# Group Project - Scene Text Recognition

Subrahmanya V Bhide (SC18B030)
*Department of Aerospace Engineering*
*Indian Institute of Space Science and Technology*
Trivandrum, India

Shashwat Gupta (SC18B027)
*Department of Aerospace Engineering*
*Indian Institute of Space Science and Technology*
Trivandrum, India

Ashrit R (SC18B125)
*Department of Electronics and Communication Engineering*
*Indian Institute of Space Science and Technology*
Trivandrum, India

Yugal Joshi (SC18B103)
*Department of Electronics and Communication Engineering*
*Indian Institute of Space Science and Technology*
Trivandrum, India

***Abstract*—This document summarises the work we have done for the Scene Text Recognition Group project and explains the ideas we went through and learnt.**

***Index Terms*—Neural network, detection, recognition, You Only Look Once, Connectionist Temporal Classification**

## I. Introduction

Given an input image with variable length labels/words, we have predicted the text in the image with a reasonable accuracy using YOLO to detect where the text is present in the image and CNN to recognise it.

## II. Initial Reading

### A. Histogram of Oriented Gradients

The image is first divided into cells and derivative masks are used to calculate gradients for pixels. These gradients are then normalised using blocks of cells. Two kinds of blocks can be chosen, i.e. rectangular and circular. A histogram of the cell orientation is made using the weighted sum of the pixels comprising the cell. These cell gradients are then sent into ML algorithms like SVM, ANN, etc. to make decisions or predictions. The main disadvantage of HOG is that it's computationally expensive and not very accurate. [1]

### B. Region Based Convolutional Neural Networks (RCNN)

Certain regions in the image are proposed as containing some objects (in our case text); initially this was done through sweep operations where a large amount of proposals were created. A selective search method is used where initial sub-segmentation of the input image is generated. Similar bounding boxes are combined into larger ones recursively and these larger boxes are used to generate region proposals for object detection.

The proposals generated by the selective search method are fed into AlexNet CNN to generate features which are then fed in to the SVM for classification and prediction. Around the same object, we would have many proposals and the best one out of them is chosen using Non-maximum suppression, where regions are eliminated using criteria such as Confidence Score and Intersection over Union. The equation used for IoU and a few examples have been depicted in Fig. 1 and Fig. 2. [2]

The major disadvantages of RCNN are that it takes a huge amount of time to train and region proposal is not trained, i.e. there is no method to train to select regions.

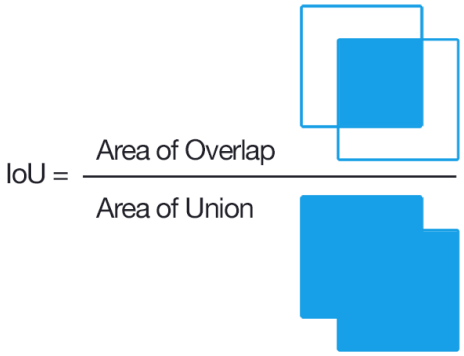


Fig. 1. Intersection over Union Equation [3]

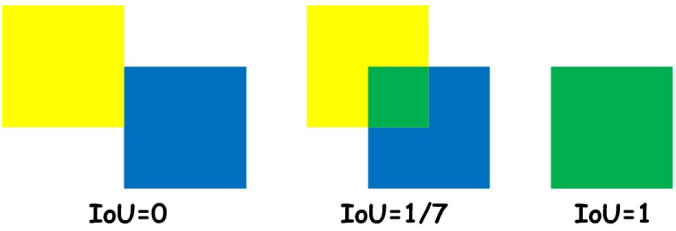


Fig. 2. Intersection over Union Examples [4]

### C. Single-Shot Detectors

Aspect ratio and size of the input image create a problem in sliding-window type of applications previously mentioned. Any Single-Shot Detector has two main components: Backbone model and SSD Head. Backbone model is a pre-trained image classification network which is typically a network like ResNet trained on ImageNet from which the final, fully connected classification layer has been removed. Thus this acts

as a feature extractor.
SSD head consists of many convolutional layers added to the backbone feature map. Due to having many convolutional layers, the layers bearing smaller receptive field can represent smaller sized objects. Because of this, SSD allows us to define a hierarchy of grid cells at different layers. For example, we could use a 4x4 grid to find smaller objects, a 2x2 grid to find mid sized objects and a 1x1 grid to find objects that cover the entire image. There are also provisions to specify aspect ratio zoom level and anchor boxes. [5]

### D. You Only Look Once

All the previously discussed algorithms give results but lack in an important field - speed. YOLO is not as accurate, but it excels in giving fast results, and is the best choice for a real-time detection system. [6]
Here, just as in SSD, a bounding box is characterized by the center of the box, width and height of the box and $p_c$ i.e. is the probability of class $c$. The input image is divided into cells, and the object is considered to lie in the cell where the center lies. After this Non-max suppression, confidence level and IoU are used to finalise the bounding box. [7]

## III. Text Recognition

### A. Convolutional Recurrent Neural Network

Convolutional Recurrent Neural Network combines Deep Convolutional Neural Networks and RNN which constructs an end-to-end system for sequence recognition.
The convolutional layers automatically extract a feature sequence from each input image.The recurrent layers are built on top of the convolutional network for making predictions for each frame of the feature sequence. The transcription layer is finally built atop the CRNN, and is adopted to translate the per-frame predictions by the recurrent layers into a label sequence. [8]
Though CRNN is composed of different kinds of network architectures, it can be jointly trained with one loss function. In the model shown in Fig. 3, the component of convolutional layers is constructed by taking the convolutional and max-pooling layers, and is used to extract a sequential feature representation from an input image. The recurrent layers predict a label distribution $y_t$ for each frame $x_t$ in the feature sequence $x = x_1; ...; xT$. Connectionist Temporal Classification (CTC) is used in our transcription process to decode the output from the RNN and convert it to a text label. [8]

### B. Connectionist Temporal Classification

The sequence labelling problem consists of input sequences $X = [x_1, x_2, ...x_T]$ and its corresponding output sequences $Y = [y_1, y_2, ..., y_U]$. An accurate mapping from X's to Y's needs to be found, but both X and Y can vary in length and we don't have an accurate correspondence of the elements (X and Y). For a given X, the CTC algorithm gives us an output distribution over all possible Y's. We can use this distribution either to infer a likely output or to assess the probability of a given output. [8]

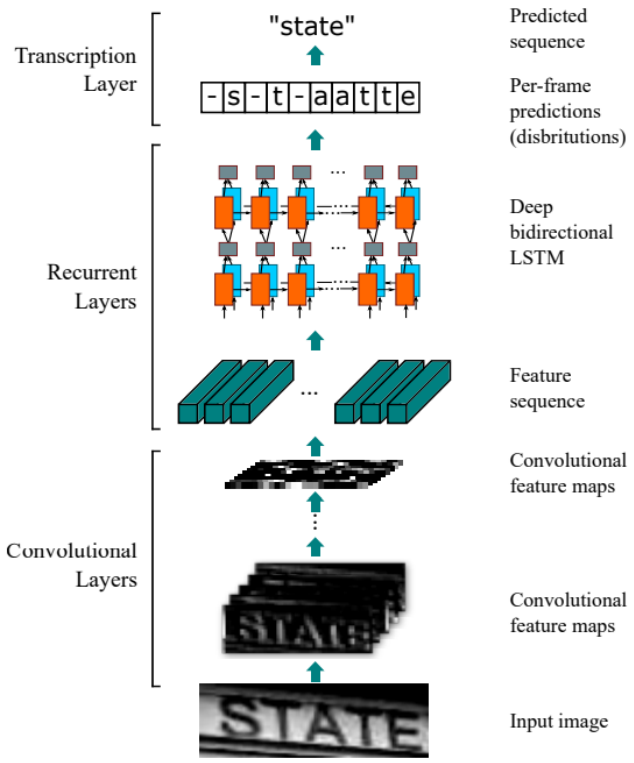


Fig. 3. Text Recognition Model [9]

## IV. Pipeline Developed for Scene Text Recognition

The problem of scene text recognition was divided into two parts for execution. Figure 4 shows the pipeline we have followed for our project.

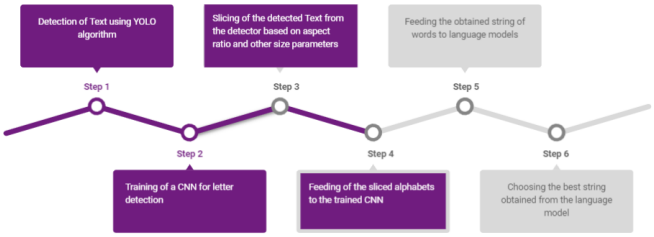


Fig. 4. Pipeline Used for Scene Text Recognition

### A. Detection

This part mainly deals with obtaining the bounding boxes for the text present in the input image. For this task, several standard algorithms such as VGG, CRAFT, YOLO and others discussed above such as SSD, AlexNet are available. Out of all these algorithms, since scene text recognition mainly deals with real time data, the algorithm that is most consistent and quick enough for such an application is YOLO. And we have used a Deep neural network trained using the YOLO loss function available here [10]. The neural network takes in the input image and provides the image with bounding

boxes around the text and also the coordinates for the bounding boxes.

### *B. Recognition*

We initially intended to use a CTC network along with CRNN for the purpose of recognition [11]. For the purpose of recognition, there are other available pre-trained models such as Google Tessaract, etc. Since for the detection purpose we had already used a pre-trained model, we intended to train our own model. But due to limitations in the amount of data we could train and the space and time it consumed, we couldn't train the CTC model and hence devised a new pipeline to do the recognition.

We then decided to train a model to recognise a letter, so that we could implement the word recognition by implementing it letter-by-letter. For this, images of words in various fonts would be required.

A standard text image would involve standard fonts and these fonts are available as images. But here again, we were limited by our limitation of space and amount of data, since data was available as images and they occupied a large amount of space (10GB; we didn't think of training on a smaller dataset). The only .csv data available was for the handwritten letters. We thus trained a CNN using this data. Since handwritten data was light in the background and dark in text. We calculated the best of the predictions of the given image and that of the inverted (bitwise not) counterpart of the image. This led to some correct classifications which were previously misclassified. For our trained CNN we obtained an validation loss of 7.23% and a validation accuracy of 97.95%. [12]

The summary of the model is shown in Fig. 5.

These parts of the pipeline could be completed before the deadline of the project. The only part left out is the automation and integration of the whole pipeline along with automatic cropping of the bounding boxes to obtain letters, which can be done using the information on the Aspect ration of bounding box and the average/mean aspect ratio of standard fonts.

| Layer (type) | Output Shape | Param # |
|---|---|---|
| conv2d_6 (Conv2D) | (None, 26, 26, 32) | 320 |
| max_pooling2d_6 (MaxPooling2 | (None, 13, 13, 32) | 0 |
| conv2d_7 (Conv2D) | (None, 13, 13, 64) | 18496 |
| max_pooling2d_7 (MaxPooling2 | (None, 6, 6, 64) | 0 |
| conv2d_8 (Conv2D) | (None, 4, 4, 128) | 73856 |
| max_pooling2d_8 (MaxPooling2 | (None, 2, 2, 128) | 0 |
| flatten_2 (Flatten) | (None, 512) | 0 |
| dense_6 (Dense) | (None, 64) | 32832 |
| dense_7 (Dense) | (None, 128) | 8320 |
| dense_8 (Dense) | (None, 26) | 3354 |

Total params: 137,178

Trainable params: 137,178

Non-trainable params: 0

Fig. 5. Model Summary for Recognition CNN

## V. Results and Discussions

The bounding boxes obtained by the YOLO model for detection of text from images are shown in Figs. 6 and 7.

Through different examples, we could see that the model works well for plane images, i.e. where text is not slanted, but encounters issues with slanted text as in Test Case 1.

Figures 8 and 9 show the results obtained for the recognition part.

We can observe that not all the predictions are accurate and errors are there too.

We conclude by listing out the shortcomings and the improvements that can be done to our pipeline.

1) Training the CNN on font data to improve recognition accuracy.
2) Since YOLO has some issues with slant text detection other models such as VGG can be tried with the cost of time.
3) Improve the Recognition CNN by adding more layers as the present one is a simplistic and minimal.
4) The pipeline is incomplete and the integration and the automation of cropping of images based on Aspect ration need to be incorporated.


## Acknowledgement

We thank the Head of Dept. of Avionics, Professor Deepak Mishra for giving us this opportunity to learn about the various types of text detection and recognition systems using neural networks and providing us actual experience in applying the concepts we learned to an actual, real-life application.


## References


[1] https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients
[2] https://www.geeksforgeeks.org/r-cnn-region-based-cnns/
[3] https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/
[4] https://towardsdatascience.com/intersection-over-union-iou-calculation-for-evaluating-an-image-segmentation-model-8b22e2e84686


Fig. 6. Text Detection Test Case 1


Fig. 7. Text Detection Test Case 2

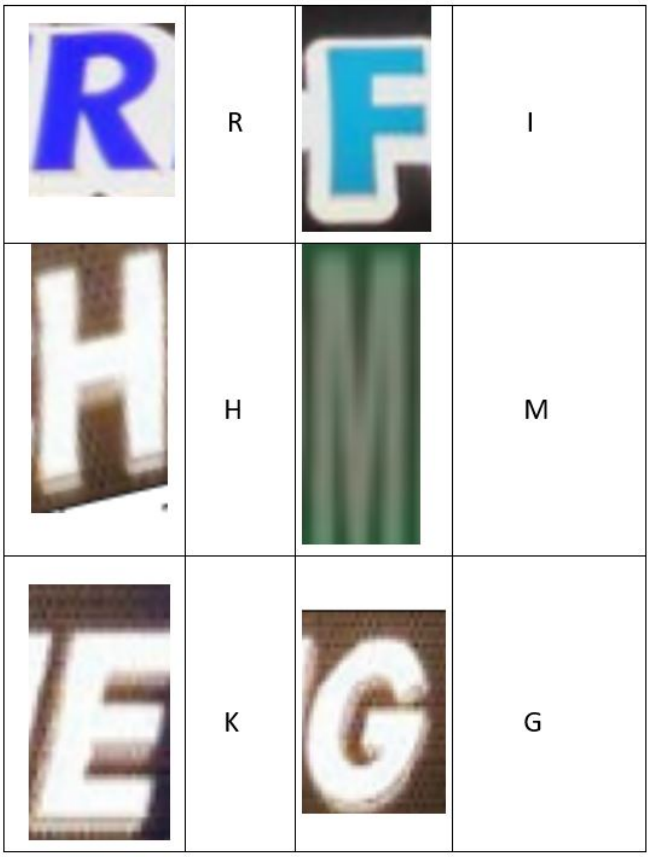


Fig. 8. Text Recognition - Test Cases with Letters


Fig. 9. Text Recognition Test Case with Words

[5] https://developers.arcgis.com/python/guide/how-ssd-works/
[6] https://towardsdatascience.com/yolo-you-only-look-once-3dbdbb608ec4
[7] https://www.analyticsvidhya.com/blog/2018/12/practical-guide-object-detection-yolo-framewor-python/
[8] https://medium.com/analytics-vidhya/image-text-recognition-738a368368f5
[9] An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition : Baoguang Shi, Xiang Bai and Cong Yao : School of Electronic Information and Communications
[10] https://github.com/Neerajj9/Text-Detection-using-Yolo-Algorithm-in-keras-tensorflow
[11] https://www.kaggle.com/sachinpatel21/az-handwritten-alphabets-in-csv-format/tasks
[12] https://data-flair.training/blogs/handwritten-character-recognition-neural-network/