Report on the application of OCR on Forms using Tensorflow

Vedha Krishna Yarasuri; vedhakrishnapanchami@gmail.com

Table Of Contents

Introduction

Optical Character Recognition (OCR) systems mainly focus on extracting text from scanned documents.[1] Optical character recognition converts images of typed, handwritten or printed text into machine-encoded text (for example the text on signs and billboards in a landscape photo). Extracting Textual characters from a given image is a challenging Task in the field of deep learning. However this acts as a base for machine level understanding of a given scenario.Text Extraction from natural scenes is more challenging due to various visual factors such as intensity variations, focal variations, etc.[1]. In this assignment work, I worked on a feature extraction and attention based model which is used for getting the text embedding of an image. The dataset was produced by IIIT Hyderabad. It contains 5000 instances of cropped images. The images have been taken from billboards, signboards, house numbers. However the images are then increased with the help of data augmentation techniques. The labels of corresponding images are converted to an array of numbers for output. After acquiring the dataset, the next step is extracting the features of the images. Inception-Resnet-V2 model gives higher accuracies outperforming InceptionV2 and Inception V3 by giving an accuracy of 84.2%[1].

The Inception-resnet-v2 model has been taken as the feature extractor and the features obtained from it are then used to train an encoder-decoder model. The features are given to the encoder and the results are obtained from the decoder in the format of an array. The array is then decoded to get the corresponding name of the image. Later validation sets have been used to validate the model.Fig 1 gives an idea of the model.

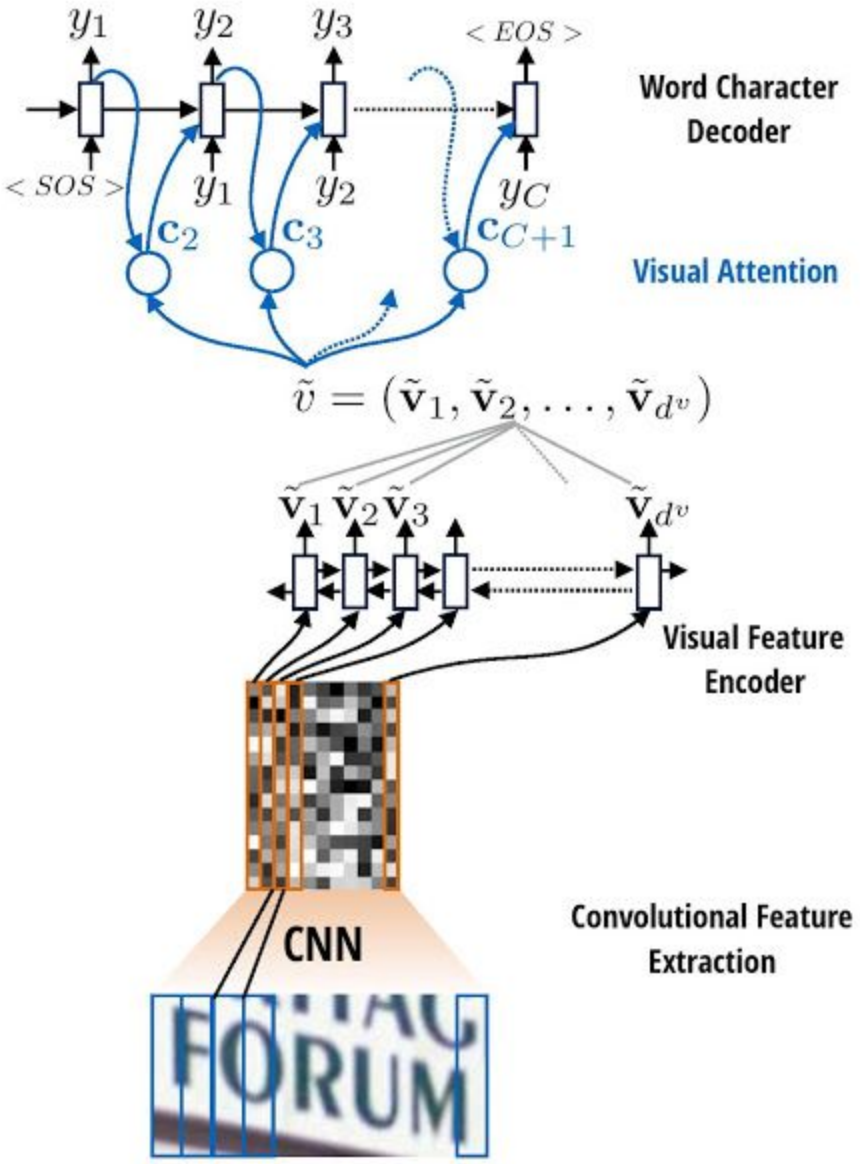

Fig 1 : The complete picture

System Architecture

1. Data Acquisition and Data Augmentation

Training a deep learning model requires lots and lots of data. For training an OCR model a large number of images and corresponding labels need to be used. IIIT Hyderabad has been working on this for a time and they generated a dataset named IIIT 5K. The IIIT 5K-word dataset is harvested from Google image search. Query words like billboards, signboard, house numbers, house name plates, movie posters were used to collect images. The dataset contains 5000 cropped word images from Scene Texts and born-digital images. The dataset is divided into train and test parts. This dataset can be used for large lexicon cropped word recognition. They also provide a lexicon of more than 0.5 million dictionary words with this dataset. However ,The dataset size is very small to train a huge model. Keeping this into consideration, 3 methods (Techniques) of data augmentation have been implemented. They are :

1. Rotating Images by + 15 and - 15 degrees.
2. Blurring Images using PIL Library.
3. Adding uniform noise to the dataset.

In the beginning the dataset size was only 5000, later on implementing the above mentioned techniques, The dataset has been increased to 25000.

Fig 2 : Few Instances From the dataset

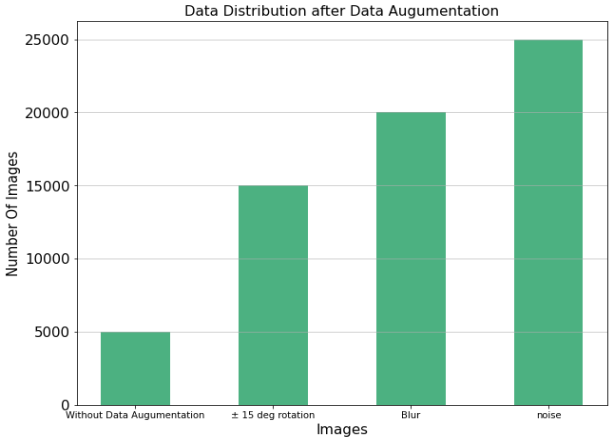

Fig 3 : Data Distribution after Data Augmentation

After Data Augmentation, the images are then passed to the feature extractor model to get features.

Fig 4 : Visualizing Images In their Different Formats

2. Feature Extraction

Inception_resnet_v2 model can give a greater accuracy with its features.[1]. Using Tensorflow, I have imported the pre-trained inception_resnet_v2 model model. I did not re-trained the model and froze the pre-trained model weights. In general, people think that the beginning layers of the deep learning model will learn the edge features. The deeper layers learn to detect various objects,etc. So we cannot completely rely on the features obtained in the beginning as the model did not learn much about the images. However The deeper layers could also cause problems based on the Use Case. So Many deep learning scholars recommend using the middle layers. From[1], the feature_extractor gave good results in the layers “block8_1_conv” and “mixed_7a”. These layers however give similar results. With the Input layer and output layer as “block8_1_conv”. I have saved the feature extractor model.

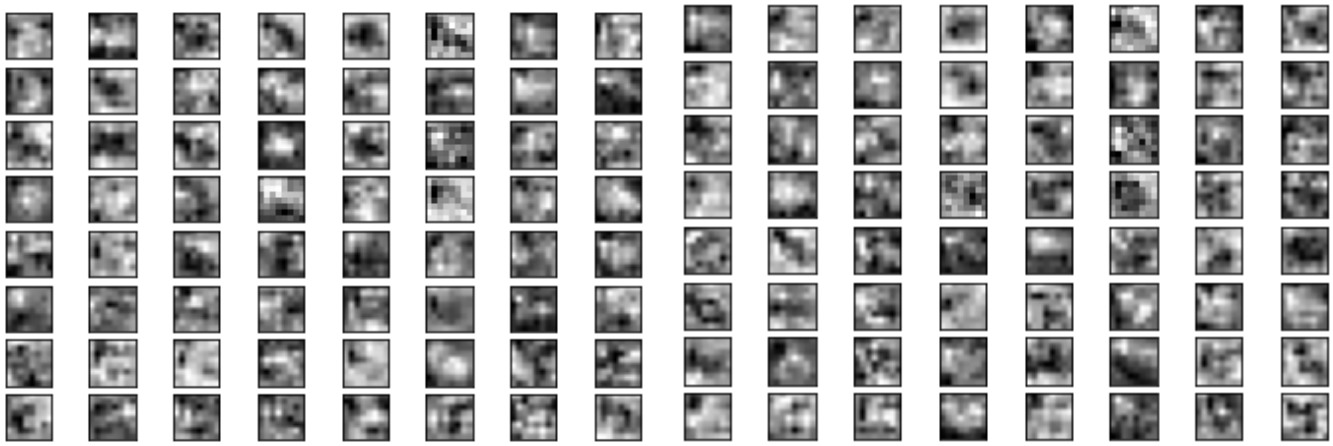

Fig 5: Extracted features of the image labelled “ You ”

These are few instances of the features that the model has extracted.The features are stored using numpy. I stored the features as this would decrease the training time of sequence models. The stored features are then loaded while training the sequence models (Encoder, Attention, Decoder). While validating the images, The images are passed through the model and the obtained features are sent to the sequence model for testing and validating.

3. Sequence Model

The sequence model follows the attention based approach which consists of an encoder, attention and a decoder. The encoder consists of a general neural network with 256 units. Its output is connected to the attention network. Attention consists of a neural network with an architecture of 1 input layer, 1 hidden layer and 1 output layer. Input and hidden layers consist of 512 units. This network computes the attention weights and context which is to be passed to the input decoder. The decoder consists of an attention network, embedding layer followed by a gru layer and 2 dense layers. The final dense layer gives the output probabilities of the predicted character.

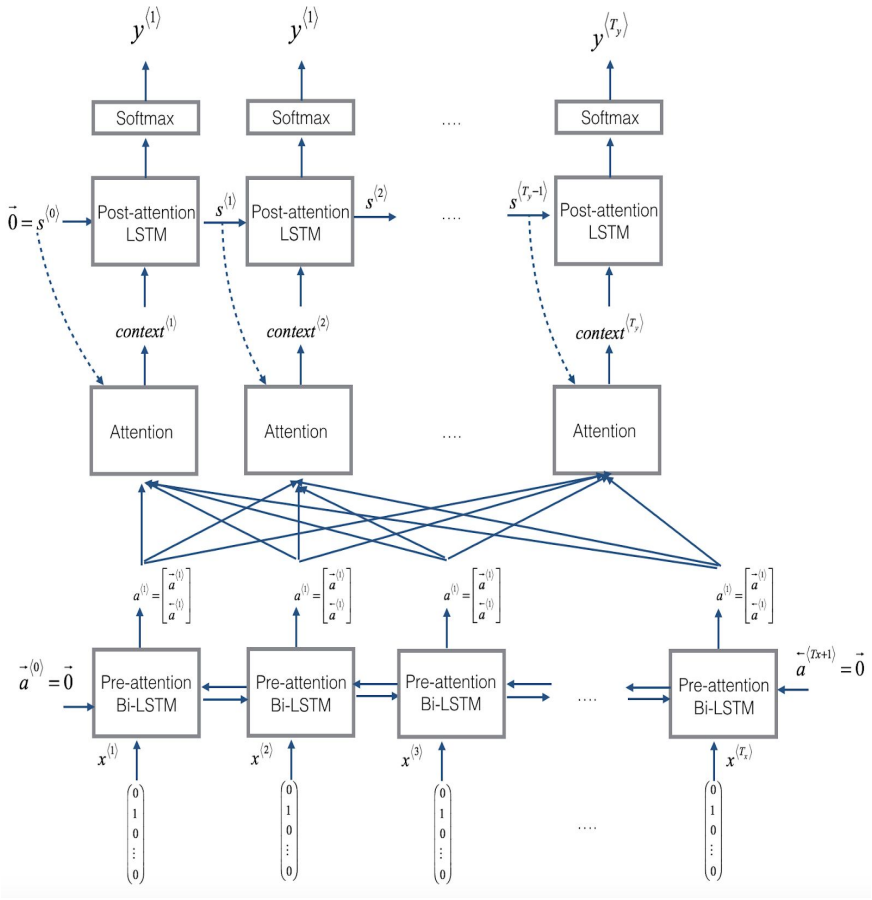

Fig 6 : General Attention Based Sequence to sequence Model

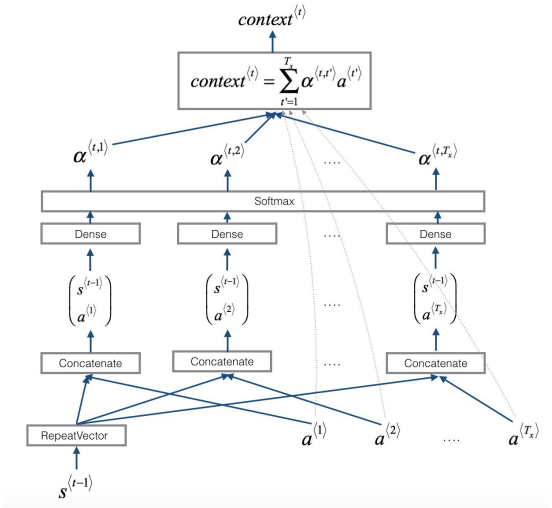

Fig 7 : Inside part of Attention Block

Fig 6 and 7 are the basic generalized versions of attention based sequence models. For the given problem statement, We can make few changes in selecting various layers but the concept remains the same. I have selected a shallow neural network as encoder, the attention network remains the same and as the decoder, I have used an embedding layer along with GRU and a dense network.

Hyperparameters
Hyperparameters are one of the main parts in designing a deep learning solution. Hyperparameters can be chosen in any way. The Ideal hyperparameter solution was very difficult to build. For this problem statement, I have chosen these values as my hyperparameters.

Hyperparameter Name	Value
BATCH_SIZE	64
BUFFER_SIZE	1000
embedding_dim	256
units	512
vocab_size	41
num_steps	609
features_shape	2080
attention_features_shape	64

Training, Saving the model weights

The model has been trained over 20 epochs and achieved an accuracy of 52%. The corresponding weights have been saved and are downloaded which can be later used for further training or converting the model to tensorflow js or tensorflow lite model.

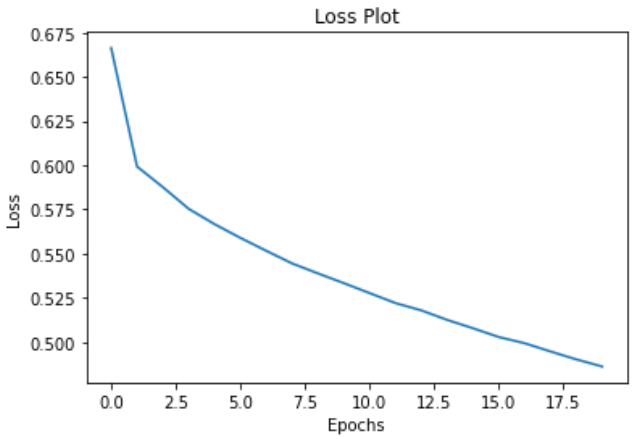

Fig 8 : Loss Plot

Web App

The Web application was created using Node.js for the backend and by bootstrap for the frontend. The web app is a simple application. It has an index.html file which displays a welcome page to the user and asks the user for an image as input. When the user selects an image and uploads the selected image, then the server receives the post request using REST api. Multer takes care of saving the image to the server and deletes it if needed. Rest api can be used to take in the input data from any from. Since the image is not like a normal text, it uses the help from multer in taking the file and saving it in the server. When the image is uploaded and stored, the plan was to take the saved image and send it to the feature extractor for extracting the features using tensorflow.js. The extracted features are then passed to the sequence model for evaluation. The result of the model is then intended to give a response back to the browser using the same REST api. However the model evaluation could not happen for me during this period. The REST api was completed but the model evaluation could not be able to complete. Fig No 9 and No 10 . Shows images of the home page of a web application .

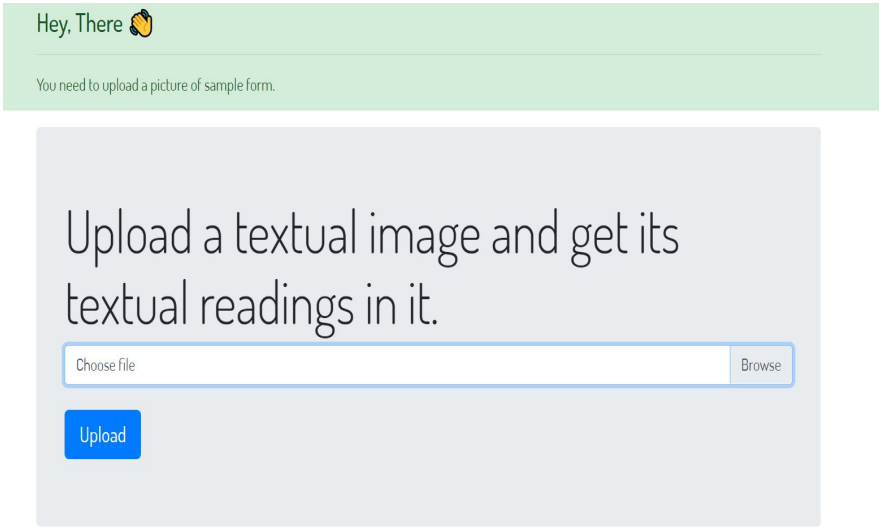

Fig 9: Home page of the web application

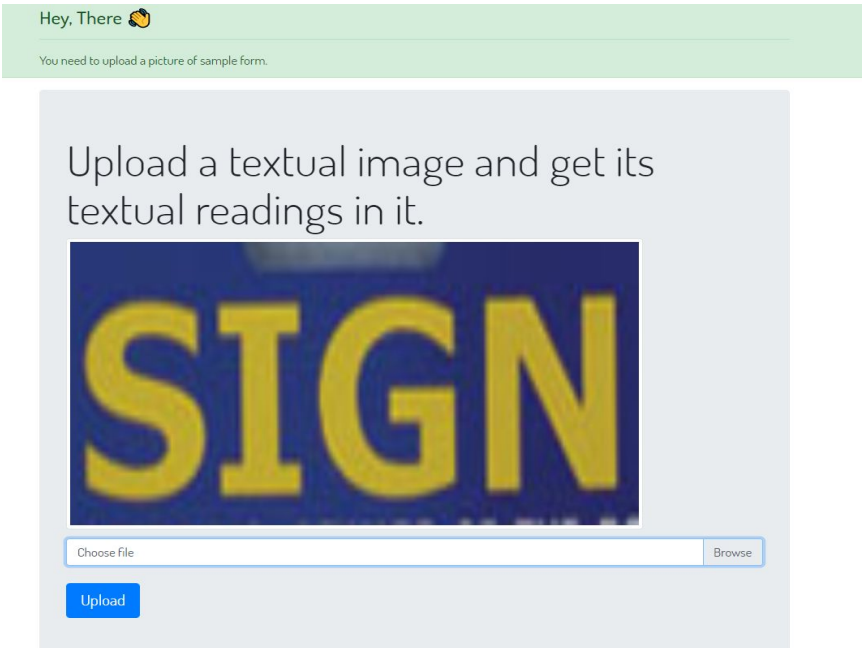

Fig 10: The Image Uploaded by the user.

Challenges Faced and Their Solutions

I have faced 4 problems and tried to overcome them as much as I can. Here is the list of the problems that I have faced during solving this assignment.

1. Lack of Experience and Practice
2. Designing Sequence Models
3. Tensorflow.js

4. Shell Scripting

Lack Of Experience and practice :

I have an year of experience in the domain of machine learning. I started learning neural networks, svm, regressions till December 2019. I got much interested in deep learning and started Deep learning in the month of January 2020. Due to the lockdown period I have taken deep learning specialization from coursera and gained a lot of the knowledge through the course. From the knowledge I gained from the course, I have learnt to deal with images and pre trained models such as VGG 19 and mobilenet. Later I have built an Object detection model using pre-trained models. But I did not work in the domain of text. I have applied for this internship due to the knowledge I have learnt from the specialization. When the assignment was announced I understood that I need to take features from the images and then do something with them. But to know what happens after extracting features, I spent around 4 days in learning sequence models. Later from the examples from tensorflow Image captioning, I have tried to replicate the same by changing many parameters.

Designing Sequence Models

As said in problem 1, I was not much experienced in text related data. I have a good experience in working with conv networks and pretrained models. So initially I could not understand the suggested paper properly. Later after spending 4 days in learning rnn and lstm and mainly attention models from sequence models in coursera. From the knowledge I gained from the course, I have tried to implement attention based encoder-decoder but I was not able to complete it. Later I have seen examples from tensorflow's documentation on language translation and image captioning and tried to replicate encoder, decoder, bahdanau attention classes.

Tensorflow.js

I have 2 years of experience in designing web pages. So I designed REST api without much problem, but I do not have much experience in bringing the trained models from tensorflow to javascript. In the past I designed small scale sequence models like ANN, using tensorflow and loaded them in browsers using tfjs functions. I have also imported pre-trained models such as mobilenet using cdn's. but this time I need to bring the whole sequence model which is an object of subclass. Also there is no InceptionResnetV2 model cdn to call through javascript. I have tried to convert the feature_extracter_model using

tensorflow js converter, but I was not able to load it on the server. Due to these reasons I was not able to complete the web app completely.

Shell Scripting

I have tried to learn many new concepts and try to complete this assignment upto a great extinction, However for me learning the sequence models took a great amount of time this week. So I was not able to learn shell scripting completely due to this time factor. I tried to implement how much ever so was possible by me.

References

1) @INPROCEEDINGS{8270074, author={Z. {Wojna} and A. N. {Gorban} and D. {Lee} and K. {Murphy} and Q. {Yu} and Y. {Li} and J. {Ibarz}}, booktitle={2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)}, title={Attention-Based Extraction of Structured Information from Street View Imagery}, year={2017}, volume={01}, number={}, pages={844-850},}

2) @InProceedings{MishraBMVC12,
  author   = "Mishra, A. and Alahari, K. and Jawahar, C.~V.",
  title   = "Scene Text Recognition using Higher Order Language Priors",
  booktitle = "BMVC",
  year   = "2012",
}