**Assignment 6**

Summary:

1/ SGD and LambdaLR

2/ Adam and LambdaLR

3/ Adam and MultiStepLR

4/ Adam and OneCycleLR

5/ AdamW and LambdaLR

6/ AdamW and OneCycleLR

7/ AdamW and MultiStepLR

In summary, LambdaLR works well for all of the three optimizers. However, MultiStepLR and OneCycleLR doesn’ work very well. It is a bit tricky to get an optimal learning rate from the eye ball test.

1/

Using the original optimizer and learning rate scheduling in the code, we got the following plots:

The loss decreases between 1.3k to 2k rounds of training, and correspondingly, the LR range is from 2.2e-3 to 0.1.

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

2/

I changed the optimizer to Adam, and keep the learning rate scheduling unchanged:

The code is as below:

The charts are as below:

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

The loss decreases from 620 to 1.32k. The learning rate range is from 3.93e-5 to 2.3e-3.

Next, I change the learning rate to 2.3e-3, and get the results as below:

The loss of test converges well.

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

3/

Next, I change the learning scheduling to MultiStepLR. The code is as below:

optimizer = torch.optim.Adam(model.parameters(), lr=2.3e-3, weight\_decay=5e-4)

lr\_milestones = list(range(50,2000,50))

lr\_scheduler = torch.optim.lr\_scheduler.MultiStepLR(optimizer, lr\_milestones, gamma=0.1, last\_epoch=-1)

The results are as below:

There is no obvious downward trend in loss/train. It fluctuates between 0.3 to 0.45.

The learning rate plunges first and keeps at a very low level.

Chart, box and whisker chart

Description automatically generated

I suspect that the fluctuation of Loss/train is caused by the high frequency of learning rate updating. Therefore, I adjusted the milestones parameter in the optimizer and have the following code:

optimizer = torch.optim.Adam(model.parameters(), lr=2.3e-3, weight\_decay=5e-4)

lr\_milestones = list(range(50,2000,200))

lr\_scheduler = torch.optim.lr\_scheduler.MultiStepLR(optimizer, lr\_milestones, gamma=0.1, last\_epoch=-1)

Then I got the following results:

Though I cannot detect the learning rate range from Loss/train plot, it seems that the currently used learning rate, 2.3e-3, is good, making Loss/test converge.

Chart, line chart

Description automatically generated

4/ Next, still using Adam optimizer, I change the scheduling to OneCycleLR.

The code is as below:

optimizer = torch.optim.Adam(model.parameters(), lr=2.3e-3, weight\_decay=5e-4)

lr\_scheduler = torch.optim.lr\_scheduler.OneCycleLR(optimizer, max\_lr=0.1, steps\_per\_epoch=len(train\_dataloader), epochs=30)

The loss decreases between 0 to 600 rounds. The learning rate range is from 4e-3 to 8.7e-3.

Chart, line chart

Description automatically generated

Next, I change the learning rate to 8e-3 and re-train the model:

optimizer = torch.optim.Adam(model.parameters(), lr=8e-3, weight\_decay=5e-4)

lr\_scheduler = torch.optim.lr\_scheduler.OneCycleLR(optimizer, max\_lr=0.1, steps\_per\_epoch=len(train\_dataloader), epochs=30)

However, we still have exploding loss during the test.

Chart, line chart

Description automatically generated

5/

Then I try AdamW optimizer and LambdaLR learning rate scheduler:

optimizer = torch.optim.AdamW(model.parameters(), lr=0.1, weight\_decay=5e-4)

lr\_scheduler = torch.optim.lr\_scheduler.LambdaLR(optimizer, lrs)

The loss decreases from 610 to 1.26k. The learning rate ranges from 4e-5 to 1.5e-3.

Chart, line chart

Description automatically generated

Set the learning rate to 1.5e-3 and retrain the model as below. It converges.

optimizer = torch.optim.AdamW(model.parameters(), lr=1.5e-3, weight\_decay=5e-4)

lr\_scheduler = torch.optim.lr\_scheduler.LambdaLR(optimizer, lrs)

Chart, line chart

Description automatically generated

6/

Then I change the scheduler to OneCycleLR:

optimizer = torch.optim.AdamW(model.parameters(), lr=1.5e-3, weight\_decay=5e-4)

lr\_scheduler = torch.optim.lr\_scheduler.OneCycleLR(optimizer, max\_lr=0.1, steps\_per\_epoch=len(train\_dataloader

), epochs=30)

Chart, line chart

Description automatically generated

According to the plots, I change the learning rate to 1.5e-3, and get:

optimizer = torch.optim.AdamW(model.parameters(), lr=1.5e-3, weight\_decay=5e-4)

lr\_scheduler = torch.optim.lr\_scheduler.OneCycleLR(optimizer, max\_lr=0.5, steps\_per\_epoch=len(train\_dataloader), epochs=10)

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

The loss decreases between 0 to 40. The corresponding learning rate is about 0.02. However, after using 0.02 and retrain, we still have exploding loss during the test.

Chart, line chart

Description automatically generated

7/ Next, I use MultiStepLR as the scheduler:

optimizer = torch.optim.AdamW(model.parameters(), lr=0.02, weight\_decay=5e-4)

lr\_milestones = list(range(50,500,50))

lr\_scheduler = torch.optim.lr\_scheduler.MultiStepLR(optimizer, lr\_milestones, gamma=0.1, last\_epoch=-1)

Chart

Description automatically generated