Everyone can use CRNN to recognize text images

Mingwei Li*

Abstract

With the increase of data and computation power, deep learning is widely adopted in multiple fields. Text images recognition is a classical and valuable task, and deep neural network is able to provide us with satisfying results. The following 4 pictures demonstrate the performance of our neural network. Our model supports **both** Chinese and English.

电子不停车收费系统(Electronic Toll Collection System,简称ETC),是通过"车载电子标签+IC卡"与ETC车道内的微

Figure 1: The Performance on Mixed-Language Text Images

原标题:腾讯回应"冻结老干妈1600万财产":系老干妈拖欠千万广告费,被迫起诉

Figure 2: The Performance on Chinese Text Images

The MA program offered by the Department of Economics is STEM-designated and offers a rigorous curriculum

```
file_path is:asset\economics.jpg
result is:The MA progam ofered by the Departnent of Econonics is STEM-designated and ofers a rigorous curicuum
elapsed time is:1.177s
```

Figure 3: The Performance on English Text Images(1)

The imported patients include a Chinese who studied in the UK, a Chinese who worked in the

```
file path is:asset\english.jpg
result is:The imported patients include a Chinese who studied in the UK, a Chinese who worked in the
elapsed time is:1.14s
```

Figure 4: The Performance on English Text Images(2)

Just as the title indicates, this essay aims to tell you how to build an OCR model from nothing. We hope that even new beginner in python can benefit from this work.

The whole passage has 3 chapters. Chapter 1 is the user guidance of the project. It tells you how to use this project to recognize your own images. Chapter 2 is the technical guidance of the project. All technical details to build the model are involved in this chapter. Chapter 3 is an introduction to the principles of CRNN.

The source code of this essay is available at here. Please star my project if you feel it's helpful, thanks!!

^{*}Mingwei Li is a master student of economics, The University of Texas at Austin

Contents

1	User Guidance	3
	1.1 File Paths	
	Technical Guidance 2.1 Data Management	
3	The Principle of CRNN	7
4	Prospect	8

1 User Guidance

The project is adequate to recognize your own images as the model weight is preserved in our github repository. You need to download train data if you want train your own CRNN model.

1.1 File Paths

At first, you need to save all your images under asset/directory. Please note! This model could only recognize single line of text. Because of the parameter sharing mechanism of RNN and CNN, our model is able to recognize single text lines of flexible length. Specifically, you can use this model to recognize a sentence consisted of more than 1000 words. However, the input image should have only one text line!



Figure 5: Asset Images

Secondly, you should execute order line **python demo_infer.py**, and the recognition results of all images under **asset**/ directory would be presented as followed.

```
file_path is:asset\economics_1.jpg
result is:2. Obtaining the backgound in econonics and mathematics required to gin admission to a high-quaity elapsed time is:1.085s

file_path is:asset\economics_2.jpg
result is: '> The progrars STEM designation neans that international students may receive a 24month extension of elapsed time is:1.123s

file_path is:asset\english.jpg
result is: The imported patients include a Chinese who studied in the UK, a Chinese who worked in the elapsed time is:1.18s

file_path is:asset\finance.jpg
result is: newpiles 采用201 3—2018年我国沪深A股非金融类公司和房地产上市公司数据,研究金融资产配置及其 elapsed time is:0.705s
```

Figure 6: Recognition Results

The model weight of this project is saved as weights/CRNN.h5.

1.2 Infer Details

- (1) Image Processing: At first, all images are converted to gray-scale images. Then, they are resized to make sure the height of input images is exactly 32 pixels. The corresponding function is at here.
- (2) Basic Division: The greatest weakness of CRNN is that such neural network can't recognize the space between words naturally. Moreover, the white space between different words may harm the performance of CRNN. Therefore, it's necessary to conduct some basic division. The corresponding function is at here. The input image is binarized twice. Cut lines would be added to spaces which have more than 5 consecutive pure black pixels. Figure

7 demonstrates the performance of division. **Please note!** CRNN doesn't need to divide a sentence into single characters. However, huge space between different words should be deleted.

原标题: 腾讯回应"冻结老干妈1600万财产": 系老干妈拖欠千万广告费,被迫起诉原标题: 腾讯回应"冻结老干妈1600万财产": 系老干妈拖欠千万广告费,被迫起诉The MA program offered by the Department of Economics is STEM-designated and offers a rigorous curriculum

Figure 7: Division Performance

(3) Recognize small pieces: The whole image is divided into small pieces after the previous 2 steps. Then, those small pieces are sent to neural network. Background color is added to the right side of each single piece, and the corresponding function is at here. The left sub-figure of figure 8 is the original piece, and the right sub-figure is the piece after color adding. According to our experiment, color adding is able to increase recognition performance slightly. The reason of this phenomenon may be the fact that some training images have blank space on their right sides.



Figure 8: The Performance of color adding

Figure 9 is a part of training data. In fact, most of the training data is Chinese rather than English. Therefore, our model performs better on Chinese recognition than English. More English data is needed to obtain a more robust model.

H) . 2005-10	推断出,共同面对庞大	。为怀不特。然后选择	能是世界第一。绿姨就	从而达到废实相生的效	总一定在电梯里看见我	侯八年、公元前815	学、淡雪尽铬。并最胜	夏商周新代工程"的基	思波说,除了认真驾驶	投資者在对市场遺傳所	遂已至家,搭建地大就
21006000_9963	21006000_2318	21006015_3015	21006015_3411	21006031_3756	21006078_4246	21006109_2097	21006140_1775	21006140_2688	21006140_3271	21006140_3529	21006156_2368
05872.jpg	770282.jpg	581194.jpg	519248.jpg	349758.jpg	507395.jpg	456779.jpg	374782.jpg	213383.jpg	972777.jpg	979434.jpg	757893.jpg
次"新势力2006个	涛的流通股股东州持来	代和下限年代的考古標	十二章说: 学了好几个	以当年辽宁省的价局批	荐机场太阳饭店 (Ai	我也管理特殊, 伊特尔	射的导弹使用的是固体	专家建议: 揭此, 后有州	其用心不难体会。神农	是国的集体祭司的悲剧	能不是岩井俊二最好的
21006203_4050	21006234_5095	21006250_1533	21006250_1845	21006265_1792	21006281_2303	21006296_1703	21006312_4155	21006343_3800	21006375_7805	21006390_2272	21006390_3509
83966.jpg	00098.jpg	094075.jpg	095682.jpg	10039.jpg	918403.jpg	314151.jpg	433018.jpg	857989.jpg	38992.jpg	115185.jpg	329702.jpg
用作物(其外类类)	题是连路。大量店铺关	副枪又列, 天下型破损	严风景盛,可是地方政	万里博士学历的本届华	是校被忠人认同校报忠	文事項: 澳王臣服于汉	你儿床上,后被捕叛变	默默无闻。有时比物质	藏品粮本说没有什么"	4] VisiblePa	理者能够对其效出21
21006406_1829	21006406_3525	21006421_8135	21006437_3618	21006500_3744	21006515_3581	21006515_3626	21006531_6387	21006578_2180	21006578_8753	21006578_3628	21006609_5406
800497.jpg	851529.jpg	63971.jpg	528285.jpg	686650.jpg	696593.jpg	198143.jpg	26560.jpg	57944.jpg	07433.jpg	674462.jpg	696.jpg

Figure 9: Part of the Training Data

2 Technical Guidance

This chapter aims to direct you to build a CRNN model from data management to image inference. The contents of this chapter is carefully designed to make sure that even new beginners in python can follow

2.1 Data Management

More than 3.6 million images are involved in the process of model training. Therefore, some special methods are required to manage data efficiently. The whole data set is uploaded at here, and the download key is **thxb**. The data on baiducloud can be used to substitute the original **data**/ directory of our project.

(1) Unpacking Data: All images are saved in 7z format as followed:

<pre>part_1.7z</pre>	2020/6/7 8:40	360压缩 7Z 文件	648,667 KB
<pre>part_2.7z</pre>	2020/6/7 8:48	360压缩 7Z 文件	648,293 KB
<pre>part_3.7z</pre>	2020/6/7 8:51	360压缩 7Z 文件	648,094 KB
<pre>part_4.7z</pre>	2020/6/7 8:55	360压缩 7Z 文件	648,062 KB
art_5.7z	2020/6/7 8:59	360压缩 7Z 文件	648,607 KB
art_6.7z	2020/6/7 9:04	360压缩 7Z 文件	648,540 KB
<pre>part_7.7z</pre>	2020/6/7 11:16	360压缩 7Z 文件	648,471 KB
<pre>part_8.7z</pre>	2020/6/7 8:44	360压缩 7Z 文件	648,079 KB
<pre>part_9.7z</pre>	2020/6/7 11:22	360压缩 7Z 文件	649,339 KB
<pre>part_10.7z</pre>	2020/6/7 11:26	360压缩 7Z 文件	713,227 KB

Figure 10: Training Data

Then, you should execute the order line **python unpacking_data.py**, and your program would execute 7z commands automatically to extract data. It may takes hours to finish this process. The whole training data is divided into 101 batches after finishing unpacking.

2020/6/6 18:36	文件夹
2020/5/5/40 02	
2020/6/6 19:03	文件夹
2020/6/6 19:30	文件夹
2020/6/6 19:58	文件夹
2020/6/6 20:25	文件夹
2020/6/6 20:55	文件夹
2020/6/6 21:24	文件夹
2020/6/6 21:52	文件夹
2020/6/6 22:21	文件夹
2020/6/6 22:42	文件夹
	2020/6/6 19:58 2020/6/6 20:25 2020/6/6 20:55 2020/6/6 21:24 2020/6/6 21:52 2020/6/6 22:21

Figure 11: Training Data

If we gather the 3.6 million images into one directory, the data loading speed of computer memory would be extremely low. Therefore, we have to divide the whole data set into 101 batches.

(2) Loading Data: The core data loading module is at here. This class inherits from tf.keras.utils.Sequence. The ___len___ method returns the total number of steps. The ___getitem___ method returns a batch of

data. Users only need to specify these 2 methods as **tf.keras.utils.Sequence** has incorporated other fundamental methods.

The following code is critical to our training process. It changes the background color of given text. 255 stands for white while 0 represents black. The ideal model should recognize both white text and black text. If the following code isn't involved, the train loss would decrease to less than 0.10 quickly. After the following code is incorporated in our program, the train loss stabilizes at 0.25. As a result, change the background color of images randomly may increase the robustness of our model.

```
for name in name_list:
   img = cv2.imread(name)
   if np.random.randint(0, 100) > 50:
        img = 255-img
```

Another important point is the use of **data generator**. The corresponding code is at here. In comparing with fitting after loading the data into memory, fit directly on data generator would double the training speed. **tensorflow** is able to synchronize reading data and training model effectively.

2.2 Net Structure

The whole structure of CRNN is at here. The input shape and output shape of every single module are specified in the annotations. The train model is as followed. There are four inputs and one output.

```
self.train_core = Model([vgg_input, labels, input_length, label_length], loss_out)
```

vgg_input: Gray-scale images whose shape is (batch_size,32,280,1).

labels: The ground-truth labels whose shape is (batch size,10).

input length: The input of CTC-loss function. Its shape is (batch size,1).

label_length: The input of CTC-loss function. Its shape is (batch_size,1).

loss_out: Return of CTC-loss function.

The details of CTC-loss would be demonstrated in chapter 3. There are 2 critical aspects concerning network structure: 1. The height of input image ought to be restricted to 32 pixels. 2. The width of input image is flexible in both training and inferring process. However, in order to train in batches, we set all training images and labels to the same shape.

The most innovative point of CRNN is at here. The pooling window of (2,1) is able to capture text features effectively.

```
12 = layers.MaxPooling2D(pool_size=(2,1), name='max3')(12)
```

CRNN.predict_core is used for further inference on new images, and **CRNN.train_core** is used for fitting on training data. **_load_weights** method demonstrates the relationship between these 2 **cores**.

```
def _load_weights(self,path):
    self.train_core.load_weights(path)
    output = self.train_core.get_layer("blstm2_out").output
    img_input = self.train_core.input[0]
    self.predict_core = Model(img_input,output)
```

3 The Principle of CRNN

CRNN is a neural network consisted of CNN and RNN. The parameter sharing mechanism of CNN and RNN enables CRNN to recognize images of flexible width. The most important aspect of CRNN is the use of **CTC-loss**. Classical OCR has to divide a word into single characters before text recognition. With the help of **CTC-loss**, CRNN don't need to conduct characters dividing task, which is much more difficult than single character recognition.

CTC-loss is a special loss different from classical loss like mean squared error. However, just like other loss function, CTC-loss is also a measurement of the difference between y_true and y_pred. As for our OCR task, the definitions of y_true and y_pred are demonstrated below.

y_true: An array of 10 integers. Each integer represents a certain Chinese or English letter. For instance, 20455828_2605100732.jpg corresponds to [263,82,29,56,35,435,890,293,126,129]. Figure 12 is the Chinese text represented by the integer array.



Figure 12: 20455828_2605100732.jpg

y_pred: A tensor whose shape is (69,5991). Dimension one(69) stands for different positions in horizontal direction(width). Dimension two(5991) is the output of softmax layer, stands for the probability of 5991 characters on 69 different positions.

CTC-loss: For any given **y_pred**, we can decode **y_pred** to get the final prediction. For instance, -s-t-aatte and -s-tta-tte would both lead to state. We assume state as the ground truth, and multiple **y_pred** can be decoded as state. Since **y_pred** is the probability of different letters, we are able to calculate the following probability by greedy algorithm.

$$P_{ctc} = \sum_{Decode(y_pred) = = y_true} P(y_pred)$$

CTC-loss is defined as followed:

$$CTCloss = -log(P_{ctc})$$

During the decode process, the blank letter - is abandoned. Therefore, CRNN is not able to recognize the space between different words naturally.

4 Prospect

The training data of our project is mainly consisted of Chinese rather than English. Therefore, the CRNN model performs better on Chinese. More English images should be incorporated into our training data to make the model more robust.

In addition, as the structure of English letter is very different from Chinese letter, it's reasonable to train multiple models to deal with various languages.