

# Extract Information From Aadhar/Pan Card Using OCR

---

---

Asmani Deep Vipul - 4170

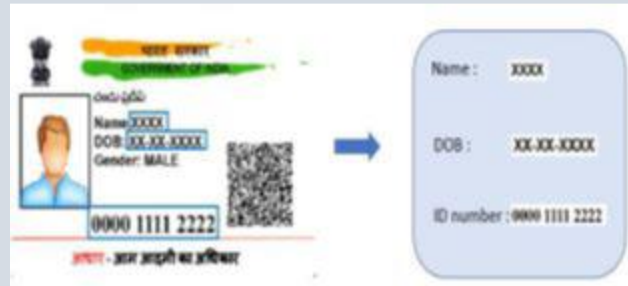
Babariya Yash Vipulbhai - 4171

## Table of Contents

1. Introduction.....	- 2 -
2. Background.....	- 3 -
3. Related work.....	- 4 -
a. Text detection.....	- 4 -
b. Text recognition.....	- 5 -
4. Dataset and Features .....	- 6 -
a. Description.....	- 6 -
b. Dataset collection method.....	- 6 -
i. Training YOLO using the Darknet framework.....	- 6 -
ii. Get data first.....	- 6 -
iii. Data Annotation .....	- 6 -
iv. Training.....	- 7 -
5. Methods.....	- 8 -
6. Results/Experiments .....	- 10 -
7. Conclusion/Future Work.....	- 13 -
Acknowledgement .....	- 13 -
8. References.....	- 14 -
9. Certificate.....	- 15 -

## 1. Introduction

OCR is one of the most critical issues in the domain of computer vision. Different techniques have been proposed in past literature but with the advancement of deep learning techniques, now it's possible to use OCR capabilities for real world environment. If we want to use OCR for real world applications, then accuracy is important concern.



Picture 1: Basic OCR process

KYC is best example for this where we want to onboard verified user. In India, for kyc user must supply Aadhar card, pan card etc., as identity document. Now there are unusual ways to confirm user using identity documents. Manual process is time taking so organizations are moving towards AI and blockchain based methods to enhance user experience. Generally different companies are providing mobile app for on boarding process where user can upload identity document. Within the app, deep learning based ocr technique is used to extract essential information from document. Now the accuracy of text extraction is important because if there exist way for editing then it's not possible completely to validate the authenticity. To avoid this specific issue, we developed ocr pipeline where we can extract and auto-fill the desired information from identity documents. We focused on three majors

techniques i.e., object detection, text detection and basic image processing to enhance the accuracy for text extraction from identity documents. Simplest way to achieve the better results includes image processing techniques like noise removal, Dilation/ erosion, rotation/Deskewing etc.

Second approach used by us is object detection, by using the same we can find the regions and then extract information from those regions. Third important method is text detection where we can design bounding boxes in input image where text is present and then extraction procedure will work.

## 2. Background

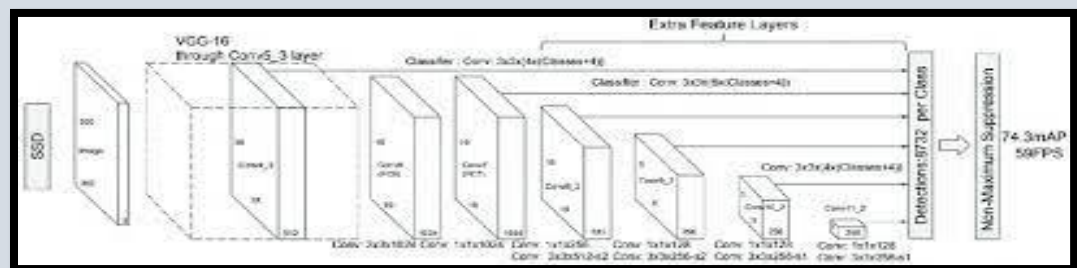
We used image processing methods and Adaptive Thresholding. There are various other image processing techniques available but for our use case, we found mentioned techniques are performing well. Contour is the process of connecting same intensity or color points. We used opencv for our task. Adaptive thresholding is the technique for finding threshold of smaller regions that can further be used for character recognition

In Text recognition, for extracting information from different identity documents such as ADHAAR, PAN, etc., we broke down the problem into four distinct stages:

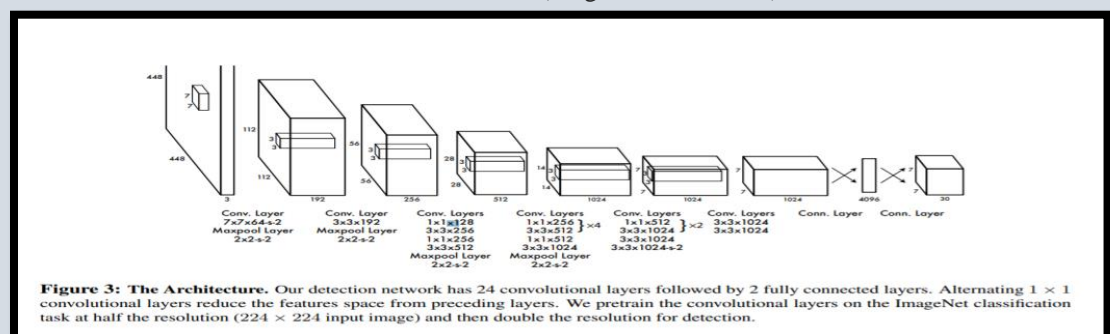
1. Collecting & creating a dataset of different IDs from various sources
2. Finding regions of interest from the acquired images and making bounding boxes around the areas of text
3. Training our deep learning algorithm to find such regions
4. Performing OCR on the identified region of interests

Different IDs such as Aadhaar, PAN, passport, etc. is collected from various sources

and a proper dataset is created. To create dataset for our deep learning algorithm we created bounding boxes around our areas of interests (text) using Labeling. The dataset created in such a way that the name of annotation files being created is the same as the image files, and the annotation files are in a different folder as to the image files and in the same sequence arranged as the image files.



Picture: SSD (Single Shot Detector) architecture



Picture: YOLO (you only look once) Architecture

### 3. Related work

OCR has two major building blocks:

- a. Text detection
- b. Text recognition

#### a. Text detection

Our first task is to detect the required text from images/documents. Often, as the need is, you don't want to read the entire document, rather just a piece of information like credit card number, Aadhaar/PAN card number, name, amount and date from bills, etc. Detecting the required text is a tough task but thanks to deep learning, we'll be able to selectively read text from an image.

Text detection or in general object detection has been an area of intensive research accelerated with deep learning. Today, object detection, and in our case, text detection, can be achieved through two approaches.

- Region-Based detectors
- Single Shot detectors

In Region-Based methods, the first goal is to find all the regions which have the objects and then pass those regions to a classifier, which gives us the locations of the required objects. So, it is a two-step process.

Firstly, it finds the bounding box and afterwards, the class of it. This approach is considered more exact but is comparatively slow as compared to the Single Shot approach. Algorithms like Faster R-CNN and R-FCN take this approach.

Single Shot detectors, however, predict both the boundary box and the class at the same time. Being a single step process, it is much faster. However, it must be noted that Single Shot detectors perform badly while detecting smaller objects. SSD and YOLO are Single Shot detectors.

Often, there is a tradeoff between speed and accuracy while choosing the object detector. For example, Faster R-CNN has the highest accuracy, while YOLO is fastest among all. Here is a great article which compares different detectors and provides comprehensive insights on how they work.

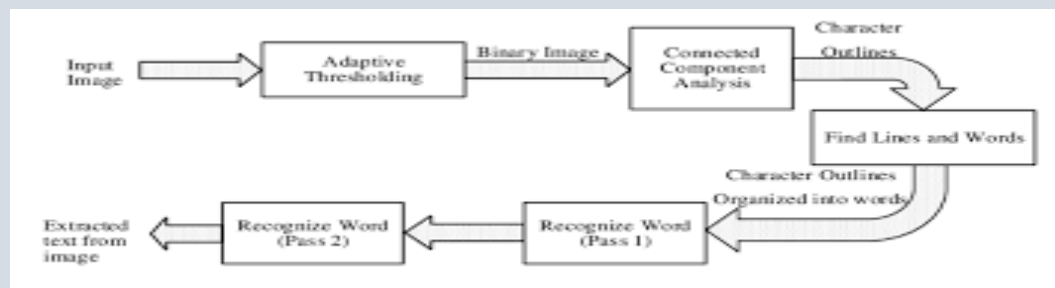
To decide which one to use, totally depends on your application. Here, we are using YOLOv3 here mainly because, No one can beat it when it comes to speed. Has good enough accuracy for our application. YOLOv3 has Feature Pyramid Network (FPN) to detect small objects better.

## b. Text recognition

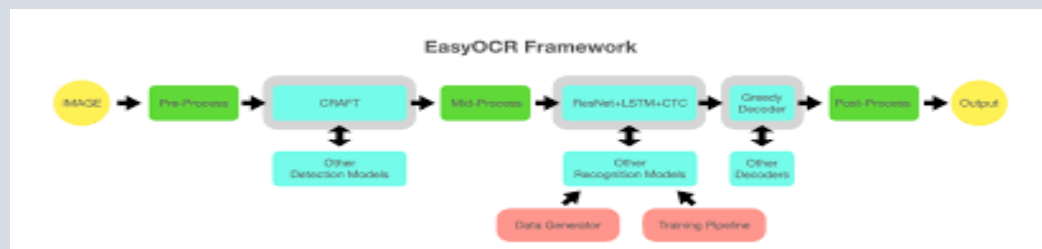
Now that we have our custom text detector implemented for text detection, we move onto the later process of Text Recognition. You can either build your own text recognizer or use an open-sourced one.

Although, it is a great practice to implement your own text recognizer, it is challenging to get the labelled data for it. However, if you already have a lot of labelled data to create your custom text recognizer, it'll certainly improve the accuracy.

In this article, however, we are going to use the Tesseract OCR, EasyOCR engine for text recognition. With only a few tweaks, the Tesseract OCR engine works wonders for our application. We are going to use Tesseract 4, which is the latest version. Thankfully, it also supports many languages. The EasyOCR package is created and supported by Jaidev AI, a company that specializes in Optical Character Recognition services. EasyOCR is implemented using Python and the PyTorch library. If you have a CUDA-capable GPU, the underlying PyTorch deep learning library can speed up your text detection and OCR speed tremendously. As of this writing, EasyOCR can OCR text in 58 languages, including English, German, Hindi, Russian, and more! The EasyOCR maintainers plan to add other languages in the future.



Tesseract Framework



EasyOCR Framework

## 4. Dataset and Features

### a. Description

YOLO is an ultramodern, real-time object detection network. There are many versions of it. YOLOv3 is the most recent and the fastest version.

YOLOv3 uses Darknet-53 as its feature extractor. It has overall 53 convolutional layers, hence the name 'Darknet-53'. It has successive  $3 \times 3$  and  $1 \times 1$  convolutional layer and has some shortcut connections.

For classification, independent logistic classifiers are used with the binary cross-entropy loss function.

### b. Dataset collection method

#### i. Training YOLO using the Darknet framework

We will use the Darknet neural network framework for training and testing. The framework uses multi-scale training, lots of data augmentation and batch normalization. It is an open-source neural network framework written in C and CUDA. It is fast, easy to install, and supports CPU and GPU computation.

#### ii. Get data first

Data is the first and most important thing in any machine learning based project. So, whatever is your application make sure you have around 100 images for it. If you have a fewer number of images, then use image augmentation to increase the size of your data. In image augmentation, we basically alter images by changing its size, orientation, light, color, etc.

There are many methods available for augmentation, and you can very easily pick any method you like. we would like to mention an image augmentation library called Augmentations, build by Kaggle Masters and Grandmaster.

We collected 50 images of PAN Card floating on the internet, and using image augmentation, created a dataset of 100 PAN card images.

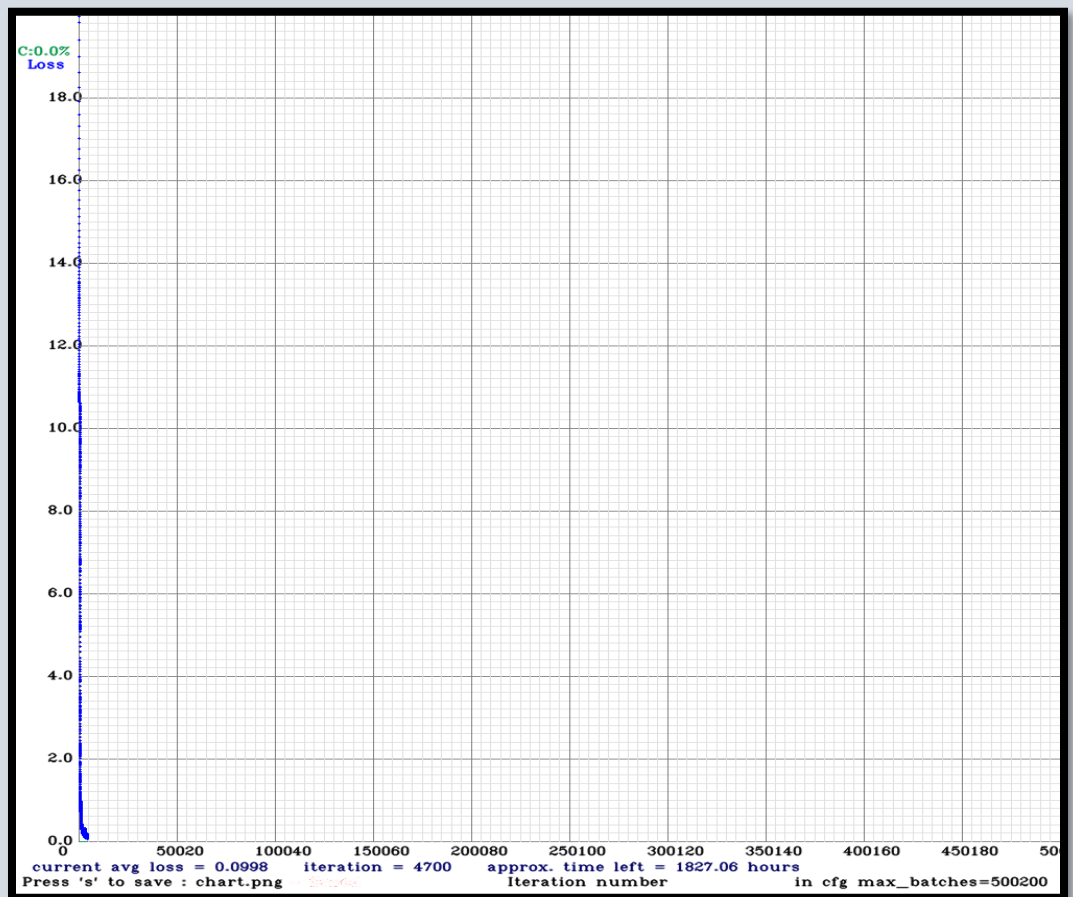
#### iii. Data Annotation

Once we have collected the data, let's move to the next step, which is to label it. There are many free data annotation tools available. we used VoTT v1 because it is a simple tool and works like a charm. it is important we tag all the text fields that we want to read from the image data. It also generates the data folders which will be needed during training.

#### iv. Training

To start training our OCR, we first need to change our config file. You will get your required config file in 'cfg' folder named 'yolov3.cfg'. Here, you need to change the batch size, subdivision, number of classes and filter parameters. Follow the required changes needed in the config file, as given in the documentation.

We will start training with pre-trained weights of darknet-53. This will help our model converge early. The best thing is it has multi GPU support. When you see that average loss '0. xxxxxx avg' no longer decreases after a certain number of iterations, you should stop training. As you can see in the below chart, we stopped at 20000 iterations as the loss became constant.

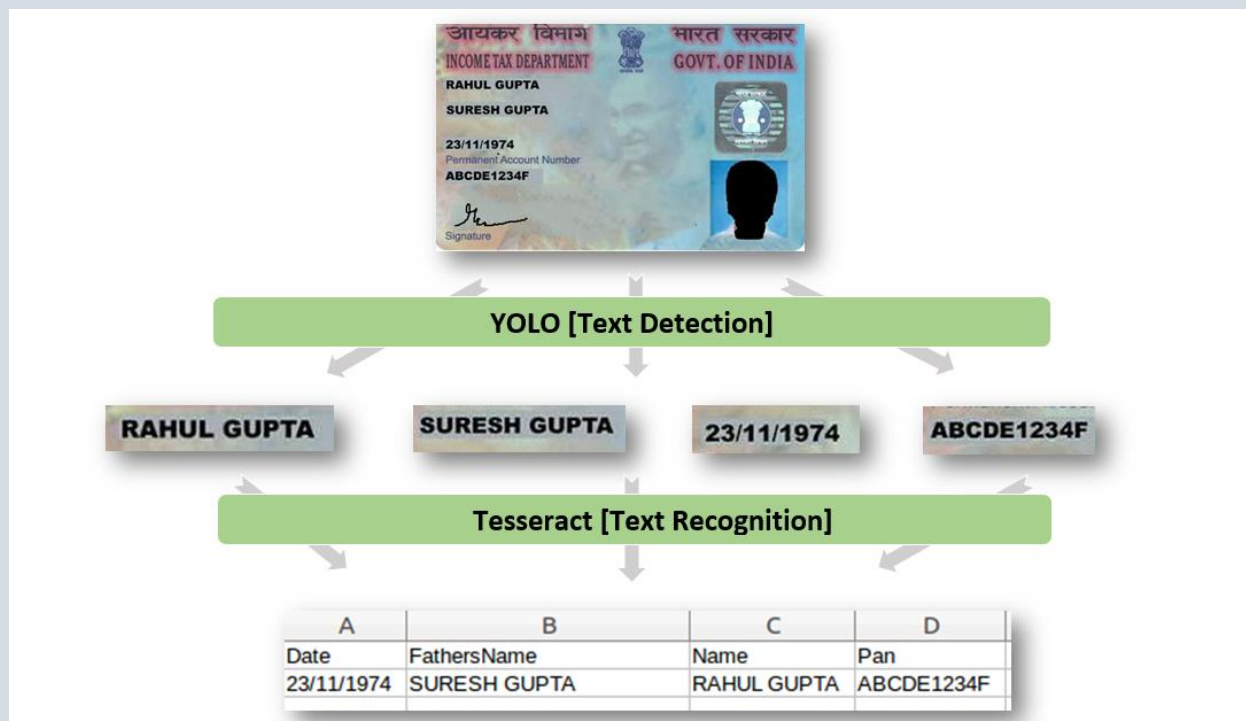




## 5. Methods

Once we've implemented the process of text detection and text recognition, it is time to combine them to achieve the following flow:

- i. Detect the required region from the image
- ii. Pass that detected regions to Tesseract
- iii. Store the results from Tesseract in your required format

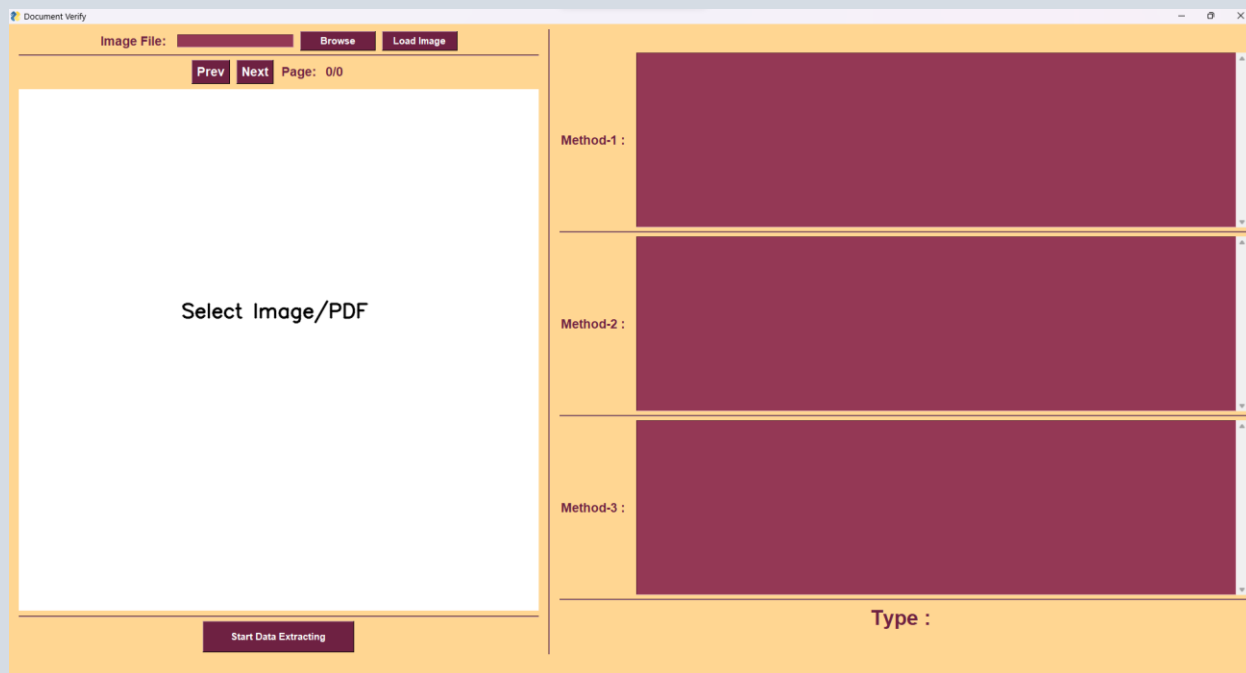


From the above diagram, you can understand that, first the image of pan card is passed into YOLO. Then, YOLO detects the required text regions and crops them out from the image. Later, we pass those regions one by one to tesseract. Tesseract reads them, and we store that information. Now, to be the results you can choose any form of choice. Here, we used a json to show the results.

Text detection algorithms are getting matured using deep learning. There are various challenges in text detection which includes image noise, viewing angles, blurring effects, Lighting conditions, resolution, non-paper objects, non-planar objects unknown layout etc. Literature of text detector algorithms is growing; we took various algorithms for analysis. Finally, we selected east text detector and CRAFT text detector for pipeline of OCR for identity documents.



## 6. Results/Experiments



Document Verify

Image File: D:/OPENCV/data/pancard18.jpg Browse Load Image

Prev Next Page: 1/1



Method-1 :

```
[[{"pan_sign": "", "pan_number": "AJDPY2700K", "pan_dob": "", "pan_father_name": "RAMNARESH YADAV", "pan_name": "BABLU PHASAD YADAV"}]]
```

Method-2 :

```
[[{"pan_sign": "aest-VQ1?3Z", "pan_number": "AJDPY2700K", "pan_dob": "01/05/1987", "pan_father_name": "RAMNARESH YADAV", "pan_name": "BABLU PRASAD YADAV"}]]
```

Method-3 :

Type :

Start Data Extracting

Save\_Data\_tesseract - Notepad

File Edit View

```
[
  {
    "pan_sign": "",
    "pan_number": "AJDPY2700K",
    "pan_dob": "",
    "pan_father_name": "RAMNARESH YADAV",
    "pan_name": "BABLU PHASAD YADAV"
  }
]
```

Save\_Data\_easyocr - Notepad


File Edit View

```
[
  {
    "pan_sign": "aest-VQ1?3Z",
    "pan_number": "AJDPY2700K",
    "pan_dob": "01/05/1987",
    "pan_father_name": "RAMNARESH YADAV",
    "pan_name": "BABLU PRASAD YADAV"
  }
]
```

Document Verify

Image File: D:\OPENCV\data\pancard\A2 Browse Load Image

Prev Next Page: 1/1



Method-1 :

```
[[{"adhar_gender": "W / MALE", "adhar_name": "Riyasat", "adhar_number": "6754 3973 8680", "adhar_dob": "01/01/1991"}]]
```

Method-2 :

```
[[{"adhar_gender": "MALE", "adhar_name": "Riyasat", "adhar_number": "6754 3973 8680", "adhar_dob": "01/01/1991"}]]
```

Method-3 :

Type :

Start Data Extracting

Save\_Data\_tesseract - Notepad

File Edit View

```
[
  {
    "adhar_gender": "W / MALE",
    "adhar_name": "Riyasat",
    "adhar_number": "6754 3973 8680",
    "adhar_dob": "01/01/1991"
  }
]
```

Save\_Data\_easyocr - Notepad

File Edit View

```
[
  {
    "adhar_gender": "MALE",
    "adhar_name": "Riyasat",
    "adhar_number": "6754 3973 8680",
    "adhar_dob": "01/01/1991"
  }
]
```

## 7. Conclusion/Future Work

In this paper we proposed images processing, identification and OCR which can effectively perform the task of character recognition for identity documents. Use of image processing techniques, objection detection algorithms (SSD) and text detection procedures (CRAFT) produced a pipeline for real world application. In future, we will create datasets by collecting more samples of identity documents in different conditions for training different models in pipeline. Deep learning architectures are evolving with extremely fast pace, which will be helpful for designing robust system. In future, we will develop deep network for our use case. To extend proposed ocr versions for different use cases is our second priority. Current work will be extremely useful for industrial or academic purpose.

### Acknowledgement

This project is fully controlled by CodeSequel Inc.



## 8. References

1. <https://signy.io/#/>, Last visited: 29 January 2020.
2. Brzeski, Adam, et al. "Evaluating performance and accuracy improvements for attention-OCR." IFIP International Conference on Computer Information Systems and Industrial Management. Springer, Cham, 2019.
3. Saluja, Rohit, et al. "OCR On-the-Go: Robust End-to-end Systems for Reading License Plates & Street Signs." 15th IAPR International Conference on Document Analysis and Recognition (ICDAR). 2019.
4. Achanta, Rakesh, and Trevor Hastie. "Telugu OCR framework using deep learning." arXiv preprint arXiv:1509.05962 (2015).
5. Namysl, Marcin, and Iuliu Konya. "Efficient, lexicon-free OCR using deep learning." arXiv preprint arXiv:1906.01969 (2019).
6. Wei, Tan Chiang, U. U. Sheikh, and Ab Al-Hadi Ab Rahman. "Improved optical character recognition with deep neural network." 2018 IEEE 14th International Colloquium on Signal Processing & Its Applications (CSPA). IEEE, 2018.
7. Kundaikar, Teja, and Jyoti D. Pawar. "Multi-font Devanagari Text Recognition Using LSTM Neural Networks." First International Conference on Sustainable Technologies for Computational Intelligence. Springer, Singapore, 2020.
8. Chandrakala, H. T., and G. Thippeswamy. "Deep Convolutional Neural Networks for Recognition of Historical Handwritten Kannada Characters." Frontiers in Intelligent Computing: Theory and Applications. Springer, Singapore, 2020. 69-77.
9. Ali, Hazrat, et al. "Pioneer dataset and automatic recognition of Urdu handwritten characters using a deep autoencoder and convolutional neural network." SN Applied Sciences 2.2 (2020): 152.
10. Springmann, Uwe, et al. "OCR of historical printings of Latin texts: problems, prospects, progress." Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage. 2014.
11. Wang, Jianfeng, and Xiaolin Hu. "Gated recurrent convolution neural network for ocr." Advances in Neural Information Processing Systems. 2017.
12. Ahmed, Saad Bin, et al. "Deep learning based isolated Arabic scene character recognition." 2017 1st International Workshop on Arabic Script Analysis and Recognition (ASAR). IEEE, 2017.
13. Naseer, Asma, and Kashif Zafar. "Meta features-based scale invariant OCR decision making using LSTM-RNN." Computational and Mathematical Organization Theory 25.2 (2019): 165-183.
14. Maalej, Rania, and Monji Kherallah. "Improving MDLSTM for offline Arabic handwriting recognition using dropout at different positions." International conference on artificial neural networks. Springer, Cham, 2016.
15. Breuel, Thomas M. "High performance text recognition using a hybrid convolutional-lstm implementation." 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). Vol. 1. IEEE, 2017.
16. Liu, Wei, et al. "Ssd: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.
17. Zhou, Xinyu, et al. "EAST: an efficient and accurate scene text detector." Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. 2017.
18. Baek, Youngmin, et al. "Character region awareness for text detection." Proceedings of the IEEE

## 9. Certificate



### PROJECT CERTIFICATE

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Asmani Deep**, student of third year M.Sc.- IT K.S School of business management, Gujarat university has successfully completed Deep Learning training in **CodeSequel Inc (Digital IT Agency)**. He worked on an online Document Processor Tool from 2/01/2022. This is an original work, and this work has not been submitted anywhere in any form.

We found him sincere, hardworking, technically sound and result oriented. We take this opportunity to thank him and wish him all the best for him future.

PREPARED BY

Krunal Dangi (founder)

CodeSequel Inc.





## PROJECT CERTIFICATE

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Babariya Yash**, student of third year M.Sc.- IT K.S School of business management, Gujarat university has successfully completed Deep Learning training in **CodeSequel Inc (Digital IT Agency)**. He worked on an online Document Processor Tool from 2/01/2022. This is an original work, and this work has not been submitted anywhere in any form.

We found him sincere, hardworking, technically sound and result oriented. We take this opportunity to thank him and wish him all the best for him future.

PREPARED BY

Krunal Dangi (founder)

CodeSequel Inc.