Text skew correction

Amir Hossein Bagheri

Introduction

It happens a lot in ORC projects that the image isn't captured correctly or it is rotated manually and we need to skew it correctly and rotate so that the lines become horizontal. In this task, we are asked to correct the image containing textual data and rotate it correctly.

Challenges

- Provide a dataset for the model to train.
- Design proper model.

Dataset

For providing training and test Dataset we use two methods.

First, we collect some pictures from the internet and also I myself took some pictures.

Then we resize the image to 512*512 format and also we use augmentation methods to increase the size of the dataset we rotate each picture with 32 different angles. The data set is provided in the following link data_set

Code for the process of augmentation and saving is in **data_generation.py**

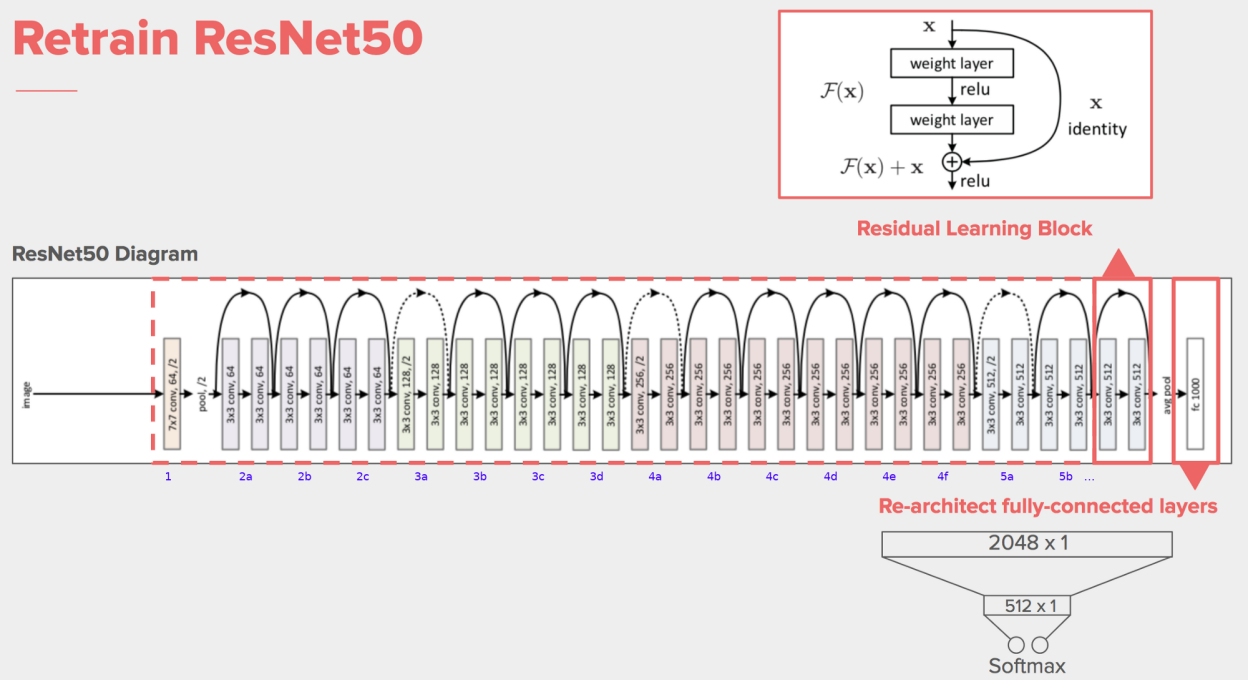
Model

We design a model that recognizes the angel that image's been rotated or captured badly.

So we can rotate the image with the opposite angel and have the corrected image.

With this definition of the problem out problem becomes the **regression** type problem.

The idea is Transfer Learning. Using ResNet50 and removing the last layer of FC we replace the last layer with FC with only one output_features with bounded Relu as activation (bounded between 0 and 359)



Instead of Softmax and 1000-FC output, we have FC with only one output, and instead of softmax we use bounded Relu activation.

We chose MSE loss for training. I tried logarithmic loss but I didn't get a good result, i think it's because logarithmic loss doesn't create a good gradient for training but MSE create a greater gradient for training.

Mode summary is like below

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
resnet50 (Functional)        (None, 16, 16, 2048)      23587712
_________________________________________________________________
flatten_2 (Flatten)          (None, 524288)            0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 524289
_________________________________________________________________
activation_2 (Activation)    (None, 1)                 0
=================================================================
Total params: 24,112,001
Trainable params: 524,289
Non-trainable params: 23,587,712
_________________________________________________________________
```

Train model

We only train the last FC and freeze the other layer in pre-trained ResNet with an ADAM optimizer.

With 20 epoch and a validation set of 10 percent.

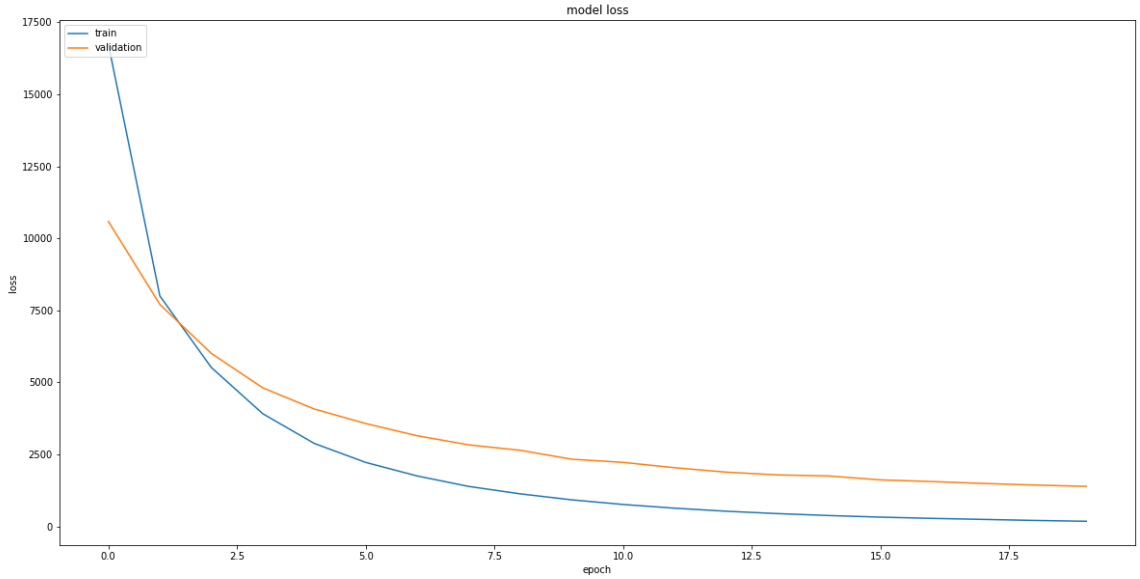
Python Code

Colab

Results

the model is trained with an ADAM optimizer with a learning rate of 10e-4.

The results plot for loss is below



As you see the learning rate is chosen properly. So we have a smooth decrease in loss.

The other thing worth mentioning is that with more 20 epochs the model would be overfitted with train data.