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

```
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/main.py b/codes/cnn/main.py
index 878d601..95d98c7 100644
--- a/codes/cnn/main.py
+++ b/codes/cnn/main.py
@@ -10,7 +10,7 @@ import torch.nn as nn
 import torch.optim as optim

 from model import Model
-from load_data import load_cifar_4d
+from load_data import load_cifar_2d

 parser = argparse.ArgumentParser()

@@ -102,35 +102,35 @@ if __name__ == '__main__':
   if not os.path.exists(args.train_dir):
     os.mkdir(args.train_dir)
   if args.is_train:
-    X_train, X_test, y_train, y_test = load_cifar_4d(args.data_dir)
+    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]
-    cnn_model = Model(drop_rate=args.drop_rate)
-    cnn_model.to(device)
-    print(cnn_model)
-    optimizer = optim.Adam(cnn_model.parameters(), lr=args.learning_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)

    # model_path = os.path.join(args.train_dir, 'checkpoint_%d.pth.tar' % args.inference_version)
    # if os.path.exists(model_path):
-   #   cnn_model = torch.load(model_path)
+   #   mlp_model = torch.load(model_path)

    pre_losses = [1e18] * 3
    best_val_acc = 0.0
    for epoch in range(1, args.num_epochs+1):
      start_time = time.time()
-     train_acc, train_loss = train_epoch(cnn_model, X_train, y_train, optimizer)
+     train_acc, train_loss = train_epoch(mlp_model, X_train, y_train, optimizer)
      X_train, y_train = shuffle(X_train, y_train, 1)

-     val_acc, val_loss = valid_epoch(cnn_model, X_val, y_val)
+     val_acc, val_loss = valid_epoch(mlp_model, X_val, y_val)

      if val_acc >= best_val_acc:
        best_val_acc = val_acc
        best_epoch = epoch
-       test_acc, test_loss = valid_epoch(cnn_model, X_test, y_test)
-       with open(os.path.join(args.train_dir, 'checkpoint_{}.pth.tar'.format(epoch)), 'wb') as fout:
-         torch.save(cnn_model, fout)
-       with open(os.path.join(args.train_dir, 'checkpoint_0.pth.tar'), 'wb') as fout:
-         torch.save(cnn_model, fout)
+       test_acc, test_loss = valid_epoch(mlp_model, X_test, y_test)
+       # with open(os.path.join(args.train_dir, 'checkpoint_{}.pth.tar'.format(epoch)), 'wb') as fout:
+       #   torch.save(mlp_model, fout)
+       # with open(os.path.join(args.train_dir, 'checkpoint_0.pth.tar'), 'wb') as fout:
+       #   torch.save(mlp_model, fout)

      epoch_time = time.time() - start_time
      print("Epoch " + str(epoch) + " of " + str(args.num_epochs) + " took " + str(epoch_time) + "s")
@@ -150,19 +150,19 @@ if __name__ == '__main__':
      pre_losses = pre_losses[1:] + [train_loss]

  else:
-   print("begin testing")
-   cnn_model = Model()
-   cnn_model.to(device)
+   mlp_model = Model()
+   mlp_model.to(device)
    model_path = os.path.join(args.train_dir, 'checkpoint_%d.pth.tar' % args.inference_version)
    if os.path.exists(model_path):
-     cnn_model = torch.load(model_path)
+     mlp_model = torch.load(model_path)

-   X_train, X_test, y_train, y_test = load_cifar_4d(args.data_dir)
+   X_train, X_test, y_train, y_test = load_cifar_2d(args.data_dir)

    count = 0
    for i in range(len(X_test)):
-     test_image = X_test[i].reshape((1, 3, 32, 32))
-     result = inference(cnn_model, test_image)[0]
+     test_image = X_test[i].reshape((1, 3 * 32 * 32))
+     result = inference(mlp_model, test_image)[0]
```

```
      if result == y_test[i]:
        count += 1
    print("test accuracy: {}".format(float(count) / len(X_test)))
+nt) / len(X_test)))
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 =

```

```
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)
```