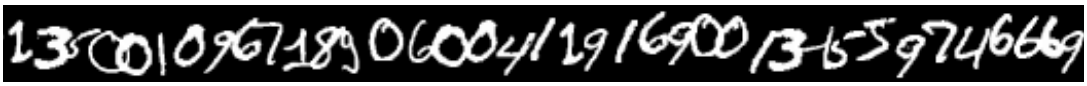
11-785 Fall 2017
Homework 03

Introduction

This homework will require you to use pyTorch, if you have not familiarized yourself with it yet start early.

You may have noticed that when you write things down you do not typically write only one character in isolation. You also probably also do not put it in a box centered on the mean location of you pencil/pen mark. With this in mind in this homework we will be revisiting image recognition in the form of sequences of digits for a more realistic scenario.

The data will consist of sequences of 100 digits from the MNIST dataset padded vertically with 4 pixels then randomly shifted vertically somewhere from 4 pixels down to 4 pixels up. Then S consisting of a total of 200 1-pixels spaces is spread out between the spaces between each two characters and overlaps O are chosen randomly from the range 10 to 18. Each digit at position I in the sequence with the exception of the first is placed overlapping the previous digit by O[i] – S[i] pixels. A portion of a datum is displayed below additionally the generator includes code for outputting image files.



This results in sequences of digits, which do not have well defined bounding boxes or locations. Note this procedure will produce images of the same size with the same number of digits. This presents several problems namely not having a well-defined mapping between portions of the image and labels.

Note you are given the data generator and the code, which will be used to evaluate your model. You can feel free to generate as much additional data as you wish however keep in mind the data yours will be tested on will be of the form which is described above and which the generator is initialized to produce (Only differing in using a different random seed).

You will be graded 50% on a correct implementation of CTC and 50% on your results on the dataset.

Part 1.

Implement a CTC cost function in the structure given in the starter code. Do not use autograd variables in this function. You can find out more about implementing a torch.autograd.Function here (http://pytorch.org/tutorials/beginner/examples_autograd/two_layer_net_custom_function.html).

Information on CTC can be found in the RNN recitation slides (http://deeplearning.cs.cmu.edu/slides/RNNrecitation.pdf) and in the CTC paper (ftp://ftp.idsia.ch/pub/juergen/icml2006.pdf).

Part 2.

Create a network for the dataset. The lower layers of your network should be convolutional taking slices, which are complete vertically, and no more than 10 pixels horizontally (required). These lower levels should output features to one or more recurrent layers. These recurrent layers should output to and MLP (optional) which will produce the final output for that time step. This network must be trained with CTC so remember to have 11 possible outputs (one for blank). The rest of the network architecture is up to you.

Make sure the model submitted is of the form specified in the starter code and is saved to TrainedModel.pt