Attention-OCR-IITB-Assignment

14.12.2019

Laxya Agrawal

E-mail： lakshagarwala@gmail.com t

Github link ：https://github.com/laksh9950/Laxya_Agarwal-IITB-Assignment

Overview

This project uses text-renderer to generate synthetic images in addition to external dataset and uses attention-ocr to train the model. Generated model will have some limitations, such as MaxHeight allowed is 48, MaxWidth allowed is 960 and MaxPrediction length is 23 characters. Supported characters

Here , we have trained the model on the generated dataset and exported the model which is giving the best results . Inherently , the model is not 100% accurate but it is giving good Inference.

Goals

- Generate Synthetic Images.
- Split them into Training and Validation Set.
- Generate TFRecords.
- Initialize training by changing some hyperparameters.
- Export the inference Graph.
- Evaluate the Model.
- Make a Web-app on localhost to serve the ML Application.

- Host the Web application on Heroku.

Specifications

Supported images are of width 960 and height 48. As the input of the model have fixed number of nodes, any image bigger than the specifications will get skipped and might not generate any result.

But most of the images according to our specification of detecting single words will be in that dimensions so for very extreme cases there might be problem but not for general cases .

Many things from the source code available from the github is being changed for achieving the goals. Folders in the repo provided are the updated ones. Initially in text renderer master directory, parse_args.py is been manipulated. In that file, image height and width are fixed, although it can be manually changed by passing appropriate arguments. Moreover, I have removed the usage of any chinese text present while generating images. Also I have added Background Images, Fonts .

Additionally, Attention-ocr model is being tweaked in the prediction region. You may find the updated file in attention-ocr-master/aocr/__main__.py . The change is present from line 267 in the same file. Code is tweaked in order to get a text file with Image_file_name alongside predicted text. Command for generation can be found in instructions.txt, as well as my_run.sh . You may get an error while executing ./my_run.sh , for instance you can use the command mentioned in instructions.txt . The arguments --dirpath will take test image folder name and --out_file_name will take the name of the output file.

Challenges Faced

- ## Generation of Synthetic Images

  Initially the standalone repo of text-renderer was only able to generate images with one kind of background image and only one kind of font. Moreover, there was only a single corpus which had only one paragraph, that too includes chinese as well as enlish characters together. I added a bunch of new background images, new fonts, changed the corpus by generating different kinds of text files using Faker in python. Merged them together and copied it to text-renderer-master/data/corpus. Detailed info about this step can be found in instructions.txt. I have also used external data from various sources to generalize better . Here for most of the conventional fonts and backgrounds the model is giving great accuracy .

- ## Training the Model

  During the training process, I was unaware about some of the parser arguments which are necessary to include, such as --max-width, --max-height,--max-prediction, as it might affect the training. Initially, I just tweaked the learning rate, which was creating a problem for me. Later I realized that the default values of parameters are responsible for the errors, as they are skipping the images. I updated the commands and passed appropriate arguments in order to achieve the goal. Commands regarding the training process which I've used are mentioned in intructions.txt. Here , I have tried many things to use pretrained models but the aocr repository structure is

very complex and due to the use of tensorflow 1.15 most of the pretrained models are available today can be accessed using keras only . I have tried a Resnet50 but it is not giving very good accuracy . so I have used only the original model for the final prediction.

- ## Testing and Prediction

  While testing the Model initially, I found that standalone attention-ocr only uses uppercase A-Z characters in their charmap. So I changed the charmap according to the needs and re-trained the model. After retraining, there was no built-in way to generate a text file as required in the assignment. So I tweaked the main.py from aocr to satisfy the requirements. Command to generate the predicted file is present in instructions.txt.

  While training some of the things I tweaked were giving better performance than provided in the original repository. Model took very long to converge more than 10 hours of training on Google colab .

- ## Web-app

  This is my first interaction with Flask. I referred to many tutorials to achieve the goals. I'll link those in the references. Implementation of web app on localhost was straight-forward. The problem I faced when I tried to implement the same on heroku. Initially the error showed there is some problem with libraries, so I tweaked the requirements.txt according to the needs. Then there was a problem with tensorflow version 1.15.0 as it's not supported by Heroku. The lowest version after 1.15.0 supported by Heroku is 1.14.0, So I re-ran the process of exporting the inference graph on tensorflow 1.14.0, Now, I started to re deploy the application on Heroku, but now the tensorflow on Heroku is unable to implement the application. I tried to make everything from start in tensorflow 1.14.0 and tested on my localhost. It runs perfectly fine. But Heroku is not able to process the files with tensorflow. Worker terminates after executing my python file. There is a issur in Heroku itself for deploying tensorflow models . My model works perfectly on the localhost .

References

- https://github.com/emedvedev/attention-ocr/issues/146
- https://github.com/emedvedev/attention-ocr/issues/155
- https://github.com/emedvedev/attention-ocr/issues/145
- https://github.com/emedvedev/attention-ocr/issues/143
- https://github.com/emedvedev/attention-ocr/issues/141
- https://github.com/emedvedev/attention-ocr/issues/89

- https://github.com/legolas123/cv-tricks.com/blob/4a6fe329f6f1aa9542901a2b363c3e55789fdd71/Tensorflow-tutorials/freeze_model_and_deploy/webapp.py