

Optical Text Translator

Shivani Joshi¹, Rajiv Kumar², Praveen Kumar Rai³, Rikendra⁴, Avinash Dwivedi⁵

^{1,2,4} GL Bajaj Institute of Technology and Management, Greater Noida, India

³ ITS Engineering College Greater Noida, India

⁵ JIMS Technical Campus, Greater Noida

ABSTRACT

In the actual world, when people go to a new location, they may have difficulties with language, signboards, or other kinds of communication. People in the digital age utilize Google Translate, Pocket Dictionary, and other similar applications. A signboard or other kind of notice may communicate important information or even act as a warning. Important information may be lost if the message is unavailable to individuals of different linguistic backgrounds; furthermore, visitors may struggle to carry out their duties in a new place if they do not comprehend the language spoken there. A portable dictionary, on the other hand, may be ineffective if users wish to translate a language in which words are not organized alphabetically. In another research, it is also understood in the same manner that users are unable to type the text of what they see. Because they are unable to comprehend the language despite the pocket dictionary and online translation service offered, people from various linguistic backgrounds may experience a breakdown in communication. The purpose of this article is to create an app that will translate information shown on a signboard into the language of the user's choice. When the user points their phone's camera at the signboard, the App takes a picture of it and then translates the text on the signboard.

Keywords: Text Recognizer, Convolutional Recurrent Neural Network (CRNN), Long short-term memory (LSTM), YOLO model.

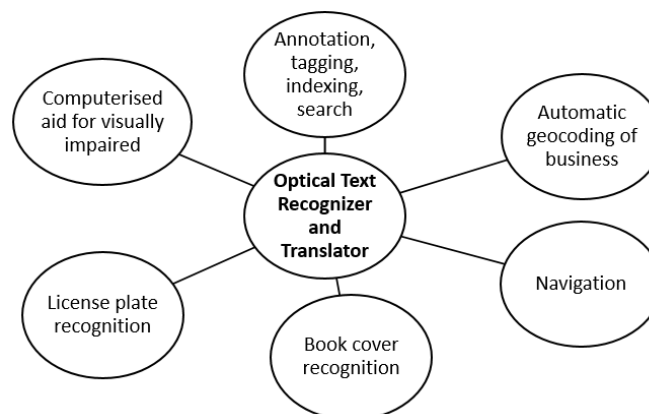
1. INTRODUCTION

The purpose of this article is to create a model that translates text on a signboard into the language of the user's choice. The user will direct the camera toward the signboard. The text on the signboard should then be translated by the App on their phone. To begin, we'll create a system that works for names (such as street names, city names, organization names, business names, and so on) that usually include 1-2 words and are seldom longer than 4-5 syllables. We will provide services in five different languages (i.e., the App can read and translate text from and to one of these five languages). The number of languages will be increased to 15 in the next phase. Finally, in the project's last phase, we will assist with the translation of lengthier texts printed on signboards, such as

“DO NOT LITTER” to “कूड़ा मत करो”

The aim of this article is to develop a lightweight, quick, and efficient program that translates the text shown on a signboard into the user's preferred language. This article will go through the App's architecture as well as its three major building components. The Optical Text Recognizer and Machine Translator is a communication tool that can recognize text in images and translate it to any language. Travel lovers, as well as anybody with a mobile phone, may utilize it. Deep Learning is a subset of a broader class of machine learning and artificial neural networks. Figure 1 depicts the primary uses of an optical text recognizer and translator.

Figure 1 Applications of Optical Text Recognizer and Translator.



2. LITERATURE REVIEW

According to, it is essential for content image analysis to identify and localise text inside scenic pictures (Y F Pan, 2011). This issue is worsened by the intricate backdrop and the wide variety of text fonts, sizes, and line orientations. The authors propose a hybrid method for reliably detecting and localising text in pictures of natural landscape. A text area detector is used to assess the text's existing confidence and scale information included inside an image pyramid, assisting in the segmentation of potential text components through local binarization. We provide a conditional random field (CRF) model with supervised parameter learning for effectively filtering out non-text components while accounting for unary component characteristics and binary contextual component connections. Finally, text components are combined into text lines/words using a learning-based energy reduction technique. Due to the fact that all three phases are based on learning, human involvement is required for just a few variables.

As previously mentioned, text identification in natural scene photographs is a crucial step in the development of many content-based image analysis systems (x. C Yin et al, 2014). We describe in this paper a method for accurately and robustly detecting handwriting in natural scene images. The concept of decreasing regularised variants is used to create a fast and efficient pruning technique for choosing Maximally Stable Extremal Regions (MSER) as character candidates. The single link clustering technique divides character candidates into text candidates using distance weights and clustering thresholds that are automatically learned, and then determines the posterior probability of text candidates belonging to other than text. The system evaluated using the ICDAR 2011 and achieved an f-measure of more than 76%.

L Sun et al. (2015) created a text recognition system using a neural network and a colour enhanced contrasting extremal region (CER). The tree is composed of six components, and the pictures are inverted from the original landscape colour images. Each picture patch is labelled in a distinct manner using divide and conquer methods.

(W. Liu, 2016) pioneered the use of a single deep neural network to detect items in images (SSD). The SSD performs the evaluation of the output by using bounding boxes of various sizes. Each itm type in each box is given a score by the network. Additionally, the network handles objects of different sizes smoothly by integrating predicted attributes from several feature maps with varying resolutions. SSD is more straightforward than object proposal techniques since it obviates the requirement for proposal creation and subsequent pixel or feature resampling phases by encapsulating all processing in a single network. As a consequence, SSD is simple to train and incorporate into systems that need some level of detection.

YOLO, a novel method for object detection, was introduced by (J. Redmon, 2016). Previously published research on object detection repurposes classifiers for the purpose of object recognition. Rather than that, we treat object identification as a regression issue with spatially distinct bounding boxes and associated class probabilities. In a single assessment, a single neural network predicts bounding boxes and class probabilities based on complete pictures. Due to the fact that the detection pipeline is comprised of a single network, detection performance may be improved from start to end. Our unified design enables us to operate at a high rate of speed. Our basic YOLO model analyses pictures at a rate of 45 frames per second in real time. Fast YOLO, a downscaled version of the network, processes an incredible 155 frames per second while retaining twice the mAP of other real-time detectors. YOLO has a higher rate of localization mistakes than cutting-edge detection algorithms but is less likely to forecast false positives based on background data.

(Q. Ren et al., 2017) developed the Area Proposal Network (RPN), which utilises the detection network's full-image convolutional features and allows almost cost-free area suggestions. A RPN is a fully convolutional network that concurrently predicts object borders and objectness ratings for each point. RPNs are trained from start to finish to provide high-quality region recommendations for Fast R-CNN identification. Through a simple alternating optimization, we can teach RPN and Fast R-CNN to share convolutional features.

(J.Q. Ma, 2018) presents a new rotation-based technique for detecting text in natural scene pictures that is orientated in any direction. The Rotation Region Proposal Networks are intended to offer inclined suggestions based on data about the text orientation angle. After that, the angle information is modified for bounding box regression to more closely match the orientation of the suggestions included inside the text area. The pooling layer Rotation Region-of-Interest is suggested for projecting arbitrary-oriented suggestions to a feature map for a text area classifier. The framework as a whole is designed using region proposals, which guarantees that arbitrary-oriented text identification is computationally efficient in comparison to prior text detection systems. They evaluated the rotation-based framework on three real-world scene text recognition datasets, demonstrating that it outperformed prior methods in terms of efficacy and efficiency.

Although the quicker R-CNN-based text recognition methods released by (Zhuoyao Zhang et al., 2019) have shown promise, the accuracy of location was not accepted in some cases due to the limitation of regression of local bounding box modules. Further it has been improved by new localized module called as LocNet which enhances the accuracy .

(Yu Zilong Zhang et al., 2021) introduced the Scene Text Recognition (STR) problem, which requires the consumption of a significant quantity of data in order to build a strong recognizer capable of combining visual data such as pictures with linguistic data such as words. On the other hand, existing techniques uses the processes of one stage training for full framework line to tine. This all depends upon annotation pictures which have not used data from two modality. This has been revised by pre-trained multi modal network(PMNN)

often use a one-stage training process to train the whole framework end-to-end, which is heavily dependent on well-annotated pictures and does not properly use data from the two modalities stated before. To solve this, we present a pre-trained multi-modal network (PMMN) that pre-trains vision and language models, respectively, using visual and linguistic data, in order to get modality-specific information for accurate scene text recognition. To begin, we convergence train the suggested off-the-shelf vision and language models. Then, for end-to-end fine-tuning, we integrate the pre-trained models into a single framework and use the learned multi-modal information to interact to generate robust features for character prediction. Extensive study is being done to establish PMMN's efficacy.

3. METHODOLOGY USED IN THE PAPER

This section contains a comprehensive description of the approach. The pre-processing of the input scene picture is followed by text detection using the YOLO method, and further text detection using the Recurrent Convolutional Neural Network. Finally, machine translation is done on this identified text using the Sequence to Sequence Encoder-Decoder Model.

3.1 Pre-processing

Using the camera, which captures text images and feeds them into the Yolo Model, users may communicate with the app via a simple camera interface. Exceptions are used to handle errors and isolate anomalous results or situations. This allows for more accurate error detection and isolation.

The collected images are pre-processed using a low pass filter (R Kumar, 2017) to remove noise before being sent to the computer. Exceptions are used to perform appropriate error handling, which allows for the isolation of aberrant outcomes or circumstances. The pictures obtained are pre-processed using a low pass filter in order to eliminate noise from the photographs.

3.2 Text detection

Text detection using the YOLO ALGORITHM (You Only Look Once), which determines the bounding box of the text to be recognized. The output of the YOLO model is then used to feed into the CRNN model.

With Deep Learning (DL) methods, You Only Look Once (YOLO) is a network that identifies objects in the environment. In order to identify items in a photograph, YOLO categorizes them inside the picture and calculates their location in respect to the picture. In real-time object detection, YOLO is a powerful convolutional neural network (CNN) (R Kumar et al., 2020), (S Joshi, 2020) that uses convolutional learning. The technique makes use of a single neural network to analyze the whole image, after which it divides it into regions and predicts bounding boxes and probabilities for each of them. The weighting of these bounding boxes is determined by the expected probability. Following is an explanation of how YOLO works: Yolo's first action is to take a photograph. The recorded image structure is then divided into grids, which is the last step (say for example 3 X 3 grid). The image is then classified and localized so that it may be used on each grid.

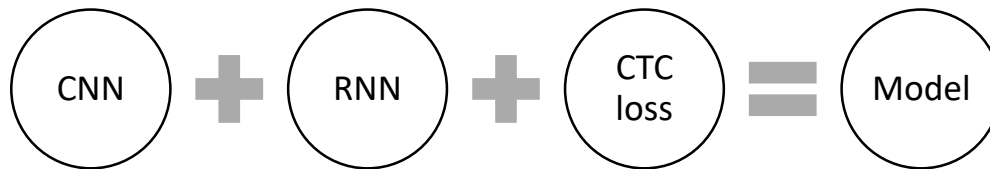
The YOLO algorithm is then used to forecast the bounding boxes and class probabilities for objects (if any are found, of course). As a consequence, the label y for each grid cell will be an eight-dimensional vector in which: pc indicates the presence of an item; y denotes the label for the cell; z denotes the label for the cell; and y denotes the label for the cell. The bounding box is defined by the numbers bx , by , bh , and bw , which are all positive integers. The classes that need to be recognized are indicated by the letters $c1$, $c2$, and $c3$.

3.3 Region detection and text detection

Recurrent Convolutional Neural Network (RCNN) for text recognition and ingestion of the text detected region It is then passed through the Seq-Seq Encoder/Decoder model, with the resultant Translation shown on the screen as a consequence of this. CRNN is Convolutional Recurrent Neural Network and is a consists of CNN, RNN, and CTC loss. The functionality of this model is to recognized image based upon sequence application like scene text recognition and optical character recognition

(OCR). One of the most interesting aspects of this neural network architecture is that it integrates feature extraction, sequence modeling, and transcription into a unified framework. This approach does not require the use of character segmentation.

In this case, the convolution neural network is used to extract one or more features from the input image (text detected region). Because of a connection between the characters, the deep bidirectional recurrent neural network predicts label sequences based on the relationship between the characters. The transcription layer converts each frame of the RNN into a label sequence, which is then sent to the next layer. Transcription may be divided into two categories: lexicon-free transcription and lexicon-based transcription (see below). The lexicon-based approach will be used to forecast the label sequence that is most likely to occur.



This model is comprised of three critical components:

- i. The CNN is used to extract features from the pictures that are being processed.
- ii. A Random Neural Network (RNN) is used to predict the consecutive result on every occasion.
- iii. The loss function is used to update the parameters of models in the third instance.

It is possible to describe the model's architecture as follows:

- i. the input images are resized as 32 x 128 pixels.
- ii. Seven convolution layers with size (3,3) and last having size(2,2) are used. The number of filter varies from 61 to 512 layer by layer.
- iii. the two max-pooling layers with size (2,2) and (2,1) are used to extract features.
- iv. Batch normalization is used to accelerates the training.
- v. Lambda functions are used to squeeze output from convolution layers and compatible with LSTM model.
- vi. Two Bidirectional LSTM with size 128 units this RNN layer gives output of number of classes used.

Loss Function

Now that we have created the model architecture, the following step is to choose a loss function for the model. Using the CTC loss function, we will attempt to recognize the text in the given situation. Text recognition issues benefit greatly from the use of CTC loss. It assists us in avoiding the need to annotate each time. When using CTC, we are able to avoid the issue where a single character may span several time steps, resulting in the requirement for additional processing if we do not utilize CTC. In order to calculate the loss, a CTC loss function needs four arguments: the predicted outputs, the ground truth labels, the input sequence length to the LSTM, and the ground truth label length. In order to do this, we must first build a custom loss function and then send it along to the model. It is necessary to build a model that accepts these four inputs and outputs the loss in order to make it consistent with our model.

3.4 Machine Translation

Encoding and decoding the text using the Sequence to Sequence Encoder-Decoder Model is being done on this recognized text. Raw text data and pictures would be the types of data that would be received by the product. The translated text would be the data that was sent out. Table 1 shows the results of a comparison analysis between three models.

Table -1 (Comparison of three models)

YOLO MODEL	CRNN MODEL	SSED MODEL
For training this model we need a dataset which has images containing text	For training this model we a dataset that contains text images parallel with their respective texts	For training this model we need a data set that contains Text parallel with their translation in English

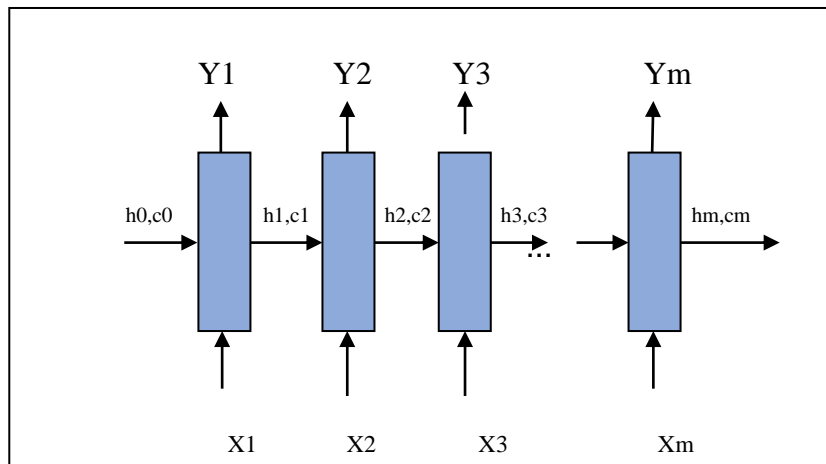
There are a variety of systems that you encounter on a daily basis that are based on the sequence-to-sequence paradigm. For example, the sequence-to-sequence paradigm is used to power applications such as Google Translate, voice-activated gadgets, and online chatbots.

In machine learning, an array of multiple recurrent units (LSTM or GRU cells for better performance) is used to take a single element of the input sequence and collect relevant information about that element before propagating it forward. When it comes to question-answering problems, the input sequence is a collection of all of the words from the inquiry.

Each word is represented by the symbol x_i where i indicates the word's position in the sentence.

The full process for using the encoder is shown in the following steps.

- i. In the first case, an encoder receives an input sequence and encodes it as internal state vectors.
- ii. It is also possible to use just internal states since the encoder's outputs are discarded. Let's look at how the model's encoder works-



This methodology uses one input at a time and checks the length m and uses steps according to the length for the entire input for reading. Then h_t and c_t become internal states and during the time frame t become for LSTM and h_t for GRU. The t and h_t are the internal states. Y_t becomes the output for time step t .



Figure 2: Text translation

Consider for translation from English to Hindi, now for the Encoder we have taken X_t then: Suppose my English sentence "My country is Great". This sentence can be used for encoding as it is consisting of four words (My -1, country-2, is -3, great-4), so this makes a sequence of the sentence like X_1 ="My", X_2 ="country", X_3 ="is" and X_4 ="great". Now LSTM will read this whole sentence in the sequence starting from X_1 and ends with X_4 . The complete text from English to Hindi word by word will be done by Sequence-Sequence Encoder-Decoder Model. In this model we have used X_t as a vector for word embedding which converts each word into a vector of fixed length.

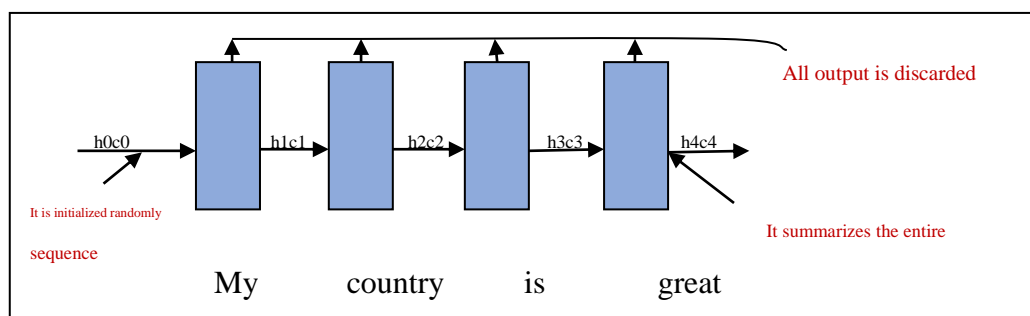


Figure 3 : Sentence Encoder

Now, let's talk about internal states (ht,ct) -

- i. It discovers what the LSTM has read up to time step t. For example, at t=2, it recalls that LSTM read “My country”
- ii. The starting states can be initialized by the user as we have used ho, co (both vectors) are set at random or with zeroes.
- iii. While using LSTM the dimension of ho, co must be equals the number of units in cell.
- iv. The last state h4,c4 includes the pivotal phrase of the whole input sequence "My country s great."

Now we have to check the output of encoder Y_t . In each step of LSTM Y_t is treated as predictions(behavior). When we do the machine translation the entire input sequence from $(X_1 \text{ to } \dots X_m)$ is taken as input for generating outputs. Y_t is used and being discarded by the encoder at each time step

While concluding the encoder part of the model:- Here, at this step the encoder will read the sentence(as we have written in English so it will use this language) word for word and save the final internal states (known as an intermediate vector) of the LSTM generated after the last time step. Because the output will be generated once the entire sequence is read, the Encoder's outputs (Y_t) at each time step are discarded.

Hidden State(Encoder) : It is the final state obtained from the encoder. It encapsulates the information for all input elements to find encoder accurate predictions.

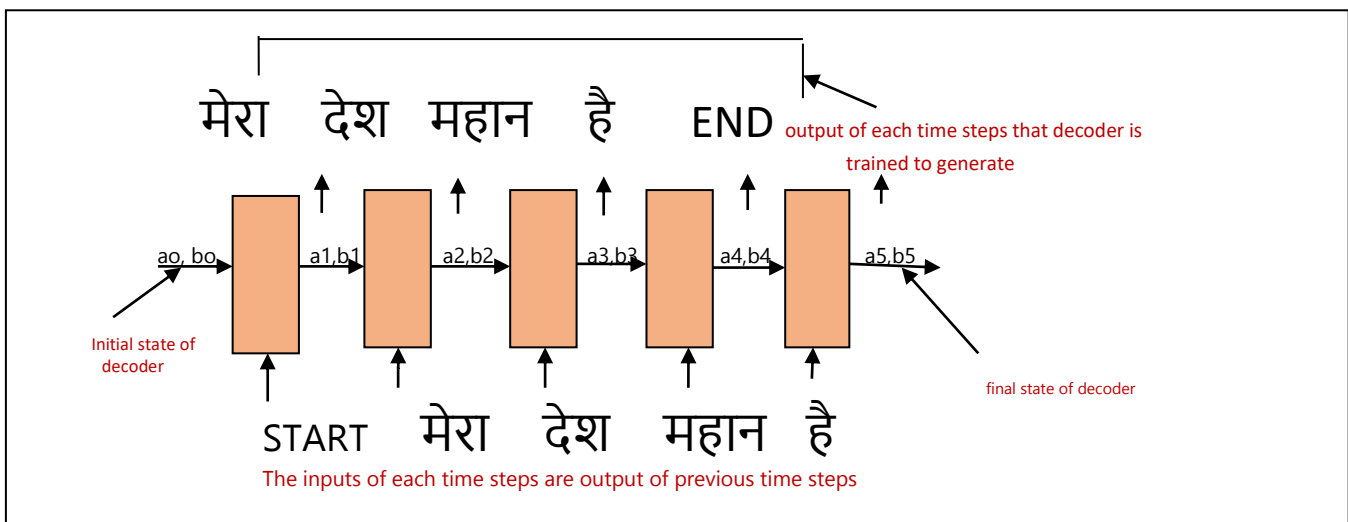
Decoder: The recurrent units, each of which predicts an output y_t at a time step t. Each recurrent unit receives a secret state from the preceding unit and generates both an output and its own hidden state.

While computing the outputs of combining the hidden state for current time step with the corresponding weight $W(S)$. SoftMax is a tool to generate a probability vector that will help in determining the final output.

The decoder operates differently throughout the training and testing phases, as opposed to the encoder, which operates in the same manner during both phases.Let's have a look at how the decoder works during the training phase. Let's take an example from English to Hindi translation “My country is great”. So encoder and decoder translates each word as an output in a sequence manner.

So, to get the desired and final output — मेरा देश महान है . Here, the decoder will work as the initial and final state of the sequence labeled as “start & end”, so we will add $START_$ at the initial stage of the output sequence and $_END$ at the final stage of output sequence.

So, the output of given sentence will be $START_$ मेरा देश महान है $_END$. Let's understand the working visually-



- v. The decoder's final states are removed because it is useless. The encoder decoder is visually shown in Figure 5.

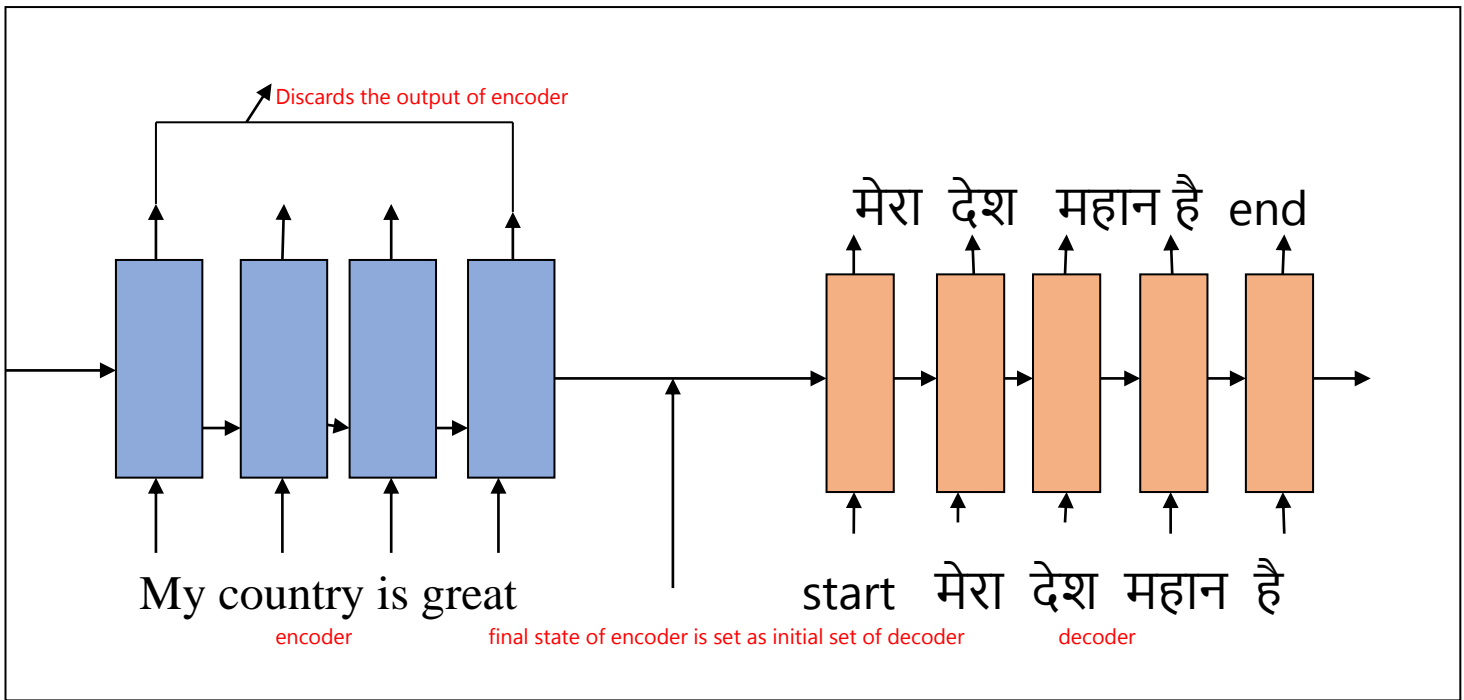


Figure 5 : Encoder Decoder together

The state chart diagram is shown in Figure 6.

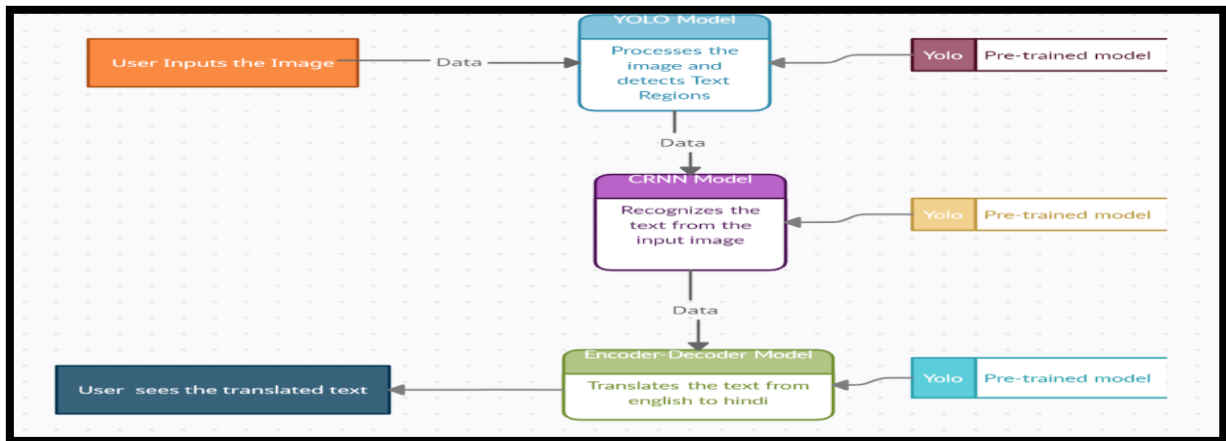


Figure 6: State chart diagram.

The processed all these steps are summarized in Figure 7.

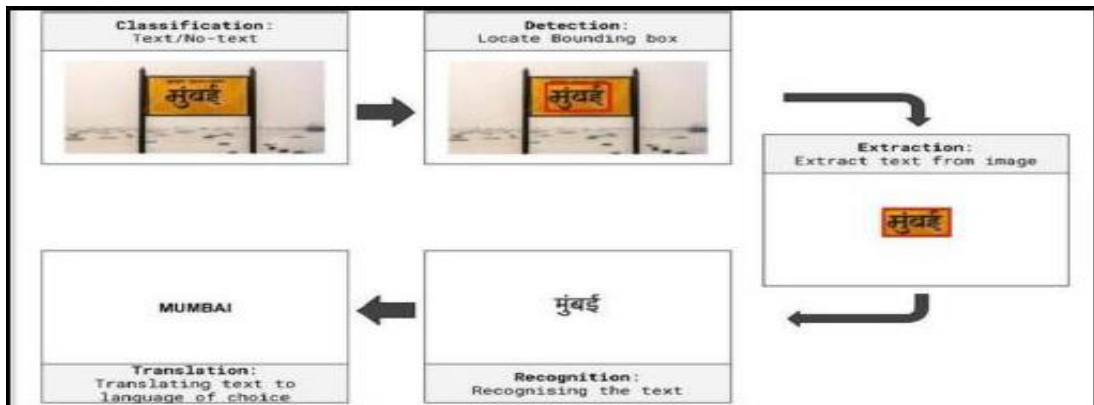


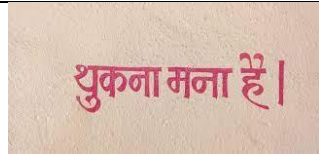
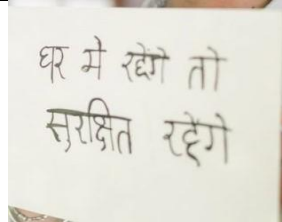


Figure 7: Processing steps used.

4. RESULT ANALYSIS

The experiments are performed on 200 different scene images. Some sample results are shown in table xx. accuracy of this model lies between 90 to 98.2 %. The techniques performed better on images with average quality images where as it provides poor results in case of blur images.

S.No	Scene Image	Recognized Text	Translator	Accuracy (%age)
1		कोई फोटोग्राफी नहीं	No Photography is	98.2
2		अंदर आना मन है	no entry	97.5
3		थूको मत	Do not spit	95
4		सुरक्षा के लिए घर पर रहें	for safety stay home	90

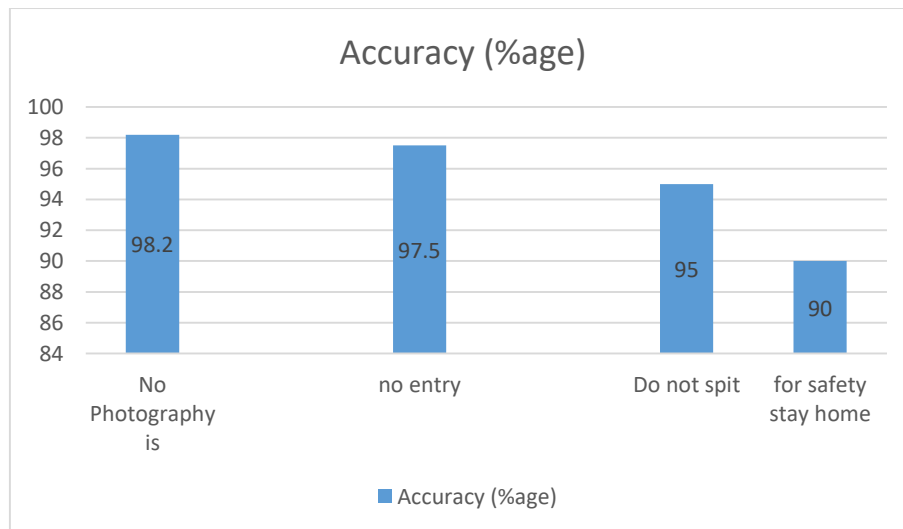


Figure 8 Recognized text with accuracy.

The detailed recognized text with accuracy is shown in Figure 8. The accuracy ranges from 90% to 98%.

Following are the challenges in these models.

- Image containing the Multi-lingual Text
- Cropped Region of Image that Contains Hindi-English Text
- English words and Sentences

5. CONCLUSION

In this paper we proposed Optical Text Recognizer and Translator using RCNN and Yolo model. The experimental results show better accuracy than other tradition methodology. The system is well suited with images with high quality whereas performance decreases in case of noisy images. The recognition accuracy ranges between 90 to 98.2 %. However the recognition accuracy also depends on availability of dataset. The system requires ample disc space at time of training the model. Still there is scope to make this system faster and more accurate. Also system lacks in translation to different mixed languages.

REFERENCES

1. Y. F. Pan, X. Hou, C.-L. Liu, A hybrid approach to detect and localize texts in natural scene images, *IEEE Trans. IP* 20 (3) (2011) 800–813.
2. X. C. Yin, X.-W. Yin, K.-Z. Huang, H.-W. Hao, Robust text detection in natural scene images, *IEEE Trans. PAMI* 36 (5) (2014) 970–983.
3. L. Sun, Q. Huo, W. Jia, K. Chen, A robust approach for text detection from natural scene images, *Pattern Recognit.* 48 (9) (2015) 2906–2920.
4. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg, SSD: single shot multiBox detector, in: *ECCV*, 2016, pp. 21–37.
5. J. Redmon, S.K. Divvala, R.B. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: *CVPR*, 2016, pp. 779–788.
6. S.-Q. Ren, K.-M. He, R.B. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, *IEEE Trans. PAMI* 39 (6) (2017) pp. 1137–1149.
7. J.-Q. Ma, W.-Y. Shao, H. Ye, L. Wang, H. Wang, Y.-B. Zheng, X.-Y. Xue, Arbitrary-oriented scene text detection via rotation proposals, *IEEE Trans. Multimedia* 20 (11) (2018) 3111–3122.
8. Zhong, Zhuoyao, Lei Sun, and Qiang Huo. "Improved localization accuracy by LocNet for Faster R-CNN based text detection in natural scene images." *Pattern recognition* 96 (2019): 106986.
9. Yu Zilong Zhang, Fu, Fuyu Huang, and Yizhi Liu. "PMMN: Pre-trained Multi-Modal Network for Scene Text Recognition." *Pattern Recognition Letters* (2021).
10. Rajiv Kumar, Shivani Joshi, and Avinash Dwivedi. "CNN-SSPSO: A Hybrid and Optimized CNN approach for peripheral blood cell image recognition and classification." *International Journal of Pattern Recognition and Artificial Intelligence* (2020): 2157004.
11. Shivani Joshi, Rajiv Kumar, and Avinash Dwivedi. "Hybrid DSSCS and convolutional neural network for peripheral blood cell recognition system." *IET Image Processing* 14, no. 17 (2020): 4450-4460.
12. Rajiv Kumar, Kiran Kumar Ravulakollu, and Rajesh Bhat, "Fuzzy-Membership Based Writer Identification from Handwritten Devnagari Script," *Journal of Information Processing Systems*, vol. 13, no. 4, pp. 893~913, 2017. DOI: 10.3745/JIPS.02.0018.