Technical Report

Backbone

Based on CVPR 2014 paper, the network could easily reach a high performance with 8 hidden layers (conv+bn+relu+maxpool+dropout) and 2 fully connected layers. With a pre-trained model, the test accuracy could reach more than 86%. The drawback of the proposed network is too many trainable parameters, twice as ResNeXt50! To improve the network, we can use ResNet, MobileNet or ResNeXt instead. There are more augmentation methodologies that we can apply, from auto augment to Cutout/Mosaic, etc. We can also switch to YOLO net. The backbone of a regular YoloV4 network is CSPDarkNet-53. It has five downsampling (pooling) layers, meaning the stride would be 32. If the resolution of input was approx. 200x100, there will be not enough feature information (low resolution) in the last layer. A potential solution should be to omit the last downsampling layer and train from scratch.

Global Non Maximum Suppression

In terms of YOLO detection, in addition to ordinary NMS, we found that a digit may be recognized as several different numbers and for a two-digit street number, there is possibility to be detected as three-digit one. In this case, a global non maximum suppression method was applied. But before implementation, think of a scenario where there is noise on the image. The noise is visually separable but it may look like a digit and may be detected as a digit. Since it often happens at the border of the image, we implemented a algorithm which looks like below:

Digit Detection Enhancement
if minus representation in coordinate: mark it as a warning and record its area and index
if warning in detections: calculate other detections' average area and compare to warning
if warning area is too small: recognize it as noise and remove it from detections

Voters

For this challenge, I tried several methods, including YoloV4-tiny with (label smoothing loss, mosaic augmentation, cosine learning scheduler) and/or (diou loss, mish activation), CRNN baseline, ResNeXt baseline, etc. For multiple model prediction, I designed two voting discriminators. First one integrates Yolo detection, CRNN and ResNeXt. The following are the details:

Voting algorithm
if r_1 and r_2: choose the longer string if both lengths not equal, otherwise randomly pick
elif not r_1 or r_2: choose the one with valid prediction
else: choose r_3
if r_3.length > cur.length: if it's just a shift (like 1159 and 159), ignore the inconsistency, otherwise, follow the longer one
if $r_3 \bigcap cur$ is empty, concatenate them

Second one is a soft voter, integrating Yolo detection and deep neural network. The testing accuracy of a single model is approx 85%. In Yolo detection, you can automatically get the likelihood of each predicted digit. In customized deep neural network, you can transform your raw outputs through Softmax activation. The trick of the voter is when you compare digit-wise possibilities, you can determine if the two detections are left-aligned or right-aligned. When two predictions are not equal in length, the principle is to always choose the longer one.