A Reprot for Deep Learning Lab Exercise 4

Ziyi Guo, MSc Data Science

zg2u21@sonton.ac.uk

1 Wider MLPs on Mnist

In the experiment, a series of MLP models of a single layer with various hidden units in are implemented, and are trained and tested on the MNIST dataset. Experiments are carried out to explore the overfitting problem of on the MNIST dataset based on the code below.

```
class BaselineModel(nn.Module):    # Define Baseline Model
    def __init__(self, input_size, hidden_size, num_classes):
        super(BaselineModel, self).__init__()
        self.fc1 = nn.Linear(input_size, hidden_size)
        self.fc2 = nn.Linear(hidden_size, num_classes)
    def forward(self, x):
        out = self.fc1(x)
        out = F.relu(out)
        out = self.fc2(out)
        if not self.training:
            out = F.softmax(out, dim=1)
        return out
model = BaselineModel(784, 784, 10) # Build the Model
@callbacks.on_end_epoch
def store(state):
    train_loss[state[torchbearer.state.EPOCH]]=state[torchbearer.state.METRICS]['loss']
    test_loss[state[torchbearer.state.EPOCH]]=state[torchbearer.state.METRICS]['val_loss']
    train_acc[state[torchbearer.state.EPOCH]]=state[torchbearer.state.METRICS]['acc']
    test_acc[state[torchbearer.state.EPOCH]]=state[torchbearer.state.METRICS]['val_acc']

loss_function = nn.CrossEntropyLoss()  # Define Loss Function
optimiser = optim.Adam(model.parameters())  # Define Optimiser
# Create Trial Object
trial = torchbearer.Trial(model, optimiser, loss_function, metrics=['loss','accuracy'], callbacks=[store])
trial.with_generators(train_generator=trainloader,val_generator=testloader,test_generator=testloader).cuda()
trial.run(epochs=128)
results = trial.evaluate(data_key=torchbearer.TEST_DATA)model = BaselineModel(784, 784, 10) # Build the Model
```

To begin with, MLP models with few hidden units are built and tested in the first several experiments. The loss and accuracy of both training and test set against epoch are plotted in the figures as below.

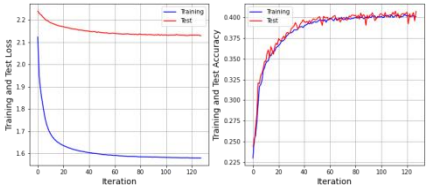

Fig 1.Training and Test Loss and Accuracy (Hidden Units=1)

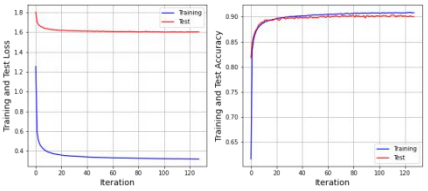

Fig 2.Training and Test Loss and Accuracy (Hidden Units=5)

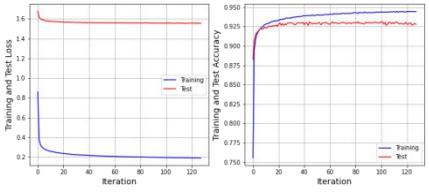

Fig 3.Training and Test Loss and Accuracy (Hidden Units=10)

It can be observed that the trained models with hidden units of 1, 5, and 10 have gradually better training and test accuracy, but generally perform poor on the classification task on MNIST dataset. More complex models are therefore tested as below.

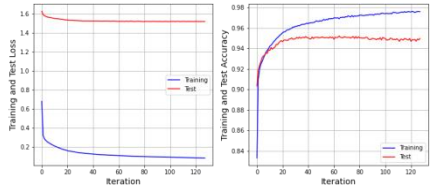

Fig 4.Training and Test Loss and Accuracy (Hidden Units=15)

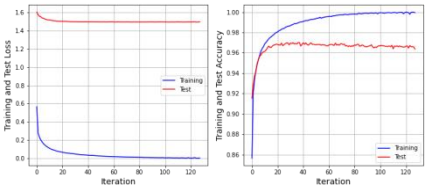

Fig 5.Training and Test Loss and Accuracy (Hidden Units=30)

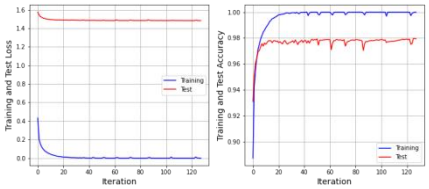

Fig 6.Training and Test Loss and Accuracy (Hidden Units=100)

It can be observed that the above models gradually have higher training accuracy and learn better but there appears comparatively greater gap between the accuracy curves in the first two figures, while this circumstance improves with the increase hidden units. Particularly worth mentioning is that models with more than 30 hidden layers get training accuracy around 1 on the task.

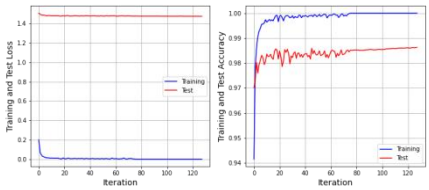

Fig 7.Training and Test Loss and Accuracy (Hidden Units=5000)

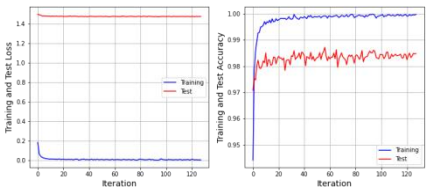

Fig 8.Training and Test Loss and Accuracy (Hidden Units=12,500)

Fig 9.Training and Test Loss and Accuracy (Hidden Units=250,000)

Further experiments build extremely more complex models with 5000, 125,000, and 250,000 hidden units. It can be observed that the training accuracy is flapping within 1 while the test accuracy is improving but has never reach 1. Limited by available GPU memory, experiments on MLP models with more hidden units are not carried out.

To explore the problem of overfitting, it is worth mentioning that overfitting happens when the validation accuracy is much lower than training accuracy based on a same model and set of parameters, or the validation accuracy is generally deceasing against epoch. And according to the current results of the experiments, no such phenomena are observed as above thus inferring that the model has not been overfitting probably due to the linearly seperability of the MNIST dataset. Further in conclusion, the MLP models are performing better with more hidden units and the divergency of training and test accuracy of the models with hidden units of [15,30] is possibly due to the unknown features in test set that are not learned in the training set. Finally, the MLP model with 250,000 hidden units hasn't completely learned the training dataset to generalize to the test set.