

修改方式

1. 把原先`codes/cnn/main.py`替换成新作业的`codes/cnn/main.py`
2. 其他有修改的文件：
 1. `code_analyze/_code/mlp/main.py`: 该文件修改影响实现
 2. `code_analyze/_code/cnn/main.py`: 该文件修改影响实现
 3. `codes/cnn/model.py`: 该文件删去了44行的一个空格

文件细节的修改，可以查看下面的changelog

Git changelog

```
1 diff --git a/code_analyze/_codes/cnn/model.py b/code_analyze/_codes/cnn/model.py
2 index 254fd46..b77c6a9 100644
3 --- a/code_analyze/_codes/cnn/model.py
4 +++ b/code_analyze/_codes/cnn/model.py
5 @@ -4,10 +4,10 @@ import torch
6     from torch import nn
7     from torch.nn import init
8     from torch.nn.parameter import Parameter
9 -class BatchNorm2d(nn.Module):
10 +class BatchNorm1d(nn.Module):
11     # TODO START
12     def __init__(self, num_features):
13 -    super(BatchNorm2d, self).__init__()
14 +    super(BatchNorm1d, self).__init__()
15     self.num_features = num_features
16
17     # Parameters
18 diff --git a/code_analyze/_codes/mlp/main.py b/code_analyze/_codes/mlp/main.py
19 index 2ec7403..d4606fb 100644
20 --- a/code_analyze/_codes/mlp/main.py
21 +++ b/code_analyze/_codes/mlp/main.py
22 @@ -105,7 +105,7 @@ if __name__ == '__main__':
23     X_train, X_test, y_train, y_test = load_cifar_2d(args.data_dir)
24     X_val, y_val = X_train[40000:], y_train[40000:]
25     X_train, y_train = X_train[:40000], y_train[:40000]
26 -    mlp_model = Model(drop_rate=args.drop_rate)
27 +    mlp_model = Model(drop_rate=drop_rate)
28     mlp_model.to(device)
29     print(mlp_model)
30     optimizer = optim.Adam(mlp_model.parameters(), lr=args.learning_rate)
31 diff --git a/codes/cnn/main.py b/codes/cnn/main.py
32 index 878d601..95d98c7 100644
33 --- a/codes/cnn/main.py
34 +++ b/codes/cnn/main.py
35 @@ -10,7 +10,7 @@ import torch.nn as nn
36     import torch.optim as optim
37
38     from model import Model
39 -from load_data import load_cifar_4d
40 +from load_data import load_cifar_2d
41
42     parser = argparse.ArgumentParser()
43
44 @@ -102,35 +102,35 @@ if __name__ == '__main__':
45     if not os.path.exists(args.train_dir):
46         os.mkdir(args.train_dir)
47     if args.is_train:
48 -    X_train, X_test, y_train, y_test = load_cifar_4d(args.data_dir)
49 +    X_train, X_test, y_train, y_test = load_cifar_2d(args.data_dir)
50     X_val, y_val = X_train[40000:], y_train[40000:]
51     X_train, y_train = X_train[:40000], y_train[:40000]
52 -    cnn_model = Model(drop_rate=args.drop_rate)
53 -    cnn_model.to(device)
54 -    print(cnn_model)
55 -    optimizer = optim.Adam(cnn_model.parameters(), lr=args.learning_rate)
56 +    mlp_model = Model(drop_rate=drop_rate)
57 +    mlp_model.to(device)
58 +    print(mlp_model)
```

```

59 + optimizer = optim.Adam(mlp_model.parameters(), lr=args.learning_rate)
60
61 # model_path = os.path.join(args.train_dir, 'checkpoint_%d.pth.tar' %
args.inference_version)
62 # if os.path.exists(model_path):
63 - #     cnn_model = torch.load(model_path)
64 + #     mlp_model = torch.load(model_path)
65
66 pre_losses = [1e18] * 3
67 best_val_acc = 0.0
68 for epoch in range(1, args.num_epochs+1):
69     start_time = time.time()
70 -     train_acc, train_loss = train_epoch(cnn_model, X_train, y_train, optimizer)
71 +     train_acc, train_loss = train_epoch(mlp_model, X_train, y_train, optimizer)
72     X_train, y_train = shuffle(X_train, y_train, 1)
73
74 -     val_acc, val_loss = valid_epoch(cnn_model, X_val, y_val)
75 +     val_acc, val_loss = valid_epoch(mlp_model, X_val, y_val)
76
77     if val_acc >= best_val_acc:
78         best_val_acc = val_acc
79         best_epoch = epoch
80 -     test_acc, test_loss = valid_epoch(cnn_model, X_test, y_test)
81 -     with open(os.path.join(args.train_dir, 'checkpoint_{}.pth.tar'.format(epoch)),
'wb') as fout:
82 -         torch.save(cnn_model, fout)
83 -         with open(os.path.join(args.train_dir, 'checkpoint_0.pth.tar'), 'wb') as fout:
84 -             torch.save(cnn_model, fout)
85 +     test_acc, test_loss = valid_epoch(mlp_model, X_test, y_test)
86 +     # with open(os.path.join(args.train_dir, 'checkpoint_{}.pth.tar'.format(epoch)),
'wb') as fout:
87 +         # torch.save(mlp_model, fout)
88 +         # with open(os.path.join(args.train_dir, 'checkpoint_0.pth.tar'), 'wb') as fout:
89 +         # torch.save(mlp_model, fout)
90
91     epoch_time = time.time() - start_time
92     print("Epoch " + str(epoch) + " of " + str(args.num_epochs) + " took " +
str(epoch_time) + "s")
93 @@ -150,19 +150,19 @@ if __name__ == '__main__':
94     pre_losses = pre_losses[1:] + [train_loss]
95
96     else:
97 -     print("begin testing")
98 -     cnn_model = Model()
99 -     cnn_model.to(device)
100 +     mlp_model = Model()
101 +     mlp_model.to(device)
102     model_path = os.path.join(args.train_dir, 'checkpoint_%d.pth.tar' %
args.inference_version)
103     if os.path.exists(model_path):
104 -     cnn_model = torch.load(model_path)
105 +     mlp_model = torch.load(model_path)
106
107 -     X_train, X_test, y_train, y_test = load_cifar_4d(args.data_dir)
108 +     X_train, X_test, y_train, y_test = load_cifar_2d(args.data_dir)
109
110     count = 0
111     for i in range(len(X_test)):
112 -     test_image = X_test[i].reshape((1, 3, 32, 32))
113 -     result = inference(cnn_model, test_image)[0]
114 +     test_image = X_test[i].reshape((1, 3 * 32 * 32))
115 +     result = inference(mlp_model, test_image)[0]

```

```

116         if result == y_test[i]:
117             count += 1
118         print("test accuracy: {}".format(float(count) / len(X_test)))
119     +nt) / len(X_test)))
120 diff --git a/codes/cnn/model.py b/codes/cnn/model.py
121 index 205e442..5e10f39 100644
122 --- a/codes/cnn/model.py
123 +++ b/codes/cnn/model.py
124 @@ -44,7 +44,7 @@ class Model(nn.Module):
125     # TODO END
126     self.loss = nn.CrossEntropyLoss()
127
128 - def forward(self, x, y=None):
129 + def forward(self, x, y=None):
130     # TODO START
131     # the 10-class prediction output is named as "logits"
132     logits =
133

```

```

diff --git a/code_analyze/_codes/cnn/model.py b/code_analyze/_codes/cnn/model.py
index 254fd46..b77c6a9 100644
--- a/code_analyze/_codes/cnn/model.py
+++ b/code_analyze/_codes/cnn/model.py
@@ -4,10 +4,10 @@ import torch
from torch import nn
from torch.nn import init
from torch.nn.parameter import Parameter
- class BatchNorm2d(nn.Module):
+ class BatchNorm1d(nn.Module):
    # TODO START
    def __init__(self, num_features):
-         super(BatchNorm2d, self).__init__()
+         super(BatchNorm1d, self).__init__()
        self.num_features = num_features

    # Parameters
diff --git a/code_analyze/_codes/mlp/main.py b/code_analyze/_codes/mlp/main.py
index 2ec7403..d4606fb 100644
--- a/code_analyze/_codes/mlp/main.py
+++ b/code_analyze/_codes/mlp/main.py
@@ -105,7 +105,7 @@ if __name__ == '__main__':
    X_train, X_test, y_train, y_test = load_cifar_2d(args.data_dir)
    X_val, y_val = X_train[40000:], y_train[40000:]
    X_train, y_train = X_train[:40000], y_train[:40000]
-    mlp_model = Model(drop_rate=args.drop_rate)
+    mlp_model = Model(drop_rate=drop_rate)
    mlp_model.to(device)
    print(mlp_model)
    optimizer = optim.Adam(mlp_model.parameters(), lr=args.learning_rate)

```

```

diff --git a/codes/cnn/model.py b/codes/cnn/model.py
index 205e442..5e10f39 100644
--- a/codes/cnn/model.py
+++ b/codes/cnn/model.py
@@ -44,7 +44,7 @@ class Model(nn.Module):
    # TODO END
    self.loss = nn.CrossEntropyLoss()

-    def forward(self, x, y=None):
+    def forward(self, x, y=None):
        # TODO START
        # the 10-class prediction output is named as "logits"
        logits =

```

(END)