PROJECT REPORT

On

TEXT DETECTION IN NATURAL SCENE IMAGES

Submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology
In
Computer Science Engineering

SUBMITTED BY:
Akshat Upadhyay (17103039)
Harsh Pandey (17103043)
Shashwat Singh (17103047)

UNDER THE SUPERVISION OF:
Dr. Anuja Arora
ASSOCIATE PROFESSOR
Dept. of C.S.E, JIIT, Sec-62, Noida

Department of Computer Science & Information Technology
JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

Table Of Contents

CERTIFICATE

This is to certify that Akshat Upadhyay (Enroll. No. 17103039), Harsh Pandey (Enroll. No. 17103043) , Shashwat Singh (Enroll. No. 17103047) have successfully completed the project titled **"TEXT DETECTION IN NATURAL SCENE IMAGES"** under my supervision and guidance in the fulfillment of requirements of Fifth Semester, Bachelor of Technology (Computer Science & Engineering) of Jaypee Institute of Information Technology, Sec-62, Noida .

_____ Date: _____

Assoc. Professor Dr. Anuja Arora

Dept. Of C.S.E, JIIT, Noida

(Project Mentor)

ACKNOWLEDGEMENT

We deem it a pleasure to acknowledge our sense of gratitude to our project guide Assoc. Prof. Dr. Anuja Arora under whom we have carried out the project work. Her incisive and objective guidance and timely advice encouraged us with constant flow of energy to continue the work. We also wish to thank her from the bottom of our hearts for guiding us and sharing with us her vast knowledge at points where we found ourselves lost in the project.

We would like to express our gratitude towards our parents & faculty members of the Institute for their kind cooperation and encouragement which helped us in completion of this project.

Date:_____

Place: Jaypee Institute of Information Technology, Noida

Akshat Upadhyay	Harsh Pandey	Shashwat Singh
17103039	17103043	17103047

General Introduction:

Text Recognition in Natural Scene is a topic of great interest for researchers and programmers from a very long time and various major breakthrough has been achieved in this field by them for recognizing text from images with constant background but not very eminent is done towards image with vivid and adverse background which makes this field yet not completely discovered. Most of the models made regarding this topic are not end to end trainable and are trained in separate module and then are tuned in single module to convert it into a proper one.

Optical Character recognition is method to extract text from images and other similar formats. It is used to transform any object that contains written text (printed, handwritten, typed) into a format easily interpreted by a computer.

CNN is a type of artificial neural network that are generally used to process images and handle and manipulate pixels of particular images. They consist of neurons which get trained liked normal human neurons. Post training, they are capable of memorizing patterns as well.

RNN is another advanced type of artificial neural network that works on a sequence of inputs. It uses previously gathered information to calculate present inputs.

Scene content discovery techniques dependent on neural systems have risen as of late and have demonstrated promising outcomes. Past strategies prepared with inflexible word-level jumping encloses show constraints speaking to the content locale in a discretionary shape. In this paper, we propose another scene content recognition technique to viably recognize content zone by investigating each character and liking between characters.

Problem Statement:

Text Recognition in natural scene is a challenging task because of the varied and inconsistent backgrounds along with text written in different fonts as well as orientations. The existing models for text detection in natural scenes do not provide much and fail to provide high accuracies in scenarios like vertical text, curved orientations or distortions. Character Recognition has been a field of research for a long time with various advancements in the field of Optical Character Recognition. Developments in this field not only open new doors of possibilities for the visual horizons of humans, but are a major scientific breakthrough as well. Natural scene text detection is still in its nascent stages with very few noticeable advancements. This project advancing in a similar fashion aims at applying the much known text recognition models in natural scene images using CRNN (CRAFT + RNN), which are some of the state-of-the-art algorithms that serve the purpose in an efficient and accurate manner.

Feasibility:

Earlier approaches to solve problems related to OCR were majorly focused on text recognition from hand-written documents ,old manuscripts and images with constant background. But in our project we are further extending these algorithms to recognize text in natural scene images like billboards,hoardings,banners also. Nowadays there has been a prominent impact of technologies like Augmented Reality, Mixed Reality, Google lens etc on our daily life. These technologies basically uses algorithms to gather information from our surroundings and mould them as per our needs. And hence, our approach will act as a pivot for further researches and technological advancement.

Research paper cited:

Reading text in the WILD using Convolutional Neural Network

This is based on region detection and recognition using deep convolutional neural networks .They have trained a very large CNN for the purpose of detection and recognition of text in images as a whole and not in parts. These networks are trained to even detect the text from image without any need of human labelled data to the input image. The network was trained on very big datasets which consist of thousands of images. At long last, we exhibit a genuine utilization of our content spotting framework to permit a huge number of long stretches of news film to be in a split second accessible by means of a book inquiry.

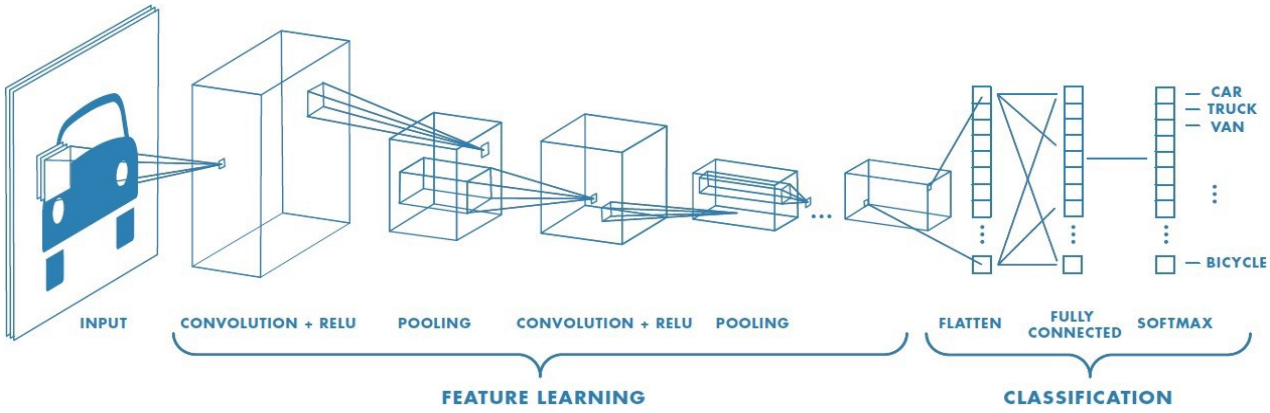


Fig 6.1 Convolutional Neural Network Architecture

Challenges faced that led to development:

Earlier OCR(Optical Character Recognition) model was utilized. The OCR model had examined characters independently, followed by cancellation of surrounding noise. Every single letter was checked pixel by pixel. The OCR model location strategy required obtuse foundation and condition for appropriate acknowledgment, which in case failed in many normal day to day environments.

Character Region awareness for Text Detection (CRAFT)

They have proposed a text detector which separates single character region and then try to merge then into a common boundary/box. They have used two terminologies:-

- Region Score : This is used to localize single characters in the image.
- Affinity Score: This is used to merge characters into single instance.

By exploiting character-level region awareness, texts in various shapes are easily represented.Datasets with huge images with text has been trained to improve accuracy and efficiency.

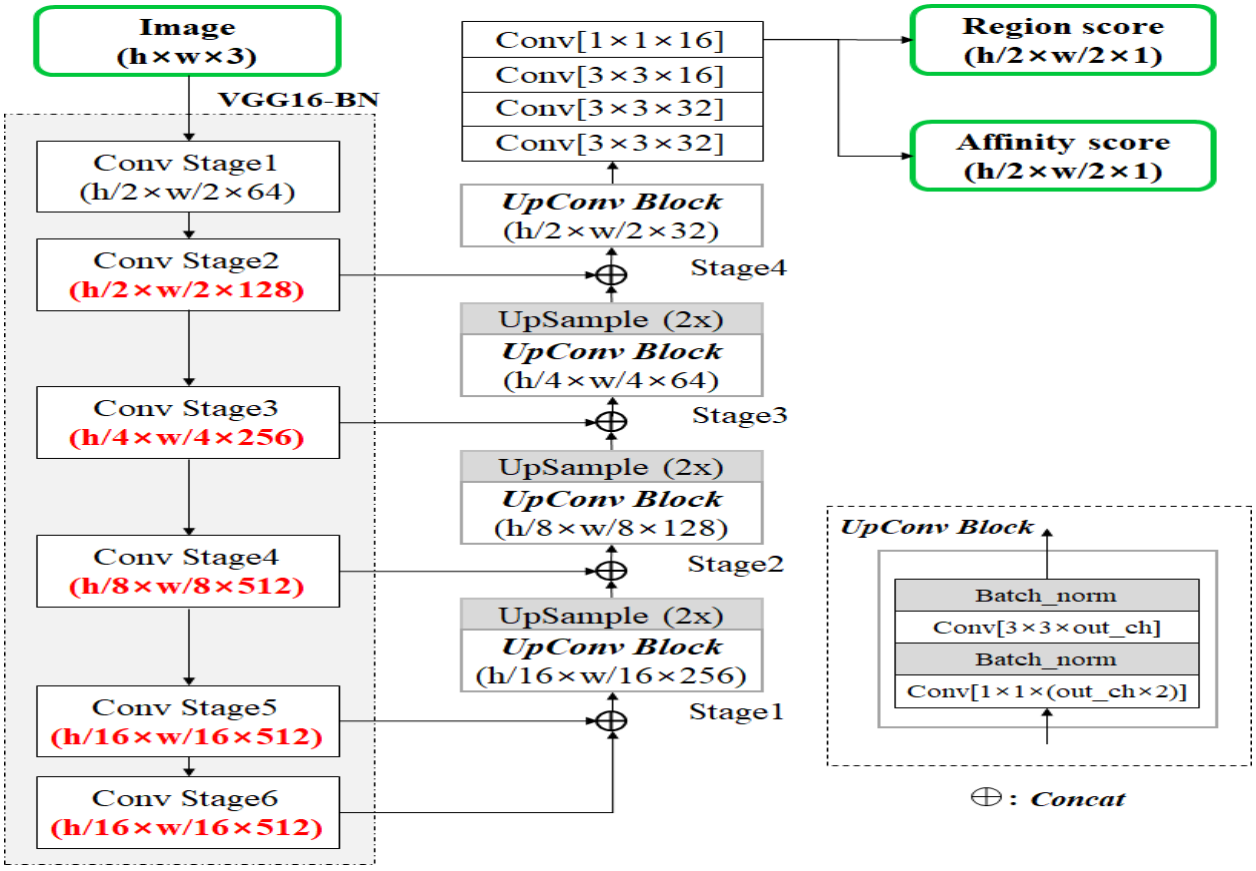
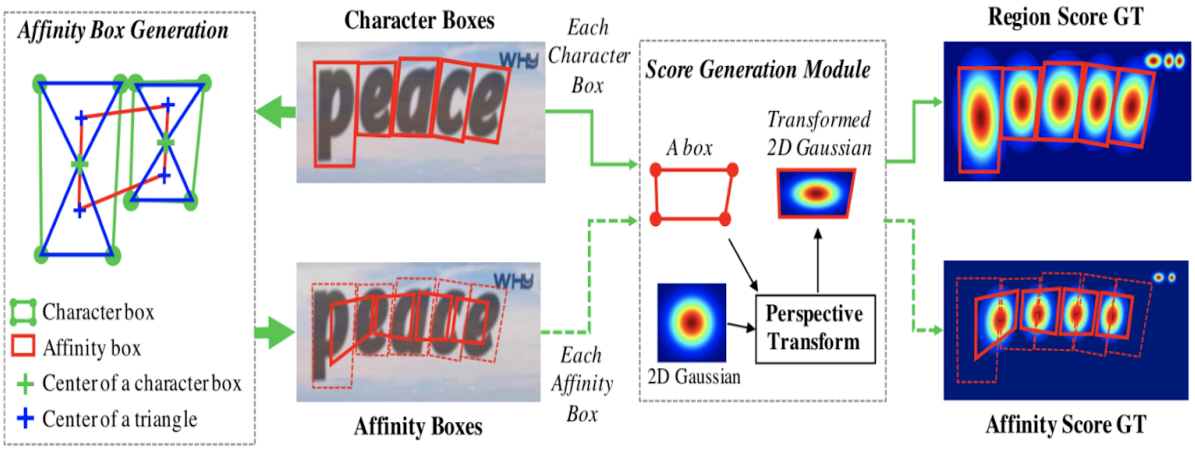


Fig 8.1 Craft Architecture

Architecture with VGG16 with batch normalization is used along with this 4 more layers with each has 2 sequational layers has been used. The final output has 2 channels namely:

1. Affinity score
2. Region score

For affinity and region score use that is basically for localizing characters and constructing detection along the border of those words around text written we had used Guassion's heat map method. Heat Map is one of the most efficient methods to differentiate the boundaries of subjects. Edges of text with background of image can be easily isolated and even in that also more focus in made on the part of heat map that has the most eminent colour scheme.



An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition

In this paper a neural architecture was proposed, called as Convolutional Recurrent Neural Network(CRNN), which has advantages of both CNN as well as RNN. CRNN was able to take inputs of variable dimensions and was able to produce output of varying lengths.

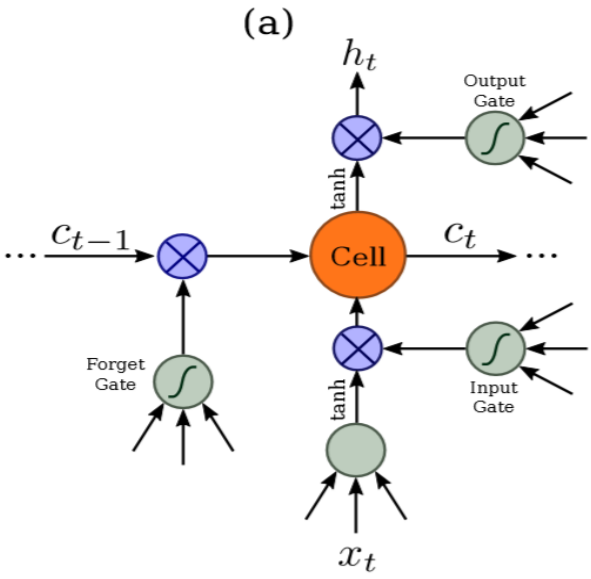


Fig 10.1 LSTM Unit

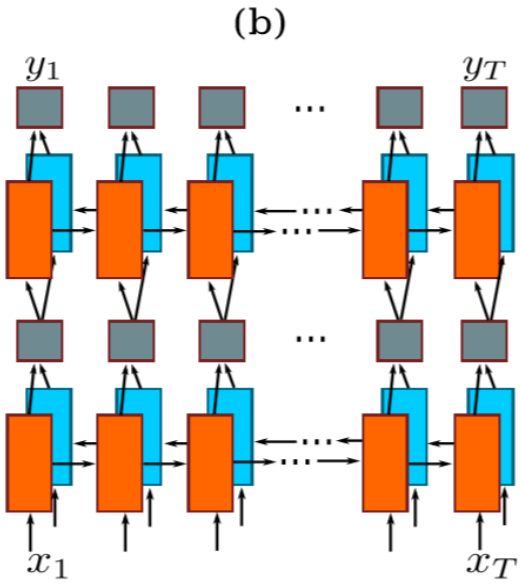


Fig 10.2 Bi-Directional LSTM

Firstly CNN layers (fully connected layers removed) were used to extract features from the image which resulted in a feature map which is then passed on to RNN.
RNN segments input into frames, each time it gets a frame it produces output based on previous as well as current input frame. LSTM a type of RNN unit is used to handle sequential data, it consists of three gates :

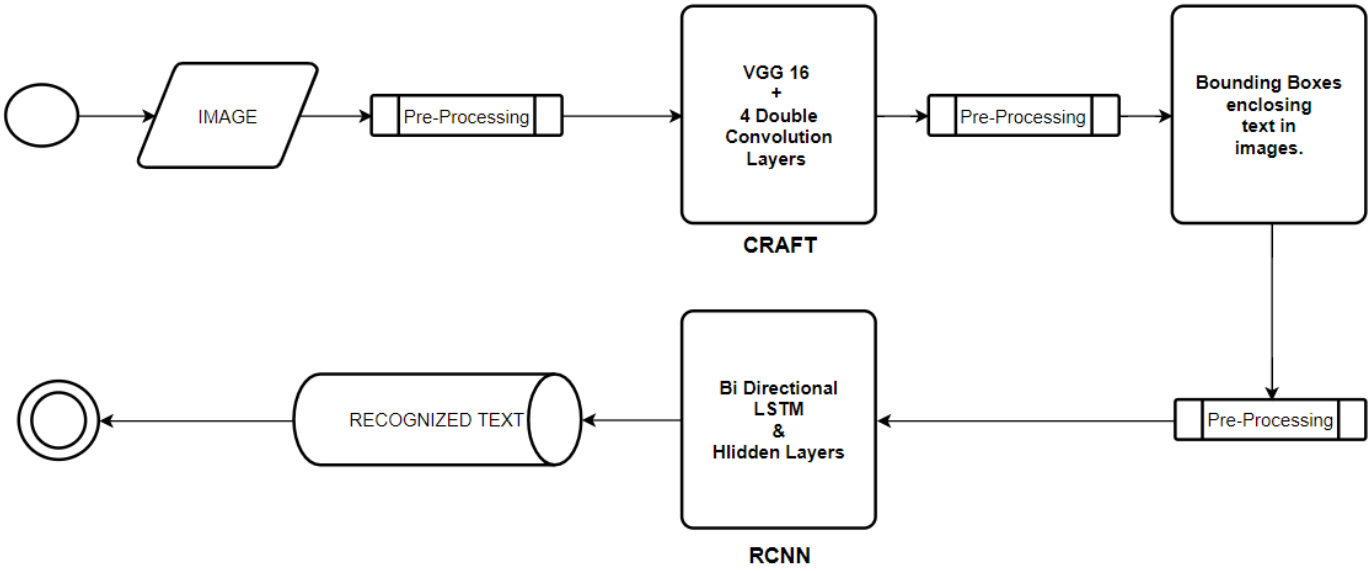
- **Forget Gate :** This gate chooses which information to use and what not to. The current input and previous input is passed through a sigmoid function, the value closer to 0 means discard and, a value close to 1 means it will keep the information.

- **Input Gate :** This gate extracts the information which is important and not redundant by passing through tanh function (created vector) and adding the regulatory filter(xt+ht-1) produced by passing through sigmoid to the created vector.

- **Output Gate :** not all information stored in the cell is important therefore it chooses what information to present as output, it creates a vector after applying tanh function then multiplies it with a regulatory filter (xt+ht-1) produced by passing through sigmoid.

Since the network is bidirectional so that it can make use of information from past as well as future contexts. Then per frame predictions are used to create a label sequence.

PROJECT WORKFLOW:



WORKING METHODOLOGY OF CRAFT MODEL:

As the pre-processed model for CRAFT Architecture, a fully connected neural network based on VGG-16 with batch normalization is adopted. There are two channels in the final production that forward the two score charts, i.e. Score of affinity and score of the region.

The area score reflects the likelihood that the given pixel is the character's center and the affinity score represents the center probability of the space between two adjacent characters. We encode the character center's probability with a Gussian heatmap]. We use the representation of the heatmap to know both the score of the area and the score of affinity.

We tend to generate character boxes from every word-level annotations during a weakly supervised manner. Once a true image with word-level annotations is provided, the

learned interim model predicts the character region score of the cropped word pictures to come up with character-level bounding boxes, so as to replicate the dependencies of the interim model's prediction.

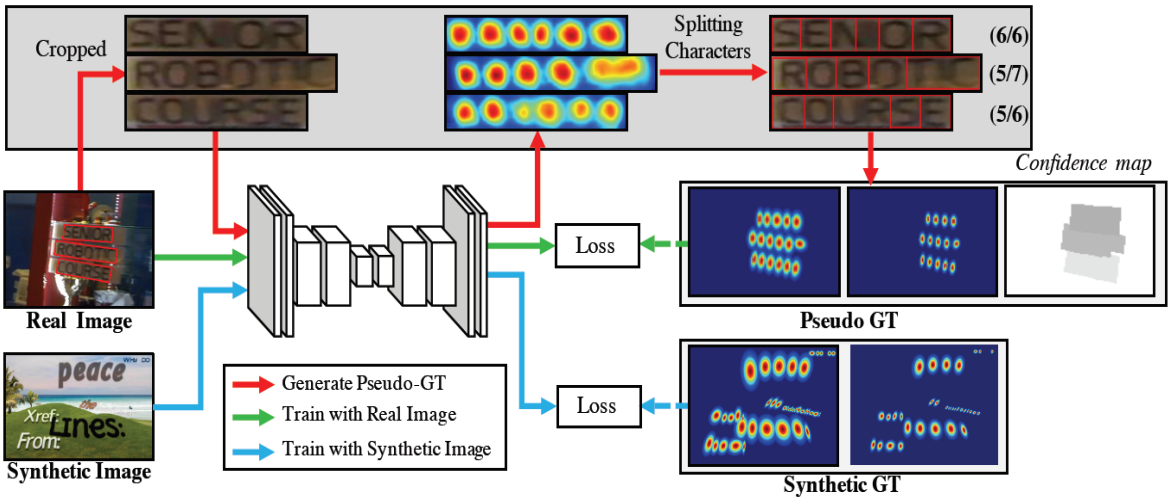


Fig 12.1 Craft Procedure for Region & Affinity Score Detection

First, the word-level images are cropped from the original image. Second, the model trained up to date predicts the region score. Third, the watershed algorithm is used to split the character regions, which is used to make the character bounding boxes covering regions. Finally, the coordinates of the character boxes are transformed back into the original image coordinates using the inverse transform from the cropping step. For a word-level annotated sample w of the training data, let R(w) and l(w) be the bounding box region and the word length of the sample w, respectively. Through the character splitting process, we can obtain the estimated character bounding boxes and their corresponding length of characters lc(w). Then the confidence score sconf (w) is:

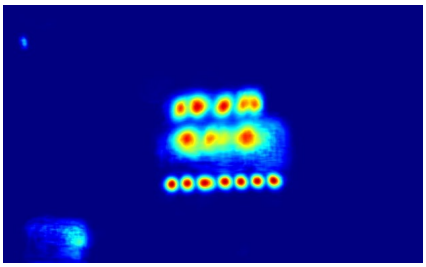
$$s_{conf}(w) = \frac{l(w) - \min(l(w), |l(w) - l^c(w)|)}{l(w)}$$

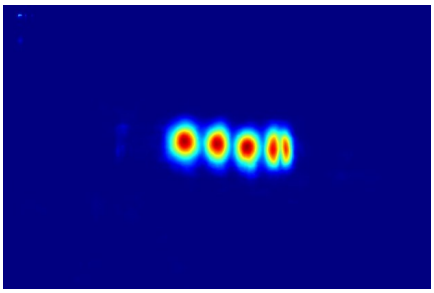
and the pixel-wise confidence map Sc for an image is computed as:

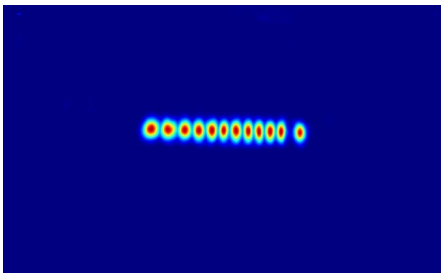
$$S_c(p) = \begin{cases} s_{conf}(w) & p \in R(w), \\ 1 & \text{otherwise}, \end{cases}$$

If the confidence score sconf (w) is below 0:5, the estimated character bounding boxes are neglected since they have adverse effects when training the model. In this case, we assume the width of the individual character is constant and compute the character-level predictions by simply dividing the word region R(w) by the number of characters l(w). Then, sconf (w) is set to 0:5 to learn unseen appearances of texts.

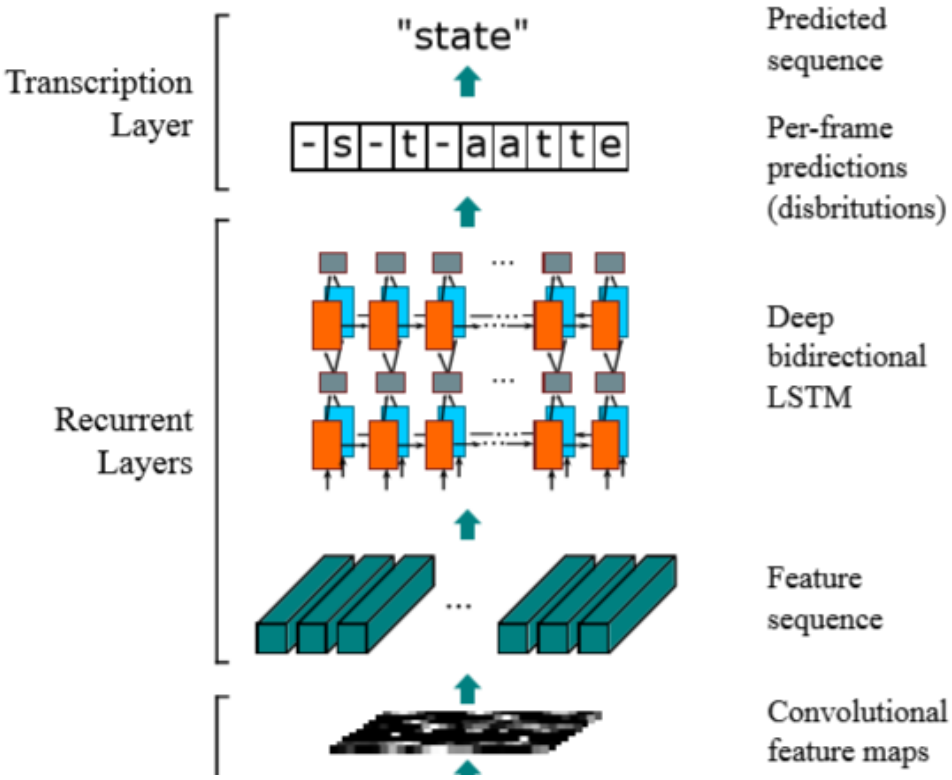
Some samples from our experiments on the ICDAR15 and SVT data sets:







WORKING OF THE RECURRENT NEURAL NETWORK:



Let's see how the above structure be used to predict the letters in the word "hello". In the above structure, RNN block, applies something called as a recurrence formula to the input vector and also its previous state. In this case, the letter "h" has nothing preceding it, let's take the letter "e". So at the time the letter "e" is supplied to the network, a recurrence formula is applied to the letter "e" and the previous state which is the letter "h". These are known as various time steps of the input. So if at time t, the input is "e", at time t-1, the input was "h". The recurrence formula is applied to e and h both. and we get a new state. The formula for the current state can be written as –

$$h_t=f(h_{t-1},x_t)$$

Here, Ht is the new state, "ht-1" is the previous state while "xt" is the current input. We now have a state of the previous input instead of the input itself, because the input neuron would have applied the transformations on our previous input. So each successive input is called a time step.

In this case we have four inputs to be given to the network, during a recurrence formula, the same function and the same weights are applied to the network at each time step.

Taking the simplest form of a recurrent neural network, let's say that the activation function is tanh, the weight at the recurrent neuron is Whh and the weight at the input neuron is Wxh, we can write the equation for the state at time t as –

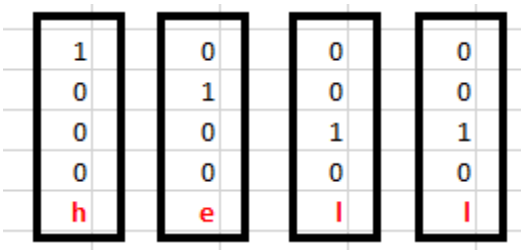
$$h_t=\tanh(W_{hh}h_{t-1}+W_{xh}x_t)$$

The Recurrent neuron in this case is just taking the immediate previous state into consideration. For longer sequences the equation can involve multiple such states. Once the final state is calculated we can go on to produce the output. Now, once the current state is calculated we can calculate the output state as-

$$Y_t = W_{hy}h_t$$

Forward Propagation in a Recurrent Neuron in Excel:

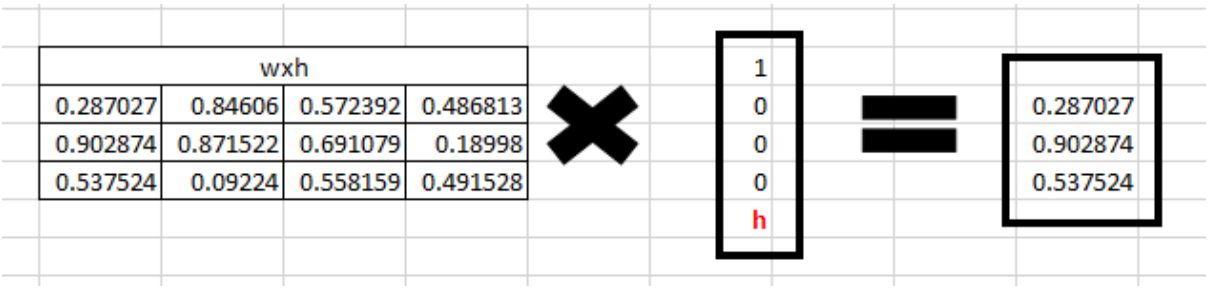
Let's take a look at the inputs first –



The inputs are one hot encoded. Our entire vocabulary is {h,e,l,o} and hence we can easily one hot encode the inputs. Now the input neuron would transform the input to the hidden state using the weight wxh. We have randomly initialized the weights as a 3*4 matrix –

wxh			
0.287027	0.84606	0.572392	0.486813
0.902874	0.871522	0.691079	0.18998
0.537524	0.09224	0.558159	0.491528

Step 1: Now for the letter "h", for the hidden state we would need Wxh*Xt. By matrix multiplication, we get it as –



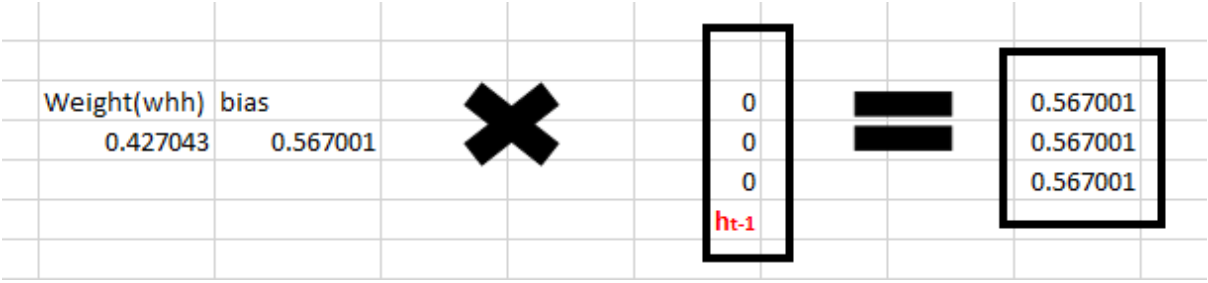
wxh			
0.287027	0.84606	0.572392	0.486813
0.902874	0.871522	0.691079	0.18998
0.537524	0.09224	0.558159	0.491528

1	0.287027
0	0.902874
0	0.537524
0	
h	

Step 2:

Now moving to the recurrent neuron, we have Whh as the weight which is a 1*1 matrix as | 0.427043 | and the bias which is also a 1*1 matrix as | 0.56700 | . For the letter "h", the previous state is [0,0,0] since there is no letter prior to it.

So to calculate -> (whh*ht-1+bias)



Weight(whh)	bias
0.427043	0.567001

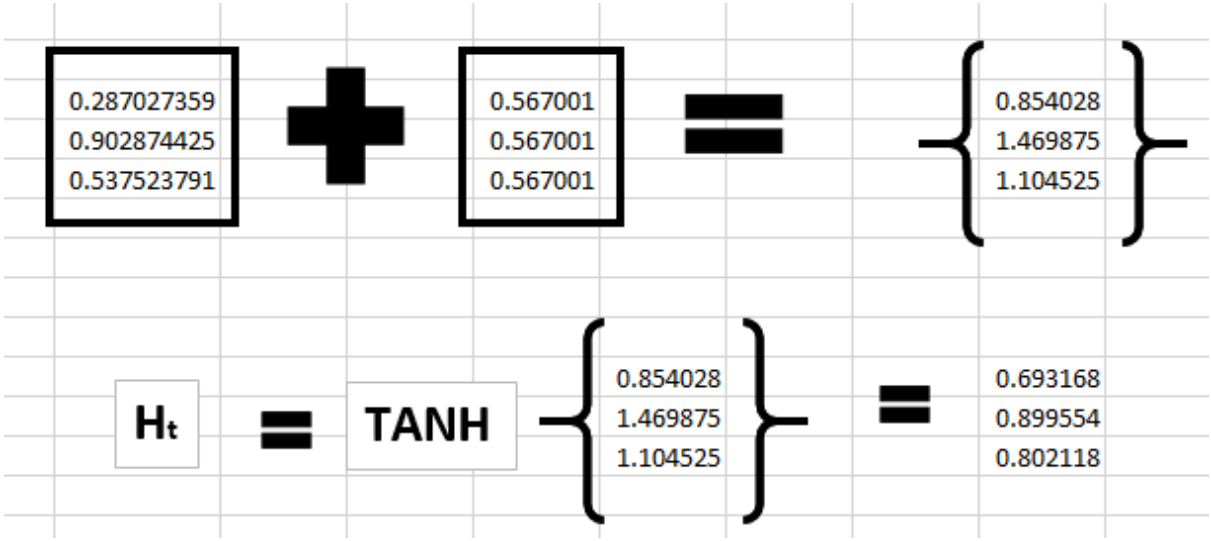
0	0.567001
0	0.567001
0	0.567001
h_{t-1}	

Step 3:

Now we can get the current state as –

$$h_t = \tanh (W_{hh}h_{t-1} + W_{xh}x_t)$$

Since for h, there is no previous hidden state we apply the tanh function to this output and get the current state –

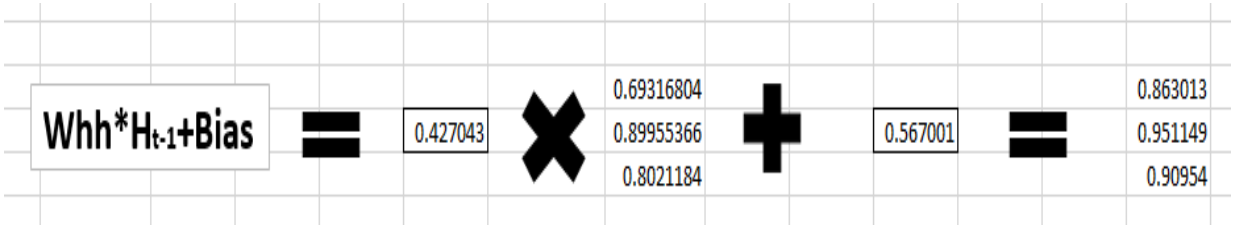


Step 4:

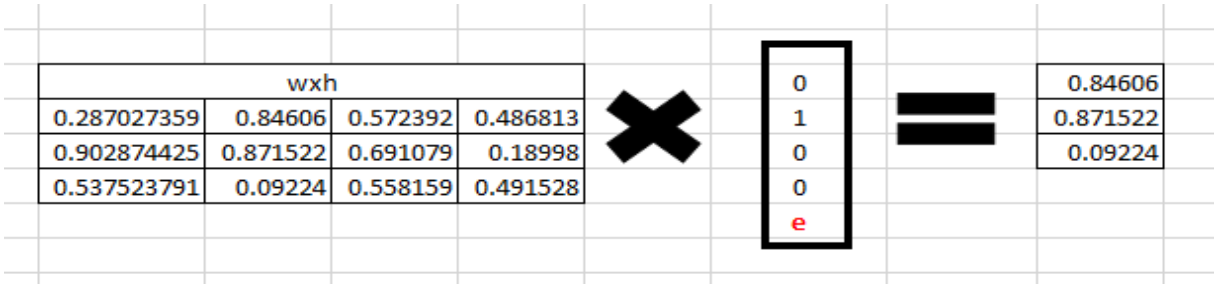
Now we go on to the next state. "e" is now supplied to the network. The processed output of ht, now becomes ht-1, while the one hot encoded e, is xt. Let's now calculate the current state ht.

$$h_t = \tanh (W_{hh}h_{t-1} + W_{xh}x_t)$$

Whh*ht-1 +bias will be –

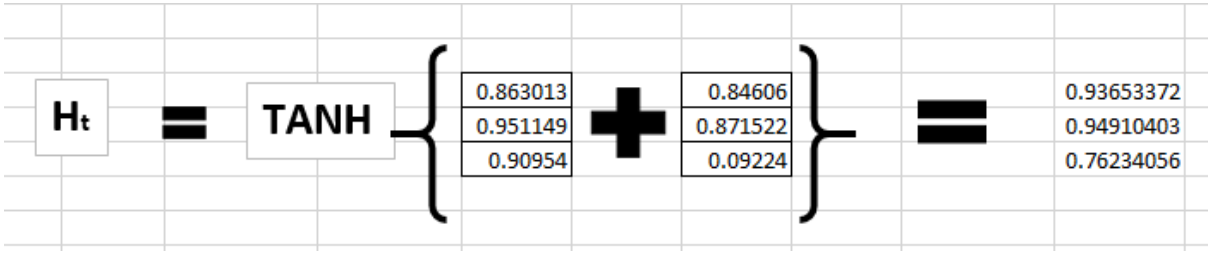


Wxh*xt will be –



Step 5:

Now calculating ht for the letter “e”,

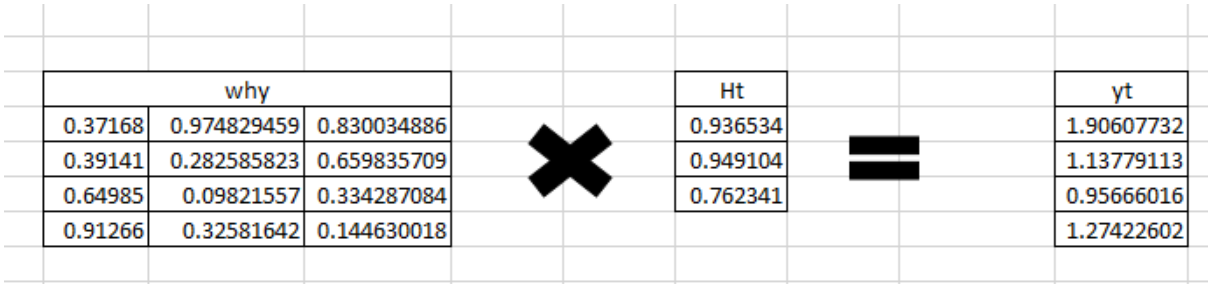


Now this would become ht-1 for the next state and the recurrent neuron would use this along with the new character to predict the next one.

Step 6:

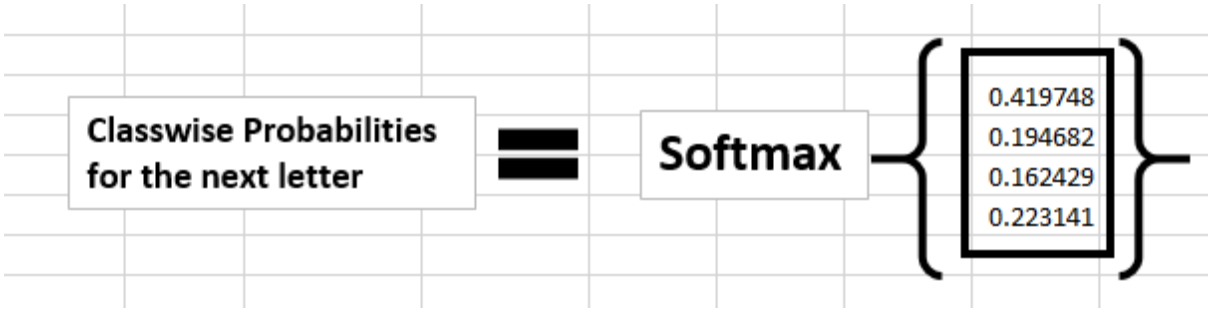
At each state, the recurrent neural network would produce the output as well. Let’s calculate yt for the letter e.

$$y_t = W_{hy} h_t$$



Step 7:

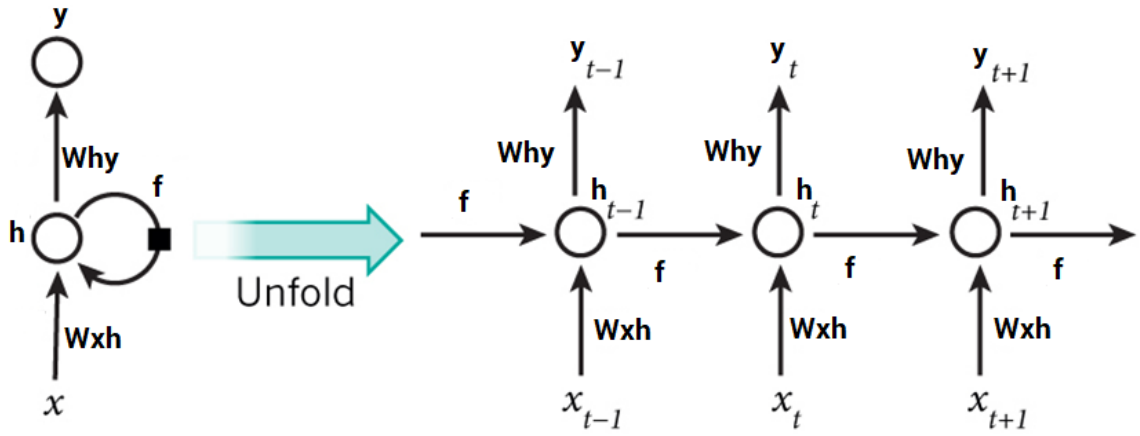
The probability for a particular letter from the vocabulary can be calculated by applying the softmax function. so we shall have softmax(yt)



If we convert these probabilities to understand the prediction, we see that the model says that the letter after “e” should be h, since the highest probability is for the letter “h”. Similarly it continues to train and make better predictions for the future characters.

Back propagation in a Recurrent Neural Network(BPTT)

To imagine how weights would be updated in case of a recurrent neural network, might be a bit of a challenge. So to understand and visualize the back propagation, let’s unroll the network at all time steps. In an RNN we may or may not have outputs at each time step.
In case of a forward propagation, the inputs enter and move forward at each time step. In case of a backward propagation in this case, we are figuratively going back in time to change the weights, hence we call it the Backpropagation through time(BPTT).



In case of an RNN, if yt is the predicted value ȳt is the actual value, the error is calculated as a cross entropy loss – **Et(ȳt,yt) = – ȳt log(yt)**

$$E(\bar{y},y) = -\sum \bar{y}t \log(yt)$$

We typically treat the full sequence (word) as one training example, so the total error is just the sum of the errors at each time step (character). The weights as we can see are the same at each time step. Let's summarize the steps for backpropagation

1. The cross entropy error is first computed using the current output and the actual output
2. Remember that the network is unrolled for all the time steps
3. For the unrolled network, the gradient is calculated for each time step with respect to the weight parameter
4. Now that the weight is the same for all the time steps the gradients can be combined together for all time steps

The unrolled network looks much like a regular neural network. And the back propagation algorithm is similar to a regular neural network, just that we combine the gradients of the error for all time steps. Now what do you think might happen, if there are 100s of time steps. This would basically take really long for the network to converge since after unrolling the network becomes really huge. Therefore it is not easy to train RNN without sufficient computational power.

RESULTS:

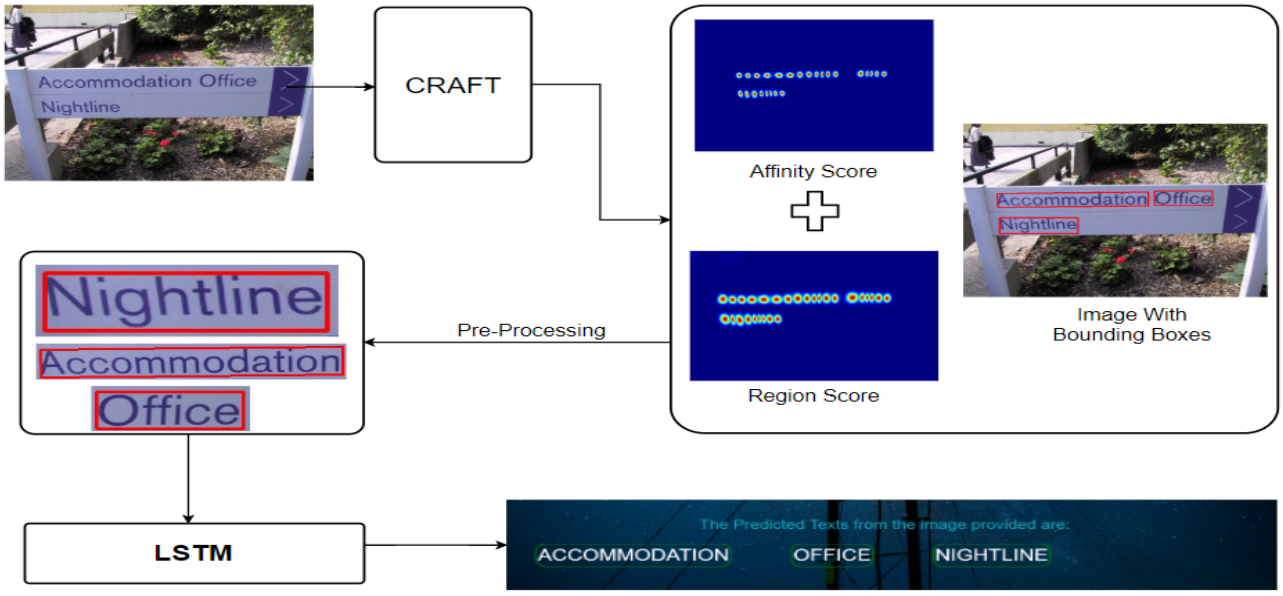


Fig 12.1 Sample 1

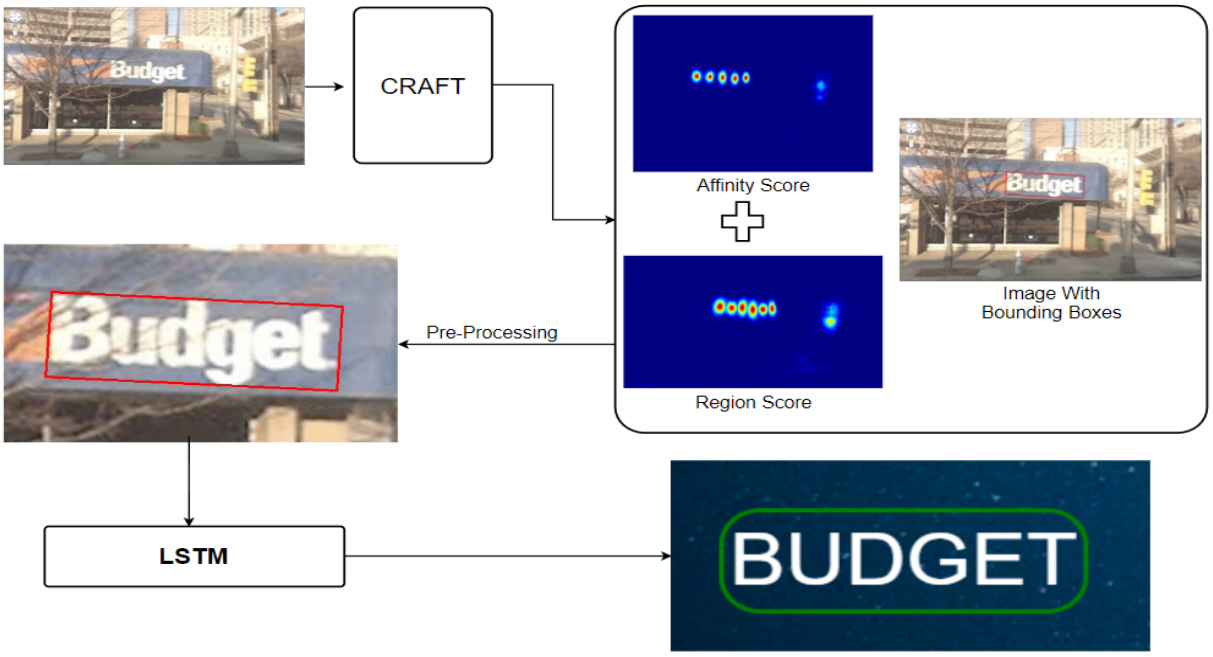
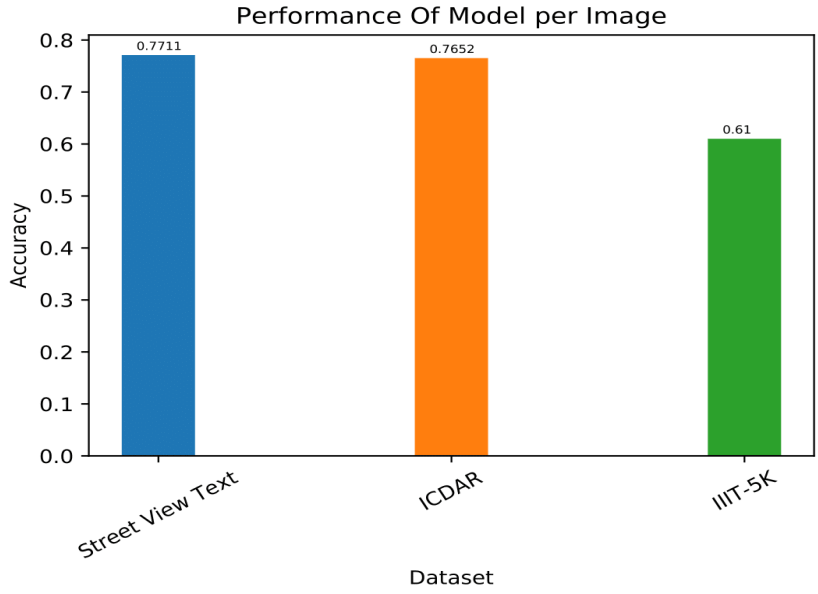
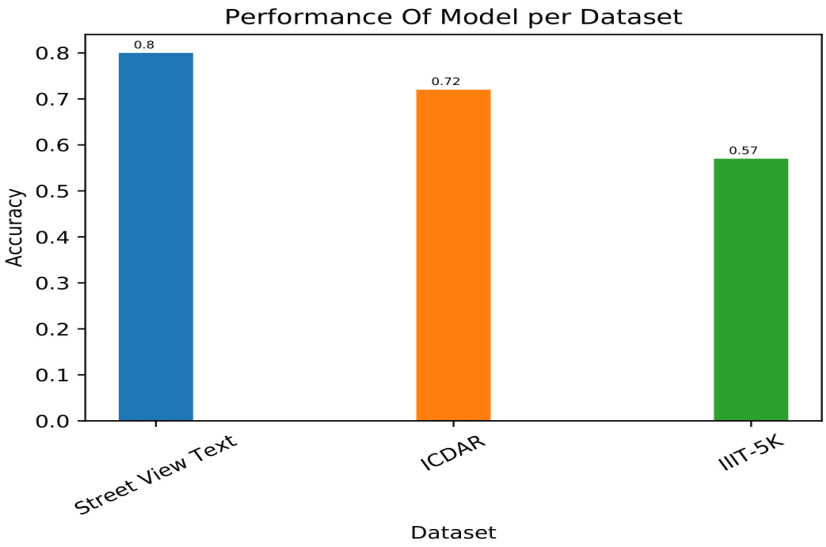


Fig 12.2 Sample 2





Conclusion

In this project we have presented an architecture that has advantages of both CNN and RNN. It is able to take input images of varying dimensions and produce outputs of varying length. It requires no detailed annotation (Character-Level). It abandons fully connected layers, therefore making the model compact and efficient. The tests run on the project show that it performs better than other CNN and RNN based projects. The project shows comparably better accuracy in datasets like SVT and ICDAR15, which are more realistic and hence can be used for future real-world applications.

APPLICATIONS

❑ Banking Sector

Handwritten cheques, Demand Drafts can be scanned. Signature can be verified and financial transactions can be done with much ease and minimal human effort. This reduces turnaround time for cheque clearance and other validations in the banking sector.

❑ HealthCare Sector

OCR enables medical sector giants to digitally saved medical history of large number of patients. Having one's entire medical history on a searchable digital store means that things like past illnesses, treatments, diagnostic tests, hospital records can be made available in one unified place.

❑ Accessibility

Paired with Text-to-Speech technology, Text recognition in natural scenes can help visually impaired people to be communicated through audible messages. This is one of the most under research people for the underprivileged sections of the society.

❑ Google Lens

Google Lens is an AI-powered technology that uses your smartphone camera and deep machine learning to not only detect an object, but understand what it detects and offer actions such as Text Translation, Text recognition, text-to speech in realtime and natural environments.

REFERENCES:

- https://medium.com/@xiaosean5408/craft%E7%B0%A1%E4%BB%8B-character-region-awareness-for-text-detection-a5c782408f00

- https://arxiv.org/abs/1904.01941/Craft-Architecture-for-Natural-Scene-Text-Detection

- https://towardsdatascience.com/an-approach-towards-convolutional-recurrent-neural-networks-f54cbeecd4a6

- https://theailearner.com/2019/05/29/creating-a-crnn-model-to-recognize-text-in-an-image-part-1/

- https://nanonets.com/blog/deep-learning-ocr/#crnn

- L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. PAMI, (6):583–598, 1991.

- M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Reading text in the wild with convolutional neural networks. IJCV (Accepted), 2015.

- B. Su and S. Lu. Accurate scene text recognition based on recurrent neural networks. In ACCV, 2014.

- Baoguang Shi, Xiang Bai and Cong Yao. An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition

- Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. Character Region Awareness for Text Detection. PAMI, (6):573–558, 2014.

DEPENDENCIES

- Python 3.5
- PIL
- Numpy

Pytorch 1.2.1
Pandas
CSV

DATASETS

1. ICDAR

2. SVT (Street View Text)

3. IIIT5K