Report for the home assignment: HAM10000 classification

Jieqiang Wei

10-05-2019

Dear ml-team at Peltarion,

First of all, thank you for giving me the chance to work on this problem. I have to say that this is my first time to use pytorch and Tensorflow has been my major force. Yet certainly it has been fun for me to use this library.

Here is the list of what I have done with the code and what I think is necessary for further explore

- The $\_\_init\_\_.py$ file is added to each subfolder of the project.
- The method $create\_train\_val\_split()$ has been modified. The original version is not good because the training data set is composed almost by the samples from just one class. The same thing holds for the test data set. In the current form, I guarantee the training dataset is composed by the samples from all the classes according to the distribution of the whole dataset. In other words, the training data is sampled according to the ratio of each class to the whole dataset. Next, the test dataset is sample in the same manner and excludes the training dataset.
- The attribute $class_weights$ are modified as the distribution of the classes in the whole dataset, i.e., ratios of the classes.
- Two $ReLU$ layers and one $Softmax$ layer are added to the CNN, since these are commonly used and the performance is better.
- The dataset we have now is large, so $train_fraction$ and $val_fraction$ are changed to a larger value 0.02. The loss is lower in general, but the running time is longer.
- The output dimension of the first linear layer and output of the second linear layer are changed to 128 to enhance the model performance.
- Since this is the first time I use pytorch, and the some figures are missed in my dataset preparation, I spent quite some time debug the code. The debugging trace is kept in the code $dataset.py$.
- I feel the metric used in the code, i.e., accuracy, is a good option.
- According to the loss plot, the epoch should stop around 2 in general. After that, the model overfits the training data, and spoils the performance on the test dataset.

- I still want to perform gridsearch over the parameters of the model, like $train_fraction$, input dimension in the CNN, etc. But the time is not enough.

Looking forward to your comments and suggestions. Have a nice weekend!

Best,
Jieqiang