: -

1) An image pre-processor such as noise removal or thresholding or both
2) A paragraph or line-level segmentation to detect the text blobs or corpuses
3) A feature extractor to extract the outlines or features of the characters in the corpuses
4) A multiclass classifier to classify characters to their requisite text04-level representations
5) Post processing and error correction of the errors made by the classifier

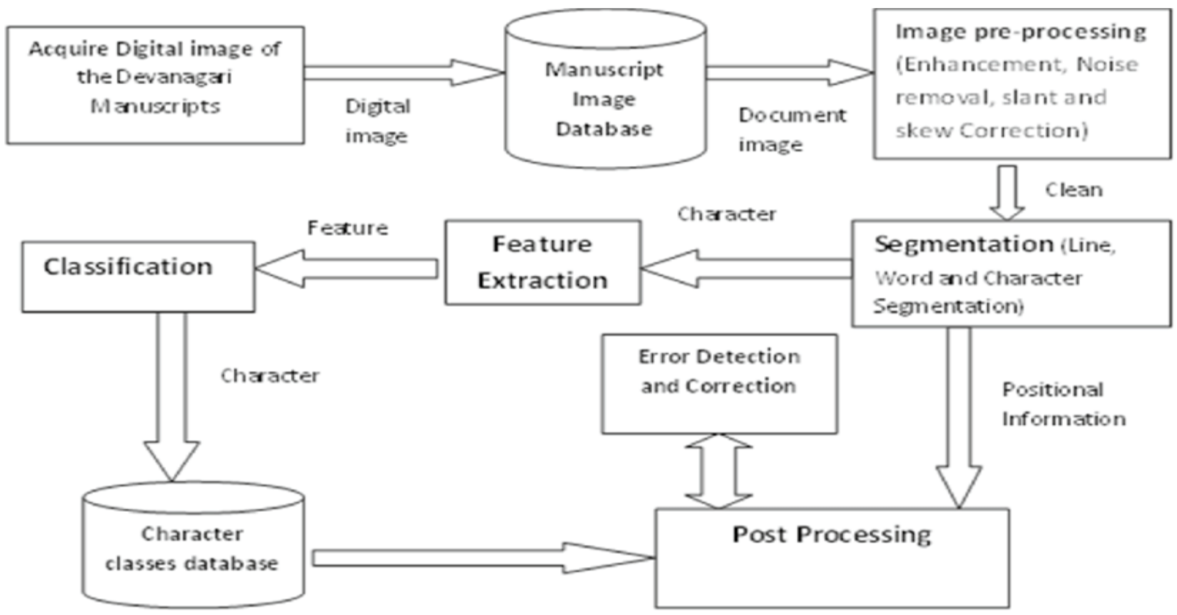

Acquire Digital image of the Devanagari Manuscripts
Digital image
Manuscript Image Database
Document image
Image pre-processing (Enhancement, Noise removal, slant and skew Correction)
Clean
Character
Feature
Classification
Feature Extraction
Segmentation (Line, Word and Character Segmentation)
Error Detection and Correction
Character
Positional Information
Character classes database
Post Processing

Fig 3.3: Example of an OCR system diagram [22]

In Fig 3.3, we can analyse the example of an OCR system diagram to get an overview of how an OCR really works.
It starts as all projects do with acquisition of the data in the form of images. Accumulation of enough of these images will help us create a database from which we can query images selectively and feed into our processing pipeline.
First, we pre-process the image to enhance the quality, facilitate noise removal and enable slant and skew correction that can occur while scanning documents
Then we perform character, line and world level segmentation to extract the pixel level regions.
Then we perform feature extraction of the segment regions and classify those features according to their requisite classes.
Our character class database can then help us enable error detection and correction and post processing to output the actual editable text.

For the purpose of this project, we will be leveraging the power of the Tesseract OCR Engine.

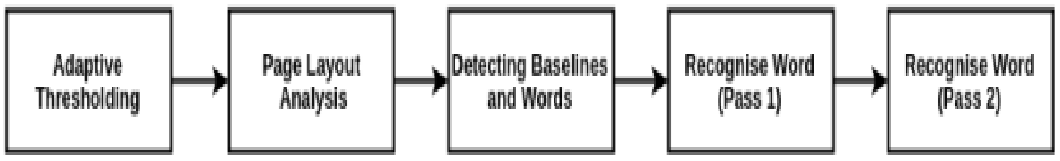


Fig 3.4: Architecture of Tesseract OCR Engine [23]

The architecture diagram/workflow/pipeline of the Tesseract OCR engine is laid out in Fig 3.4.

The first step in the pipeline is adaptive thresholding, which converts our input image into a binarized image using Otsu’s thresholding [24].
The next step in the pipeline performs page layout analysis which is used to extract text blobs/blocks/corpuses from the document as shown in Fig 3.5
Now, from the extracted blocks, we proceed at a line level and extract baselines for our given lines using two methods that are known as definite spaces and fuzzy spaces [25].
Next, we must extract the outlines of the characters from the words. Hence, we can perform word recognition as a two-phase process: -

1) In the first phase, we try to recognise words using a static classifier. Each satisfactory classification is passed to an adaptive classifier as training data

2) In our second phase, an entire run over the document is performed by our latest newly learned classifier in order to correct for characters that were classified incorrectly in the form of false positives and false negatives.

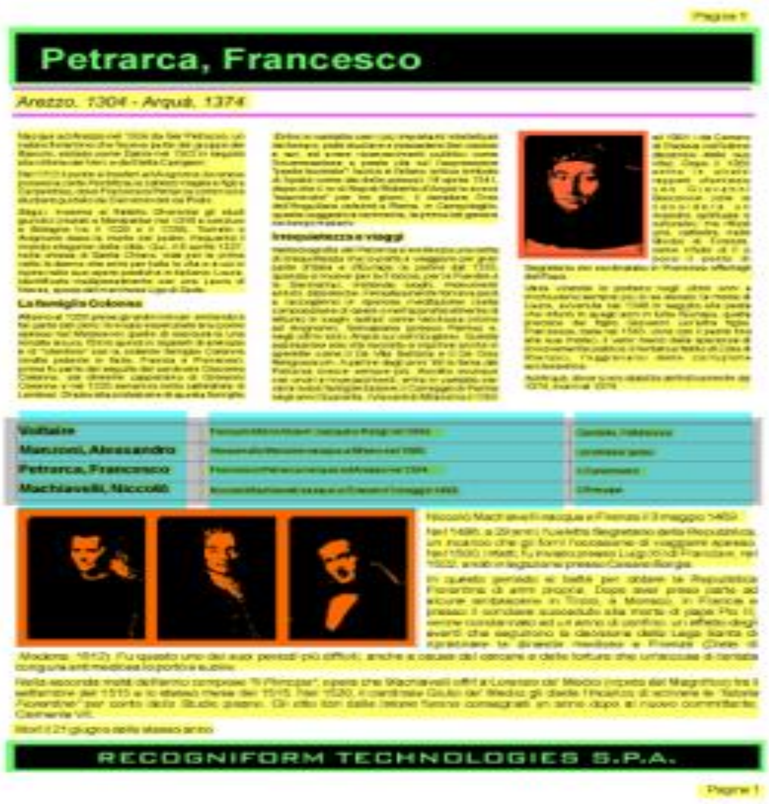


Fig 3.5: Demonstration of page layout analysis

So, for the purpose of our project we will be using the Python distribution of the Tesseract OCR Engine, called Pytesseract.
Pytesseract is an optical character recognition library built for Python. It has the ability to recognize, extract and "read" the text that is embedded in images of documents.
Pytesseract is a Python wrapper for Google's Tesseract OCR engine. It can also be used as a standalone invocation script to tesseract and has the ability to read images read with all kinds of imaging libraries including Pillow, Leptonica and OpenCV of all formats including but not limited to jpeg, png and tiff.
It also has the ability to print the recognized text instead of writing it to a file.

3.2.3 Deployment

Most machine learning or data science ideas do not make it to production. Those ideas or models that do not make it to production will not make an impact in the real world thereby rendering them insignificant.
In order to avoid such outages between development, staging and production, we must create a set of best practices to test, debug and monitor our ML models before eventually deploying them in production.
Instead of managing and creating our own infrastructure for the purpose of deployment, ready made and managed services exist for pushing our machine learning models into production in the form of products and frameworks developed by the likes of Amazon, Google and Microsoft which are named as Amazon Web Services, Google Cloud Platform and Microsoft Azure respectively.
For the purpose of this project and our consequent report, and as per the directives of the organisation allotting this project, we seek to employ the services of Google Cloud Platform or GCP for creating an end-to-end machine learning solution out of this automated Income Tax Return Acknowledgement form field extraction pipeline.
Various services exist on GCP, such as Vertex AI for managed training, deployment and serving for our machine learning models, leveraging containerisation in order to make our code manageable, composable, portable and scalable.
We also have Cloud Build and Cloud Run to support activities like Continuous Integration and Continuous Deployment which is a level of automation beyond the pipeline itself – it seeks to

seamlessly integrate our code changes and version changes without manually having re-deploy and re-integrate the code in our environment.

Finally, we shall also discuss Google App Engine which seeks to build scalable web applications which are sandboxed and hosted within servers sitting in Google's data centres.

We can easily create user interface or UI, for our application front end that would allow the user to upload a file and consequently, retrieve the results of the filed detection in the form of a two-column table. This can be deployed on App Engine.

We will also explore deploying the final model as an API endpoint as it gives external clients the ability to use JSON payloads in the form of API requests and responses to transfer and receive inputs and out puts respectively.

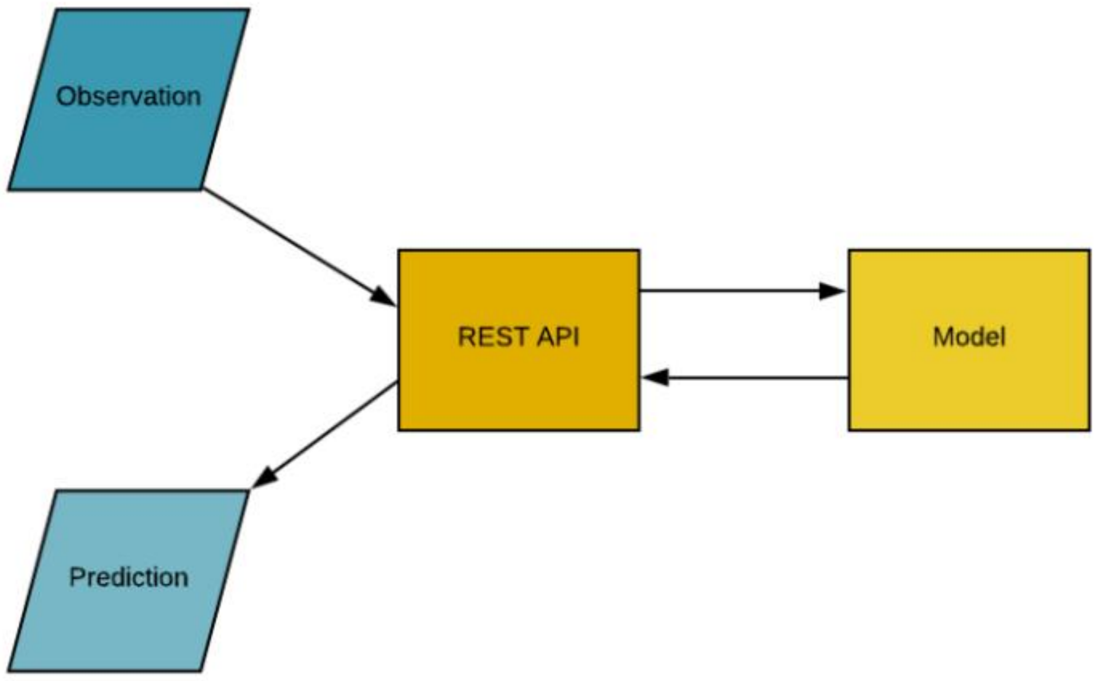


Fig 3.6: A REST API endpoint in practice

Fig 3.6 illustrates exactly what is meant by a REST API call. It acts as an intermediary between incoming/outgoing data and our machine learning/model. The client has the ability to post requests to our API endpoint, the machine learning model/extraction pipeline would have the ability to serve those requests, and the API would communicate the result of the served request back to the client. The medium of communication for the request-response workflow is often done through the means of a JSON file. A JSON file is nothing but a dictionary – it is a collection of key-value pairs ordered in a specific manner, where the ordering represents one or more key traits of the data that is being processed in the API call. The advantage of using such an architecture is that the external client can use this API endpoint and integrate it seamlessly into their own internal tools, applications or interfaces for internal use and documentation.

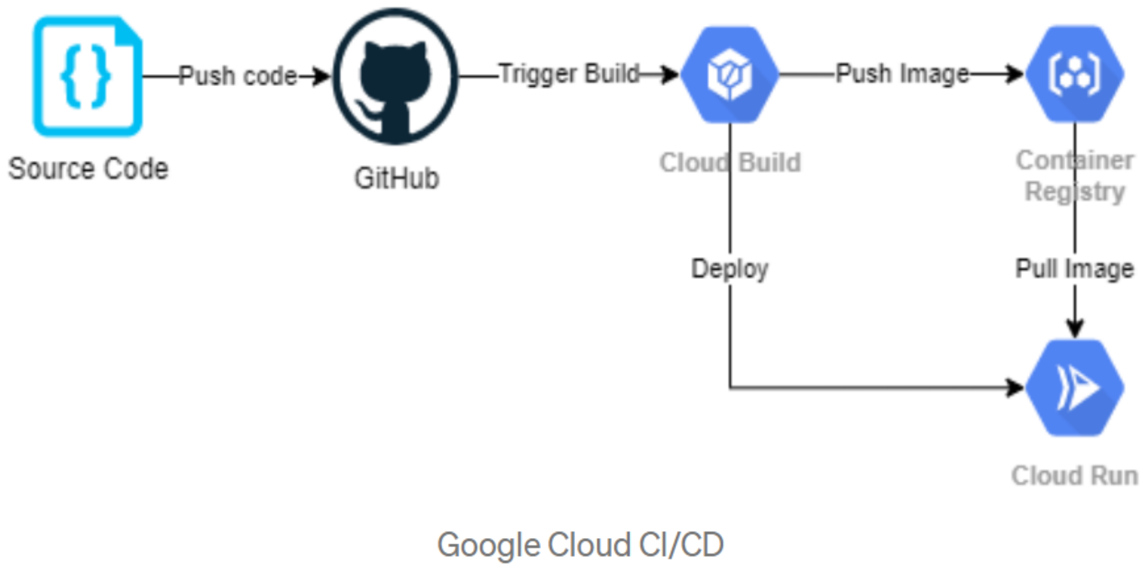


Fig 3.7: Architecture of a CI/CD pipeline deployed on Google Cloud

Fig 3.7 shows the complete architecture of a CI/CD pipeline created with the help of Cloud Build and Cloud Run. When maintaining a machine learning web app in production, a lot of things are subject to change – we may want to add stuff to our webapp UI or in contrast, remove features that create an overhead in terms of latency. We may want to change the traffic split of our machine learning model and integrate some sort of A/B test. We may have discovered a new development in the research community as a result of which we want to integrate a newly developed and trained machine learning model into the production environment and replace the old model. A lot of these changes at a high frequency can complicate re-deployment and maintenance of our codebase.

Developers will be stuck hunting through different versions of their code and will tear themselves apart if asked to manually re-integrate and deploy every step of the pipeline after the changes that are made.

Hence arrives the value of a Continuous Integration and Continuous Delivery pipeline – it strives to integrate changes to our code and automate re-deployment on demand without the advent of human intervention. Any change to our source code, say, on a developer's local environment or laptop, when pushed to a source code repository such as Github or Bitbucket, will be automatically integrated with the codebase through a Cloud Build Trigger – which would consequently lead to re-running of each of the individual steps in the pipeline and hence, re-deployment. Thus, a CI/CD pipeline effectively reduces headaches in maintaining production-ready code.

An interesting concept of note in our CI/CD pipeline is the pushing and pulling of docker images.

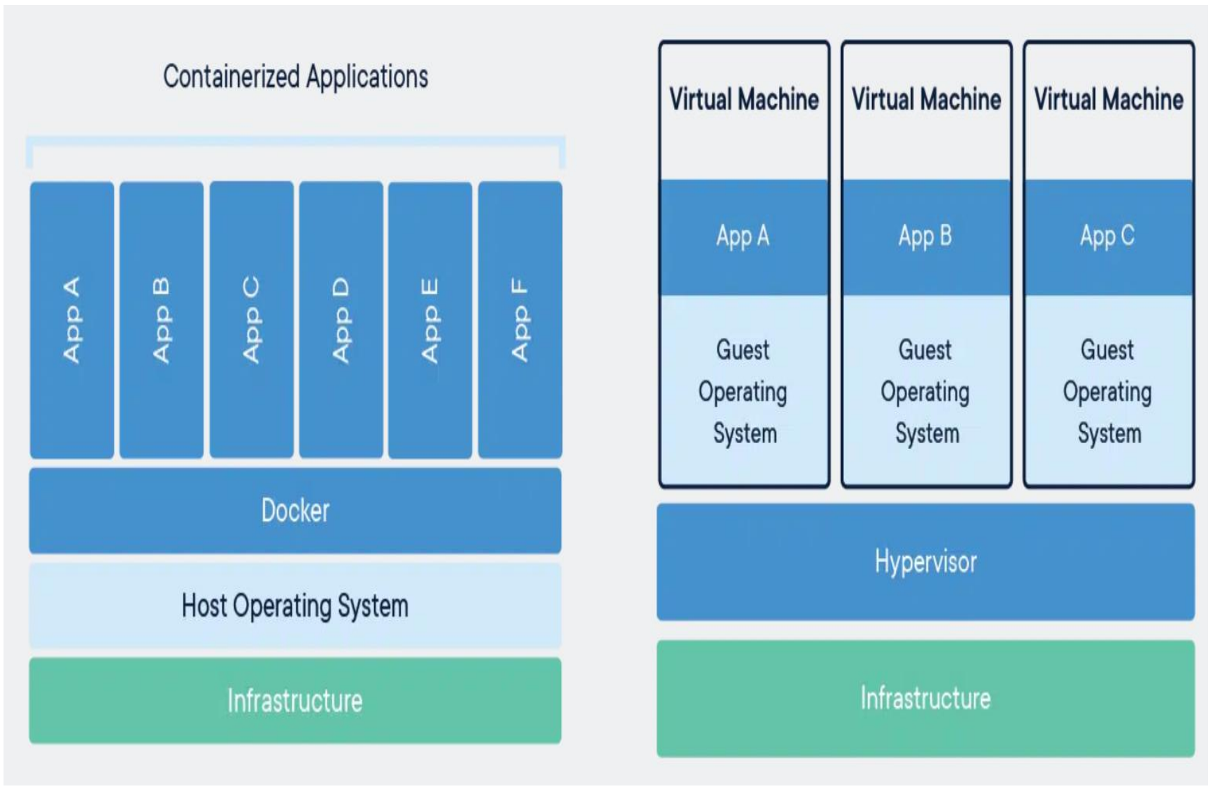


Fig 3.8: Docker containerisation

On the right half in Fig 3.8, is the template for software delivery prior to the arrival of containerisation tools – a hardware architecture such as a hypervisor would be responsible for provisioning compute resources for every set of deployments and it would do so by requisitioning one application or service on every VM with specific OS for each VM. This led to two huge burdens: -

1) One app or service per VM leads to large amounts of utilisation waste i.e., we waste a large amount of the compute resources allotted to us.

2) We would have to tailor OS-specific deployments and take special care with respect to our dependencies as there may be dependency issues between each OS.
   For example, darknet53, an object detection backbone developed by Facebook is readily available and easily installable for macOS and Linux operating systems,

however, the same object detection backbone is not available on Windows. This would lead to dependency issues if an app or a service with darknet53 object detection backbone as one of its dependencies were to be deployed on a VM running Windows.

Another figure that can explain dockerization is listed below in Fig 3.9: -

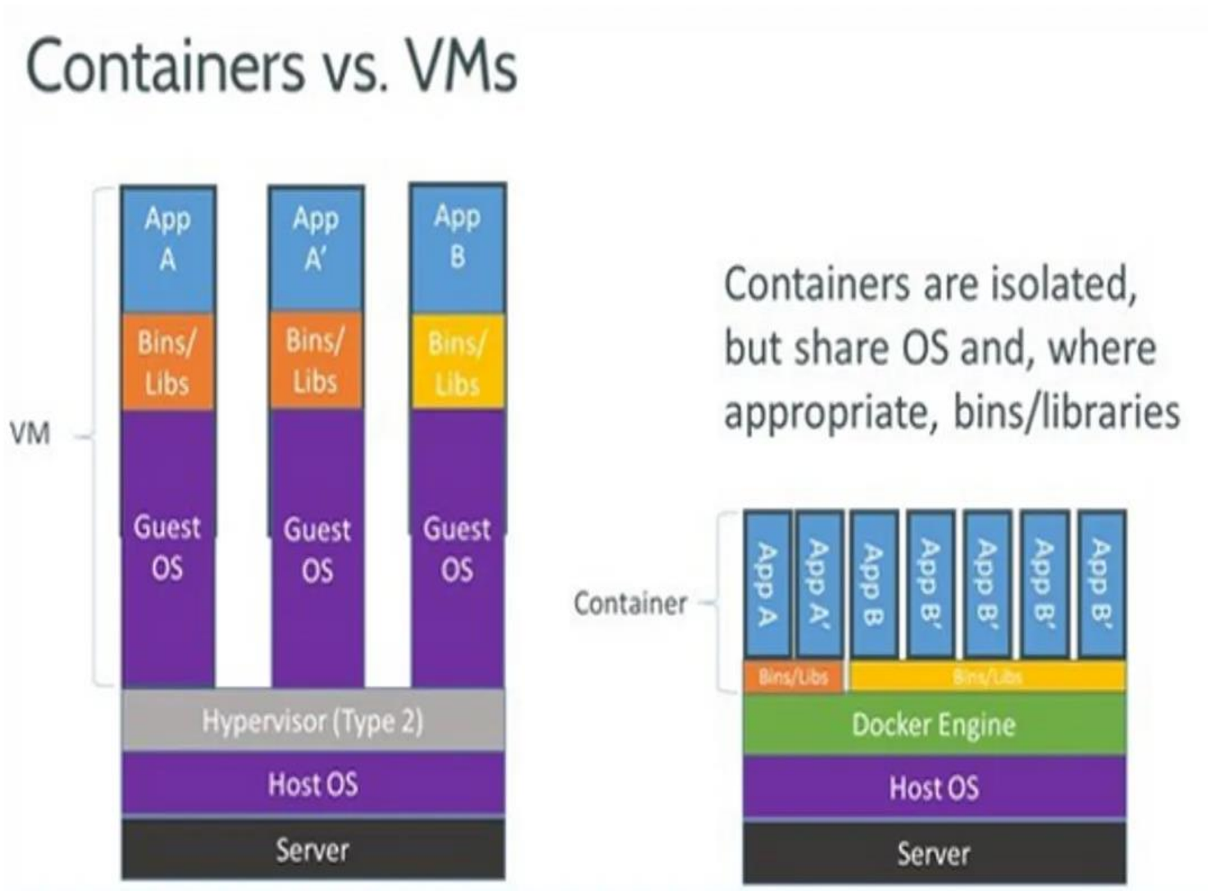


Fig 3.9: Docker containers in the real world

CHAPTER 4
RESULT ANALYSIS

4.1 Introduction

In this chapter, we shall introduce the learning curves that we have obtained after training and validation of the neural network in Fig 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8:

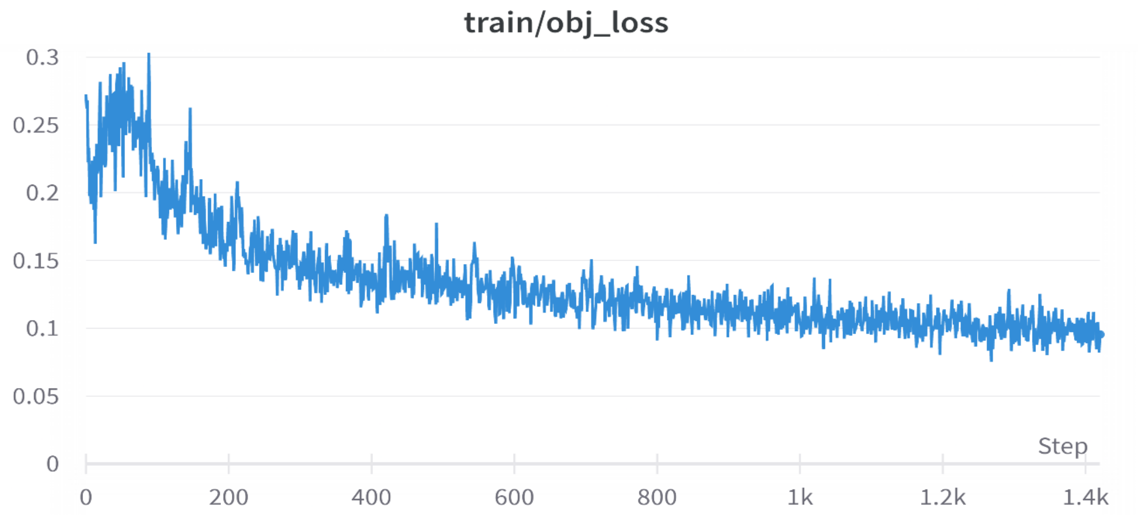


Fig 4.1: Train object or confidence loss

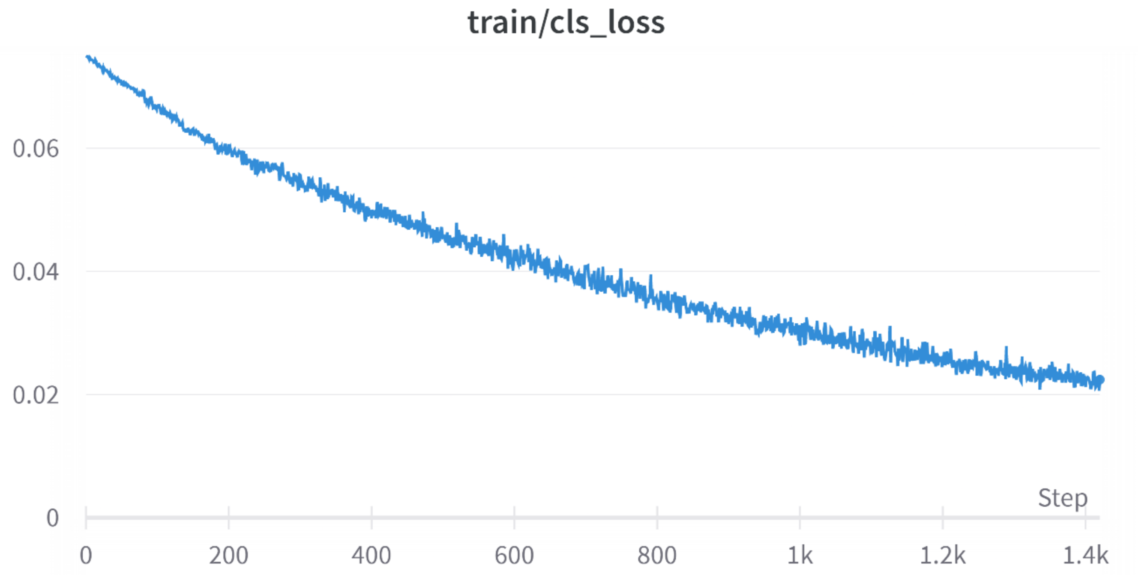


Figure 4.2: Train class or label loss

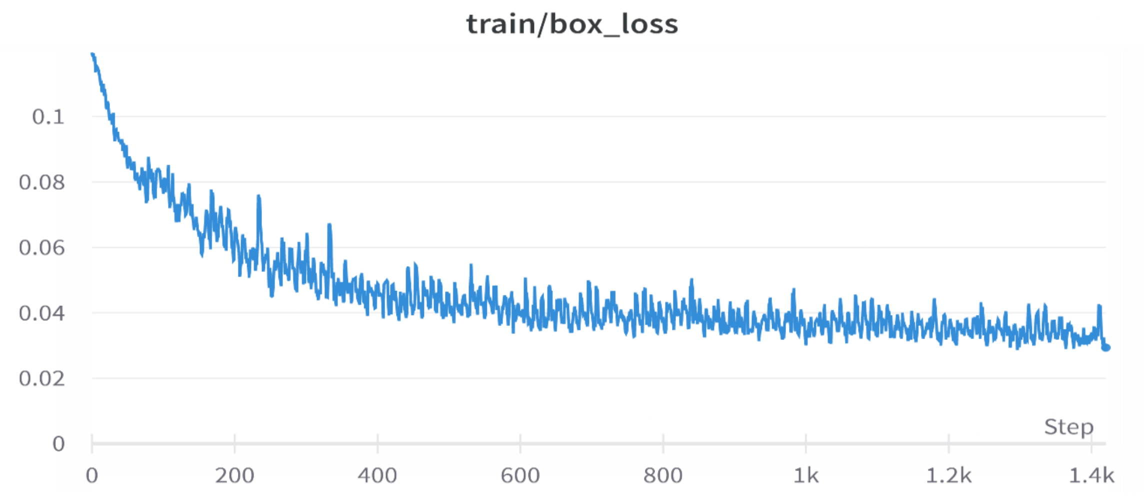


Fig 4.3: Train bounding box regression loss

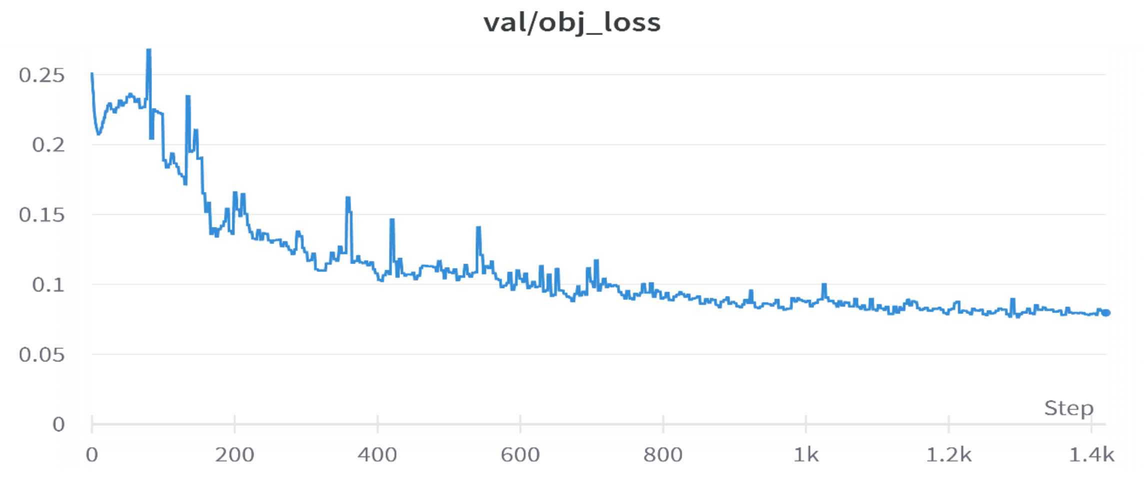


Fig 4.4: Validation object or confidence loss

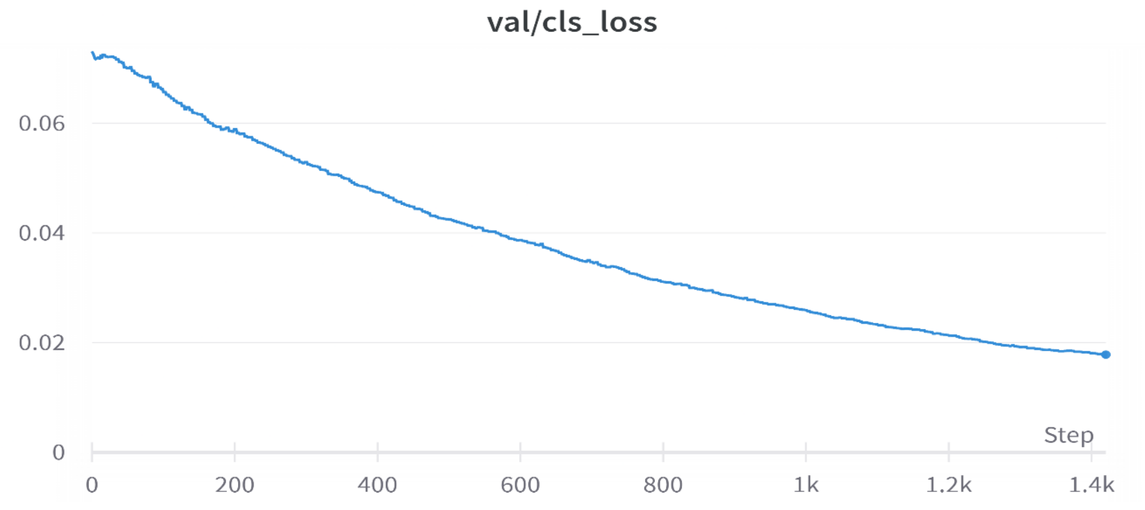


Fig 4.5: Validation class or label loss

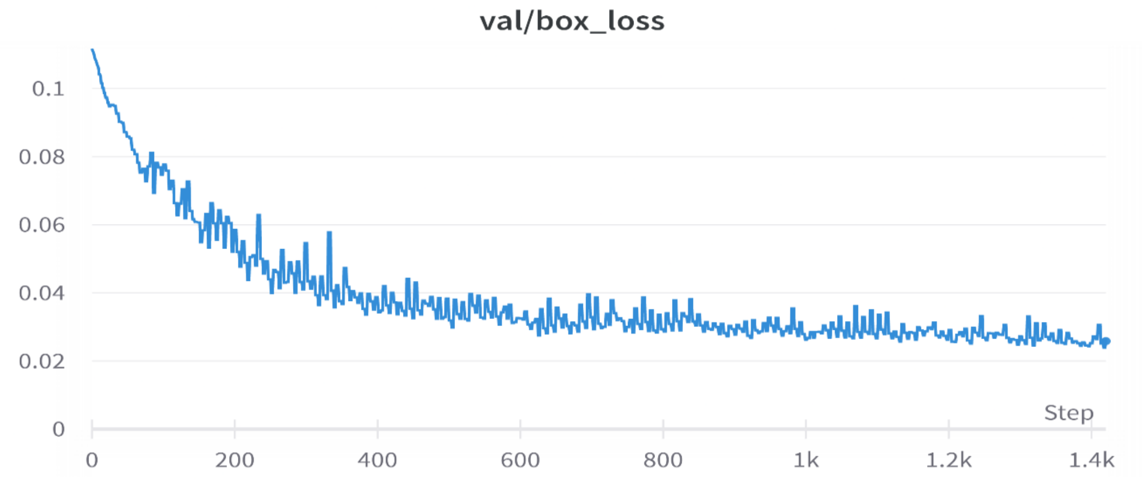


Fig 4.6: Validation bounding box regression loss

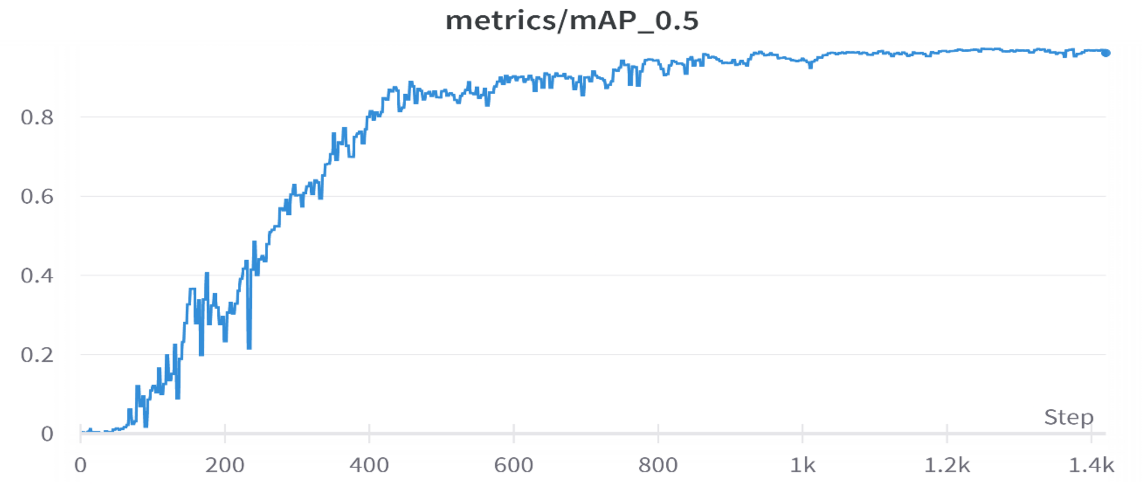


Fig 4.7: mean average precision @ 0.5 IOU [26]

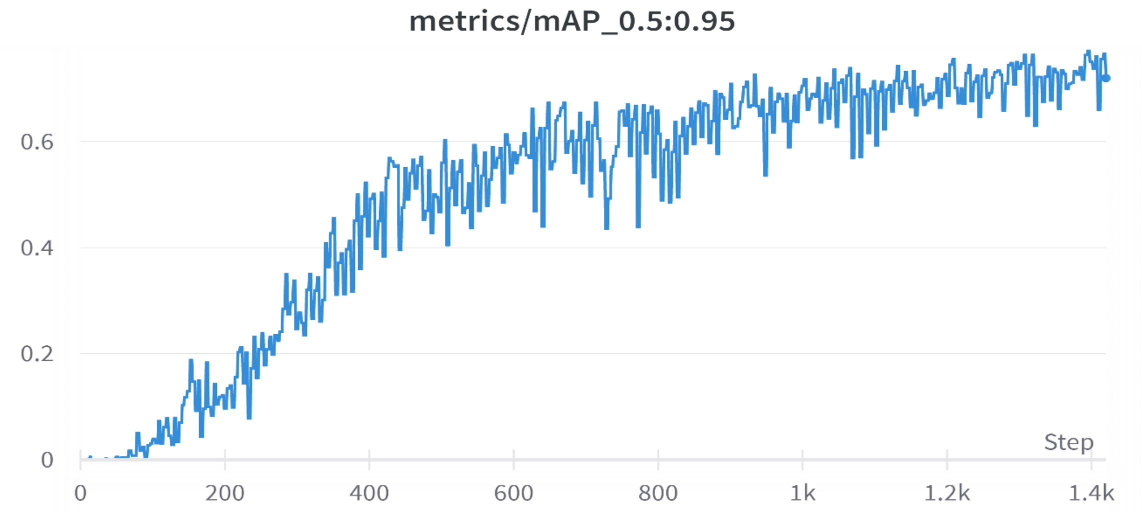


Fig 4.8: mean average precision @ 0.5:0.95 IOU [27]

4.2 Conclusions from learning curves and final results

From the given learning curves, we can conclude that our training process was carried out as we expected it to theoretically – there is a gradual decrease in loss values for both training and validation and a gradual increase in precision on the validation set which signifies that our network generalizes well to unseen data. The oscillations in said learning curves are indicative of stochastic gradient descent.

The final results of mean average precision and inference speed on both validation and test set are reported as follows in tables 4.1 and 4.2: -

Table 4.1: Mean average precision

Data set	mAP@0.5	mAP@0.5:0.95
Validation (all classes)	96.9	77.2
Testing (all classes)	92.1	68.3

Table 4.2: Inference speed

Data set	Inference speed (s/image)
Validation	1.21
Testing	1.05

Thus, our model is able to maintain a high degree of precision at a good speed both of which generalize well for unseen data.

After integrating the tesseract OCR and text extraction part of our pipeline, the results are listed in table 4.3: -

Table 4.3: Final result

Data set	Mean extraction speed (s/image)
Validation	3.47
Testing	3.43

So not only are we achieving a high degree of mean average precision but we are also well within our mandated limit of 10s.

Thus, we have delivered on key performance metrics for a valuable business outcome.

CHAPTER 5
CONCLUSION AND FUTURE SCOPE OF WORK

5.1 Summary and conclusions

The problem statement defined for the purpose of this report was to automate the extraction of 15 specific fields from IT Return Acknowledgement forms.
In order to creative an effective and accurate pipeline for the same, an object detection approach was proposed in order to make region proposals for the detected fields, which upon extraction was fed into SOTA OCR in order to facilitate text extraction.
In conclusion, not only has the work so far met the target specifications but it is also generalizable and robust. It enables us to automate document processing and comprehension and drive business value by reducing latency and avoiding customer drop-offs.

5.2 Future scope of work

The following changes can be accommodated to this extraction pipeline over time in order to make it truly robust:

1) Hyperparameter tuning: In order to leverage better performance from our trained model we could explore the possibility of kickstarting a hyperparameter tuning job on Google's Vertex AI service in order to gain incrementally better performance for this automated approach

2) Integrate Continuous training: The format/ template of these ITRA forms might change with time, so we would have to explore the possibility of integrating a continuous training pipeline with performance monitoring and drift detection in order to detect any skew whether it be on the side of the model or the data. In such cases, a retraining approach is employed on the completion of which, the re-trained model is re-deployed into the production pipeline

3) Eventually migrate from Cloud Run to Google Kubernetes Engine: At its core, Cloud Run is high-level and "serverless" which means the infrastructure is managed for us and not under our complete control – while this is a great way to make your architecture accessible, in the long run, we need service reliability for our clients with the ability for our compute instances to scale-up or scale-down as the case may be.

Thus, we will eventually create deployment manifests in order to move our entire workload to a Kubernetes cluster and integrate this with our CI/CD pipeline.
This will help us manage scalability and demand as the warranted.

REFERENCES

[1]. Document AI Industry, edgeverve.com (Web reference)

[2]. Zhiqiang, W., & Jun, L., “A review of object detection based on convolutional neural network.”, In 2017 36th Chinese Control Conference (CCC), pp. 11104-11109

[3]. Tesseract OCR, research.google.com (Web reference)

[4]. YOLOv5 data augmentation (Web reference)

[5]. Zhang, Shifeng, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z. Li, "Single-shot refinement neural network for object detection.", In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4203-4212. 2018.

[6]. Recognic documentation, recognic.ai (Web reference)

[7]. B. Hariharan et al, ´ “Simultaneous detection and segmentation,” in European Conference on Computer Vision. Springer, 2014, pp. 297– 312.

[8]. Architecture for tracking football players, digital-library.theiet.org (Web reference)

[9]. Moving object detection and counting, ieeexplore.ieee.com (Web reference)

[10]. S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in Advances in neural information processing systems, 2015, pp. 91–99

[11]. Shafiee, M., Chywl, B., Li, F. and Wong, A., 2022, “Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video.”

[12]. Peiyuan Jiang, Daji Ergu, Fangyao Liu, Ying Cai, Bo Ma, “A Review of Yolo Algorithm Developments”, Procedia Computer Science,Volume 199,2022, Pages 1066-1073

[13]. Yu, Jimin, and Wei Zhang. 2021. "Face Mask Wearing Detection Algorithm Based on Improved YOLO-v4" Sensors 21, no. 9: 3263.

[14]. Zou, Zhengxia, et al. "Object detection in 20 years: A survey." arXiv preprint arXiv*:*1905.05055, 2019.

[15]. Empirical Risk Minimization, arxiv.org (Web reference)

[16]. Torch Vision Models, pytorch.org (Web reference)

[17]. YOLOv5 documentation, docs.ultralytics.com (Web reference)

[18]. YOLOv5 models, github.com (Web reference)

[19]. YOLOv5 performance, github.com (Web reference)

[20]. CVAT, cvat.org (Web reference)

[21]. Stochastic Gradient Descent, datascience-enthusiast.com (Web reference)

[22]. OCR Template and Block Diagram, researchgate.net (Web reference)

[23]. Akhil, S. "An overview of tesseract OCR engine.", A seminar report, Department of Computer Science and Engineering National Institute of Technology, Calicut Monsoon, 2016.

[24]. Yousefi, Jamileh, "Image binarization using otsu thresholding algorithm.", Ontario, Canada: University of Guelph, 2011.

[25]. Smith, Ray. "An overview of the Tesseract OCR engine." In Ninth international conference on document analysis and recognition, ICDAR 2007, vol. 2, pp. 629-633. IEEE, 2007.

[26]. Mean Average Precision, towardsdatascience.com (Web reference)

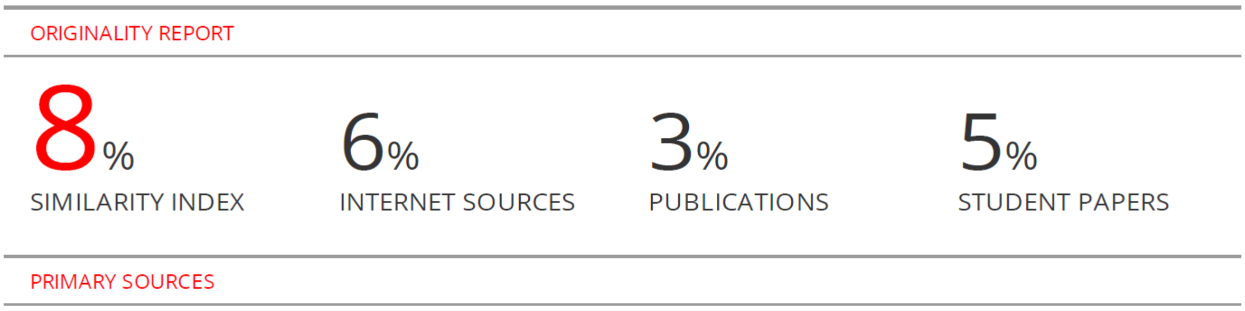
[27]. COCO Metrics, cocodataset.org (Web reference)

PROJECT DETAILS

Student Details			
Student Name	**ABHIMANYU BORTHAKUR**		
Register Number	180907742	Section / Roll No	D/53
Email Address	abhimanyuborthakur@hotmail.com	Phone No (M)	9007063439
Student Name	**N/A**		
Register Number		Section / Roll No	
Email Address		Phone No (M)	
Project Details			
Project Title	**AUTOMATING DOCUMENT EXTRACTION OF ITRA FORMS WITH ML**		
Project Duration	6 Months	Date of reporting	10/01/2022
Expected date of completion of project	July 2022		
Organization Details			
Organization Name	**Searce Cosourcing Services Pvt Ltd.**		
Full postal address with pin code	6th Floor, Qubix Business Park-Phase 1, Hinjewadi, Pune - 411057, Maharashtra, India		
Website address	www.searce.com		
Supervisor Details			
Supervisor Name	**MR SAMIR DADIA**		
Designation	VP – PS, ADMIN AND IT		
Full contact address with pin code	samir.dadia@searce.com		

Email address	samir.dadia@searce.com	Phone No (M)	N/A
Internal Guide Details			
Faculty Name	**Dr PRAMOD KUMAR**		
Full contact address with pin code	Dept. of E&C Engg., Manipal Institute of Technology, Manipal – 576 104 (Karnataka State), INDIA		
Email address	p.kumar@manipal.edu		

PLAGIARISM CHECK REPORT



ORIGINALITY REPORT

8%	6%	3%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES