Data Augmentation

May 23, 2023

[1]:
```
import matplotlib.pyplot as plt
import numpy as np
import torch
import torch.nn as nn
from torch.optim import Adam
from torch.utils.data import DataLoader
from torchvision.datasets import CIFAR10
from torchvision.models import resnet18, ResNet18_Weights
from torchvision.transforms import Compose, ToTensor, Lambda, Normalize
```

1 Task - 1

[2]:
```
def _create_batch(unbatched_data, unbatched_label, unbatched_test_data,␣
 ↪unbatched_test_label):
    unbatched_data = torch.split(unbatched_data, 100)
    unbatched_label = torch.split(unbatched_label, 100)
    unbatched_test_data = torch.split(unbatched_test_data, 100)
    unbatched_test_label = torch.split(unbatched_test_label, 100)
    return unbatched_data, unbatched_label, unbatched_test_data,␣
 ↪unbatched_test_label


def get_dataset():
    # dataset
    transform = Compose([
        ToTensor(),
        Normalize((0.4914, 0.4822, 0.4465), (0.247, 0.243, 0.261)),
    ])

    target_transform = Lambda(lambda y: torch.zeros(10, dtype=torch.float).
 ↪scatter_(0, torch.tensor(y), 1))

    train_dataset = CIFAR10('./data', train=True, download=True,␣
 ↪transform=transform, target_transform=target_transform)
    test_dataset = CIFAR10('./data', train=False, download=True,␣
 ↪transform=transform)
```

```
  train_loader = DataLoader(train_dataset, batch_size=len(train_dataset))
  test_loader = DataLoader(test_dataset, batch_size=len(test_dataset))

  train_data, train_label = next(iter(train_loader))
  test_data, test_label = next(iter(test_loader))

  print('train data: {}, train label: {}'.format(train_data.size(),␣
↪train_label.size()))
  print('test data: {}, test label: {}'.format(test_data.size(), test_label.
↪size()))

  sorted_train_label_arg = torch.argsort(torch.argmax(train_label, dim=1))
  sorted_train_label = train_label[sorted_train_label_arg]
  sorted_train_data = train_data[sorted_train_label_arg]

  train_data_sampled = []
  train_label_sampled = []
  for class_idx in range(10):
      class_idx = class_idx * 5000
      train_data_sampled.append(sorted_train_data[class_idx:(class_idx +␣
↪1000)])
      train_label_sampled.append(sorted_train_label[class_idx:(class_idx +␣
↪1000)])

  rand_idx = torch.randperm(10000)
  train_data_sampled = torch.concat(train_data_sampled, dim=0)[rand_idx]
  train_label_sampled = torch.concat(train_label_sampled, dim=0)[rand_idx]
  print(
      'train_data_sampled: {}, train_label_sampled: {}'.
↪format(train_data_sampled.size(), train_label_sampled.size()))

  train_data_sampled, train_label_sampled, test_data, test_label =␣
↪_create_batch(train_data_sampled,
                                                                    ␣
↪   train_label_sampled, test_data,
                                                                    ␣
↪   test_label)
  print('train_data_sampled: {}, train_label_sampled: {}'.
↪format(len(train_data_sampled), len(train_label_sampled)))
  print('train_data_sampled: {}, train_label_sampled: {}'.
↪format(train_data_sampled[0].size(),
                                                                ␣
↪train_label_sampled[0].size()))

  return train_data_sampled, train_label_sampled, test_data, test_label
```

[3]:
```
def train(train_data_sampled, train_label_sampled, test_data, test_label,␣
↪augmentation=None, num_epoch=100, lr=0.0001):
    model = resnet18(weights=ResNet18_Weights.DEFAULT)
    model.fc = nn.Linear(512, 10)
    device = 'cuda' if torch.cuda.is_available() else 'cpu'
    optimizer = Adam(model.parameters(), lr=lr)
    criterion = nn.MSELoss()
    model = model.to(device)

    running_loss = []
    running_acc = []
    running_train_acc = []
    loss = None

    for epoch in range(num_epoch):
        for idx, (data, label) in enumerate(zip(train_data_sampled,␣
↪train_label_sampled)):
            model.train()
            if augmentation is not None:
                data, label = augmentation(data, label)
            data, label = data.to(device), label.to(device)

            optimizer.zero_grad()
            preds = model(data)
            loss = criterion(preds, label)
            loss.backward()
            optimizer.step()
            running_loss.append(loss.item())

            # test
            # if (idx + 1) % 10 == 0:
        model.eval()
        tot_acc = torch.zeros(1).to(device)
        test_data_size = 0
        with torch.no_grad():
            for test_data_batch, test_label_batch in zip(test_data, test_label):
                test_data_batch, test_label_batch = test_data_batch.to(device),␣
↪test_label_batch.to(device)
                test_preds = model(test_data_batch)
                test_preds = torch.argmax(test_preds, dim=1)
                tot_acc = tot_acc + torch.count_nonzero((test_preds ==␣
↪test_label_batch).long())
                test_data_size += test_data_batch.size(0)
            running_acc.append(tot_acc.item() / test_data_size)

        tot_acc_train = torch.zeros(1).to(device)
        train_data_size = 0
```

```
        with torch.no_grad():
            for train_data_batch, train_label_batch in zip(train_data_sampled,␣
↪train_label_sampled):
                train_data_batch, train_label_batch = train_data_batch.
↪to(device), train_label_batch.to(device)
                train_preds = model(train_data_batch)
                train_preds = torch.argmax(train_preds, dim=1)
                ground_truth = torch.argmax(train_label_batch, dim=1)
                tot_acc_train = tot_acc_train + torch.
↪count_nonzero((train_preds == ground_truth).long())
                train_data_size += train_data_batch.size(0)
            running_train_acc.append(tot_acc_train.item() / train_data_size)
            # print(tot_acc)
            # print(test_data_size)
        print('epoch: {}, loss: {}, test acc: {}, train acc: {}'.format(epoch +␣
↪1, loss, tot_acc.item() / test_data_size, tot_acc_train.item() /␣
↪train_data_size))

    plt.figure(figsize=(10, 5))
    plt.plot(list(range(len(running_loss))), running_loss)
    plt.xlabel('iteration')
    plt.ylabel('loss')
    plt.title('loss vs. iteration')
    plt.show()

    plt.figure(figsize=(10, 5))
    plt.plot(list(range(len(running_acc))), running_acc)
    plt.plot(list(range(len(running_train_acc))), running_train_acc)
    plt.xlabel('iteration')
    plt.ylabel('acc')
    plt.title('acc vs. iteration')
    plt.legend(['test acc', 'train acc'], loc='lower right')
    plt.show()

    return running_loss, running_acc, running_train_acc
```

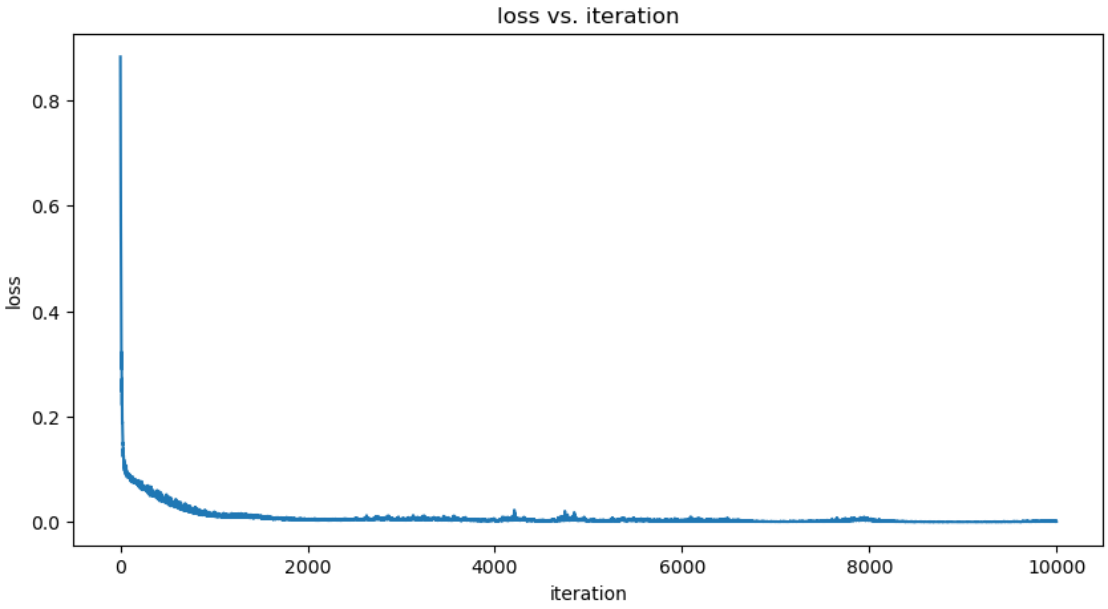
[5]:
```
train_X, train_y, test_X, test_y = get_dataset()
running_loss_without_aug, running_acc_without_aug,␣
↪running_train_acc_without_aug = train(train_X, train_y, test_X, test_y)
```

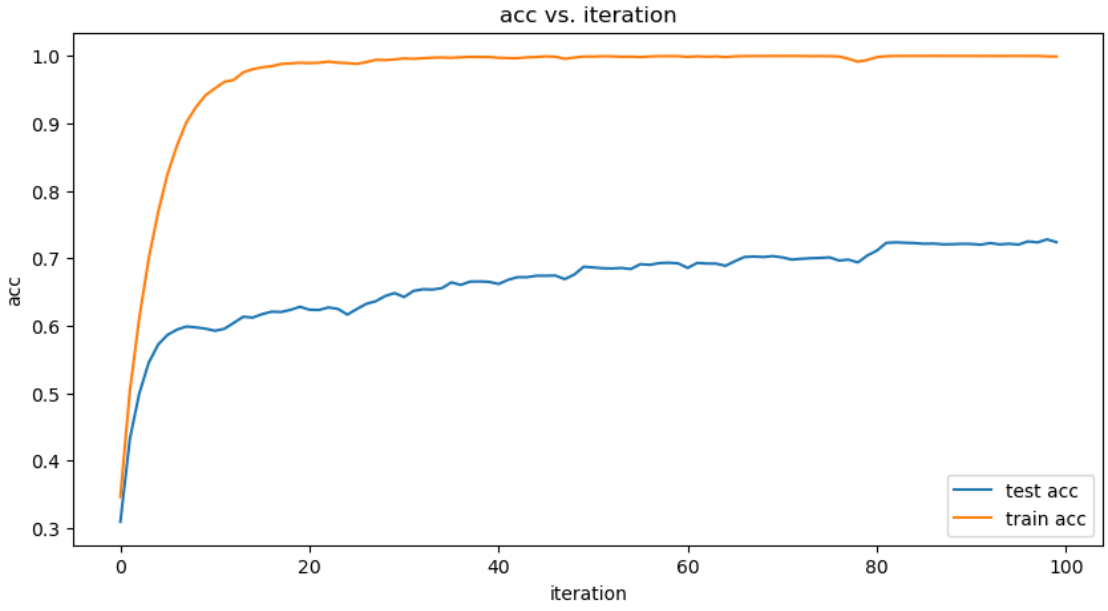
```
Files already downloaded and verified
Files already downloaded and verified
train data: torch.Size([50000, 3, 32, 32]), train label: torch.Size([50000, 10])
test data: torch.Size([10000, 3, 32, 32]), test label: torch.Size([10000])
train_data_sampled: torch.Size([10000, 3, 32, 32]), train_label_sampled:
torch.Size([10000, 10])
train_data_sampled: 100, train_label_sampled: 100
```

```
train_data_sampled: torch.Size([100, 3, 32, 32]), train_label_sampled:
torch.Size([100, 10])
epoch: 1, loss: 0.08269018679857254, test acc: 0.3095, train acc: 0.3464
epoch: 2, loss: 0.06839144229888916, test acc: 0.4322, train acc: 0.5029
epoch: 3, loss: 0.05558766424655914, test acc: 0.4997, train acc: 0.6115
epoch: 4, loss: 0.045817699283361435, test acc: 0.5453, train acc: 0.6998
epoch: 5, loss: 0.037445105612277985, test acc: 0.5722, train acc: 0.7686
epoch: 6, loss: 0.029781831428408623, test acc: 0.5866, train acc: 0.8254
epoch: 7, loss: 0.023174161091446877, test acc: 0.5944, train acc: 0.8676
epoch: 8, loss: 0.01800876297056675, test acc: 0.5989, train acc: 0.9021
epoch: 9, loss: 0.013292535208165646, test acc: 0.5975, train acc: 0.924
epoch: 10, loss: 0.011358753778040409, test acc: 0.5957, train acc: 0.9416
epoch: 11, loss: 0.009671524167060852, test acc: 0.5926, train acc: 0.9518
epoch: 12, loss: 0.009712215512990952, test acc: 0.5955, train acc: 0.9615
epoch: 13, loss: 0.010245075449347496, test acc: 0.6043, train acc: 0.9644
epoch: 14, loss: 0.00968923605978489, test acc: 0.6134, train acc: 0.9756
epoch: 15, loss: 0.007300151977688074, test acc: 0.6121, train acc: 0.9803
epoch: 16, loss: 0.00810390803962946, test acc: 0.6172, train acc: 0.983
epoch: 17, loss: 0.006043017841875553, test acc: 0.6209, train acc: 0.9847
epoch: 18, loss: 0.005448923911899328, test acc: 0.6204, train acc: 0.988
epoch: 19, loss: 0.004856435116380453, test acc: 0.6236, train acc: 0.9888
epoch: 20, loss: 0.004425158724188805, test acc: 0.6281, train acc: 0.9898
epoch: 21, loss: 0.003999474924057722, test acc: 0.6238, train acc: 0.9893
epoch: 22, loss: 0.004028269089758396, test acc: 0.6234, train acc: 0.9898
epoch: 23, loss: 0.004078140016645193, test acc: 0.6271, train acc: 0.9914
epoch: 24, loss: 0.004708396270871162, test acc: 0.6251, train acc: 0.9899
epoch: 25, loss: 0.004293549340218306, test acc: 0.6166, train acc: 0.9893
epoch: 26, loss: 0.00530210230499506, test acc: 0.6247, train acc: 0.9881
epoch: 27, loss: 0.00467159878462553, test acc: 0.6324, train acc: 0.9907
epoch: 28, loss: 0.0052609192207455635, test acc: 0.6363, train acc: 0.9942
epoch: 29, loss: 0.006438563577830791 5, test acc: 0.644, train acc: 0.9937
epoch: 30, loss: 0.005469070747494698, test acc: 0.6485, train acc: 0.9948
epoch: 31, loss: 0.004018892999738455, test acc: 0.6425, train acc: 0.9962
epoch: 32, loss: 0.003771614981815219, test acc: 0.6516, train acc: 0.9956
epoch: 33, loss: 0.003965006209909916, test acc: 0.6541, train acc: 0.9965
epoch: 34, loss: 0.0032552448101341724, test acc: 0.6537, train acc: 0.9973
epoch: 35, loss: 0.004925399553030729, test acc: 0.6557, train acc: 0.9976
epoch: 36, loss: 0.005569250788539648, test acc: 0.6642, train acc: 0.9971
epoch: 37, loss: 0.005269808229058981, test acc: 0.6606, train acc: 0.998
epoch: 38, loss: 0.00335780275054276, test acc: 0.6655, train acc: 0.9985
epoch: 39, loss: 0.002259480534121394, test acc: 0.6657, train acc: 0.9983
epoch: 40, loss: 0.005760362837463617, test acc: 0.6651, train acc: 0.9983
epoch: 41, loss: 0.004598166793584824, test acc: 0.6619, train acc: 0.9971
epoch: 42, loss: 0.005980531685054302, test acc: 0.668, train acc: 0.9966
epoch: 43, loss: 0.0050547304563224316, test acc: 0.672, train acc: 0.9965
epoch: 44, loss: 0.0028263807762414217, test acc: 0.672, train acc: 0.9978
epoch: 45, loss: 0.0018822288839146495, test acc: 0.6741, train acc: 0.9983
epoch: 46, loss: 0.0019843673799186945, test acc: 0.6741, train acc: 0.9992
```

epoch: 47, loss: 0.0029090377502143383, test acc: 0.6744, train acc: 0.9987
epoch: 48, loss: 0.003525268053635955, test acc: 0.6691, train acc: 0.9956
epoch: 49, loss: 0.0031137929763644934, test acc: 0.6758, train acc: 0.9975
epoch: 50, loss: 0.0017594328382983804, test acc: 0.6875, train acc: 0.999
epoch: 51, loss: 0.0014901439426466823, test acc: 0.6864, train acc: 0.9989
epoch: 52, loss: 0.0014038366498425603, test acc: 0.6853, train acc: 0.9994
epoch: 53, loss: 0.0016317104455083609, test acc: 0.6849, train acc: 0.9992
epoch: 54, loss: 0.0018221321515738964, test acc: 0.6856, train acc: 0.9987
epoch: 55, loss: 0.0024186044465750456, test acc: 0.6841, train acc: 0.9988
epoch: 56, loss: 0.0036178959999233484, test acc: 0.6913, train acc: 0.9982
epoch: 57, loss: 0.0023230859078466892, test acc: 0.6903, train acc: 0.9991
epoch: 58, loss: 0.001673982129432261, test acc: 0.6928, train acc: 0.9995
epoch: 59, loss: 0.0015983434859663248, test acc: 0.6934, train acc: 0.9996
epoch: 60, loss: 0.0029248297214508057, test acc: 0.6924, train acc: 0.9995
epoch: 61, loss: 0.004313305020332336, test acc: 0.6857, train acc: 0.9985
epoch: 62, loss: 0.0016589189181104302, test acc: 0.6931, train acc: 0.9993
epoch: 63, loss: 0.0015713373431935906, test acc: 0.6922, train acc: 0.9987
epoch: 64, loss: 0.001694574486464262, test acc: 0.6921, train acc: 0.9992
epoch: 65, loss: 0.0016527273692190647, test acc: 0.6889, train acc: 0.9982
epoch: 66, loss: 0.0014814800815656781, test acc: 0.6957, train acc: 0.9993
epoch: 67, loss: 0.0009156903252005577, test acc: 0.7019, train acc: 0.9996
epoch: 68, loss: 0.0009283950785174966, test acc: 0.7026, train acc: 0.9998
epoch: 69, loss: 0.000956649542786181, test acc: 0.7019, train acc: 0.9997
epoch: 70, loss: 0.0007068121922202408, test acc: 0.7032, train acc: 0.9999
epoch: 71, loss: 0.0006486732163466513, test acc: 0.7014, train acc: 0.9998
epoch: 72, loss: 0.0006420854479074478, test acc: 0.6981, train acc: 0.9999
epoch: 73, loss: 0.0010578504297882318, test acc: 0.6992, train acc: 0.9998
epoch: 74, loss: 0.000706540304236114, test acc: 0.7001, train acc: 0.9996
epoch: 75, loss: 0.0008729228866286576, test acc: 0.7005, train acc: 0.9997
epoch: 76, loss: 0.0009297134238295257, test acc: 0.7014, train acc: 0.9995
epoch: 77, loss: 0.0028363787569105625, test acc: 0.6967, train acc: 0.9991
epoch: 78, loss: 0.004476973786950111, test acc: 0.698, train acc: 0.9957
epoch: 79, loss: 0.0040770941413939, test acc: 0.6939, train acc: 0.9914
epoch: 80, loss: 0.0042197210714221, test acc: 0.7041, train acc: 0.9939
epoch: 81, loss: 0.0024831220507621765, test acc: 0.7113, train acc: 0.9981
epoch: 82, loss: 0.0018150366377085447, test acc: 0.7228, train acc: 0.9995
epoch: 83, loss: 0.0009056587587110698, test acc: 0.7236, train acc: 0.9998
epoch: 84, loss: 0.000496978871524334, test acc: 0.723, train acc: 0.9999
epoch: 85, loss: 0.0004217819659970701, test acc: 0.7225, train acc: 0.9999
epoch: 86, loss: 0.00035369620309211314, test acc: 0.7215, train acc: 0.9999
epoch: 87, loss: 0.0003027221537195146, test acc: 0.7218, train acc: 1.0
epoch: 88, loss: 0.0002794950851239264, test acc: 0.7207, train acc: 0.9999
epoch: 89, loss: 0.00024988423683680594, test acc: 0.7209, train acc: 0.9999
epoch: 90, loss: 0.00026774953585118055, test acc: 0.7214, train acc: 0.9999
epoch: 91, loss: 0.00028830175870098174, test acc: 0.7212, train acc: 0.9999
epoch: 92, loss: 0.0002599008730612695, test acc: 0.7202, train acc: 0.9998
epoch: 93, loss: 0.00020335218869149685, test acc: 0.7225, train acc: 0.9999
epoch: 94, loss: 0.00022070706472732127, test acc: 0.7206, train acc: 0.9997

```
epoch: 95, loss: 0.00025583113892935216, test acc: 0.7215, train acc: 0.9999
epoch: 96, loss: 0.0003169855335727334, test acc: 0.7204, train acc: 0.9999
epoch: 97, loss: 0.0007783197797834873, test acc: 0.725, train acc: 0.9998
epoch: 98, loss: 0.001185030909255147, test acc: 0.7236, train acc: 0.9999
epoch: 99, loss: 0.0010571167804300785, test acc: 0.728, train acc: 0.9992
epoch: 100, loss: 0.000805065908934921, test acc: 0.724, train acc: 0.999
```





The final results of the training without augmentation is the following: - test accuracy: 0.724 - train accuracy: 0.999 - loss: 0.0008

2 Task - 2

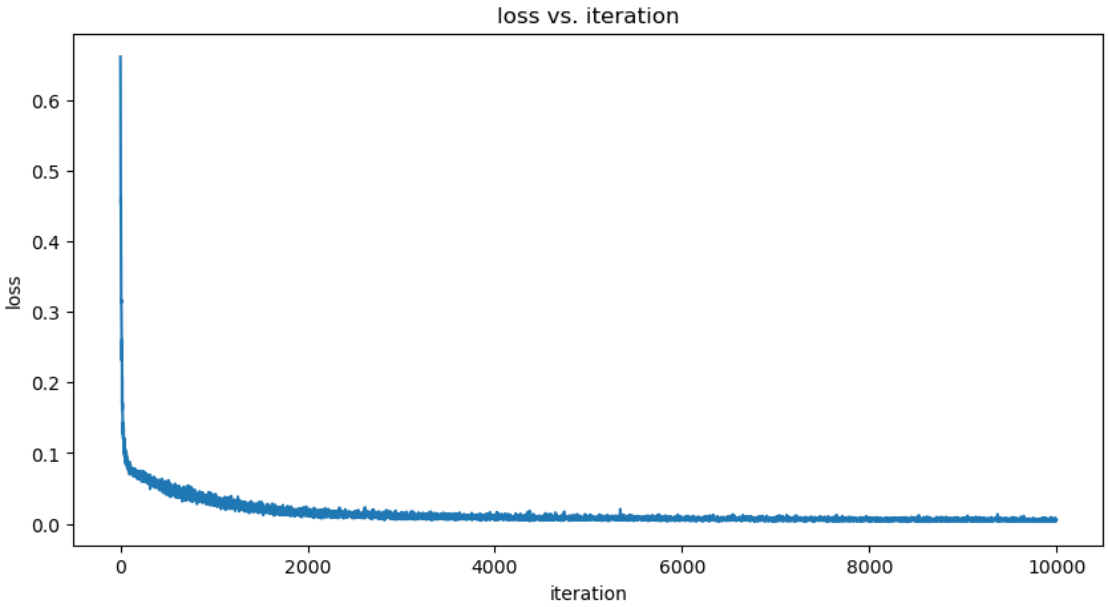
[10]:
```
def mixup(data_batch, label_batch, alpha):
    mixup_idx = np.random.choice(data_batch.size(0), data_batch.size(0))
    mixup_samples = data_batch[mixup_idx]
    mixup_labels = label_batch[mixup_idx]
    lambda_arr = torch.tensor(np.random.beta(alpha, alpha, size=data_batch.
 ↪size(0)))
    for sample_idx, (data_sample, label_sample) in enumerate(zip(data_batch,␣
 ↪label_batch)):
        mixup_samples[sample_idx] = lambda_arr[sample_idx] * data_sample + (1 -␣
 ↪lambda_arr[sample_idx]) * mixup_samples[
            sample_idx]
        mixup_labels[sample_idx] = lambda_arr[sample_idx] * label_sample + (1 -␣
 ↪lambda_arr[sample_idx]) * mixup_labels[
            sample_idx]
    return mixup_samples, mixup_labels
```

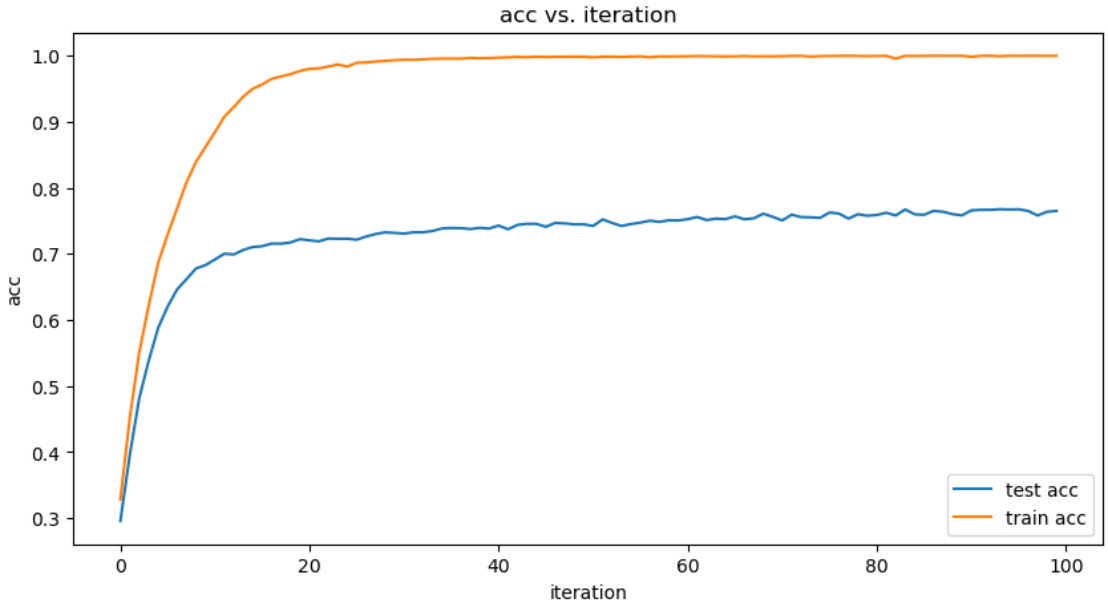
[6]:
```
train_X, train_y, test_X, test_y = get_dataset()
running_loss_mixup_2, running_acc_mixup_2, running_train_acc_mixup_2 =␣
 ↪train(train_X, train_y, test_X, test_y, augmentation=lambda x, y: mixup(x,␣
 ↪y, 0.2))
train_X, train_y, test_X, test_y = get_dataset()
running_loss_mixup_4, running_acc_mixup_4, running_train_acc_mixup_4 =␣
 ↪train(train_X, train_y, test_X, test_y, augmentation=lambda x, y: mixup(x,␣
 ↪y, 0.4))
```

```
Files already downloaded and verified
Files already downloaded and verified
train data: torch.Size([50000, 3, 32, 32]), train label: torch.Size([50000, 10])
test data: torch.Size([10000, 3, 32, 32]), test label: torch.Size([10000])
train_data_sampled: torch.Size([10000, 3, 32, 32]), train_label_sampled:
torch.Size([10000, 10])
train_data_sampled: 100, train_label_sampled: 100
train_data_sampled: torch.Size([100, 3, 32, 32]), train_label_sampled:
torch.Size([100, 10])
epoch: 1, loss: 0.06990260630846024, test acc: 0.2959, train acc: 0.3283
epoch: 2, loss: 0.06399860233068466, test acc: 0.3962, train acc: 0.4523
epoch: 3, loss: 0.05819224938750267, test acc: 0.4819, train acc: 0.5513
epoch: 4, loss: 0.05867232754826546, test acc: 0.5386, train acc: 0.6223
epoch: 5, loss: 0.0491856150329113, test acc: 0.5883, train acc: 0.6867
epoch: 6, loss: 0.04320608079433441, test acc: 0.6203, train acc: 0.7292
epoch: 7, loss: 0.042134735733270645, test acc: 0.6463, train acc: 0.7689
epoch: 8, loss: 0.03895001485943794, test acc: 0.6619, train acc: 0.8086
```

epoch: 9, loss: 0.04038247466087341, test acc: 0.678, train acc: 0.8394
epoch: 10, loss: 0.028294013813138008, test acc: 0.683, train acc: 0.8617
epoch: 11, loss: 0.029467925429344177, test acc: 0.6916, train acc: 0.8843
epoch: 12, loss: 0.029151761904358864, test acc: 0.7003, train acc: 0.9077
epoch: 13, loss: 0.020557798445224762, test acc: 0.6992, train acc: 0.9224
epoch: 14, loss: 0.022855358198285103, test acc: 0.706, train acc: 0.9381
epoch: 15, loss: 0.019298668950796127, test acc: 0.7104, train acc: 0.9499
epoch: 16, loss: 0.020935578271746635, test acc: 0.7117, train acc: 0.9565
epoch: 17, loss: 0.0156543031334877, test acc: 0.7156, train acc: 0.9649
epoch: 18, loss: 0.017471980303525925, test acc: 0.7155, train acc: 0.9686
epoch: 19, loss: 0.021055294200778008, test acc: 0.7174, train acc: 0.972
epoch: 20, loss: 0.01922060362994671, test acc: 0.7223, train acc: 0.9769
epoch: 21, loss: 0.014427981339395046, test acc: 0.7205, train acc: 0.9801
epoch: 22, loss: 0.012666899710893631, test acc: 0.719, train acc: 0.9808
epoch: 23, loss: 0.01760138012468815, test acc: 0.7233, train acc: 0.9836
epoch: 24, loss: 0.012028587982058525, test acc: 0.7229, train acc: 0.9866
epoch: 25, loss: 0.011018653400242329, test acc: 0.723, train acc: 0.9835
epoch: 26, loss: 0.008789804764091969, test acc: 0.7215, train acc: 0.9893
epoch: 27, loss: 0.012163756415247917, test acc: 0.7263, train acc: 0.9897
epoch: 28, loss: 0.014896933920681477, test acc: 0.7301, train acc: 0.9912
epoch: 29, loss: 0.010685714893043041, test acc: 0.7326, train acc: 0.9921
epoch: 30, loss: 0.01617460325360298, test acc: 0.7318, train acc: 0.9932
epoch: 31, loss: 0.007763540837913752, test acc: 0.7308, train acc: 0.9939
epoch: 32, loss: 0.010971849784255028, test acc: 0.7327, train acc: 0.9937
epoch: 33, loss: 0.011824049986898899, test acc: 0.7327, train acc: 0.9945
epoch: 34, loss: 0.010598092339932919, test acc: 0.7346, train acc: 0.9952
epoch: 35, loss: 0.013463246636092663, test acc: 0.7385, train acc: 0.9955
epoch: 36, loss: 0.01355811394751072, test acc: 0.7392, train acc: 0.9955
epoch: 37, loss: 0.010622275061905384, test acc: 0.739, train acc: 0.9955
epoch: 38, loss: 0.007913278415799141, test acc: 0.7376, train acc: 0.9966
epoch: 39, loss: 0.008232880383729935, test acc: 0.7394, train acc: 0.9961
epoch: 40, loss: 0.010141539387404919, test acc: 0.7384, train acc: 0.9964
epoch: 41, loss: 0.00860197376459837, test acc: 0.7429, train acc: 0.9971
epoch: 42, loss: 0.006725256331264973, test acc: 0.7373, train acc: 0.9975
epoch: 43, loss: 0.008079849183559418, test acc: 0.7439, train acc: 0.9982
epoch: 44, loss: 0.00860842876136303, test acc: 0.7455, train acc: 0.9977
epoch: 45, loss: 0.007876666262745857, test acc: 0.7454, train acc: 0.9984
epoch: 46, loss: 0.0076199485920369625, test acc: 0.7411, train acc: 0.998
epoch: 47, loss: 0.007646444719284773, test acc: 0.7469, train acc: 0.9983
epoch: 48, loss: 0.006134714465588331, test acc: 0.7462, train acc: 0.9983
epoch: 49, loss: 0.01037922129034996, test acc: 0.7447, train acc: 0.9985
epoch: 50, loss: 0.009813770651817322, test acc: 0.7448, train acc: 0.9984
epoch: 51, loss: 0.0057187872007489204, test acc: 0.7423, train acc: 0.9975
epoch: 52, loss: 0.012150523252785206, test acc: 0.7522, train acc: 0.9984
epoch: 53, loss: 0.008140780963003635, test acc: 0.7469, train acc: 0.9986
epoch: 54, loss: 0.00829895306378603, test acc: 0.7423, train acc: 0.998
epoch: 55, loss: 0.009309233166277409, test acc: 0.7452, train acc: 0.9986
epoch: 56, loss: 0.010488580912351608, test acc: 0.7474, train acc: 0.999

epoch: 57, loss: 0.008695347234606743, test acc: 0.7503, train acc: 0.9976
epoch: 58, loss: 0.010019233450293541, test acc: 0.7485, train acc: 0.9989
epoch: 59, loss: 0.0050649926997721195, test acc: 0.7508, train acc: 0.9987
epoch: 60, loss: 0.007034052163362503, test acc: 0.7506, train acc: 0.9989
epoch: 61, loss: 0.005797507241368294, test acc: 0.7526, train acc: 0.9991
epoch: 62, loss: 0.008512381464242935, test acc: 0.7559, train acc: 0.9994
epoch: 63, loss: 0.007684632670134306, test acc: 0.7512, train acc: 0.9993
epoch: 64, loss: 0.005980576854199171, test acc: 0.7534, train acc: 0.9991
epoch: 65, loss: 0.006230719853192568, test acc: 0.7527, train acc: 0.9988
epoch: 66, loss: 0.007445056457072496, test acc: 0.7568, train acc: 0.999
epoch: 67, loss: 0.00978008285164833, test acc: 0.7525, train acc: 0.9995
epoch: 68, loss: 0.00885844323784113, test acc: 0.7539, train acc: 0.9989
epoch: 69, loss: 0.006828161422163248, test acc: 0.7609, train acc: 0.9991
epoch: 70, loss: 0.006645433604717255, test acc: 0.7561, train acc: 0.999
epoch: 71, loss: 0.006022353190928698, test acc: 0.7507, train acc: 0.9992
epoch: 72, loss: 0.006742055993527174, test acc: 0.7594, train acc: 0.9997
epoch: 73, loss: 0.004122141283005476, test acc: 0.7557, train acc: 0.9997
epoch: 74, loss: 0.008295712061226368, test acc: 0.7554, train acc: 0.9987
epoch: 75, loss: 0.005498307291418314, test acc: 0.7545, train acc: 0.9993
epoch: 76, loss: 0.007596911396831274, test acc: 0.7627, train acc: 0.9996
epoch: 77, loss: 0.006315879989415407, test acc: 0.7611, train acc: 0.9997
epoch: 78, loss: 0.0053792414255440235, test acc: 0.7534, train acc: 0.9998
epoch: 79, loss: 0.004663547966629267, test acc: 0.7601, train acc: 0.9997
epoch: 80, loss: 0.003257066709920764, test acc: 0.7578, train acc: 0.9993
epoch: 81, loss: 0.006298436317592859, test acc: 0.7589, train acc: 0.9996
epoch: 82, loss: 0.005930742714554071, test acc: 0.7623, train acc: 0.9998
epoch: 83, loss: 0.00577149773016572, test acc: 0.7582, train acc: 0.9955
epoch: 84, loss: 0.007446452043950558, test acc: 0.7672, train acc: 0.9997
epoch: 85, loss: 0.005257091019302607, test acc: 0.76, train acc: 0.9996
epoch: 86, loss: 0.004544971510767937, test acc: 0.7592, train acc: 0.9997
epoch: 87, loss: 0.006269799079746008, test acc: 0.7651, train acc: 0.9999
epoch: 88, loss: 0.007595733739435673, test acc: 0.7638, train acc: 0.9999
epoch: 89, loss: 0.007846420630812645, test acc: 0.7602, train acc: 0.9998
epoch: 90, loss: 0.006870228331536055, test acc: 0.7583, train acc: 0.9999
epoch: 91, loss: 0.006406405009329319, test acc: 0.7659, train acc: 0.9983
epoch: 92, loss: 0.005937369540333748, test acc: 0.7667, train acc: 0.9998
epoch: 93, loss: 0.00495437765493989, test acc: 0.7667, train acc: 0.9999
epoch: 94, loss: 0.005106732714921236, test acc: 0.7676, train acc: 0.9992
epoch: 95, loss: 0.005415619350969791, test acc: 0.7672, train acc: 0.9998
epoch: 96, loss: 0.006282527931034565, test acc: 0.7675, train acc: 0.9998
epoch: 97, loss: 0.004731628578156233, test acc: 0.7651, train acc: 0.9999
epoch: 98, loss: 0.005173301789909601, test acc: 0.7582, train acc: 0.9999
epoch: 99, loss: 0.005042733158916235, test acc: 0.7637, train acc: 0.9998
epoch: 100, loss: 0.00508588133379817, test acc: 0.7651, train acc: 0.9998



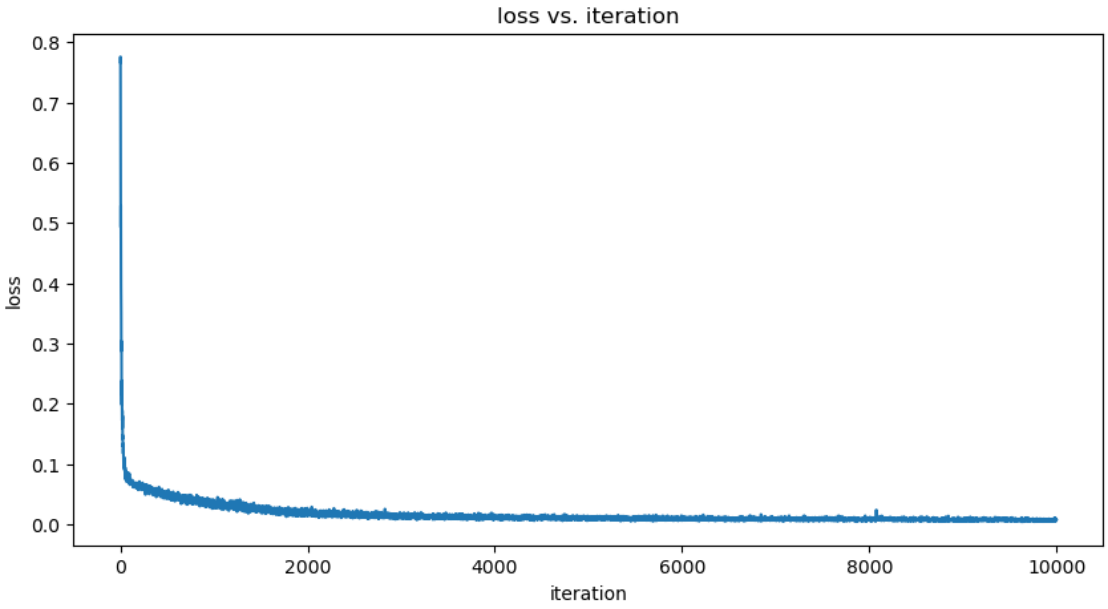


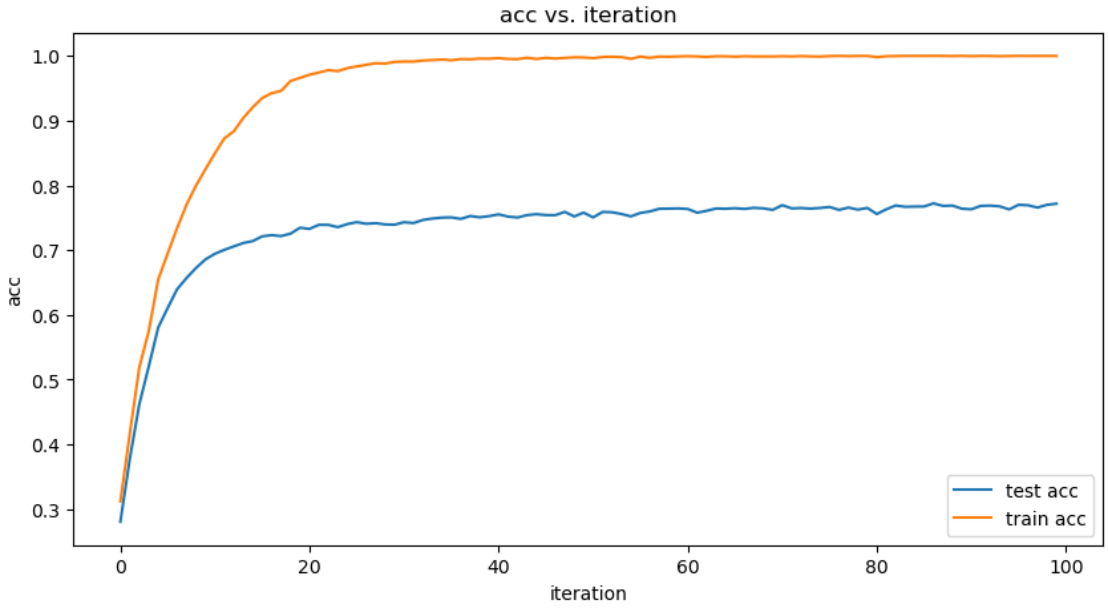

```
Files already downloaded and verified
Files already downloaded and verified
train data: torch.Size([50000, 3, 32, 32]), train label: torch.Size([50000, 10])
test data: torch.Size([10000, 3, 32, 32]), test label: torch.Size([10000])
train_data_sampled: torch.Size([10000, 3, 32, 32]), train_label_sampled:
torch.Size([10000, 10])
train_data_sampled: 100, train_label_sampled: 100
```

```
train_data_sampled: torch.Size([100, 3, 32, 32]), train_label_sampled:
torch.Size([100, 10])
epoch: 1, loss: 0.07491793483495712, test acc: 0.2807, train acc: 0.312
epoch: 2, loss: 0.06374610215425491, test acc: 0.3781, train acc: 0.417
epoch: 3, loss: 0.055627405643463135, test acc: 0.4621, train acc: 0.519
epoch: 4, loss: 0.05430176109075546, test acc: 0.5204, train acc: 0.575
epoch: 5, loss: 0.04955452308058739, test acc: 0.5805, train acc: 0.6548
epoch: 6, loss: 0.04990536719560623, test acc: 0.6106, train acc: 0.6948
epoch: 7, loss: 0.04007405787706375, test acc: 0.6397, train acc: 0.7341
epoch: 8, loss: 0.038220033049583435, test acc: 0.657, train acc: 0.7704
epoch: 9, loss: 0.040372248739004135, test acc: 0.6725, train acc: 0.7998
epoch: 10, loss: 0.03394496440887451, test acc: 0.6857, train acc: 0.825
epoch: 11, loss: 0.02789261005818844, test acc: 0.6944, train acc: 0.8497
epoch: 12, loss: 0.027137700468301773, test acc: 0.7005, train acc: 0.8726
epoch: 13, loss: 0.02661474421620369, test acc: 0.7058, train acc: 0.8835
epoch: 14, loss: 0.023620719090104103, test acc: 0.711, train acc: 0.9041
epoch: 15, loss: 0.023221604526042938, test acc: 0.714, train acc: 0.9206
epoch: 16, loss: 0.023854820057749748, test acc: 0.7213, train acc: 0.9348
epoch: 17, loss: 0.018495827913284302, test acc: 0.7231, train acc: 0.9424
epoch: 18, loss: 0.01442231610417366, test acc: 0.7217, train acc: 0.9458
epoch: 19, loss: 0.014133330434560776, test acc: 0.7255, train acc: 0.9612
epoch: 20, loss: 0.01722422055900097, test acc: 0.7346, train acc: 0.9659
epoch: 21, loss: 0.015292219817638397, test acc: 0.7328, train acc: 0.9708
epoch: 22, loss: 0.014703410677611828, test acc: 0.7391, train acc: 0.9743
epoch: 23, loss: 0.016472110524773598, test acc: 0.739, train acc: 0.9778
epoch: 24, loss: 0.014986475929617882, test acc: 0.7355, train acc: 0.9763
epoch: 25, loss: 0.012513908557593822, test acc: 0.7402, train acc: 0.9809
epoch: 26, loss: 0.014186632819473743, test acc: 0.7432, train acc: 0.9837
epoch: 27, loss: 0.013791984878480434, test acc: 0.7407, train acc: 0.9861
epoch: 28, loss: 0.010662726126611233, test acc: 0.7416, train acc: 0.9885
epoch: 29, loss: 0.01740904711186886, test acc: 0.7397, train acc: 0.988
epoch: 30, loss: 0.009794371202588081, test acc: 0.7393, train acc: 0.9905
epoch: 31, loss: 0.01236952468752861, test acc: 0.7431, train acc: 0.9911
epoch: 32, loss: 0.014453453943133354, test acc: 0.7418, train acc: 0.9911
epoch: 33, loss: 0.010614958591759205, test acc: 0.7466, train acc: 0.9927
epoch: 34, loss: 0.00914190523326397, test acc: 0.7488, train acc: 0.9934
epoch: 35, loss: 0.01112157478928566, test acc: 0.7502, train acc: 0.9942
epoch: 36, loss: 0.012181958183646202, test acc: 0.7506, train acc: 0.9931
epoch: 37, loss: 0.01400761492550373, test acc: 0.7482, train acc: 0.9949
epoch: 38, loss: 0.015802817419171333, test acc: 0.7527, train acc: 0.9946
epoch: 39, loss: 0.013186005875468254, test acc: 0.7507, train acc: 0.9956
epoch: 40, loss: 0.01381335873156786, test acc: 0.7525, train acc: 0.9955
epoch: 41, loss: 0.008359283208847046, test acc: 0.7552, train acc: 0.9964
epoch: 42, loss: 0.009452230297029018, test acc: 0.7518, train acc: 0.9951
epoch: 43, loss: 0.009239623323082924, test acc: 0.7504, train acc: 0.9948
epoch: 44, loss: 0.013287489302456379, test acc: 0.754, train acc: 0.997
epoch: 45, loss: 0.00991004891693592, test acc: 0.7555, train acc: 0.9951
epoch: 46, loss: 0.010622729547321796, test acc: 0.7541, train acc: 0.9968
```

epoch: 47, loss: 0.012665116228163242, test acc: 0.754, train acc: 0.9957
epoch: 48, loss: 0.008731372654438019, test acc: 0.7591, train acc: 0.9967
epoch: 49, loss: 0.010126256383955479, test acc: 0.752, train acc: 0.9976
epoch: 50, loss: 0.006997264921665192, test acc: 0.7579, train acc: 0.9974
epoch: 51, loss: 0.01150534301996231, test acc: 0.7505, train acc: 0.9963
epoch: 52, loss: 0.010926640592515469, test acc: 0.759, train acc: 0.9982
epoch: 53, loss: 0.009108432568609715, test acc: 0.7585, train acc: 0.9985
epoch: 54, loss: 0.007158889900892973, test acc: 0.7558, train acc: 0.998
epoch: 55, loss: 0.010137894190847874, test acc: 0.7521, train acc: 0.9953
epoch: 56, loss: 0.009575883857905865, test acc: 0.7575, train acc: 0.9988
epoch: 57, loss: 0.009521456435322762, test acc: 0.7597, train acc: 0.9969
epoch: 58, loss: 0.009729796089231968, test acc: 0.7639, train acc: 0.9988
epoch: 59, loss: 0.010306259617209435, test acc: 0.7641, train acc: 0.9985
epoch: 60, loss: 0.008771911263465881, test acc: 0.7644, train acc: 0.999
epoch: 61, loss: 0.007922545075416565, test acc: 0.7636, train acc: 0.9994
epoch: 62, loss: 0.012205599807202816, test acc: 0.7578, train acc: 0.999
epoch: 63, loss: 0.011109442450106144, test acc: 0.7607, train acc: 0.9983
epoch: 64, loss: 0.009320690296590328, test acc: 0.7643, train acc: 0.9993
epoch: 65, loss: 0.008842707611620426, test acc: 0.7639, train acc: 0.9992
epoch: 66, loss: 0.007240456063300371, test acc: 0.7647, train acc: 0.9987
epoch: 67, loss: 0.007379146292805672, test acc: 0.7637, train acc: 0.9993
epoch: 68, loss: 0.007517263758927584, test acc: 0.7653, train acc: 0.999
epoch: 69, loss: 0.006432157941162586, test acc: 0.7645, train acc: 0.999
epoch: 70, loss: 0.0077491938136518, test acc: 0.7622, train acc: 0.999
epoch: 71, loss: 0.009936437010765076, test acc: 0.7694, train acc: 0.9993
epoch: 72, loss: 0.009076383896172047, test acc: 0.7645, train acc: 0.999
epoch: 73, loss: 0.007070276886224747, test acc: 0.765, train acc: 0.9995
epoch: 74, loss: 0.00821662601083517, test acc: 0.7641, train acc: 0.9991
epoch: 75, loss: 0.007936238311231136, test acc: 0.7651, train acc: 0.9988
epoch: 76, loss: 0.006570979952812195, test acc: 0.7666, train acc: 0.9996
epoch: 77, loss: 0.00850741472095251, test acc: 0.7622, train acc: 0.9998
epoch: 78, loss: 0.007071048021316528, test acc: 0.7657, train acc: 0.9995
epoch: 79, loss: 0.0077611422166228294, test acc: 0.7626, train acc: 0.9997
epoch: 80, loss: 0.009177954867482185, test acc: 0.7651, train acc: 0.9998
epoch: 81, loss: 0.0073374416679143906, test acc: 0.7558, train acc: 0.998
epoch: 82, loss: 0.009241148829460144, test acc: 0.763, train acc: 0.9993
epoch: 83, loss: 0.0077719190157949924, test acc: 0.7689, train acc: 0.9996
epoch: 84, loss: 0.008720850571990013, test acc: 0.7669, train acc: 0.9998
epoch: 85, loss: 0.00853673554956913, test acc: 0.7673, train acc: 0.9998
epoch: 86, loss: 0.008180550299584866, test acc: 0.7674, train acc: 0.9998
epoch: 87, loss: 0.008054685778915882, test acc: 0.7722, train acc: 0.9999
epoch: 88, loss: 0.005552793852984905, test acc: 0.7683, train acc: 0.9998
epoch: 89, loss: 0.009409700520336628, test acc: 0.7688, train acc: 0.9996
epoch: 90, loss: 0.00683743879199028, test acc: 0.764, train acc: 0.9998
epoch: 91, loss: 0.00661622267216444, test acc: 0.7631, train acc: 0.9995
epoch: 92, loss: 0.005683603696525097, test acc: 0.7683, train acc: 0.9998
epoch: 93, loss: 0.00805608183145523, test acc: 0.7687, train acc: 0.9997
epoch: 94, loss: 0.010204089805483818, test acc: 0.7676, train acc: 0.9994

```
epoch: 95, loss: 0.006772044580429792, test acc: 0.7629, train acc: 0.9996
epoch: 96, loss: 0.00826518889516592, test acc: 0.77, train acc: 0.9998
epoch: 97, loss: 0.007529132068157196, test acc: 0.7693, train acc: 0.9997
epoch: 98, loss: 0.00766611797735095, test acc: 0.7657, train acc: 0.9998
epoch: 99, loss: 0.00835186056792736, test acc: 0.77, train acc: 0.9998
epoch: 100, loss: 0.008235705085098743, test acc: 0.7717, train acc: 0.9997
```





The final results of the training with mixup (alpha = 0.2) is the following: - test accuracy: 0.7651 - train accuracy: 0.998 - loss: 0.0005

The final results of the training with mixup (alpha = 0.4) is the following: - test accuracy: 0.7717 - train accuracy: 0.997 - loss: 0.0008

3 Task - 3

[11]:
```
def cutout(data_batch, label_batch, k=16):
    cutout_mask = np.random.choice(2, data_batch.size(0)) # if zero no cutoff␣
 ↪if one cutoff
    cutout_samples = torch.zeros(data_batch.size())
    for sample_idx, (data_sample, label_sample) in enumerate(zip(data_batch,␣
 ↪label_batch)):
        if cutout_mask[sample_idx] == 1:
            random_row = np.random.choice(data_sample.size(1), 1)[0]
            random_col = np.random.choice(data_sample.size(2), 1)[0]
            if k % 2 == 0:
                cutout_row_min, cutout_row_max = int(max(0, random_row - ((k /␣
 ↪2) - 1))), int(min(data_sample.size(1) - 1, random_row + (k / 2)))
                cutout_col_min, cutout_col_max = int(max(0, random_col - ((k /␣
 ↪2) - 1))), int(min(data_sample.size(1) - 1, random_col + (k / 2)))
            else:
                cutout_row_min, cutout_row_max = int(max(0, random_row - ((k -␣
 ↪1) / 2))), int(min(data_sample.size(1) - 1, random_row + ((k - 1) / 2)))
                cutout_col_min, cutout_col_max = int(max(0, random_col - ((k -␣
 ↪1) / 2))), int(min(data_sample.size(1) - 1, random_col + ((k - 1) / 2)))
            zero_filter = torch.zeros((data_sample.size(0), cutout_row_max -␣
 ↪cutout_row_min + 1, cutout_col_max - cutout_col_min + 1))
            cutout_samples[sample_idx] = data_sample
            cutout_samples[sample_idx, :, cutout_row_min:(cutout_row_max + 1),␣
 ↪cutout_col_min:(cutout_col_max + 1)] = zero_filter

    return cutout_samples, label_batch
```

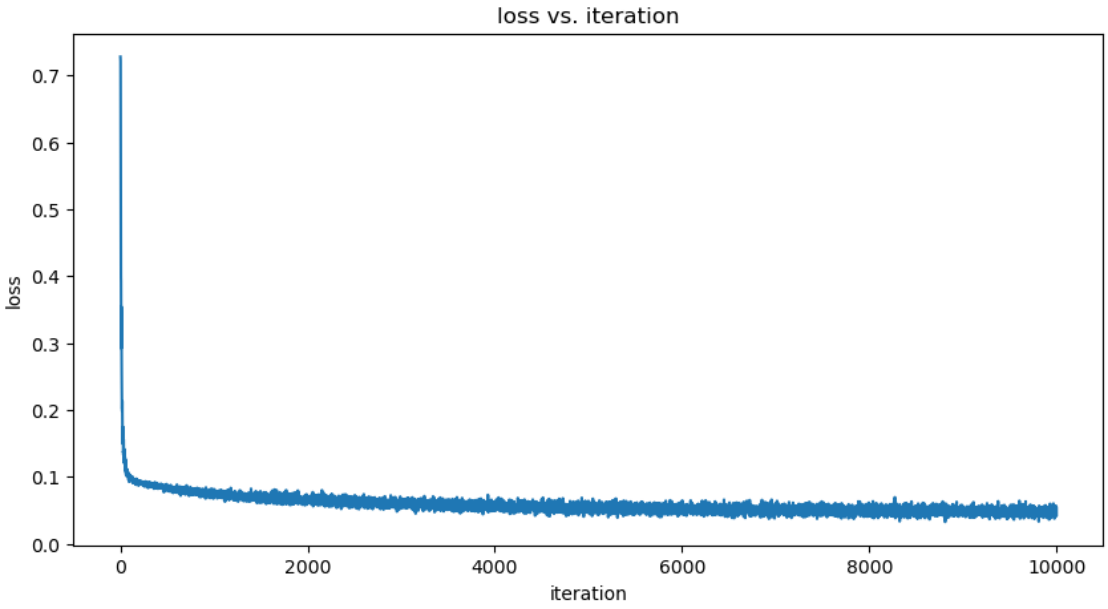
[7]:
```
train_X, train_y, test_X, test_y = get_dataset()
running_loss_cutout, running_acc_cutout, running_train_acc_cutout =␣
 ↪train(train_X, train_y, test_X, test_y, augmentation=cutout)
```

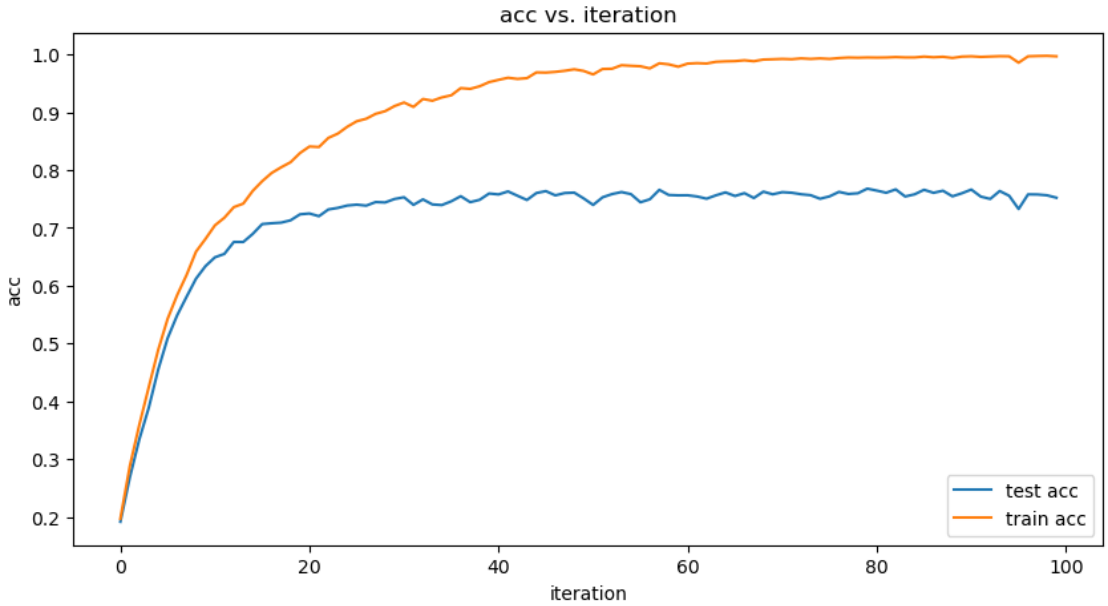
```
Files already downloaded and verified
Files already downloaded and verified
train data: torch.Size([50000, 3, 32, 32]), train label: torch.Size([50000, 10])
test data: torch.Size([10000, 3, 32, 32]), test label: torch.Size([10000])
train_data_sampled: torch.Size([10000, 3, 32, 32]), train_label_sampled:
torch.Size([10000, 10])
train_data_sampled: 100, train_label_sampled: 100
train_data_sampled: torch.Size([100, 3, 32, 32]), train_label_sampled:
```

```
torch.Size([100, 10])
epoch: 1, loss: 0.09869825094938278, test acc: 0.1917, train acc: 0.197
epoch: 2, loss: 0.08894947916269302, test acc: 0.2686, train acc: 0.288
epoch: 3, loss: 0.08612920343875885, test acc: 0.3341, train acc: 0.3591
epoch: 4, loss: 0.0845990777015686, test acc: 0.3889, train acc: 0.4244
epoch: 5, loss: 0.0819743275642395, test acc: 0.455, train acc: 0.489
epoch: 6, loss: 0.08564885705709457, test acc: 0.5094, train acc: 0.5429
epoch: 7, loss: 0.08172920346260071, test acc: 0.5487, train acc: 0.5836
epoch: 8, loss: 0.07927829027175903, test acc: 0.5809, train acc: 0.6186
epoch: 9, loss: 0.07522102445363998, test acc: 0.6123, train acc: 0.659
epoch: 10, loss: 0.07391782104969025, test acc: 0.6339, train acc: 0.681
epoch: 11, loss: 0.07897448539733887, test acc: 0.6492, train acc: 0.7045
epoch: 12, loss: 0.0721789076924324, test acc: 0.6551, train acc: 0.7177
epoch: 13, loss: 0.07726304233074188, test acc: 0.6758, train acc: 0.736
epoch: 14, loss: 0.06939802318811417, test acc: 0.6757, train acc: 0.7421
epoch: 15, loss: 0.06592607498168945, test acc: 0.6897, train acc: 0.7641
epoch: 16, loss: 0.06551659107208252, test acc: 0.7067, train acc: 0.7809
epoch: 17, loss: 0.06760089844465256, test acc: 0.7081, train acc: 0.7951
epoch: 18, loss: 0.06759130209684372, test acc: 0.7092, train acc: 0.8049
epoch: 19, loss: 0.06990764290094376, test acc: 0.7132, train acc: 0.8137
epoch: 20, loss: 0.06531646847724915, test acc: 0.7236, train acc: 0.8296
epoch: 21, loss: 0.06770215183496475, test acc: 0.725, train acc: 0.8409
epoch: 22, loss: 0.0719475969672203, test acc: 0.7203, train acc: 0.84
epoch: 23, loss: 0.06240691617131233, test acc: 0.732, train acc: 0.8558
epoch: 24, loss: 0.06619412451982498, test acc: 0.7347, train acc: 0.8632
epoch: 25, loss: 0.05751859396696091, test acc: 0.7388, train acc: 0.8751
epoch: 26, loss: 0.06517964601516724, test acc: 0.7403, train acc: 0.8846
epoch: 27, loss: 0.05798964947462082, test acc: 0.7386, train acc: 0.8889
epoch: 28, loss: 0.059940699487924576, test acc: 0.7448, train acc: 0.8974
epoch: 29, loss: 0.06800143420696259, test acc: 0.7441, train acc: 0.902
epoch: 30, loss: 0.06415283679962158, test acc: 0.7501, train acc: 0.9109
epoch: 31, loss: 0.055476024746894836, test acc: 0.7531, train acc: 0.917
epoch: 32, loss: 0.05871293321251869, test acc: 0.7399, train acc: 0.9094
epoch: 33, loss: 0.0579301193356514, test acc: 0.7494, train acc: 0.9231
epoch: 34, loss: 0.05639321357011795, test acc: 0.7407, train acc: 0.9201
epoch: 35, loss: 0.06106572598218918, test acc: 0.7396, train acc: 0.9258
epoch: 36, loss: 0.06225500628352165, test acc: 0.7461, train acc: 0.9295
epoch: 37, loss: 0.058533553034067154, test acc: 0.7549, train acc: 0.9419
epoch: 38, loss: 0.05888955295085907, test acc: 0.7446, train acc: 0.9406
epoch: 39, loss: 0.060350604355335236, test acc: 0.7487, train acc: 0.9451
epoch: 40, loss: 0.05010366067290306, test acc: 0.7596, train acc: 0.9523
epoch: 41, loss: 0.05513612553477287, test acc: 0.758, train acc: 0.9562
epoch: 42, loss: 0.053337205201387405, test acc: 0.7632, train acc: 0.9598
epoch: 43, loss: 0.056175362318754196, test acc: 0.7557, train acc: 0.9578
epoch: 44, loss: 0.062331125140190125, test acc: 0.7484, train acc: 0.9591
epoch: 45, loss: 0.0569499172270298, test acc: 0.7604, train acc: 0.9691
epoch: 46, loss: 0.054143521934747696, test acc: 0.7637, train acc: 0.9688
epoch: 47, loss: 0.06010737270116806, test acc: 0.7564, train acc: 0.97
```

epoch: 48, loss: 0.05825473368167877, test acc: 0.7604, train acc: 0.9719
epoch: 49, loss: 0.06180543825030327, test acc: 0.7611, train acc: 0.9744
epoch: 50, loss: 0.05444752052426338, test acc: 0.7511, train acc: 0.9716
epoch: 51, loss: 0.05563029646873474, test acc: 0.7398, train acc: 0.9656
epoch: 52, loss: 0.051579248160123825, test acc: 0.753, train acc: 0.975
epoch: 53, loss: 0.056677211076021194, test acc: 0.7586, train acc: 0.9754
epoch: 54, loss: 0.053661949932575226, test acc: 0.7621, train acc: 0.9815
epoch: 55, loss: 0.05488939955830574, test acc: 0.7585, train acc: 0.9805
epoch: 56, loss: 0.058740392327308655, test acc: 0.7444, train acc: 0.9797
epoch: 57, loss: 0.05505458265542984, test acc: 0.7496, train acc: 0.976
epoch: 58, loss: 0.057436998933553696, test acc: 0.7659, train acc: 0.9848
epoch: 59, loss: 0.05677672103047371, test acc: 0.757, train acc: 0.983
epoch: 60, loss: 0.054082948714494705, test acc: 0.7564, train acc: 0.9789
epoch: 61, loss: 0.04566636309027672, test acc: 0.7565, train acc: 0.9843
epoch: 62, loss: 0.05081859230995178, test acc: 0.7544, train acc: 0.9851
epoch: 63, loss: 0.04864325001835823, test acc: 0.7506, train acc: 0.9845
epoch: 64, loss: 0.054381296038627625, test acc: 0.7565, train acc: 0.9874
epoch: 65, loss: 0.051750972867012024, test acc: 0.7615, train acc: 0.9882
epoch: 66, loss: 0.04666256904602051, test acc: 0.7553, train acc: 0.9886
epoch: 67, loss: 0.05571414902806282, test acc: 0.7602, train acc: 0.99
epoch: 68, loss: 0.05492745339870453, test acc: 0.7518, train acc: 0.9884
epoch: 69, loss: 0.05467253550887108, test acc: 0.7628, train acc: 0.9913
epoch: 70, loss: 0.04535382241010666, test acc: 0.7582, train acc: 0.9918
epoch: 71, loss: 0.05208577215671539, test acc: 0.7619, train acc: 0.9924
epoch: 72, loss: 0.0532839372754097, test acc: 0.7609, train acc: 0.9919
epoch: 73, loss: 0.05134724825620651, test acc: 0.7583, train acc: 0.9933
epoch: 74, loss: 0.058595795184373856, test acc: 0.7567, train acc: 0.9924
epoch: 75, loss: 0.05831894278526306, test acc: 0.7505, train acc: 0.9933
epoch: 76, loss: 0.05308462679386139, test acc: 0.7546, train acc: 0.9924
epoch: 77, loss: 0.053802844136953354, test acc: 0.7624, train acc: 0.9939
epoch: 78, loss: 0.047006942331790924, test acc: 0.7588, train acc: 0.9948
epoch: 79, loss: 0.06305711716413498, test acc: 0.76, train acc: 0.9945
epoch: 80, loss: 0.04909297823905945, test acc: 0.768, train acc: 0.9949
epoch: 81, loss: 0.04798582196235657, test acc: 0.7646, train acc: 0.9947
epoch: 82, loss: 0.05481545254588127, test acc: 0.7609, train acc: 0.9949
epoch: 83, loss: 0.05115797743201256, test acc: 0.7667, train acc: 0.9956
epoch: 84, loss: 0.05015217140316963, test acc: 0.7544, train acc: 0.995
epoch: 85, loss: 0.05282312259078026, test acc: 0.7582, train acc: 0.995
epoch: 86, loss: 0.05328960716724396, test acc: 0.766, train acc: 0.9964
epoch: 87, loss: 0.0418008454144001, test acc: 0.7608, train acc: 0.9953
epoch: 88, loss: 0.04117252677679062, test acc: 0.7644, train acc: 0.9961
epoch: 89, loss: 0.05389820784330368, test acc: 0.7547, train acc: 0.9941
epoch: 90, loss: 0.05068975314497948, test acc: 0.76, train acc: 0.9964
epoch: 91, loss: 0.04648975655436516, test acc: 0.7664, train acc: 0.9969
epoch: 92, loss: 0.05615836754441261, test acc: 0.7542, train acc: 0.9959
epoch: 93, loss: 0.043193258345127106, test acc: 0.7502, train acc: 0.9965
epoch: 94, loss: 0.05237484723329544, test acc: 0.7639, train acc: 0.9971
epoch: 95, loss: 0.041562922298908234, test acc: 0.7557, train acc: 0.9969

```
epoch: 96, loss: 0.05258917808532715, test acc: 0.7328, train acc: 0.9859
epoch: 97, loss: 0.04942133650183678, test acc: 0.7581, train acc: 0.9969
epoch: 98, loss: 0.04889083281159401, test acc: 0.7579, train acc: 0.9974
epoch: 99, loss: 0.04777270555496216, test acc: 0.7565, train acc: 0.9977
epoch: 100, loss: 0.05342172086238861, test acc: 0.7523, train acc: 0.9969
```





The final results of the training with cutout is the following: - test accuracy: 0.7523 - train accuracy:

0.9969 - loss: 0.05

4 Task - 4

[12]:
```
def standard_augmentation(data_batch, label_batch, k=4):
    standard_samples = torch.zeros(data_batch.size())
    for sample_idx, (data_sample, label_sample) in enumerate(zip(data_batch,␣
 ↪label_batch)):
        upward_k, rightward_k = np.random.choice(list(range(-1*k, k+1)), 2).
 ↪astype(int)
        if upward_k > 0:
            standard_samples[sample_idx, :, :(data_sample.size(1) - upward_k), :
 ↪] = data_sample[:, upward_k:, :]
        else:
            upward_k = -1 * upward_k
            standard_samples[sample_idx, :, upward_k:, :] = data_sample[:, :
 ↪(data_sample.size(1) - upward_k), :]

        if rightward_k > 0:
            standard_samples[sample_idx, :, :, rightward_k:] = data_sample[:, :
 ↪, :(data_sample.size(2) - rightward_k)]
        else:
            rightward_k = -1 * rightward_k
            standard_samples[sample_idx, :, :, :(data_sample.size(2) -␣
 ↪rightward_k)] = data_sample[:, :, rightward_k:]

        flip_or_not = np.random.choice(2, 1) # if zero not flip if one flip
        if flip_or_not == 1:
            flip_r = torch.fliplr(standard_samples[sample_idx, 0, :, :])
            flip_g = torch.fliplr(standard_samples[sample_idx, 1, :, :])
            flip_b = torch.fliplr(standard_samples[sample_idx, 2, :, :])
            standard_samples[sample_idx] = torch.stack([flip_r, flip_g, flip_b])
    return standard_samples, label_batch
```

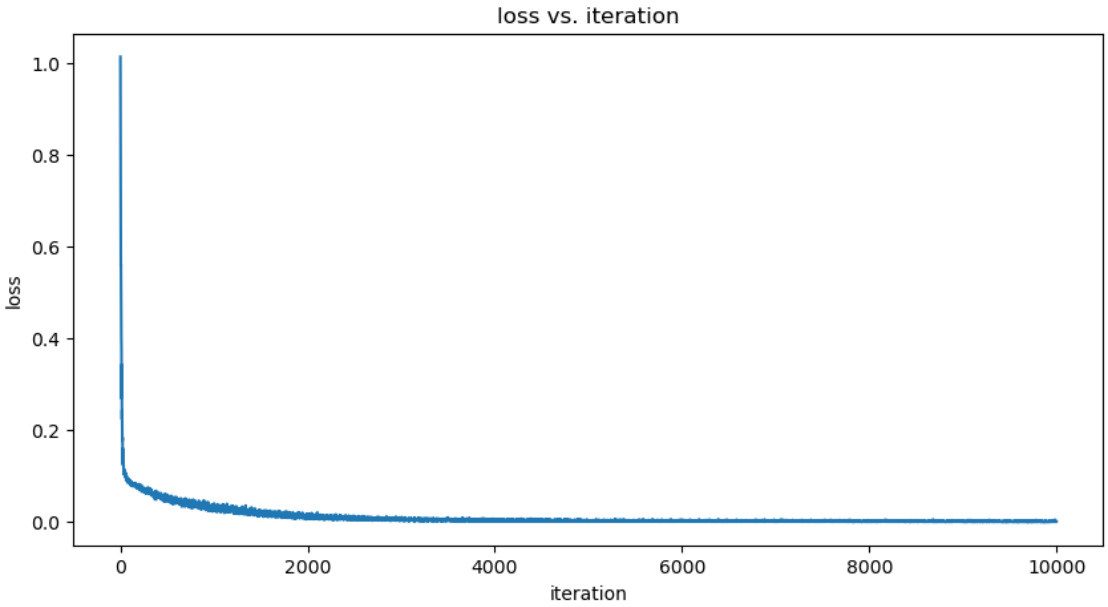
[8]:
```
train_X, train_y, test_X, test_y = get_dataset()
running_loss_standard, running_acc_standard, running_train_acc_standard =␣
 ↪train(train_X, train_y, test_X, test_y, augmentation=standard_augmentation)
```

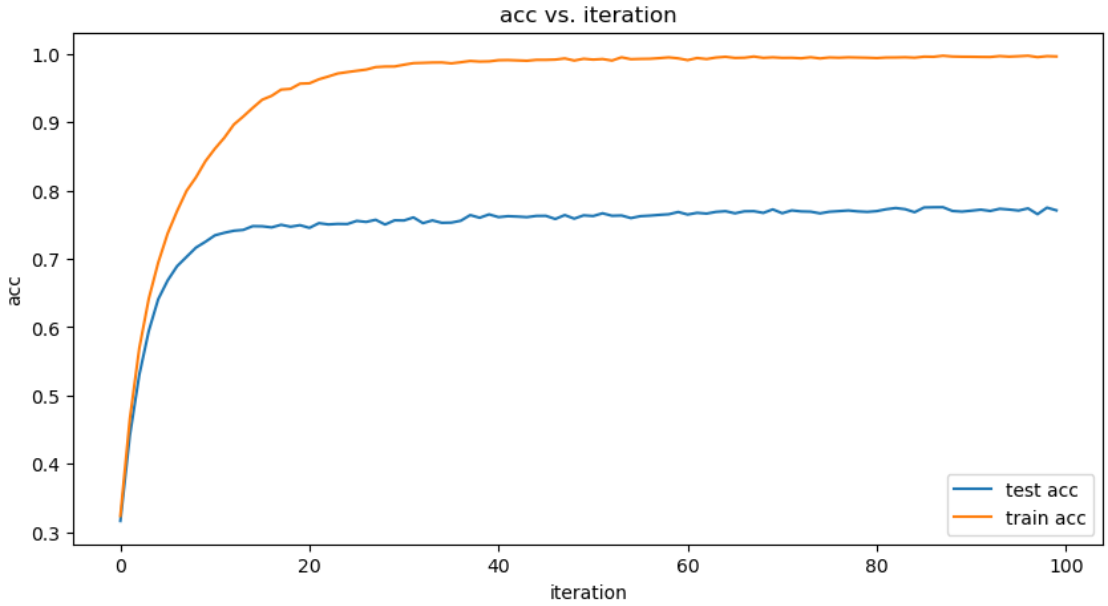
```
Files already downloaded and verified
Files already downloaded and verified
train data: torch.Size([50000, 3, 32, 32]), train label: torch.Size([50000, 10])
test data: torch.Size([10000, 3, 32, 32]), test label: torch.Size([10000])
train_data_sampled: torch.Size([10000, 3, 32, 32]), train_label_sampled:
torch.Size([10000, 10])
train_data_sampled: 100, train_label_sampled: 100
train_data_sampled: torch.Size([100, 3, 32, 32]), train_label_sampled:
torch.Size([100, 10])
```

```
epoch: 1, loss: 0.0879349485039711, test acc: 0.3166, train acc: 0.3239
epoch: 2, loss: 0.07401793450117111, test acc: 0.4416, train acc: 0.4655
epoch: 3, loss: 0.06605496257543564, test acc: 0.5302, train acc: 0.569
epoch: 4, loss: 0.0590493343770504, test acc: 0.5943, train acc: 0.6414
epoch: 5, loss: 0.05434347689151764, test acc: 0.6408, train acc: 0.6943
epoch: 6, loss: 0.05159439146518707, test acc: 0.6683, train acc: 0.737
epoch: 7, loss: 0.043481454253196716, test acc: 0.6892, train acc: 0.7699
epoch: 8, loss: 0.04408739134669304, test acc: 0.7028, train acc: 0.7994
epoch: 9, loss: 0.03796207532286644, test acc: 0.7164, train acc: 0.8192
epoch: 10, loss: 0.033165376633405685, test acc: 0.7249, train acc: 0.8428
epoch: 11, loss: 0.030968395993113518, test acc: 0.7343, train acc: 0.861
epoch: 12, loss: 0.026470445096492767, test acc: 0.738, train acc: 0.8771
epoch: 13, loss: 0.025108547881245613, test acc: 0.741, train acc: 0.8963
epoch: 14, loss: 0.02486906200647354, test acc: 0.7422, train acc: 0.9081
epoch: 15, loss: 0.022069908678531647, test acc: 0.7477, train acc: 0.9207
epoch: 16, loss: 0.02162429876625538, test acc: 0.7475, train acc: 0.9326
epoch: 17, loss: 0.015907863155007362, test acc: 0.7459, train acc: 0.9383
epoch: 18, loss: 0.017329929396510124, test acc: 0.7497, train acc: 0.9474
epoch: 19, loss: 0.017098277807235718, test acc: 0.7469, train acc: 0.9485
epoch: 20, loss: 0.013699951581656933, test acc: 0.7491, train acc: 0.9561
epoch: 21, loss: 0.012308248318731785, test acc: 0.7452, train acc: 0.9566
epoch: 22, loss: 0.009447853080928326, test acc: 0.7521, train acc: 0.9625
epoch: 23, loss: 0.008813097141683102, test acc: 0.7502, train acc: 0.9666
epoch: 24, loss: 0.008671049028635025, test acc: 0.751, train acc: 0.971
epoch: 25, loss: 0.009108972735702991, test acc: 0.7508, train acc: 0.973
epoch: 26, loss: 0.008427761495113373, test acc: 0.7554, train acc: 0.9749
epoch: 27, loss: 0.009106003679335117, test acc: 0.7539, train acc: 0.9767
epoch: 28, loss: 0.0055327811278402805, test acc: 0.757, train acc: 0.9803
epoch: 29, loss: 0.006833572406321764, test acc: 0.7501, train acc: 0.9811
epoch: 30, loss: 0.00936303474009037, test acc: 0.7562, train acc: 0.9812
epoch: 31, loss: 0.006988380569964647, test acc: 0.756, train acc: 0.9838
epoch: 32, loss: 0.005965442396700382, test acc: 0.7605, train acc: 0.9861
epoch: 33, loss: 0.004350312519818544, test acc: 0.7521, train acc: 0.9865
epoch: 34, loss: 0.004744218662381172, test acc: 0.7561, train acc: 0.987
epoch: 35, loss: 0.009936727583408356, test acc: 0.7526, train acc: 0.9871
epoch: 36, loss: 0.006778798531740904, test acc: 0.7529, train acc: 0.9858
epoch: 37, loss: 0.0077829137444496155, test acc: 0.7556, train acc: 0.9874
epoch: 38, loss: 0.005054045934230089, test acc: 0.7639, train acc: 0.9893
epoch: 39, loss: 0.004426872823387384, test acc: 0.76, train acc: 0.9884
epoch: 40, loss: 0.003879308234900236, test acc: 0.7649, train acc: 0.9887
epoch: 41, loss: 0.005692426115274429, test acc: 0.7609, train acc: 0.9904
epoch: 42, loss: 0.00385400652885437, test acc: 0.7623, train acc: 0.9906
epoch: 43, loss: 0.004982029087841511, test acc: 0.7617, train acc: 0.9901
epoch: 44, loss: 0.00519225001335144, test acc: 0.7609, train acc: 0.9896
epoch: 45, loss: 0.004568636883050203, test acc: 0.7627, train acc: 0.9909
epoch: 46, loss: 0.0042817844077944756, test acc: 0.7628, train acc: 0.9909
epoch: 47, loss: 0.003724011592566967, test acc: 0.7582, train acc: 0.9913
epoch: 48, loss: 0.0026745477225631475, test acc: 0.7639, train acc: 0.993
```

epoch: 49, loss: 0.003749055555090308, test acc: 0.7587, train acc: 0.9899
epoch: 50, loss: 0.0030358373187482357, test acc: 0.7635, train acc: 0.9925
epoch: 51, loss: 0.003378787310793996, test acc: 0.7625, train acc: 0.9913
epoch: 52, loss: 0.003833663184195757, test acc: 0.7665, train acc: 0.9921
epoch: 53, loss: 0.005015059839934111, test acc: 0.7629, train acc: 0.9899
epoch: 54, loss: 0.0028799260035157204, test acc: 0.7633, train acc: 0.9946
epoch: 55, loss: 0.002474428853020072, test acc: 0.7595, train acc: 0.9919
epoch: 56, loss: 0.0037355164531618357, test acc: 0.7623, train acc: 0.9923
epoch: 57, loss: 0.0025193199981004, test acc: 0.7631, train acc: 0.9925
epoch: 58, loss: 0.002228055614978075, test acc: 0.7642, train acc: 0.9934
epoch: 59, loss: 0.0030023884028196335, test acc: 0.7651, train acc: 0.9945
epoch: 60, loss: 0.003395533887669444, test acc: 0.7684, train acc: 0.9931
epoch: 61, loss: 0.0020127310417592525, test acc: 0.7647, train acc: 0.9905
epoch: 62, loss: 0.0019254542421549559, test acc: 0.7672, train acc: 0.9936
epoch: 63, loss: 0.0051668500527739525, test acc: 0.7661, train acc: 0.9921
epoch: 64, loss: 0.0018517057178542018, test acc: 0.7686, train acc: 0.9943
epoch: 65, loss: 0.0019340633880347013, test acc: 0.7696, train acc: 0.9954
epoch: 66, loss: 0.0021100437734276056, test acc: 0.7665, train acc: 0.9937
epoch: 67, loss: 0.0016127193812280893, test acc: 0.7694, train acc: 0.994
epoch: 68, loss: 0.0010641587432473898, test acc: 0.7696, train acc: 0.9958
epoch: 69, loss: 0.0031510433182120323, test acc: 0.7671, train acc: 0.9938
epoch: 70, loss: 0.0026846490800380707, test acc: 0.7722, train acc: 0.9946
epoch: 71, loss: 0.0027267641853541136, test acc: 0.7666, train acc: 0.9938
epoch: 72, loss: 0.0017106859013438225, test acc: 0.7707, train acc: 0.9939
epoch: 73, loss: 0.0029396095778793097, test acc: 0.7693, train acc: 0.9932
epoch: 74, loss: 0.0016160738887265325, test acc: 0.7689, train acc: 0.9947
epoch: 75, loss: 0.0016968444688245654, test acc: 0.7664, train acc: 0.993
epoch: 76, loss: 0.0013293777592480183, test acc: 0.7688, train acc: 0.9944
epoch: 77, loss: 0.0008771327557042241, test acc: 0.7697, train acc: 0.994
epoch: 78, loss: 0.0006159388576634228, test acc: 0.7706, train acc: 0.9946
epoch: 79, loss: 0.0010156116914004087, test acc: 0.7693, train acc: 0.9943
epoch: 80, loss: 0.0011802760418504477, test acc: 0.7686, train acc: 0.994
epoch: 81, loss: 0.0013703290605917573, test acc: 0.7697, train acc: 0.9935
epoch: 82, loss: 0.0008053391356952488, test acc: 0.7724, train acc: 0.9943
epoch: 83, loss: 0.0010806707432493567, test acc: 0.7742, train acc: 0.9944
epoch: 84, loss: 0.0013668467290699482, test acc: 0.7725, train acc: 0.9947
epoch: 85, loss: 0.0011246155481785536, test acc: 0.7681, train acc: 0.9941
epoch: 86, loss: 0.001424836227670312, test acc: 0.775, train acc: 0.9956
epoch: 87, loss: 0.0008899846579879522, test acc: 0.7753, train acc: 0.9954
epoch: 88, loss: 0.0016760416328907013, test acc: 0.7754, train acc: 0.9969
epoch: 89, loss: 0.001816102652810514, test acc: 0.7699, train acc: 0.9958
epoch: 90, loss: 0.0008085351437330246, test acc: 0.769, train acc: 0.9955
epoch: 91, loss: 0.001097466447390616, test acc: 0.7703, train acc: 0.9954
epoch: 92, loss: 0.0009940052404999733, test acc: 0.7718, train acc: 0.9952
epoch: 93, loss: 0.0014265998033806682, test acc: 0.7699, train acc: 0.9951
epoch: 94, loss: 0.0008116996614262462, test acc: 0.7731, train acc: 0.9964
epoch: 95, loss: 0.0020163364242762327, test acc: 0.772, train acc: 0.9956
epoch: 96, loss: 0.0019353991374373436, test acc: 0.7705, train acc: 0.9962

```
epoch: 97, loss: 0.0014067931333556771, test acc: 0.7737, train acc: 0.9969
epoch: 98, loss: 0.000888130918610841, test acc: 0.7652, train acc: 0.9949
epoch: 99, loss: 0.0009489298681728542, test acc: 0.7748, train acc: 0.9963
epoch: 100, loss: 0.0008496436639688909, test acc: 0.7708, train acc: 0.9959
```





The final results of the training with standard augmentation is the following: - test accuracy: 0.7708 - train accuracy: 0.9959 - loss: 0.0008

5 Task - 5

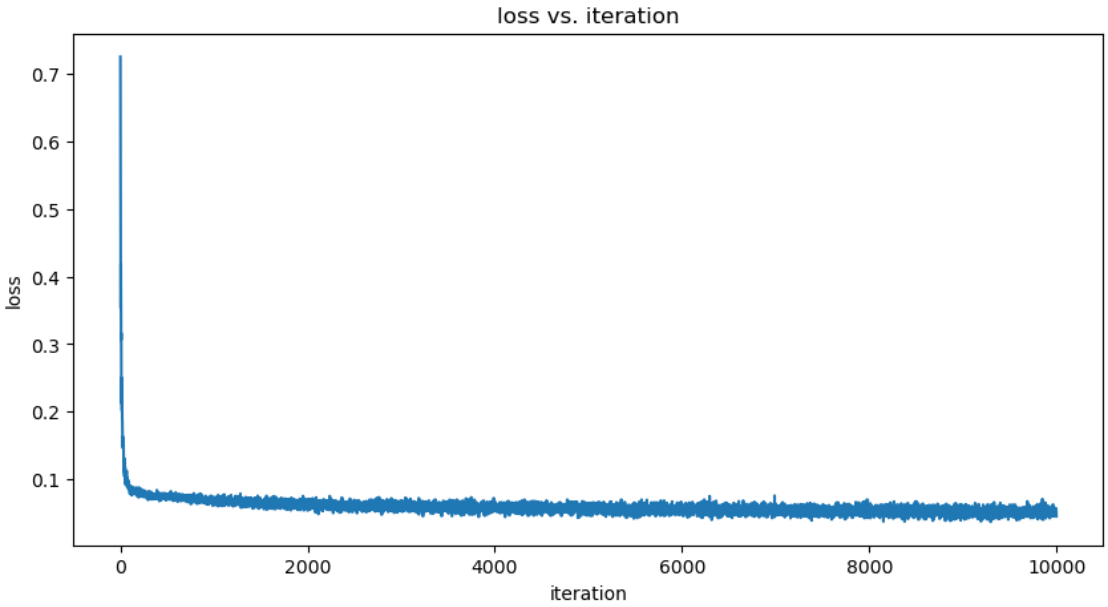
[13]:
```
def combined_augmentation(data_batch, label_batch, alpha, k_cutout=16,␣
↪k_standard=4):
    data_batch, label_batch = standard_augmentation(data_batch, label_batch,␣
↪k=k_standard)
    data_batch, label_batch = cutout(data_batch, label_batch, k=k_cutout)
    return mixup(data_batch, label_batch, alpha)
```

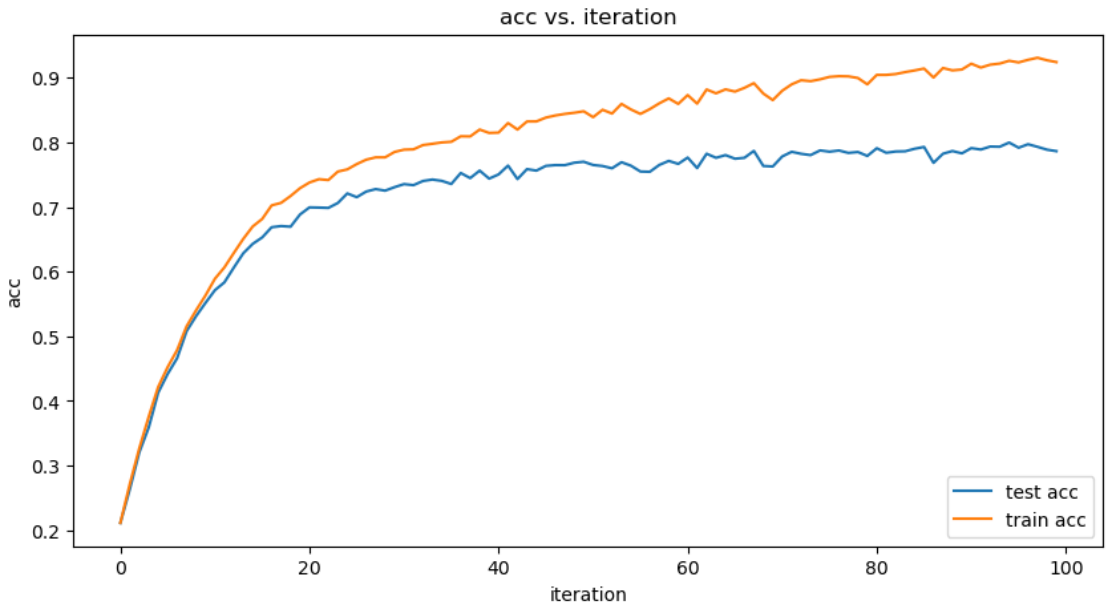
[9]:
```
train_X, train_y, test_X, test_y = get_dataset()
running_loss_combined_2, running_acc_combined_2, running_train_acc_combined_2 =␣
↪train(train_X, train_y, test_X, test_y, augmentation=lambda x, y:␣
↪combined_augmentation(x, y, 0.2))
train_X, train_y, test_X, test_y = get_dataset()
running_loss_combined_4, running_acc_combined_4, running_train_acc_combined_4 =␣
↪train(train_X, train_y, test_X, test_y, augmentation=lambda x, y:␣
↪combined_augmentation(x, y, 0.4))
```

```
Files already downloaded and verified
Files already downloaded and verified
train data: torch.Size([50000, 3, 32, 32]), train label: torch.Size([50000, 10])
test data: torch.Size([10000, 3, 32, 32]), test label: torch.Size([10000])
train_data_sampled: torch.Size([10000, 3, 32, 32]), train_label_sampled:
torch.Size([10000, 10])
train_data_sampled: 100, train_label_sampled: 100
train_data_sampled: torch.Size([100, 3, 32, 32]), train_label_sampled:
torch.Size([100, 10])
epoch: 1, loss: 0.08883503079414368, test acc: 0.2113, train acc: 0.2114
epoch: 2, loss: 0.07770182937383652, test acc: 0.2636, train acc: 0.2715
epoch: 3, loss: 0.08223831653594971, test acc: 0.3205, train acc: 0.3266
epoch: 4, loss: 0.07607375085353851, test acc: 0.3591, train acc: 0.3757
epoch: 5, loss: 0.07492589205503464, test acc: 0.4129, train acc: 0.4217
epoch: 6, loss: 0.07338246703147888, test acc: 0.4418, train acc: 0.4524
epoch: 7, loss: 0.07091926783323288, test acc: 0.4661, train acc: 0.4787
epoch: 8, loss: 0.06984097510576248, test acc: 0.5079, train acc: 0.5154
epoch: 9, loss: 0.06880027055740356, test acc: 0.5311, train acc: 0.5401
epoch: 10, loss: 0.07004135102033615, test acc: 0.5517, train acc: 0.5632
epoch: 11, loss: 0.06636924296617508, test acc: 0.5714, train acc: 0.5891
epoch: 12, loss: 0.0711989775300026, test acc: 0.5835, train acc: 0.6066
epoch: 13, loss: 0.06742154806852341, test acc: 0.6064, train acc: 0.629
epoch: 14, loss: 0.06748563051223755, test acc: 0.6285, train acc: 0.6508
epoch: 15, loss: 0.06578253954648972, test acc: 0.6431, train acc: 0.67
epoch: 16, loss: 0.06498897820711136, test acc: 0.653, train acc: 0.6816
epoch: 17, loss: 0.06798457354307175, test acc: 0.6688, train acc: 0.7026
epoch: 18, loss: 0.06618783622980118, test acc: 0.6706, train acc: 0.7065
epoch: 19, loss: 0.06865017116069794, test acc: 0.6696, train acc: 0.7171
epoch: 20, loss: 0.06294132024049759, test acc: 0.6885, train acc: 0.7292
```

epoch: 21, loss: 0.06447906792163849, test acc: 0.6995, train acc: 0.738
epoch: 22, loss: 0.06474050879478455, test acc: 0.6992, train acc: 0.7431
epoch: 23, loss: 0.06555159389972687, test acc: 0.6987, train acc: 0.7418
epoch: 24, loss: 0.05468203127384186, test acc: 0.7063, train acc: 0.7549
epoch: 25, loss: 0.06521056592464447, test acc: 0.7211, train acc: 0.7583
epoch: 26, loss: 0.061153050512075424, test acc: 0.7153, train acc: 0.7665
epoch: 27, loss: 0.0693720281124115, test acc: 0.724, train acc: 0.7733
epoch: 28, loss: 0.06051688641309738, test acc: 0.7278, train acc: 0.7769
epoch: 29, loss: 0.053024593740701675, test acc: 0.7254, train acc: 0.7769
epoch: 30, loss: 0.061072662472724915, test acc: 0.7308, train acc: 0.7853
epoch: 31, loss: 0.07125304639339447, test acc: 0.7355, train acc: 0.7888
epoch: 32, loss: 0.06307921558618546, test acc: 0.7339, train acc: 0.7893
epoch: 33, loss: 0.0656859427690506, test acc: 0.7403, train acc: 0.7959
epoch: 34, loss: 0.05361231043934822, test acc: 0.7426, train acc: 0.7979
epoch: 35, loss: 0.0625823512673378, test acc: 0.7405, train acc: 0.8
epoch: 36, loss: 0.06633774191141129, test acc: 0.7357, train acc: 0.801
epoch: 37, loss: 0.06827980279922485, test acc: 0.7527, train acc: 0.8095
epoch: 38, loss: 0.065948486328125, test acc: 0.7448, train acc: 0.8093
epoch: 39, loss: 0.06062628701329231, test acc: 0.7564, train acc: 0.8199
epoch: 40, loss: 0.05981326475739479, test acc: 0.744, train acc: 0.8147
epoch: 41, loss: 0.054642681032419205, test acc: 0.7505, train acc: 0.8152
epoch: 42, loss: 0.06306076794862747, test acc: 0.7642, train acc: 0.83
epoch: 43, loss: 0.06181052699685097, test acc: 0.7433, train acc: 0.8199
epoch: 44, loss: 0.06381070613861084, test acc: 0.7587, train acc: 0.8324
epoch: 45, loss: 0.06275196373462677, test acc: 0.7564, train acc: 0.8325
epoch: 46, loss: 0.046050477772951126, test acc: 0.7637, train acc: 0.8384
epoch: 47, loss: 0.06202676519751549, test acc: 0.7649, train acc: 0.8417
epoch: 48, loss: 0.059341173619031906, test acc: 0.7648, train acc: 0.8441
epoch: 49, loss: 0.05554230511188507, test acc: 0.7686, train acc: 0.8459
epoch: 50, loss: 0.05706915631890297, test acc: 0.7701, train acc: 0.8482
epoch: 51, loss: 0.053772080689668655, test acc: 0.7651, train acc: 0.8391
epoch: 52, loss: 0.055133357644081116, test acc: 0.7635, train acc: 0.8505
epoch: 53, loss: 0.06598980724811554, test acc: 0.76, train acc: 0.8447
epoch: 54, loss: 0.06078745424747467, test acc: 0.7694, train acc: 0.8596
epoch: 55, loss: 0.06341523677110672, test acc: 0.764, train acc: 0.851
epoch: 56, loss: 0.06517845392227173, test acc: 0.755, train acc: 0.8441
epoch: 57, loss: 0.05495835468173027, test acc: 0.7546, train acc: 0.8516
epoch: 58, loss: 0.05383579432964325, test acc: 0.7652, train acc: 0.8605
epoch: 59, loss: 0.05846569314599037, test acc: 0.7714, train acc: 0.8681
epoch: 60, loss: 0.06238729879260063, test acc: 0.7668, train acc: 0.8596
epoch: 61, loss: 0.05600805580615997, test acc: 0.7766, train acc: 0.8733
epoch: 62, loss: 0.06380461156368256, test acc: 0.7603, train acc: 0.86
epoch: 63, loss: 0.048900093883275986, test acc: 0.7825, train acc: 0.8819
epoch: 64, loss: 0.054884497076272964, test acc: 0.7762, train acc: 0.876
epoch: 65, loss: 0.06315581500530243, test acc: 0.7804, train acc: 0.8821
epoch: 66, loss: 0.05615776777267456, test acc: 0.7748, train acc: 0.8787
epoch: 67, loss: 0.060679975897073746, test acc: 0.7762, train acc: 0.8844
epoch: 68, loss: 0.05413272604346275, test acc: 0.7869, train acc: 0.8917

```
epoch: 69, loss: 0.05168966203927994, test acc: 0.7636, train acc: 0.8756
epoch: 70, loss: 0.05176544561982155, test acc: 0.7628, train acc: 0.8655
epoch: 71, loss: 0.05967666208744049, test acc: 0.7783, train acc: 0.88
epoch: 72, loss: 0.06210869550704956, test acc: 0.7855, train acc: 0.8899
epoch: 73, loss: 0.056496601551771164, test acc: 0.7824, train acc: 0.8962
epoch: 74, loss: 0.059599388390779495, test acc: 0.7804, train acc: 0.8948
epoch: 75, loss: 0.054474055767059326, test acc: 0.7876, train acc: 0.8976
epoch: 76, loss: 0.05610758066177368, test acc: 0.7857, train acc: 0.9014
epoch: 77, loss: 0.06029016897082329, test acc: 0.7875, train acc: 0.9025
epoch: 78, loss: 0.06213924661278725, test acc: 0.7838, train acc: 0.9022
epoch: 79, loss: 0.05507457256317139, test acc: 0.7852, train acc: 0.8997
epoch: 80, loss: 0.056847985833883286, test acc: 0.7791, train acc: 0.8899
epoch: 81, loss: 0.05722690746188164, test acc: 0.791, train acc: 0.9045
epoch: 82, loss: 0.053345248103141785, test acc: 0.784, train acc: 0.9045
epoch: 83, loss: 0.05960209295153618, test acc: 0.786, train acc: 0.9058
epoch: 84, loss: 0.056801654398441315, test acc: 0.7863, train acc: 0.9088
epoch: 85, loss: 0.05238722637295723, test acc: 0.7902, train acc: 0.9114
epoch: 86, loss: 0.05825529992580414, test acc: 0.793, train acc: 0.9143
epoch: 87, loss: 0.06177987530827522, test acc: 0.7684, train acc: 0.9004
epoch: 88, loss: 0.05399254336953163, test acc: 0.7825, train acc: 0.9151
epoch: 89, loss: 0.05960458889603615, test acc: 0.7867, train acc: 0.9115
epoch: 90, loss: 0.052845247089862823, test acc: 0.7831, train acc: 0.9128
epoch: 91, loss: 0.05367514118552208, test acc: 0.7912, train acc: 0.9218
epoch: 92, loss: 0.054020144045352936, test acc: 0.789, train acc: 0.9158
epoch: 93, loss: 0.055617716163396835, test acc: 0.7936, train acc: 0.9203
epoch: 94, loss: 0.06518898159265518, test acc: 0.7933, train acc: 0.922
epoch: 95, loss: 0.06253798305988312, test acc: 0.7998, train acc: 0.9263
epoch: 96, loss: 0.05637548491358757, test acc: 0.7918, train acc: 0.9238
epoch: 97, loss: 0.04724191129207611, test acc: 0.7972, train acc: 0.9278
epoch: 98, loss: 0.05630214139819145, test acc: 0.7932, train acc: 0.931
epoch: 99, loss: 0.05126243829727173, test acc: 0.7888, train acc: 0.9271
epoch: 100, loss: 0.045108139514923096, test acc: 0.7865, train acc: 0.9243
```



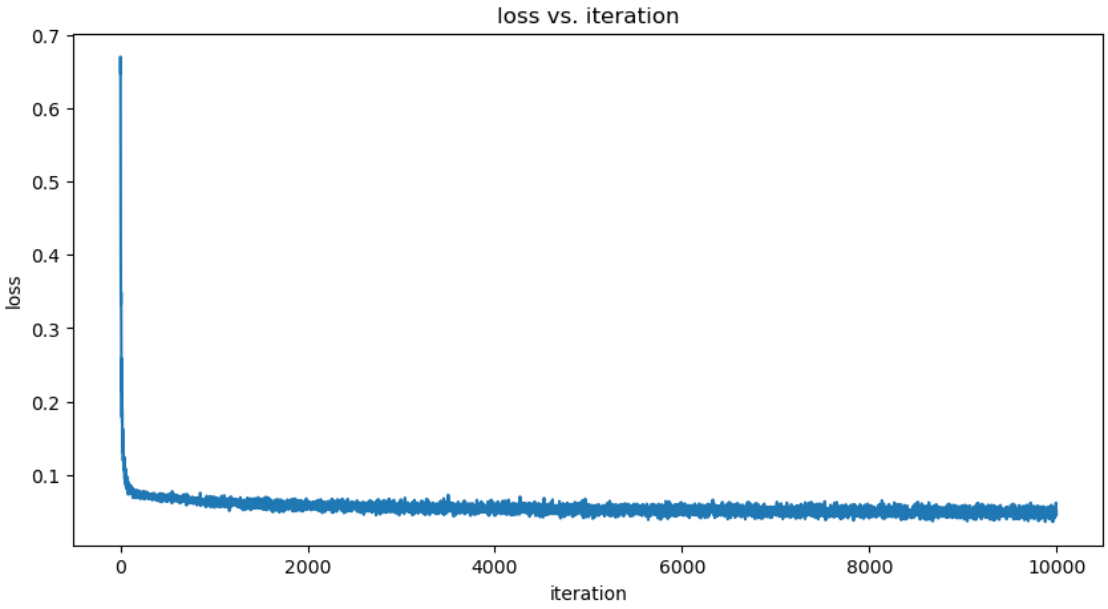


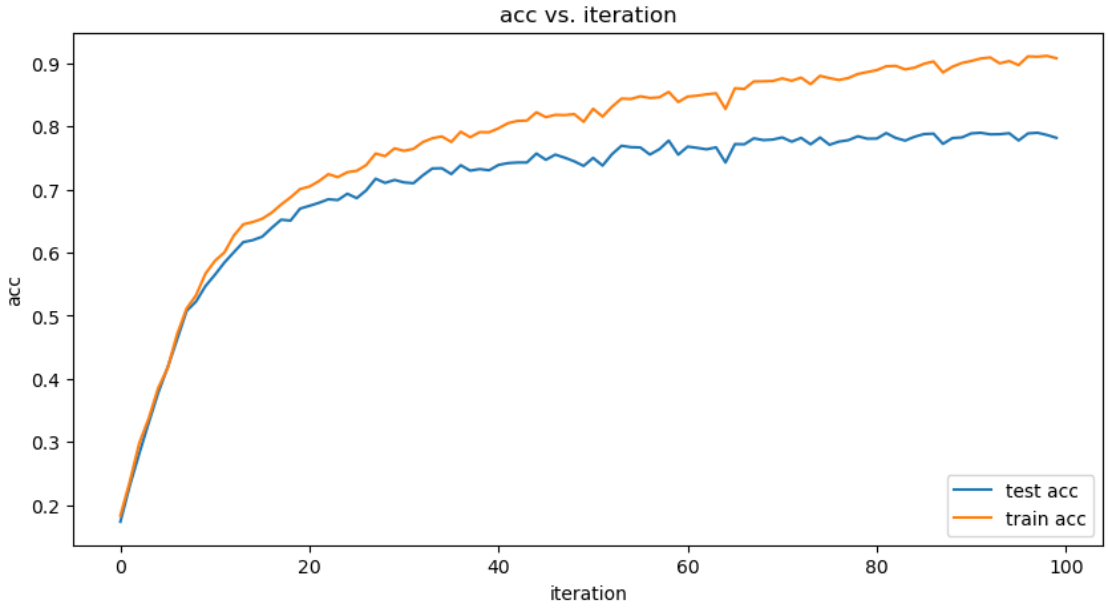

```
Files already downloaded and verified
Files already downloaded and verified
train data: torch.Size([50000, 3, 32, 32]), train label: torch.Size([50000, 10])
test data: torch.Size([10000, 3, 32, 32]), test label: torch.Size([10000])
train_data_sampled: torch.Size([10000, 3, 32, 32]), train_label_sampled:
torch.Size([10000, 10])
train_data_sampled: 100, train_label_sampled: 100
```

```
train_data_sampled: torch.Size([100, 3, 32, 32]), train_label_sampled:
torch.Size([100, 10])
epoch: 1, loss: 0.08228155970573425, test acc: 0.1736, train acc: 0.1833
epoch: 2, loss: 0.07363782823085785, test acc: 0.2311, train acc: 0.2375
epoch: 3, loss: 0.07395341992378235, test acc: 0.2817, train acc: 0.2982
epoch: 4, loss: 0.07137056440114975, test acc: 0.3307, train acc: 0.3375
epoch: 5, loss: 0.06691642850637436, test acc: 0.3779, train acc: 0.3855
epoch: 6, loss: 0.06868121773004532, test acc: 0.4181, train acc: 0.4171
epoch: 7, loss: 0.0687163695693016, test acc: 0.4624, train acc: 0.4706
epoch: 8, loss: 0.060711588710546494, test acc: 0.5075, train acc: 0.5112
epoch: 9, loss: 0.06378047913312912, test acc: 0.5223, train acc: 0.5323
epoch: 10, loss: 0.06415041536092758, test acc: 0.5469, train acc: 0.5666
epoch: 11, loss: 0.06211591511964798, test acc: 0.5648, train acc: 0.5868
epoch: 12, loss: 0.059435151517391205, test acc: 0.5841, train acc: 0.6001
epoch: 13, loss: 0.061488550156354904, test acc: 0.6005, train acc: 0.6268
epoch: 14, loss: 0.055039506405591965, test acc: 0.6165, train acc: 0.6447
epoch: 15, loss: 0.05759706348180771, test acc: 0.6196, train acc: 0.6483
epoch: 16, loss: 0.05805952847003937, test acc: 0.6251, train acc: 0.6535
epoch: 17, loss: 0.04845019802451134, test acc: 0.639, train acc: 0.6631
epoch: 18, loss: 0.05317457765340805, test acc: 0.652, train acc: 0.676
epoch: 19, loss: 0.053585004061460495, test acc: 0.6506, train acc: 0.6874
epoch: 20, loss: 0.05792556330561638, test acc: 0.6695, train acc: 0.7005
epoch: 21, loss: 0.057512179017066956, test acc: 0.674, train acc: 0.7045
epoch: 22, loss: 0.05316260829567909, test acc: 0.6787, train acc: 0.7131
epoch: 23, loss: 0.05203010141849518, test acc: 0.6844, train acc: 0.7241
epoch: 24, loss: 0.05354033038020134, test acc: 0.6831, train acc: 0.7194
epoch: 25, loss: 0.05658300593495369, test acc: 0.6931, train acc: 0.7272
epoch: 26, loss: 0.045337021350860596, test acc: 0.6862, train acc: 0.7295
epoch: 27, loss: 0.050524402409791946, test acc: 0.6985, train acc: 0.7385
epoch: 28, loss: 0.05246048048138618 5, test acc: 0.7169, train acc: 0.7568
epoch: 29, loss: 0.05490614473819733, test acc: 0.7103, train acc: 0.7527
epoch: 30, loss: 0.0586620457470417, test acc: 0.715, train acc: 0.765
epoch: 31, loss: 0.05870237946510315, test acc: 0.7111, train acc: 0.7611
epoch: 32, loss: 0.049904964864254, test acc: 0.7098, train acc: 0.7642
epoch: 33, loss: 0.05839971452951431, test acc: 0.7224, train acc: 0.775
epoch: 34, loss: 0.04644785821437836, test acc: 0.7332, train acc: 0.781
epoch: 35, loss: 0.05277642980217934, test acc: 0.7335, train acc: 0.7839
epoch: 36, loss: 0.05657822638750076, test acc: 0.7243, train acc: 0.7751
epoch: 37, loss: 0.05056845024228096, test acc: 0.7383, train acc: 0.7914
epoch: 38, loss: 0.058203279972076416, test acc: 0.7297, train acc: 0.7826
epoch: 39, loss: 0.054858773946762085, test acc: 0.7322, train acc: 0.7907
epoch: 40, loss: 0.044354118406772614, test acc: 0.7303, train acc: 0.7905
epoch: 41, loss: 0.04829545319080353, test acc: 0.7388, train acc: 0.7968
epoch: 42, loss: 0.05247977003455162, test acc: 0.7416, train acc: 0.8049
epoch: 43, loss: 0.053655099123716354, test acc: 0.7426, train acc: 0.8085
epoch: 44, loss: 0.05170534923672676, test acc: 0.7426, train acc: 0.8091
epoch: 45, loss: 0.05159560963511467, test acc: 0.7568, train acc: 0.8221
epoch: 46, loss: 0.053280849009752274, test acc: 0.7469, train acc: 0.8144
```

epoch: 47, loss: 0.05355805903673172, test acc: 0.7551, train acc: 0.8181
epoch: 48, loss: 0.05687672644853592, test acc: 0.7502, train acc: 0.8177
epoch: 49, loss: 0.05540113151073456, test acc: 0.7445, train acc: 0.8192
epoch: 50, loss: 0.04862458258867264, test acc: 0.7372, train acc: 0.8071
epoch: 51, loss: 0.05584293603897095, test acc: 0.7499, train acc: 0.8277
epoch: 52, loss: 0.0577007532119751, test acc: 0.7377, train acc: 0.8153
epoch: 53, loss: 0.05296028405427933, test acc: 0.7558, train acc: 0.8309
epoch: 54, loss: 0.05160136520862579, test acc: 0.7691, train acc: 0.8438
epoch: 55, loss: 0.04961596429347992, test acc: 0.7669, train acc: 0.8432
epoch: 56, loss: 0.053086958825588226, test acc: 0.7664, train acc: 0.8475
epoch: 57, loss: 0.050577521324157715, test acc: 0.7552, train acc: 0.8448
epoch: 58, loss: 0.04972139000892639, test acc: 0.7636, train acc: 0.8458
epoch: 59, loss: 0.050269290804862976, test acc: 0.7773, train acc: 0.8544
epoch: 60, loss: 0.05917983502149582, test acc: 0.755, train acc: 0.8384
epoch: 61, loss: 0.052001845091581345, test acc: 0.7678, train acc: 0.847
epoch: 62, loss: 0.053788572549819946, test acc: 0.7658, train acc: 0.8484
epoch: 63, loss: 0.051329128444194794, test acc: 0.7635, train acc: 0.8506
epoch: 64, loss: 0.05161783844232559, test acc: 0.7666, train acc: 0.8522
epoch: 65, loss: 0.048067476600408554, test acc: 0.7426, train acc: 0.8274
epoch: 66, loss: 0.0479382649064064, test acc: 0.7718, train acc: 0.8602
epoch: 67, loss: 0.052082620561122894, test acc: 0.7715, train acc: 0.8592
epoch: 68, loss: 0.04531478509306908, test acc: 0.7809, train acc: 0.871
epoch: 69, loss: 0.049809832125902176, test acc: 0.7781, train acc: 0.8712
epoch: 70, loss: 0.05246725678443909, test acc: 0.7789, train acc: 0.8717
epoch: 71, loss: 0.05076068639755249, test acc: 0.7823, train acc: 0.8758
epoch: 72, loss: 0.051228493452072144, test acc: 0.7756, train acc: 0.8721
epoch: 73, loss: 0.047669366002082825, test acc: 0.7816, train acc: 0.877
epoch: 74, loss: 0.046460382640361786, test acc: 0.7714, train acc: 0.8666
epoch: 75, loss: 0.05153299868106842, test acc: 0.7822, train acc: 0.8799
epoch: 76, loss: 0.052546482533216476, test acc: 0.7707, train acc: 0.8762
epoch: 77, loss: 0.0487208366394043, test acc: 0.7757, train acc: 0.8733
epoch: 78, loss: 0.04498043283820152, test acc: 0.778, train acc: 0.8764
epoch: 79, loss: 0.04891689866781235, test acc: 0.7842, train acc: 0.8828
epoch: 80, loss: 0.040531057864427567, test acc: 0.7803, train acc: 0.8859
epoch: 81, loss: 0.04877858981490135, test acc: 0.7805, train acc: 0.8889
epoch: 82, loss: 0.049054477363824844, test acc: 0.7892, train acc: 0.895
epoch: 83, loss: 0.054230764508247375, test acc: 0.7815, train acc: 0.8956
epoch: 84, loss: 0.05456836149096489, test acc: 0.7773, train acc: 0.8901
epoch: 85, loss: 0.04776258394122124, test acc: 0.7834, train acc: 0.8928
epoch: 86, loss: 0.05428894981741905, test acc: 0.7876, train acc: 0.899
epoch: 87, loss: 0.04707546532154083, test acc: 0.7883, train acc: 0.9026
epoch: 88, loss: 0.05148984119296074, test acc: 0.7721, train acc: 0.8852
epoch: 89, loss: 0.0512889139354229, test acc: 0.7813, train acc: 0.8943
epoch: 90, loss: 0.046312782913446426, test acc: 0.7823, train acc: 0.9002
epoch: 91, loss: 0.04495115950703621, test acc: 0.7887, train acc: 0.9034
epoch: 92, loss: 0.04747500270605087, test acc: 0.7897, train acc: 0.9073
epoch: 93, loss: 0.04671771451830864, test acc: 0.7872, train acc: 0.909
epoch: 94, loss: 0.04937950521707535, test acc: 0.7874, train acc: 0.8996

```
epoch: 95, loss: 0.04511893540620804, test acc: 0.789, train acc: 0.9033
epoch: 96, loss: 0.0461997464299202, test acc: 0.7775, train acc: 0.8969
epoch: 97, loss: 0.03954138606786728, test acc: 0.7887, train acc: 0.9108
epoch: 98, loss: 0.05749434977769852, test acc: 0.7897, train acc: 0.9101
epoch: 99, loss: 0.04236231744289398, test acc: 0.7862, train acc: 0.9115
epoch: 100, loss: 0.0495268739759922, test acc: 0.7817, train acc: 0.9077
```





The final results of the training with all augmentations (alpha = 0.2) is the following: - test accuracy: 0.7865 - train accuracy: 0.9243 - loss: 0.04

The final results of the training with all augmentations (alpha = 0.4) is the following: - test accuracy: 0.7817 - train accuracy: 0.9077 - loss: 0.04

Combining the augmentations increased the test accuracy for the training. Also, augmentations reduce the gap between train accuracy and test accuracy. Based on the convergence of the loss, it similar to the convergence with other augmentation methods which is very sharp.

6 Task 6

Based on the results the convergence of the loss is similar in the experiments with mixup and standard augmentation and without augmentation experiment. However, for the cutout and combined experiments the loss is not close as other experiments. It could be because the cutout method remove some information from the image. Also, we can see this in the training accuracy. The training accuracy is not good as other experiments in cutout augmentation experiment. However, even if the training accuracy is dropped and the loss is increased in the cutout augmentation it works better than the experiment without augmentation. Also, when applying all the augmentation techniques the final test accuracy is increased.