Includes and network parameters

```
import time
import shutil
import numpy as np
import torch
import torch.nn as nn
from torch.utils.data import TensorDataset, DataLoader
import matplotlib.pyplot as plt
from matplotlib import rcParams
```

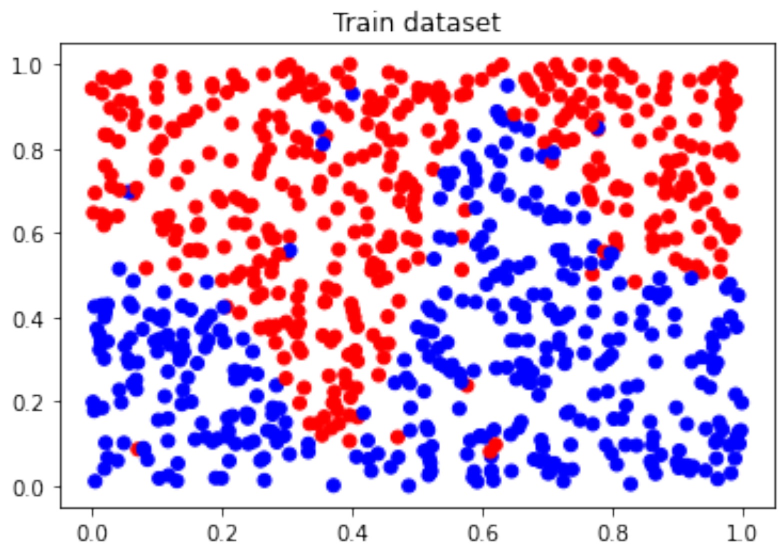
```
# Network parameters
'''
num_neurons : TYPE list
            DESCRIPTION. list of neurons in each layer.
                This should have a minimum length of 3.
                First element represents the dimension of input vector.
                Last element represents the dimension of the output vector.
                middle elements represent the number of neurons in hidden layers.
activations : TYPE, option list where each element can be either 'relu' or 'sigmoic
            DESCRIPTION. The default is ['relu'].
            If len(activations)==1:
                same activation function is applied across all hidden layers.
            else:
                len(activations) should be equal to the number of hidden layers.
'''
num_neurons = [2,20,10,10,2] # list of neurons in each layer of NN.
activations = ['relu'] # represents the activation function used at the hidden laye

# optimizer parameters
lr = 0.01
lr_step = [500]
weight_decay = 1e-3

#  training parameters
num_epochs = 200
batch_size = 256

#
print_freq = 10

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
```

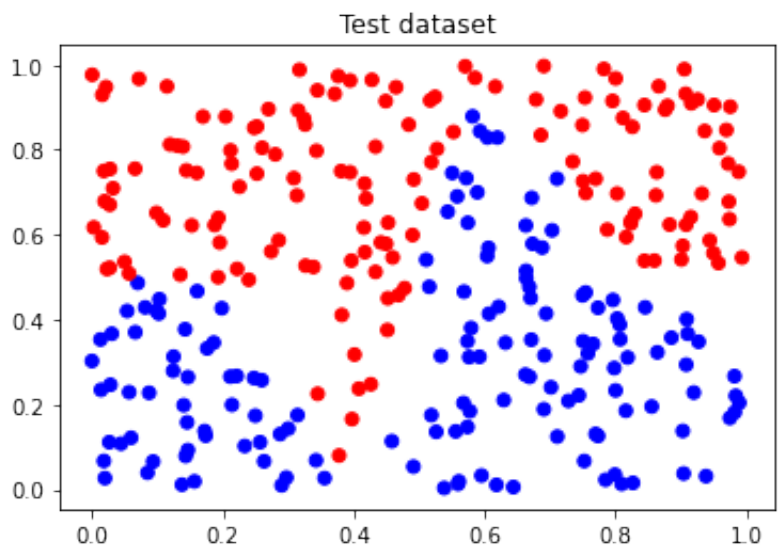
Create and plot data set

Do not change this cell!

```
# DO NOT change this cell.
ns = 800
np.random.seed(0)
X_train = np.random.rand(ns,2)
x1 = X_train[:,0]
x2 = X_train[:,1]
```

```
y_train = ((np.exp(-((x1-0.5)*6)**2)*2*((x1-0.5)*6)+1)/2-x2)>0

idx = np.random.choice(range(ns),size=(int(ns*0.03),))
y_train[idx] = ~y_train[idx]

ns = 300
np.random.seed(1)
X_val = np.random.rand(ns,2)
x1 = X_val[:,0]
x2 = X_val[:,1]
y_val = ((np.exp(-((x1-0.5)*6)**2)*2*((x1-0.5)*6)+1)/2-x2)>0

def plot(X,y,title="Dataset"):
    colors = np.where(y==0, 'r', 'b')
    plt.figure()
    plt.scatter(X[:,0],X[:,1],color=colors)
    plt.title(title)
    plt.show()

plot(X_train,y_train,"Train dataset")
plot(X_val,y_val,"Test dataset")
```





Load data set into Torch dataloader

```
X_train_tensor = torch.Tensor(X_train) # transform to torch tensor
y_train_tensor = torch.Tensor(y_train)

train_dataset = TensorDataset(X_train_tensor,y_train_tensor) # create your datset
train_loader = DataLoader(train_dataset,batch_size=batch_size,shuffle=True,drop_las

X_val_tensor = torch.Tensor(X_val) # transform to torch tensor
y_val_tensor = torch.Tensor(y_val)

val_dataset = TensorDataset(X_val_tensor,y_val_tensor) # create your datset
val_loader = DataLoader(val_dataset,batch_size=batch_size,shuffle=True,drop_last=Fa
```

Model: Feedforward neural network

```
class LinearNN(nn.Module):

    def __init__(self,num_neurons,activations=['relu']):
        '''
        Parameters
        ----------
        num_neurons : TYPE list
            DESCRIPTION. list of neurons in each layer.
                This should have a minimum length of 3.
                First element represents the dimension of input vector.
                Last element represents the dimension of the output vector.
                middle elements represent the number of neurons in hidden layers.

        activations : TYPE, optional list.
            DESCRIPTION. The default is ['relu'].
            If len(activations)==1:
                same activation function is applied across all hidden layers.
            else:
                len(actiavtions) should be equal to the number of hidden layers.

        Returns
        -------
        None.

        '''

        super(LinearNN,self).__init__()
        assert isinstance(num_neurons,list)
```

```
        assert np.all([isinstance(neurons,int) for neurons in num_neurons])
        assert np.all([neurons>=1 for neurons in num_neurons])
        assert len(num_neurons)>=3
        if activations is not None:
            assert isinstance(activations,(list))
            assert (len(activations)==len(num_neurons)-2) or (len(activations)==1)


        def activation_layer(act_func):
            '''



            Parameters
            ----------
            act_func : TYPE should be one from {'relu','sigmoid','tanh'}.
                DESCRIPTION.

            Raises
            ------
            NotImplementedError
                DESCRIPTION.

            Returns
            -------
            TYPE
                DESCRIPTION.

            '''
            if act_func=='relu':
                return nn.ReLU(inplace=True)
            elif act_func=='sigmoid':
                return nn.Sigmoid()
            elif act_func=='tanh':
                return nn.Tanh()
            else:
                raise NotImplementedError


        layers = []
        for idx,_ in enumerate(num_neurons[:-1]):
            layers.append(nn.Linear(in_features=num_neurons[idx],
                                    out_features=num_neurons[idx+1],
                                    bias=True))

            if idx!=len(num_neurons)-2: # add activation for all layers except the
```

```
                if len(activations)==1:
                    layers.append(activation_layer(activations[0]))
                else:
                    layers.append(activation_layer(activations[idx]))

        self.network = nn.Sequential(*layers)


    def forward(self,x):
        x = self.network(x)
        return x


def linear_nn(num_neurons,activations=['relu']):
    model = LinearNN(num_neurons,activations)
    return model
```

Define training function

```
def train(train_loader, model, criterion, optimizer, epoch):
    batch_time = AverageMeter()
    data_time = AverageMeter()
    losses = AverageMeter()
    top1 = AverageMeter()

    # switch to train mode
    model.train()

    end = time.time()
    for i, (input, target) in enumerate(train_loader):
        # measure data loading time
        data_time.update(time.time() - end)

        target = target.to(device)
        input_var = torch.autograd.Variable(input).to(device)
        target_var = torch.autograd.Variable(target).to(device)
        # target_var = torch.squeeze(target_var)
        # compute output
        output = model(input_var)

        # compute loss
```

```
        loss = criterion(output, target_var.long())

        # measure accuracy and record loss
        prec1 = accuracy(output.data, target)
        losses.update(loss.item(), input.size(0))
        top1.update(prec1[0][0], input.size(0))


        # compute gradient and do SGD step
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

        # measure elapsed time
        batch_time.update(time.time() - end)
        end = time.time()

        if i % print_freq == 0:
            curr_lr = optimizer.param_groups[0]['lr']
            print('Epoch: [{0}/{1}][{2}/{3}]\t'
                  'LR: {4}\t'
                  'Loss {loss.val:.4f} ({loss.avg:.4f})\t'
                  'Train Acc {top1.val:.3f} ({top1.avg:.3f})'.format(
                   epoch, num_epochs, i, len(train_loader), curr_lr,
                   loss=losses, top1=top1))


    print(' * Train Acc {top1.avg:.3f}'.format(top1=top1))
    # compute and return train loss and accuracy
    train_loss = losses.avg
    train_acc = top1.avg
    return train_loss, train_acc
```

Define validation and prediction functions

```
def validate(val_loader, model, criterion):
    batch_time = AverageMeter()
    losses = AverageMeter()
    top1 = AverageMeter()

    # switch to evaluate mode
    model.eval()
```

```
    end = time.time()
    for i, (input, target) in enumerate(val_loader):
        target = target.to(device)
        input_var = torch.autograd.Variable(input, volatile=True).to(device)
        target_var = torch.autograd.Variable(target, volatile=True).to(device)

        # compute output
        output = model(input_var)
        # loss = criterion(output, target_var[:,None])
        loss = criterion(output, target_var.long())

        # measure accuracy and record loss
        prec1 = accuracy(output.data, target)
        losses.update(loss.item(), input.size(0))
        top1.update(prec1[0][0], input.size(0))

        # measure elapsed time
        batch_time.update(time.time() - end)
        end = time.time()

        if i % print_freq == 0:
            print('Test: [{0}/{1}]\t'
                  'Loss {loss.val:.4f} ({loss.avg:.4f})\t'
                  'Prec@1 {top1.val:.3f} ({top1.avg:.3f})'.format(
                   i, len(val_loader), loss=losses,
                   top1=top1))


    print(' * Test Acc {top1.avg:.3f}'.format(top1=top1))

    return top1.avg

def predict(dataloader,model):
    y_pred = []
    y_true = []
    x = []
    with torch.no_grad():
        for i, (input, target) in enumerate(dataloader):
            # target = target.to(device)
            input_var = torch.autograd.Variable(input, volatile=True).to(device)
            # target_var = torch.autograd.Variable(target, volatile=True).to(device

            # compute output
            output = model(input_var)
```

```
            labels = torch.argmax(output,axis=1)
            y_pred.extend(list(labels.data.detach().cpu().numpy()))
            y_true.extend(list(target.numpy()))
            x.extend(list(input_var.data.detach().cpu().numpy()))
    return np.array(x),np.array(y_true),np.array(y_pred)
```

Function to plot the decision boundary of the neural network

```
def plot_decision_boundary(model):
    h = 0.005
    x_min, x_max = 0,1
    y_min, y_max = 0,1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                         np.arange(y_min, y_max, h))
    x1 = xx.ravel()
    x2 = yy.ravel()
    y = ((np.exp(-((x1-0.5)*6)**2)*2*((x1-0.5)*6)+1)/2-x2)>0

    X_train_tensor = torch.Tensor(np.c_[xx.ravel(), yy.ravel()]) # transform to tor
    y_train_tensor = torch.Tensor(y)

    dataset = TensorDataset(X_train_tensor,y_train_tensor) # create your datset
    dataloader = DataLoader(dataset,batch_size=batch_size,shuffle=False,drop_last=F


    x,y_true,y_pred = predict(dataloader,model)
    Z = y_pred.reshape(xx.shape)
    plt.figure()
    plt.contourf(x[:,0].reshape(xx.shape), x[:,1].reshape(xx.shape), Z, cmap=plt.cm
    plt.axis('tight')

    # scatter plot of data points with colors corresponding to the correct labels.
    ns = 500
    np.random.seed(0)
    X_test = np.random.rand(ns,2)
    x1 = X_test[:,0]
    x2 = X_test[:,1]
    y_test = ((np.exp(-((x1-0.5)*6)**2)*2*((x1-0.5)*6)+1)/2-x2)>0
    colors = np.where(y_test==0, 'r', 'b')
    plt.scatter(x1,x2,color=colors)
    # plt.scatter(x[:,0],x[:,1],colors=)
    plt.show()
```

Functions to track the model performance and save the desired model state

```
def save_checkpoint(state, is_best, filename='checkpoint.pth.tar'):
    torch.save(state, filename)
    if is_best:
        shutil.copyfile(filename, 'model_best.pth.tar')


class AverageMeter(object):
    """Computes and stores the average and current value"""
    def __init__(self):
        self.reset()

    def reset(self):
        self.val = 0
        self.avg = 0
        self.sum = 0
        self.count = 0

    def update(self, val, n=1):
        self.val = val
        self.sum += val * n
        self.count += n
        self.avg = self.sum / self.count


def accuracy(output, target, topk=(1,)):
    """Computes the precision@k for the specified values of k"""
    maxk = max(topk)
    batch_size = target.size(0)

    _, pred = output.topk(maxk, 1, True, True)
    pred = pred.t()
    correct = pred.eq(target.view(1, -1).expand_as(pred))

    res = []
    for k in topk:
        correct_k = correct[:k].view(-1).float().sum(0, keepdim=True)
        res.append(correct_k.mul_(100.0 / batch_size))
    return res
```

Create model instance; define loss function and optimizer

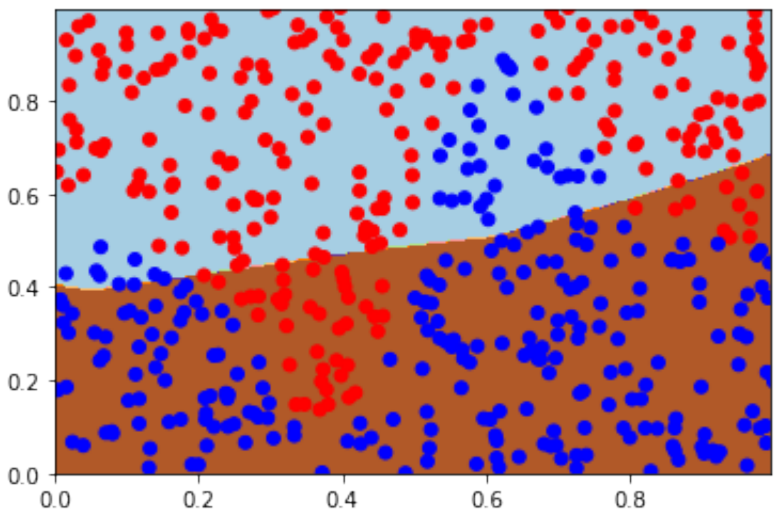
```
torch.manual_seed(999)
model = linear_nn(num_neurons,activations).to(device)

# define loss function (criterion) and optimizer
# criterion = nn.BCEWithLogitsLoss().to(device)
criterion = nn.CrossEntropyLoss().to(device)

optimizer = torch.optim.Adam(model.parameters(),lr=lr,weight_decay=weight_decay)
```

Train model and validate

```
train_losses·=·[]
train_accuracies·=·[]
test_accuracies·=·[]

best_prec1·=·0
for·epoch·in·range(num_epochs):
····if·epoch·in·lr_step:
········for·param_group·in·optimizer.param_groups:
············param_group['lr']·*=·0.1

····#·train·for·one·epoch
····#·train·for·one·epoch
····train_loss,·train_acc·=·train(train_loader,·model,·criterion,·optimizer,·epoch)
····train_losses.append(train_loss)
····train_accuracies.append(train_acc)
··
····#·evaluate·on·validation·set
····#·prec1·=·0
····prec1·=·validate(val_loader,·model,·criterion)
····test_accuracies.append(prec1)

····#·remember·best·prec@1·and·save·checkpoint
····is_best·=·prec1·>·best_prec1
····best_prec1·=·max(prec1,·best_prec1)
····save_checkpoint({
········'epoch':·epoch·+·1,
········'state_dict':·model.state_dict(),
········'best_prec1':·best_prec1,
········'optimizer':·optimizer.state_dict(),
····},·is_best,filename="checkpoint.pth.tar")
```

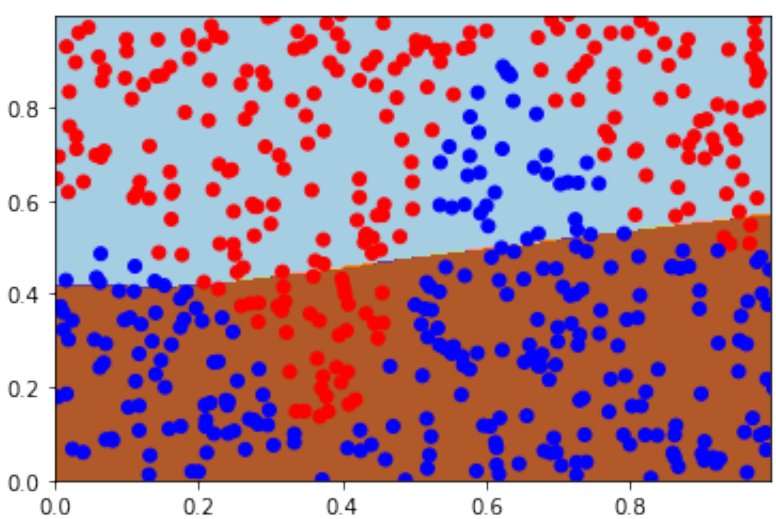
```
····print("-----------------------------------------------")
····
····if·epoch%print_freq==0:
········plot_decision_boundary(model)

plot_decision_boundary(model)
```

```
Epoch: [0/200][0/4]     LR: 0.01        Loss 0.6985 (0.6985)    Train Acc 51.5
 * Train Acc 52.750
Test: [0/2]     Loss 0.6820 (0.6820)    Prec@1 83.594 (83.594)
 * Test Acc 83.000
-----------------------------------------------
<ipython-input-7-56ce71016660>:12: UserWarning: volatile was removed and now h
  input_var = torch.autograd.Variable(input, volatile=True).to(device)
<ipython-input-7-56ce71016660>:13: UserWarning: volatile was removed and now h
  target_var = torch.autograd.Variable(target, volatile=True).to(device)
<ipython-input-7-56ce71016660>:48: UserWarning: volatile was removed and now h
  input_var = torch.autograd.Variable(input, volatile=True).to(device)
```



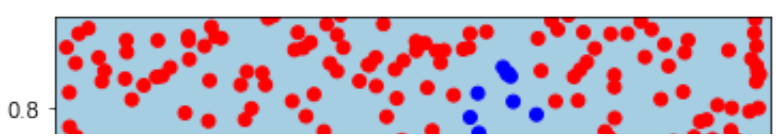

```
Epoch: [1/200][0/4]     LR: 0.01        Loss 0.6831 (0.6831)    Train Acc 81.2
 * Train Acc 78.250
Test: [0/2]     Loss 0.6595 (0.6595)    Prec@1 77.344 (77.344)
 * Test Acc 75.333
-----------------------------------------------
Epoch: [2/200][0/4]     LR: 0.01        Loss 0.6630 (0.6630)    Train Acc 73.0
 * Train Acc 75.875
Test: [0/2]     Loss 0.6293 (0.6293)    Prec@1 78.125 (78.125)
 * Test Acc 78.667
-----------------------------------------------
Epoch: [3/200][0/4]     LR: 0.01        Loss 0.6298 (0.6298)    Train Acc 74.6
 * Train Acc 76.500
Test: [0/2]     Loss 0.5832 (0.5832)    Prec@1 77.344 (77.344)
 * Test Acc 78.333
-----------------------------------------------
Epoch: [4/200][0/4]     LR: 0.01        Loss 0.5832 (0.5832)    Train Acc 75.0
 * Train Acc 76.500
```

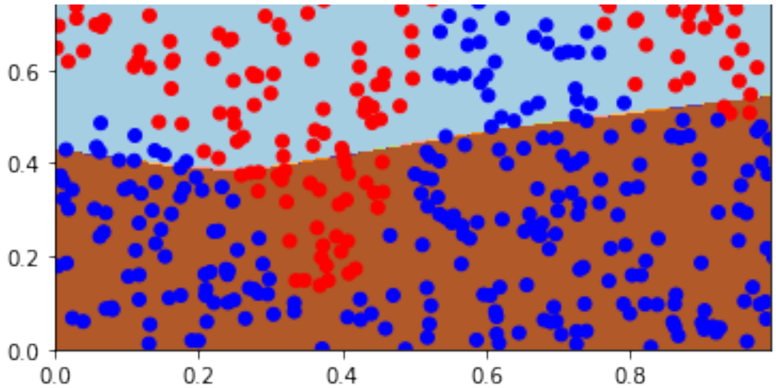
```
Test: [0/2]     Loss 0.5181 (0.5181)    Prec@1 79.688 (79.688)
 * Test Acc 80.333
---------------------------------------------
Epoch: [5/200][0/4]     LR: 0.01        Loss 0.5337 (0.5337)    Train Acc 75.7
 * Train Acc 77.625
Test: [0/2]     Loss 0.4754 (0.4754)    Prec@1 82.422 (82.422)
 * Test Acc 82.667
---------------------------------------------
Epoch: [6/200][0/4]     LR: 0.01        Loss 0.4728 (0.4728)    Train Acc 82.4
 * Train Acc 79.500
Test: [0/2]     Loss 0.4169 (0.4169)    Prec@1 83.984 (83.984)
 * Test Acc 83.667
---------------------------------------------
Epoch: [7/200][0/4]     LR: 0.01        Loss 0.4718 (0.4718)    Train Acc 79.2
 * Train Acc 80.750
Test: [0/2]     Loss 0.4137 (0.4137)    Prec@1 83.984 (83.984)
 * Test Acc 85.333
---------------------------------------------
Epoch: [8/200][0/4]     LR: 0.01        Loss 0.4030 (0.4030)    Train Acc 85.5
 * Train Acc 82.125
Test: [0/2]     Loss 0.3937 (0.3937)    Prec@1 84.375 (84.375)
 * Test Acc 85.333
---------------------------------------------
Epoch: [9/200][0/4]     LR: 0.01        Loss 0.4551 (0.4551)    Train Acc 79.2
 * Train Acc 80.750
Test: [0/2]     Loss 0.3609 (0.3609)    Prec@1 87.109 (87.109)
 * Test Acc 86.000
---------------------------------------------
Epoch: [10/200][0/4]    LR: 0.01        Loss 0.4544 (0.4544)    Train Acc 80.0
 * Train Acc 81.500
Test: [0/2]     Loss 0.3511 (0.3511)    Prec@1 85.938 (85.938)
 * Test Acc 85.667
---------------------------------------------
```




```
Epoch: [11/200][0/4]    LR: 0.01        Loss 0.4563 (0.4563)    Train Acc 82.0
 * Train Acc 81.625
Test: [0/2]     Loss 0.3819 (0.3819)    Prec@1 84.375 (84.375)
```

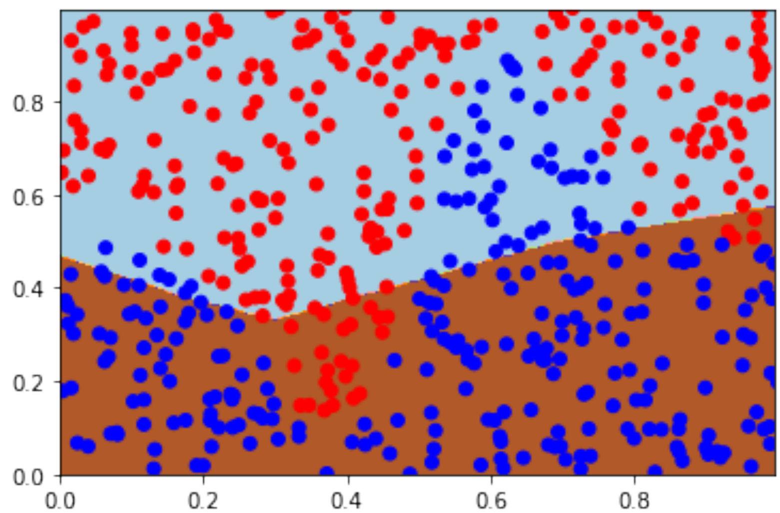
```
 * Test Acc 86.333
-----------------------------------------------
Epoch: [12/200][0/4]    LR: 0.01       Loss 0.4123 (0.4123)    Train Acc 81.6
 * Train Acc 82.500
Test: [0/2]     Loss 0.3636 (0.3636)    Prec@1 86.328 (86.328)
 * Test Acc 86.667
-----------------------------------------------
Epoch: [13/200][0/4]    LR: 0.01       Loss 0.4833 (0.4833)    Train Acc 80.0
 * Train Acc 82.125
Test: [0/2]     Loss 0.3691 (0.3691)    Prec@1 85.938 (85.938)
 * Test Acc 86.333
-----------------------------------------------
Epoch: [14/200][0/4]    LR: 0.01       Loss 0.4464 (0.4464)    Train Acc 80.8
 * Train Acc 82.375
Test: [0/2]     Loss 0.3491 (0.3491)    Prec@1 88.281 (88.281)
 * Test Acc 87.333
-----------------------------------------------
Epoch: [15/200][0/4]    LR: 0.01       Loss 0.4394 (0.4394)    Train Acc 80.4
 * Train Acc 81.875
Test: [0/2]     Loss 0.3434 (0.3434)    Prec@1 88.281 (88.281)
 * Test Acc 87.000
-----------------------------------------------
Epoch: [16/200][0/4]    LR: 0.01       Loss 0.4237 (0.4237)    Train Acc 82.0
 * Train Acc 81.375
Test: [0/2]     Loss 0.3487 (0.3487)    Prec@1 87.500 (87.500)
 * Test Acc 86.667
-----------------------------------------------
Epoch: [17/200][0/4]    LR: 0.01       Loss 0.4092 (0.4092)    Train Acc 82.0
 * Train Acc 83.375
Test: [0/2]     Loss 0.3718 (0.3718)    Prec@1 85.547 (85.547)
 * Test Acc 84.667
-----------------------------------------------
Epoch: [18/200][0/4]    LR: 0.01       Loss 0.4691 (0.4691)    Train Acc 80.8
 * Train Acc 81.375
Test: [0/2]     Loss 0.3514 (0.3514)    Prec@1 87.109 (87.109)
 * Test Acc 86.667
-----------------------------------------------
Epoch: [19/200][0/4]    LR: 0.01       Loss 0.3969 (0.3969)    Train Acc 82.0
 * Train Acc 81.000
Test: [0/2]     Loss 0.3477 (0.3477)    Prec@1 87.109 (87.109)
 * Test Acc 85.000
-----------------------------------------------
Epoch: [20/200][0/4]    LR: 0.01       Loss 0.4609 (0.4609)    Train Acc 78.5
 * Train Acc 81.875
Test: [0/2]     Loss 0.3456 (0.3456)    Prec@1 87.500 (87.500)
 * Test Acc 87.000
-----------------------------------------------
```





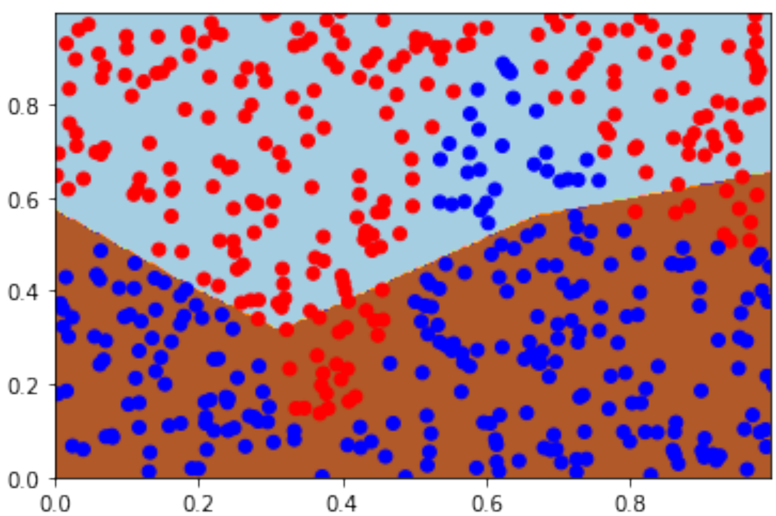
```
Epoch: [21/200][0/4]    LR: 0.01        Loss 0.3950 (0.3950)    Train Acc 83.2
 * Train Acc 82.750
Test: [0/2]     Loss 0.3424 (0.3424)    Prec@1 86.328 (86.328)
 * Test Acc 86.333
----------------------------------------------
Epoch: [22/200][0/4]    LR: 0.01        Loss 0.3874 (0.3874)    Train Acc 85.9
 * Train Acc 83.625
Test: [0/2]     Loss 0.3519 (0.3519)    Prec@1 87.109 (87.109)
 * Test Acc 86.667
----------------------------------------------
Epoch: [23/200][0/4]    LR: 0.01        Loss 0.4373 (0.4373)    Train Acc 80.8
 * Train Acc 82.250
Test: [0/2]     Loss 0.3739 (0.3739)    Prec@1 86.328 (86.328)
 * Test Acc 86.667
----------------------------------------------
Epoch: [24/200][0/4]    LR: 0.01        Loss 0.3884 (0.3884)    Train Acc 83.5
 * Train Acc 82.875
Test: [0/2]     Loss 0.3718 (0.3718)    Prec@1 86.328 (86.328)
 * Test Acc 86.667
----------------------------------------------
Epoch: [25/200][0/4]    LR: 0.01        Loss 0.4272 (0.4272)    Train Acc 83.9
 * Train Acc 83.875
Test: [0/2]     Loss 0.3533 (0.3533)    Prec@1 86.328 (86.328)
 * Test Acc 86.667
----------------------------------------------
Epoch: [26/200][0/4]    LR: 0.01        Loss 0.4068 (0.4068)    Train Acc 82.4
 * Train Acc 82.750
Test: [0/2]     Loss 0.3306 (0.3306)    Prec@1 87.109 (87.109)
 * Test Acc 86.667
----------------------------------------------
Epoch: [27/200][0/4]    LR: 0.01        Loss 0.4174 (0.4174)    Train Acc 80.8
 * Train Acc 82.625
Test: [0/2]     Loss 0.3461 (0.3461)    Prec@1 86.719 (86.719)
 * Test Acc 87.000
----------------------------------------------
Epoch: [28/200][0/4]    LR: 0.01        Loss 0.3789 (0.3789)    Train Acc 85.1
 * Train Acc 84.000
Test: [0/2]     Loss 0.3315 (0.3315)    Prec@1 86.328 (86.328)
 * Test Acc 85.000
```

```
-----------------------------------------------
Epoch: [29/200][0/4]    LR: 0.01        Loss 0.4062 (0.4062)    Train Acc 82.4
 * Train Acc 82.250
Test: [0/2]     Loss 0.3159 (0.3159)    Prec@1 88.281 (88.281)
 * Test Acc 86.667
-----------------------------------------------
Epoch: [30/200][0/4]    LR: 0.01        Loss 0.4190 (0.4190)    Train Acc 83.9
 * Train Acc 83.500
Test: [0/2]     Loss 0.3421 (0.3421)    Prec@1 86.719 (86.719)
 * Test Acc 86.333
-----------------------------------------------
```



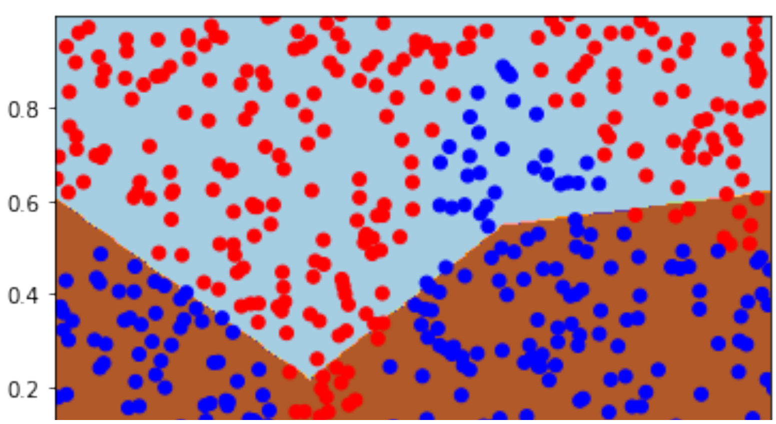

```
Epoch: [31/200][0/4]    LR: 0.01        Loss 0.4032 (0.4032)    Train Acc 83.9
 * Train Acc 84.000
Test: [0/2]     Loss 0.3291 (0.3291)    Prec@1 87.891 (87.891)
 * Test Acc 87.000
-----------------------------------------------
Epoch: [32/200][0/4]    LR: 0.01        Loss 0.4202 (0.4202)    Train Acc 82.4
 * Train Acc 84.250
Test: [0/2]     Loss 0.3620 (0.3620)    Prec@1 82.422 (82.422)
 * Test Acc 84.333
-----------------------------------------------
Epoch: [33/200][0/4]    LR: 0.01        Loss 0.4116 (0.4116)    Train Acc 82.4
 * Train Acc 82.125
Test: [0/2]     Loss 0.3392 (0.3392)    Prec@1 84.766 (84.766)
 * Test Acc 85.000
-----------------------------------------------
Epoch: [34/200][0/4]    LR: 0.01        Loss 0.4168 (0.4168)    Train Acc 82.4
 * Train Acc 84.500
Test: [0/2]     Loss 0.3234 (0.3234)    Prec@1 86.719 (86.719)
 * Test Acc 86.333
-----------------------------------------------
Epoch: [35/200][0/4]    LR: 0.01        Loss 0.3590 (0.3590)    Train Acc 86.7
 * Train Acc 84.750
Test: [0/2]     Loss 0.3336 (0.3336)    Prec@1 84.375 (84.375)
 * Test Acc 84.333
-----------------------------------------------
```

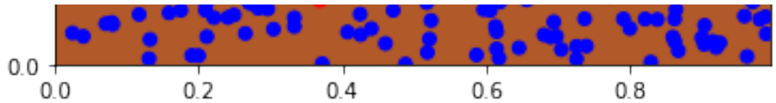
```
Epoch: [36/200][0/4]    LR: 0.01        Loss 0.3884 (0.3884)    Train Acc 83.2
 * Train Acc 82.375
Test: [0/2]     Loss 0.3301 (0.3301)    Prec@1 85.938 (85.938)
 * Test Acc 86.667
----------------------------------------------
Epoch: [37/200][0/4]    LR: 0.01        Loss 0.3337 (0.3337)    Train Acc 87.5
 * Train Acc 85.000
Test: [0/2]     Loss 0.3401 (0.3401)    Prec@1 89.453 (89.453)
 * Test Acc 88.667
----------------------------------------------
Epoch: [38/200][0/4]    LR: 0.01        Loss 0.3774 (0.3774)    Train Acc 85.9
 * Train Acc 85.500
Test: [0/2]     Loss 0.3371 (0.3371)    Prec@1 83.984 (83.984)
 * Test Acc 83.333
----------------------------------------------
Epoch: [39/200][0/4]    LR: 0.01        Loss 0.4059 (0.4059)    Train Acc 82.4
 * Train Acc 83.375
Test: [0/2]     Loss 0.3326 (0.3326)    Prec@1 86.328 (86.328)
 * Test Acc 86.667
----------------------------------------------
Epoch: [40/200][0/4]    LR: 0.01        Loss 0.3531 (0.3531)    Train Acc 88.2
 * Train Acc 85.250
Test: [0/2]     Loss 0.3345 (0.3345)    Prec@1 84.375 (84.375)
 * Test Acc 85.667
----------------------------------------------
```




```
Epoch: [41/200][0/4]    LR: 0.01        Loss 0.3988 (0.3988)    Train Acc 83.2
 * Train Acc 84.625
Test: [0/2]     Loss 0.3034 (0.3034)    Prec@1 85.938 (85.938)
 * Test Acc 85.000
----------------------------------------------
Epoch: [42/200][0/4]    LR: 0.01        Loss 0.3236 (0.3236)    Train Acc 88.2
 * Train Acc 85.750
Test: [0/2]     Loss 0.3217 (0.3217)    Prec@1 84.375 (84.375)
 * Test Acc 85.667
----------------------------------------------
Epoch: [43/200][0/4]    LR: 0.01        Loss 0.3836 (0.3836)    Train Acc 86.3
```

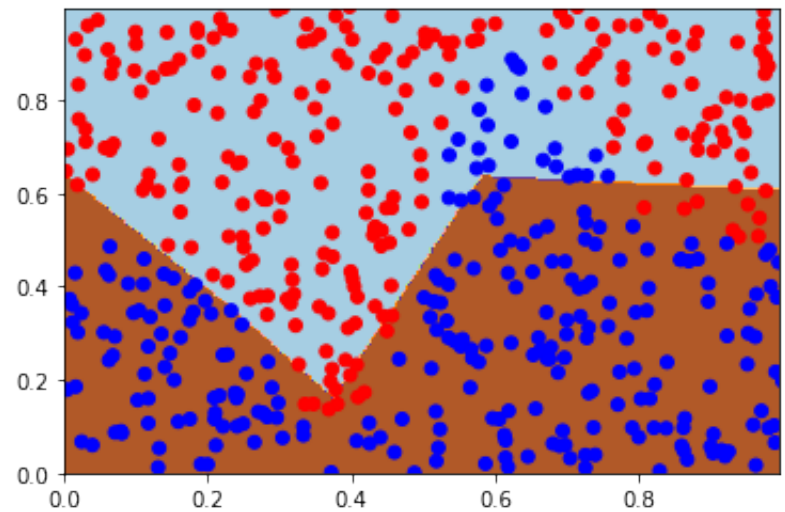
```
 * Train Acc 85.625
Test: [0/2]     Loss 0.3284 (0.3284)    Prec@1 84.375 (84.375)
 * Test Acc 85.000
----------------------------------------------
Epoch: [44/200][0/4]    LR: 0.01        Loss 0.3232 (0.3232)    Train Acc 85.9
 * Train Acc 85.000
Test: [0/2]     Loss 0.3267 (0.3267)    Prec@1 85.938 (85.938)
 * Test Acc 87.000
----------------------------------------------
Epoch: [45/200][0/4]    LR: 0.01        Loss 0.3142 (0.3142)    Train Acc 88.2
 * Train Acc 85.750
Test: [0/2]     Loss 0.3149 (0.3149)    Prec@1 87.109 (87.109)
 * Test Acc 87.000
----------------------------------------------
Epoch: [46/200][0/4]    LR: 0.01        Loss 0.3162 (0.3162)    Train Acc 87.5
 * Train Acc 86.000
Test: [0/2]     Loss 0.3169 (0.3169)    Prec@1 84.766 (84.766)
 * Test Acc 84.667
----------------------------------------------
Epoch: [47/200][0/4]    LR: 0.01        Loss 0.3247 (0.3247)    Train Acc 86.3
 * Train Acc 86.250
Test: [0/2]     Loss 0.3140 (0.3140)    Prec@1 84.766 (84.766)
 * Test Acc 85.000
----------------------------------------------
Epoch: [48/200][0/4]    LR: 0.01        Loss 0.3255 (0.3255)    Train Acc 87.1
 * Train Acc 85.375
Test: [0/2]     Loss 0.3155 (0.3155)    Prec@1 86.719 (86.719)
 * Test Acc 87.000
----------------------------------------------
Epoch: [49/200][0/4]    LR: 0.01        Loss 0.3221 (0.3221)    Train Acc 87.5
 * Train Acc 86.125
Test: [0/2]     Loss 0.3120 (0.3120)    Prec@1 85.938 (85.938)
 * Test Acc 86.667
----------------------------------------------
Epoch: [50/200][0/4]    LR: 0.01        Loss 0.3203 (0.3203)    Train Acc 87.5
 * Train Acc 85.375
Test: [0/2]     Loss 0.2990 (0.2990)    Prec@1 86.719 (86.719)
 * Test Acc 87.333
----------------------------------------------
```





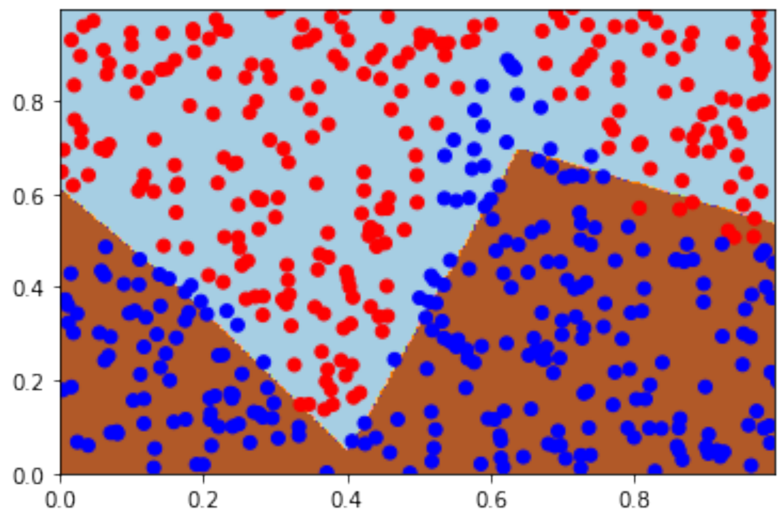
```
Epoch: [51/200][0/4]    LR: 0.01        Loss 0.3558 (0.3558)    Train Acc 85.9
 * Train Acc 86.375
Test: [0/2]     Loss 0.2820 (0.2820)    Prec@1 88.281 (88.281)
 * Test Acc 88.000
-----------------------------------------------
Epoch: [52/200][0/4]    LR: 0.01        Loss 0.3316 (0.3316)    Train Acc 87.8
 * Train Acc 86.000
Test: [0/2]     Loss 0.2940 (0.2940)    Prec@1 84.766 (84.766)
 * Test Acc 85.333
-----------------------------------------------
Epoch: [53/200][0/4]    LR: 0.01        Loss 0.3018 (0.3018)    Train Acc 87.8
 * Train Acc 87.250
Test: [0/2]     Loss 0.2890 (0.2890)    Prec@1 87.891 (87.891)
 * Test Acc 87.667
-----------------------------------------------
Epoch: [54/200][0/4]    LR: 0.01        Loss 0.3188 (0.3188)    Train Acc 87.1
 * Train Acc 87.375
Test: [0/2]     Loss 0.2824 (0.2824)    Prec@1 87.109 (87.109)
 * Test Acc 84.000
-----------------------------------------------
Epoch: [55/200][0/4]    LR: 0.01        Loss 0.2939 (0.2939)    Train Acc 87.5
 * Train Acc 85.500
Test: [0/2]     Loss 0.2927 (0.2927)    Prec@1 87.109 (87.109)
 * Test Acc 87.667
-----------------------------------------------
Epoch: [56/200][0/4]    LR: 0.01        Loss 0.3167 (0.3167)    Train Acc 89.8
 * Train Acc 89.000
Test: [0/2]     Loss 0.2841 (0.2841)    Prec@1 85.938 (85.938)
 * Test Acc 86.333
-----------------------------------------------
Epoch: [57/200][0/4]    LR: 0.01        Loss 0.2756 (0.2756)    Train Acc 88.6
 * Train Acc 87.250
Test: [0/2]     Loss 0.2894 (0.2894)    Prec@1 85.156 (85.156)
 * Test Acc 86.333
-----------------------------------------------
Epoch: [58/200][0/4]    LR: 0.01        Loss 0.3267 (0.3267)    Train Acc 87.1
 * Train Acc 88.625
Test: [0/2]     Loss 0.2654 (0.2654)    Prec@1 87.891 (87.891)
 * Test Acc 86.333
-----------------------------------------------
Epoch: [59/200][0/4]    LR: 0.01        Loss 0.3155 (0.3155)    Train Acc 88.6
 * Train Acc 88.375
Test: [0/2]     Loss 0.2665 (0.2665)    Prec@1 87.109 (87.109)
 * Test Acc 86.667
-----------------------------------------------
Epoch: [60/200][0/4]    LR: 0.01        Loss 0.2866 (0.2866)    Train Acc 88.6
 * Train Acc 88.750
```

```
Test: [0/2]     Loss 0.2734 (0.2734)    Prec@1 87.500 (87.500)
 * Test Acc 88.667
---------------------------------------------
```



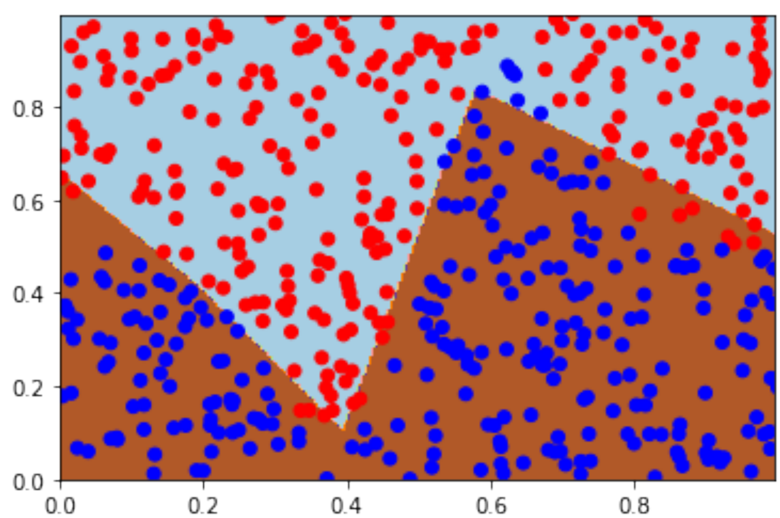

```
Epoch: [61/200][0/4]    LR: 0.01        Loss 0.2987 (0.2987)    Train Acc 89.4
 * Train Acc 87.875
Test: [0/2]     Loss 0.2847 (0.2847)    Prec@1 90.625 (90.625)
 * Test Acc 91.333
---------------------------------------------
Epoch: [62/200][0/4]    LR: 0.01        Loss 0.2657 (0.2657)    Train Acc 91.0
 * Train Acc 88.625
Test: [0/2]     Loss 0.2421 (0.2421)    Prec@1 88.281 (88.281)
 * Test Acc 87.000
---------------------------------------------
Epoch: [63/200][0/4]    LR: 0.01        Loss 0.2714 (0.2714)    Train Acc 87.5
 * Train Acc 89.500
Test: [0/2]     Loss 0.2764 (0.2764)    Prec@1 86.328 (86.328)
 * Test Acc 87.333
---------------------------------------------
Epoch: [64/200][0/4]    LR: 0.01        Loss 0.2966 (0.2966)    Train Acc 89.4
 * Train Acc 89.000
Test: [0/2]     Loss 0.2550 (0.2550)    Prec@1 90.625 (90.625)
 * Test Acc 90.667
---------------------------------------------
Epoch: [65/200][0/4]    LR: 0.01        Loss 0.2593 (0.2593)    Train Acc 90.6
 * Train Acc 88.750
Test: [0/2]     Loss 0.2340 (0.2340)    Prec@1 89.844 (89.844)
 * Test Acc 89.333
---------------------------------------------
Epoch: [66/200][0/4]    LR: 0.01        Loss 0.2323 (0.2323)    Train Acc 92.5
 * Train Acc 90.375
Test: [0/2]     Loss 0.2509 (0.2509)    Prec@1 87.891 (87.891)
 * Test Acc 88.000
---------------------------------------------
Epoch: [67/200][0/4]    LR: 0.01        Loss 0.2620 (0.2620)    Train Acc 91.0
 * Train Acc 90.750
Test: [0/2]     Loss 0.2198 (0.2198)    Prec@1 92.578 (92.578)
```

```
 * Test Acc 91.000
----------------------------------------------
Epoch: [68/200][0/4]    LR: 0.01        Loss 0.2527 (0.2527)    Train Acc 92.1
 * Train Acc 91.125
Test: [0/2]     Loss 0.2512 (0.2512)    Prec@1 89.062 (89.062)
 * Test Acc 89.333
----------------------------------------------
Epoch: [69/200][0/4]    LR: 0.01        Loss 0.2756 (0.2756)    Train Acc 88.6
 * Train Acc 89.250
Test: [0/2]     Loss 0.2324 (0.2324)    Prec@1 91.406 (91.406)
 * Test Acc 91.667
----------------------------------------------
Epoch: [70/200][0/4]    LR: 0.01        Loss 0.2216 (0.2216)    Train Acc 91.7
 * Train Acc 90.625
Test: [0/2]     Loss 0.2327 (0.2327)    Prec@1 90.234 (90.234)
 * Test Acc 90.000
----------------------------------------------
```



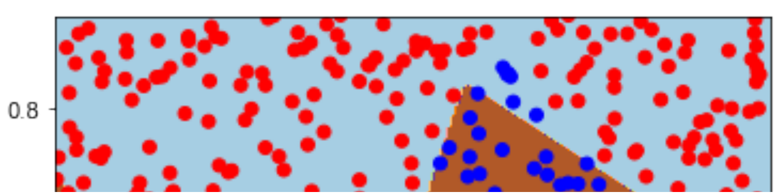

```
Epoch: [71/200][0/4]    LR: 0.01        Loss 0.2311 (0.2311)    Train Acc 91.4
 * Train Acc 88.625
Test: [0/2]     Loss 0.2598 (0.2598)    Prec@1 87.500 (87.500)
 * Test Acc 88.000
----------------------------------------------
Epoch: [72/200][0/4]    LR: 0.01        Loss 0.2017 (0.2017)    Train Acc 91.4
 * Train Acc 90.250
Test: [0/2]     Loss 0.2499 (0.2499)    Prec@1 91.016 (91.016)
 * Test Acc 91.667
----------------------------------------------
Epoch: [73/200][0/4]    LR: 0.01        Loss 0.3405 (0.3405)    Train Acc 87.5
 * Train Acc 90.125
Test: [0/2]     Loss 0.2289 (0.2289)    Prec@1 91.797 (91.797)
 * Test Acc 91.000
----------------------------------------------
Epoch: [74/200][0/4]    LR: 0.01        Loss 0.2434 (0.2434)    Train Acc 87.8
 * Train Acc 90.375
Test: [0/2]     Loss 0.2075 (0.2075)    Prec@1 92.578 (92.578)
 * Test Acc 92.667
```

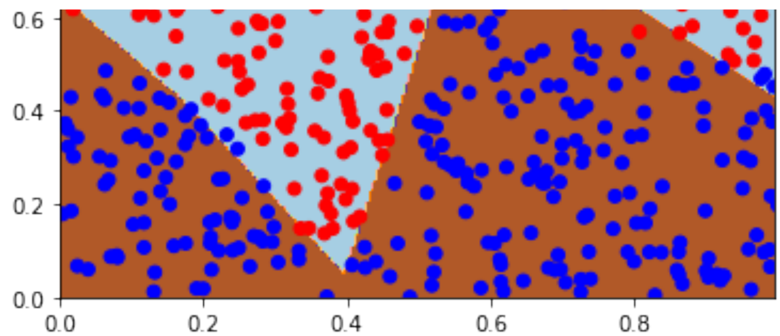
```
-----------------------------------------------
Epoch: [75/200][0/4]    LR: 0.01        Loss 0.2588 (0.2588)    Train Acc 90.2
 * Train Acc 91.375
Test: [0/2]     Loss 0.1999 (0.1999)    Prec@1 91.797 (91.797)
 * Test Acc 92.000
-----------------------------------------------
Epoch: [76/200][0/4]    LR: 0.01        Loss 0.2279 (0.2279)    Train Acc 91.7
 * Train Acc 90.750
Test: [0/2]     Loss 0.2308 (0.2308)    Prec@1 90.234 (90.234)
 * Test Acc 90.000
-----------------------------------------------
Epoch: [77/200][0/4]    LR: 0.01        Loss 0.3019 (0.3019)    Train Acc 87.8
 * Train Acc 90.875
Test: [0/2]     Loss 0.2012 (0.2012)    Prec@1 92.188 (92.188)
 * Test Acc 92.000
-----------------------------------------------
Epoch: [78/200][0/4]    LR: 0.01        Loss 0.2712 (0.2712)    Train Acc 90.2
 * Train Acc 91.500
Test: [0/2]     Loss 0.1954 (0.1954)    Prec@1 92.578 (92.578)
 * Test Acc 92.000
-----------------------------------------------
Epoch: [79/200][0/4]    LR: 0.01        Loss 0.2237 (0.2237)    Train Acc 94.1
 * Train Acc 92.750
Test: [0/2]     Loss 0.2235 (0.2235)    Prec@1 90.625 (90.625)
 * Test Acc 90.667
-----------------------------------------------
Epoch: [80/200][0/4]    LR: 0.01        Loss 0.2457 (0.2457)    Train Acc 91.0
 * Train Acc 91.875
Test: [0/2]     Loss 0.2159 (0.2159)    Prec@1 91.406 (91.406)
 * Test Acc 91.667
-----------------------------------------------
```




```
Epoch: [81/200][0/4]    LR: 0.01        Loss 0.2309 (0.2309)    Train Acc 92.5
 * Train Acc 93.000
Test: [0/2]     Loss 0.1937 (0.1937)    Prec@1 92.188 (92.188)
 * Test Acc 92.333
-----------------------------------------------
```

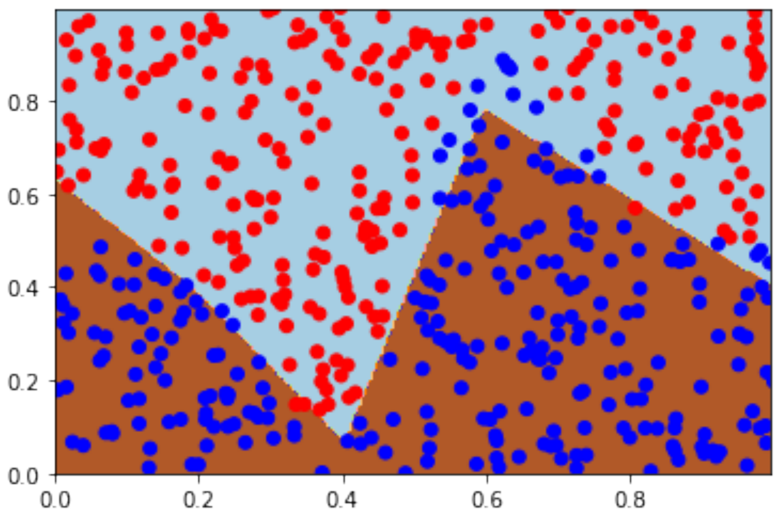
```
Epoch: [82/200][0/4]    LR: 0.01        Loss 0.2288 (0.2288)    Train Acc 91.4
 * Train Acc 92.625
Test: [0/2]     Loss 0.2010 (0.2010)    Prec@1 93.359 (93.359)
 * Test Acc 92.333
----------------------------------------------
Epoch: [83/200][0/4]    LR: 0.01        Loss 0.2317 (0.2317)    Train Acc 93.7
 * Train Acc 93.000
Test: [0/2]     Loss 0.1862 (0.1862)    Prec@1 92.578 (92.578)
 * Test Acc 91.667
----------------------------------------------
Epoch: [84/200][0/4]    LR: 0.01        Loss 0.2331 (0.2331)    Train Acc 92.9
 * Train Acc 92.875
Test: [0/2]     Loss 0.1901 (0.1901)    Prec@1 93.750 (93.750)
 * Test Acc 92.333
----------------------------------------------
Epoch: [85/200][0/4]    LR: 0.01        Loss 0.2234 (0.2234)    Train Acc 94.1
 * Train Acc 92.500
Test: [0/2]     Loss 0.1790 (0.1790)    Prec@1 93.750 (93.750)
 * Test Acc 92.667
----------------------------------------------
Epoch: [86/200][0/4]    LR: 0.01        Loss 0.2092 (0.2092)    Train Acc 92.1
 * Train Acc 93.125
Test: [0/2]     Loss 0.2544 (0.2544)    Prec@1 86.719 (86.719)
 * Test Acc 88.333
----------------------------------------------
Epoch: [87/200][0/4]    LR: 0.01        Loss 0.2562 (0.2562)    Train Acc 90.2
 * Train Acc 90.875
Test: [0/2]     Loss 0.1953 (0.1953)    Prec@1 91.797 (91.797)
 * Test Acc 92.000
----------------------------------------------
Epoch: [88/200][0/4]    LR: 0.01        Loss 0.2375 (0.2375)    Train Acc 91.0
 * Train Acc 90.625
Test: [0/2]     Loss 0.1911 (0.1911)    Prec@1 94.531 (94.531)
 * Test Acc 93.333
----------------------------------------------
Epoch: [89/200][0/4]    LR: 0.01        Loss 0.2631 (0.2631)    Train Acc 92.1
 * Train Acc 92.625
Test: [0/2]     Loss 0.1955 (0.1955)    Prec@1 93.359 (93.359)
 * Test Acc 94.000
----------------------------------------------
Epoch: [90/200][0/4]    LR: 0.01        Loss 0.1982 (0.1982)    Train Acc 92.9
 * Train Acc 93.250
Test: [0/2]     Loss 0.1783 (0.1783)    Prec@1 94.141 (94.141)
 * Test Acc 94.333
----------------------------------------------
```





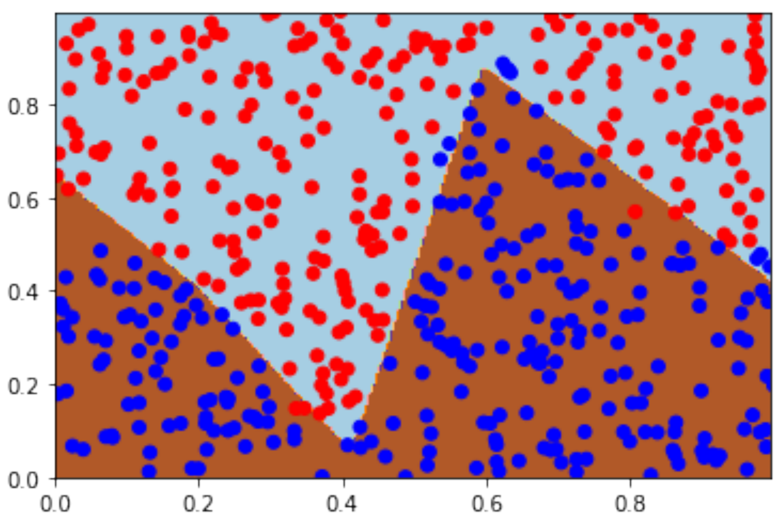
```
Epoch: [91/200][0/4]    LR: 0.01        Loss 0.2449 (0.2449)    Train Acc 93.7
 * Train Acc 94.625
Test: [0/2]     Loss 0.1771 (0.1771)    Prec@1 92.969 (92.969)
 * Test Acc 93.000
-----------------------------------------------
Epoch: [92/200][0/4]    LR: 0.01        Loss 0.2798 (0.2798)    Train Acc 93.3
 * Train Acc 93.625
Test: [0/2]     Loss 0.2085 (0.2085)    Prec@1 92.188 (92.188)
 * Test Acc 92.667
-----------------------------------------------
Epoch: [93/200][0/4]    LR: 0.01        Loss 0.1995 (0.1995)    Train Acc 93.3
 * Train Acc 93.000
Test: [0/2]     Loss 0.1823 (0.1823)    Prec@1 92.188 (92.188)
 * Test Acc 92.333
-----------------------------------------------
Epoch: [94/200][0/4]    LR: 0.01        Loss 0.2719 (0.2719)    Train Acc 87.5
 * Train Acc 91.500
Test: [0/2]     Loss 0.2128 (0.2128)    Prec@1 91.016 (91.016)
 * Test Acc 91.667
-----------------------------------------------
Epoch: [95/200][0/4]    LR: 0.01        Loss 0.2311 (0.2311)    Train Acc 91.7
 * Train Acc 93.125
Test: [0/2]     Loss 0.1841 (0.1841)    Prec@1 91.797 (91.797)
 * Test Acc 92.333
-----------------------------------------------
Epoch: [96/200][0/4]    LR: 0.01        Loss 0.2118 (0.2118)    Train Acc 92.1
 * Train Acc 92.500
Test: [0/2]     Loss 0.1930 (0.1930)    Prec@1 94.922 (94.922)
 * Test Acc 94.000
-----------------------------------------------
Epoch: [97/200][0/4]    LR: 0.01        Loss 0.2069 (0.2069)    Train Acc 93.3
 * Train Acc 92.250
Test: [0/2]     Loss 0.1985 (0.1985)    Prec@1 94.922 (94.922)
 * Test Acc 95.333
-----------------------------------------------
Epoch: [98/200][0/4]    LR: 0.01        Loss 0.2464 (0.2464)    Train Acc 92.5
 * Train Acc 93.000
Test: [0/2]     Loss 0.1898 (0.1898)    Prec@1 92.969 (92.969)
 * Test Acc 92.000
-----------------------------------------------
Epoch: [99/200][0/4]    LR: 0.01        Loss 0.3229 (0.3229)    Train Acc 85.9
```

```
 * Train Acc 90.125
Test: [0/2]     Loss 0.2559 (0.2559)    Prec@1 88.672 (88.672)
 * Test Acc 87.667
----------------------------------------------
Epoch: [100/200][0/4]   LR: 0.01        Loss 0.3159 (0.3159)    Train Acc 85.1
 * Train Acc 90.375
Test: [0/2]     Loss 0.1631 (0.1631)    Prec@1 93.750 (93.750)
 * Test Acc 93.333
----------------------------------------------
```



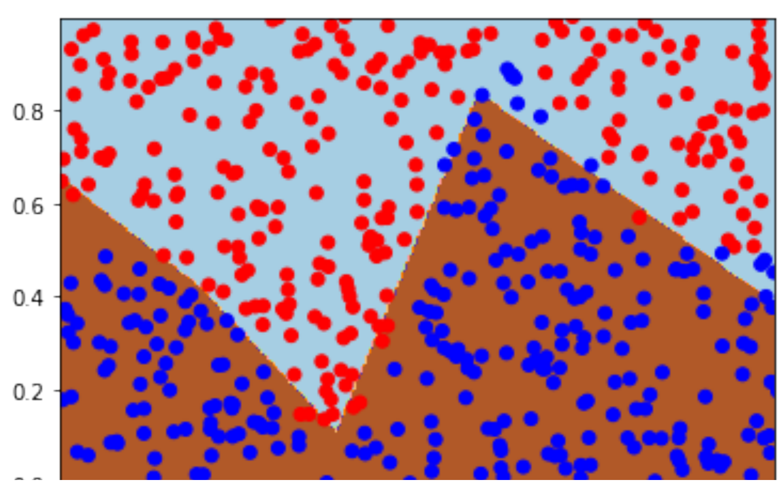

```
Epoch: [101/200][0/4]   LR: 0.01        Loss 0.2095 (0.2095)    Train Acc 91.7
 * Train Acc 91.750
Test: [0/2]     Loss 0.1544 (0.1544)    Prec@1 92.969 (92.969)
 * Test Acc 93.000
----------------------------------------------
Epoch: [102/200][0/4]   LR: 0.01        Loss 0.2062 (0.2062)    Train Acc 94.5
 * Train Acc 92.750
Test: [0/2]     Loss 0.1914 (0.1914)    Prec@1 92.188 (92.188)
 * Test Acc 92.667
----------------------------------------------
Epoch: [103/200][0/4]   LR: 0.01        Loss 0.2126 (0.2126)    Train Acc 91.7
 * Train Acc 92.375
Test: [0/2]     Loss 0.1773 (0.1773)    Prec@1 92.578 (92.578)
 * Test Acc 93.000
----------------------------------------------
Epoch: [104/200][0/4]   LR: 0.01        Loss 0.1794 (0.1794)    Train Acc 95.7
 * Train Acc 93.625
Test: [0/2]     Loss 0.1691 (0.1691)    Prec@1 94.141 (94.141)
 * Test Acc 93.000
----------------------------------------------
Epoch: [105/200][0/4]   LR: 0.01        Loss 0.2334 (0.2334)    Train Acc 93.7
 * Train Acc 93.875
Test: [0/2]     Loss 0.1536 (0.1536)    Prec@1 95.312 (95.312)
 * Test Acc 94.000
----------------------------------------------
Epoch: [106/200][0/4]   LR: 0.01        Loss 0.2093 (0.2093)    Train Acc 94.1
 * Train Acc 93.875
```

```
Test: [0/2]     Loss 0.1983 (0.1983)    Prec@1 91.406 (91.406)
 * Test Acc 92.000
----------------------------------------------
Epoch: [107/200][0/4]   LR: 0.01        Loss 0.1906 (0.1906)    Train Acc 92.5
 * Train Acc 92.250
Test: [0/2]     Loss 0.1407 (0.1407)    Prec@1 94.922 (94.922)
 * Test Acc 94.000
----------------------------------------------
Epoch: [108/200][0/4]   LR: 0.01        Loss 0.2046 (0.2046)    Train Acc 94.5
 * Train Acc 93.375
Test: [0/2]     Loss 0.2218 (0.2218)    Prec@1 90.625 (90.625)
 * Test Acc 91.000
----------------------------------------------
Epoch: [109/200][0/4]   LR: 0.01        Loss 0.2761 (0.2761)    Train Acc 91.7
 * Train Acc 93.000
Test: [0/2]     Loss 0.1738 (0.1738)    Prec@1 91.406 (91.406)
 * Test Acc 92.000
----------------------------------------------
Epoch: [110/200][0/4]   LR: 0.01        Loss 0.2502 (0.2502)    Train Acc 88.6
 * Train Acc 90.750
Test: [0/2]     Loss 0.1766 (0.1766)    Prec@1 93.750 (93.750)
 * Test Acc 94.333
----------------------------------------------
```



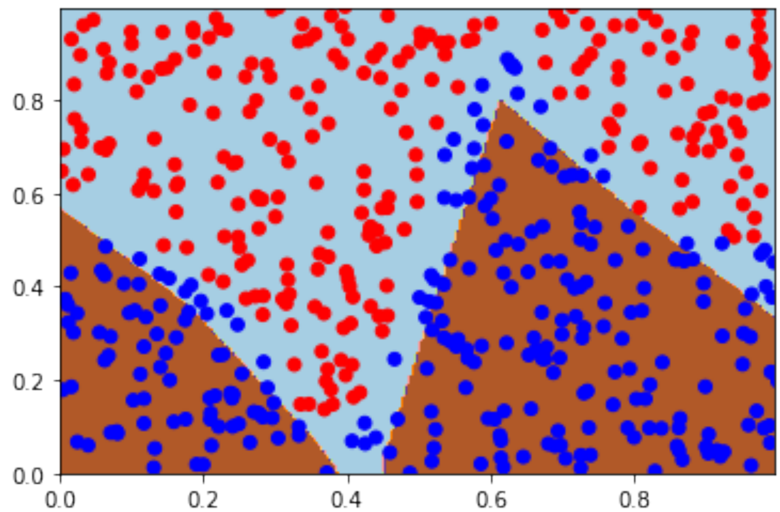

```
Epoch: [111/200][0/4]   LR: 0.01        Loss 0.1669 (0.1669)    Train Acc 94.5
 * Train Acc 92.250
Test: [0/2]     Loss 0.2485 (0.2485)    Prec@1 87.500 (87.500)
 * Test Acc 87.333
----------------------------------------------
Epoch: [112/200][0/4]   LR: 0.01        Loss 0.2313 (0.2313)    Train Acc 89.0
 * Train Acc 88.875
Test: [0/2]     Loss 0.1682 (0.1682)    Prec@1 92.969 (92.969)
 * Test Acc 93.333
----------------------------------------------
Epoch: [113/200][0/4]   LR: 0.01        Loss 0.2760 (0.2760)    Train Acc 90.6
 * Train Acc 92.375
Test: [0/2]     Loss 0.1560 (0.1560)    Prec@1 95.312 (95.312)
```

```
 * Test Acc 95.000
---------------------------------------------
Epoch: [114/200][0/4]   LR: 0.01        Loss 0.2031 (0.2031)    Train Acc 95.3
 * Train Acc 92.875
Test: [0/2]     Loss 0.1655 (0.1655)    Prec@1 94.922 (94.922)
 * Test Acc 95.000
---------------------------------------------
Epoch: [115/200][0/4]   LR: 0.01        Loss 0.2300 (0.2300)    Train Acc 94.1
 * Train Acc 93.000
Test: [0/2]     Loss 0.1631 (0.1631)    Prec@1 92.969 (92.969)
 * Test Acc 92.667
---------------------------------------------
Epoch: [116/200][0/4]   LR: 0.01        Loss 0.2162 (0.2162)    Train Acc 91.0
 * Train Acc 91.000
Test: [0/2]     Loss 0.2540 (0.2540)    Prec@1 87.891 (87.891)
 * Test Acc 88.667
---------------------------------------------
Epoch: [117/200][0/4]   LR: 0.01        Loss 0.2365 (0.2365)    Train Acc 90.2
 * Train Acc 90.375
Test: [0/2]     Loss 0.1669 (0.1669)    Prec@1 93.750 (93.750)
 * Test Acc 93.000
---------------------------------------------
Epoch: [118/200][0/4]   LR: 0.01        Loss 0.2721 (0.2721)    Train Acc 91.7
 * Train Acc 93.125
Test: [0/2]     Loss 0.1640 (0.1640)    Prec@1 93.359 (93.359)
 * Test Acc 93.333
---------------------------------------------
Epoch: [119/200][0/4]   LR: 0.01        Loss 0.1832 (0.1832)    Train Acc 92.9
 * Train Acc 92.625
Test: [0/2]     Loss 0.1498 (0.1498)    Prec@1 93.359 (93.359)
 * Test Acc 93.667
---------------------------------------------
Epoch: [120/200][0/4]   LR: 0.01        Loss 0.2413 (0.2413)    Train Acc 91.7
 * Train Acc 92.875
Test: [0/2]     Loss 0.1736 (0.1736)    Prec@1 93.359 (93.359)
 * Test Acc 94.333
---------------------------------------------
```



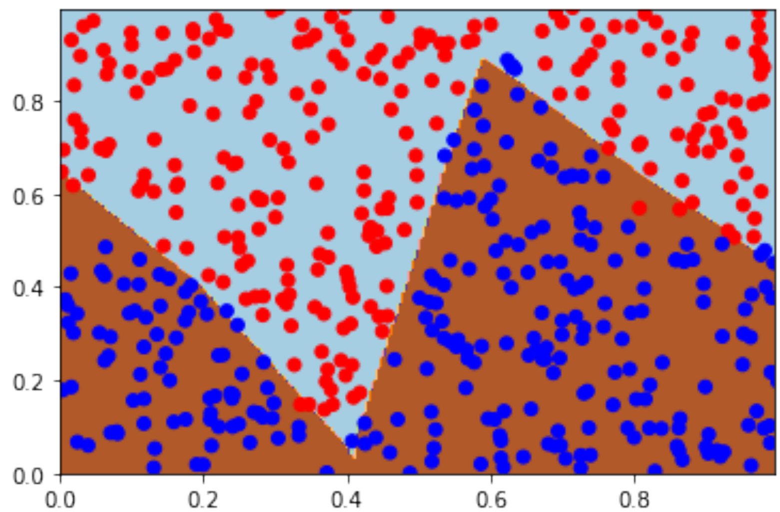

```
Epoch: [121/200][0/4]   LR: 0.01        Loss 0.2542 (0.2542)    Train Acc 90.2
 * Train Acc 92.875
Test: [0/2]     Loss 0.1699 (0.1699)    Prec@1 93.750 (93.750)
 * Test Acc 94.000
-----------------------------------------------
Epoch: [122/200][0/4]   LR: 0.01        Loss 0.1860 (0.1860)    Train Acc 95.3
 * Train Acc 93.000
Test: [0/2]     Loss 0.1388 (0.1388)    Prec@1 94.531 (94.531)
 * Test Acc 93.667
-----------------------------------------------
Epoch: [123/200][0/4]   LR: 0.01        Loss 0.2100 (0.2100)    Train Acc 93.3
 * Train Acc 93.625
Test: [0/2]     Loss 0.1729 (0.1729)    Prec@1 94.531 (94.531)
 * Test Acc 94.667
-----------------------------------------------
Epoch: [124/200][0/4]   LR: 0.01        Loss 0.2182 (0.2182)    Train Acc 92.9
 * Train Acc 93.625
Test: [0/2]     Loss 0.1569 (0.1569)    Prec@1 94.531 (94.531)
 * Test Acc 95.000
-----------------------------------------------
Epoch: [125/200][0/4]   LR: 0.01        Loss 0.1824 (0.1824)    Train Acc 94.1
 * Train Acc 93.500
Test: [0/2]     Loss 0.1737 (0.1737)    Prec@1 95.312 (95.312)
 * Test Acc 95.333
-----------------------------------------------
Epoch: [126/200][0/4]   LR: 0.01        Loss 0.2122 (0.2122)    Train Acc 93.7
 * Train Acc 94.000
Test: [0/2]     Loss 0.1649 (0.1649)    Prec@1 91.797 (91.797)
 * Test Acc 92.667
-----------------------------------------------
Epoch: [127/200][0/4]   LR: 0.01        Loss 0.1731 (0.1731)    Train Acc 92.1
 * Train Acc 92.625
Test: [0/2]     Loss 0.1514 (0.1514)    Prec@1 95.312 (95.312)
 * Test Acc 93.333
-----------------------------------------------
Epoch: [128/200][0/4]   LR: 0.01        Loss 0.1642 (0.1642)    Train Acc 94.5
 * Train Acc 93.000
Test: [0/2]     Loss 0.1505 (0.1505)    Prec@1 92.969 (92.969)
 * Test Acc 92.667
-----------------------------------------------
Epoch: [129/200][0/4]   LR: 0.01        Loss 0.2348 (0.2348)    Train Acc 92.1
 * Train Acc 92.875
Test: [0/2]     Loss 0.2075 (0.2075)    Prec@1 91.797 (91.797)
 * Test Acc 92.000
-----------------------------------------------
Epoch: [130/200][0/4]   LR: 0.01        Loss 0.2080 (0.2080)    Train Acc 93.3
 * Train Acc 91.250
Test: [0/2]     Loss 0.1632 (0.1632)    Prec@1 92.578 (92.578)
 * Test Acc 92.000
```

```
---------------------------------------------
```



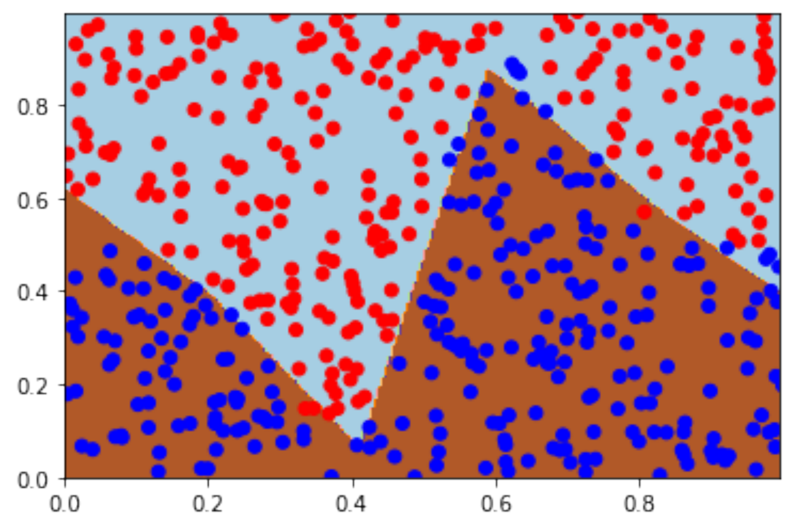

```
Epoch: [131/200][0/4]   LR: 0.01        Loss 0.2349 (0.2349)    Train Acc 91.7
 * Train Acc 90.875
Test: [0/2]     Loss 0.2854 (0.2854)    Prec@1 83.984 (83.984)
 * Test Acc 83.667
---------------------------------------------
Epoch: [132/200][0/4]   LR: 0.01        Loss 0.2978 (0.2978)    Train Acc 85.1
 * Train Acc 88.000
Test: [0/2]     Loss 0.1985 (0.1985)    Prec@1 91.797 (91.797)
 * Test Acc 92.333
---------------------------------------------
Epoch: [133/200][0/4]   LR: 0.01        Loss 0.2997 (0.2997)    Train Acc 89.(
 * Train Acc 90.125
Test: [0/2]     Loss 0.2181 (0.2181)    Prec@1 90.234 (90.234)
 * Test Acc 88.000
---------------------------------------------
Epoch: [134/200][0/4]   LR: 0.01        Loss 0.2108 (0.2108)    Train Acc 92.1
 * Train Acc 91.000
Test: [0/2]     Loss 0.2256 (0.2256)    Prec@1 91.797 (91.797)
 * Test Acc 92.000
---------------------------------------------
Epoch: [135/200][0/4]   LR: 0.01        Loss 0.2784 (0.2784)    Train Acc 87.8
 * Train Acc 88.000
Test: [0/2]     Loss 0.2309 (0.2309)    Prec@1 87.500 (87.500)
 * Test Acc 87.667
---------------------------------------------
Epoch: [136/200][0/4]   LR: 0.01        Loss 0.3065 (0.3065)    Train Acc 86.7
 * Train Acc 88.125
Test: [0/2]     Loss 0.1431 (0.1431)    Prec@1 94.141 (94.141)
 * Test Acc 93.333
---------------------------------------------
Epoch: [137/200][0/4]   LR: 0.01        Loss 0.1887 (0.1887)    Train Acc 92.9
 * Train Acc 92.125
Test: [0/2]     Loss 0.1718 (0.1718)    Prec@1 92.969 (92.969)
 * Test Acc 93.333
---------------------------------------------
```

```
Epoch: [138/200][0/4]   LR: 0.01        Loss 0.1924 (0.1924)    Train Acc 94.9
 * Train Acc 93.750
Test: [0/2]     Loss 0.1838 (0.1838)    Prec@1 94.141 (94.141)
 * Test Acc 95.000
----------------------------------------------
Epoch: [139/200][0/4]   LR: 0.01        Loss 0.1694 (0.1694)    Train Acc 94.9
 * Train Acc 93.875
Test: [0/2]     Loss 0.1718 (0.1718)    Prec@1 93.750 (93.750)
 * Test Acc 94.000
----------------------------------------------
Epoch: [140/200][0/4]   LR: 0.01        Loss 0.2138 (0.2138)    Train Acc 92.1
 * Train Acc 93.250
Test: [0/2]     Loss 0.1732 (0.1732)    Prec@1 93.750 (93.750)
 * Test Acc 94.000
----------------------------------------------
```



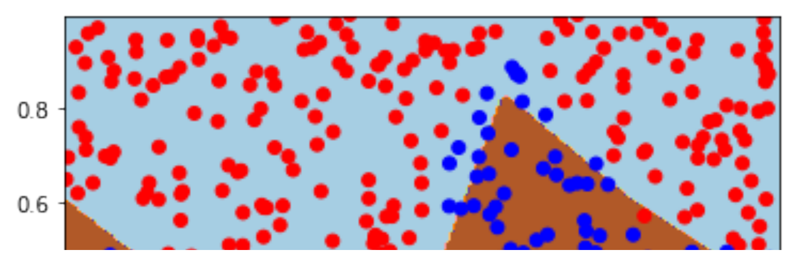

```
Epoch: [141/200][0/4]   LR: 0.01        Loss 0.2254 (0.2254)    Train Acc 92.5
 * Train Acc 94.125
Test: [0/2]     Loss 0.1582 (0.1582)    Prec@1 94.531 (94.531)
 * Test Acc 94.000
----------------------------------------------
Epoch: [142/200][0/4]   LR: 0.01        Loss 0.1883 (0.1883)    Train Acc 92.5
 * Train Acc 93.875
Test: [0/2]     Loss 0.1633 (0.1633)    Prec@1 95.703 (95.703)
 * Test Acc 95.333
----------------------------------------------
Epoch: [143/200][0/4]   LR: 0.01        Loss 0.2159 (0.2159)    Train Acc 93.3
 * Train Acc 94.125
Test: [0/2]     Loss 0.1945 (0.1945)    Prec@1 93.359 (93.359)
 * Test Acc 94.333
----------------------------------------------
Epoch: [144/200][0/4]   LR: 0.01        Loss 0.2090 (0.2090)    Train Acc 92.9
 * Train Acc 92.500
Test: [0/2]     Loss 0.1378 (0.1378)    Prec@1 93.750 (93.750)
 * Test Acc 93.000
----------------------------------------------
Epoch: [145/200][0/4]   LR: 0.01        Loss 0.1923 (0.1923)    Train Acc 92.5
```

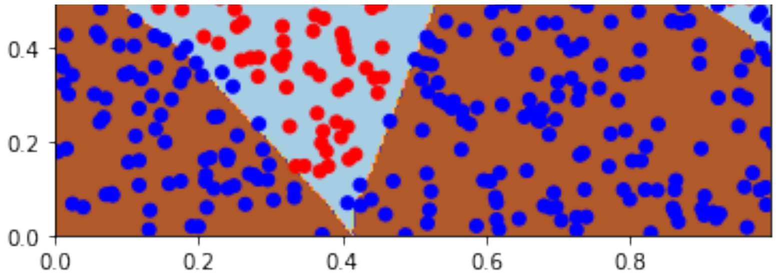
```
Epoch: [145/200][0/4]   LR: 0.01        Loss 0.1925 (0.1925)    Train Acc 92.
 * Train Acc 93.625
Test: [0/2]     Loss 0.1408 (0.1408)    Prec@1 92.969 (92.969)
 * Test Acc 93.333
-----------------------------------------------
Epoch: [146/200][0/4]   LR: 0.01        Loss 0.1997 (0.1997)    Train Acc 95.7
 * Train Acc 93.125
Test: [0/2]     Loss 0.1442 (0.1442)    Prec@1 94.141 (94.141)
 * Test Acc 93.667
-----------------------------------------------
Epoch: [147/200][0/4]   LR: 0.01        Loss 0.2138 (0.2138)    Train Acc 92.9
 * Train Acc 93.375
Test: [0/2]     Loss 0.1537 (0.1537)    Prec@1 95.312 (95.312)
 * Test Acc 95.333
-----------------------------------------------
Epoch: [148/200][0/4]   LR: 0.01        Loss 0.2596 (0.2596)    Train Acc 92.1
 * Train Acc 94.125
Test: [0/2]     Loss 0.1394 (0.1394)    Prec@1 96.094 (96.094)
 * Test Acc 95.667
-----------------------------------------------
Epoch: [149/200][0/4]   LR: 0.01        Loss 0.2196 (0.2196)    Train Acc 94.1
 * Train Acc 94.250
Test: [0/2]     Loss 0.1614 (0.1614)    Prec@1 93.750 (93.750)
 * Test Acc 94.667
-----------------------------------------------
Epoch: [150/200][0/4]   LR: 0.01        Loss 0.1910 (0.1910)    Train Acc 93.3
 * Train Acc 92.750
Test: [0/2]     Loss 0.1438 (0.1438)    Prec@1 95.312 (95.312)
 * Test Acc 95.333
-----------------------------------------------
```




```
Epoch: [151/200][0/4]   LR: 0.01        Loss 0.2042 (0.2042)    Train Acc 94.9
 * Train Acc 93.750
Test: [0/2]     Loss 0.1388 (0.1388)    Prec@1 96.094 (96.094)
 * Test Acc 95.333
-----------------------------------------------
Epoch: [152/200][0/4]   LR: 0.01        Loss 0.2171 (0.2171)    Train Acc 93.7
 * Train Acc 95.000
```

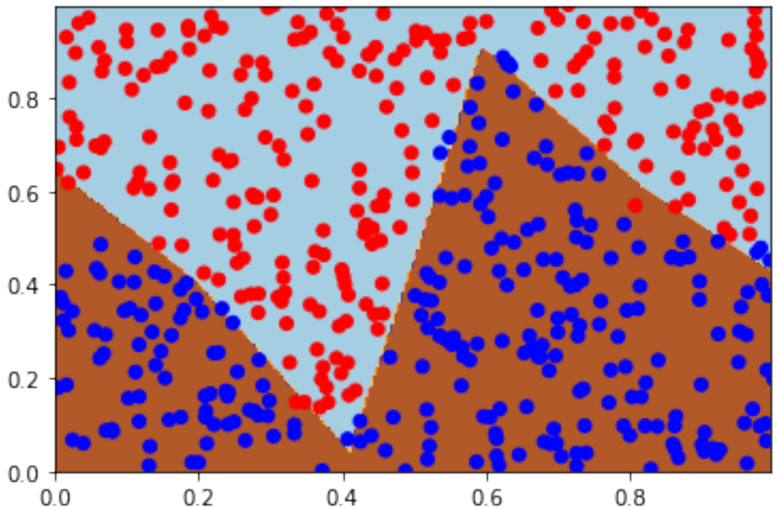
```
 * Train Acc 95.000
Test: [0/2]     Loss 0.1498 (0.1498)    Prec@1 95.312 (95.312)
 * Test Acc 95.000
----------------------------------------------
Epoch: [153/200][0/4]   LR: 0.01        Loss 0.1870 (0.1870)    Train Acc 94.1
 * Train Acc 93.750
Test: [0/2]     Loss 0.2010 (0.2010)    Prec@1 90.234 (90.234)
 * Test Acc 90.333
----------------------------------------------
Epoch: [154/200][0/4]   LR: 0.01        Loss 0.1872 (0.1872)    Train Acc 93.7
 * Train Acc 91.000
Test: [0/2]     Loss 0.1505 (0.1505)    Prec@1 94.531 (94.531)
 * Test Acc 93.667
----------------------------------------------
Epoch: [155/200][0/4]   LR: 0.01        Loss 0.2420 (0.2420)    Train Acc 87.5
 * Train Acc 89.375
Test: [0/2]     Loss 0.2146 (0.2146)    Prec@1 90.234 (90.234)
 * Test Acc 91.000
----------------------------------------------
Epoch: [156/200][0/4]   LR: 0.01        Loss 0.1973 (0.1973)    Train Acc 92.5
 * Train Acc 90.250
Test: [0/2]     Loss 0.1585 (0.1585)    Prec@1 94.141 (94.141)
 * Test Acc 93.333
----------------------------------------------
Epoch: [157/200][0/4]   LR: 0.01        Loss 0.2145 (0.2145)    Train Acc 91.7
 * Train Acc 89.625
Test: [0/2]     Loss 0.1835 (0.1835)    Prec@1 92.578 (92.578)
 * Test Acc 92.333
----------------------------------------------
Epoch: [158/200][0/4]   LR: 0.01        Loss 0.2387 (0.2387)    Train Acc 92.5
 * Train Acc 91.125
Test: [0/2]     Loss 0.1578 (0.1578)    Prec@1 92.578 (92.578)
 * Test Acc 92.333
----------------------------------------------
Epoch: [159/200][0/4]   LR: 0.01        Loss 0.2176 (0.2176)    Train Acc 91.4
 * Train Acc 92.375
Test: [0/2]     Loss 0.1748 (0.1748)    Prec@1 93.750 (93.750)
 * Test Acc 94.333
----------------------------------------------
Epoch: [160/200][0/4]   LR: 0.01        Loss 0.2122 (0.2122)    Train Acc 92.5
 * Train Acc 93.125
Test: [0/2]     Loss 0.1581 (0.1581)    Prec@1 92.969 (92.969)
 * Test Acc 93.667
----------------------------------------------
```





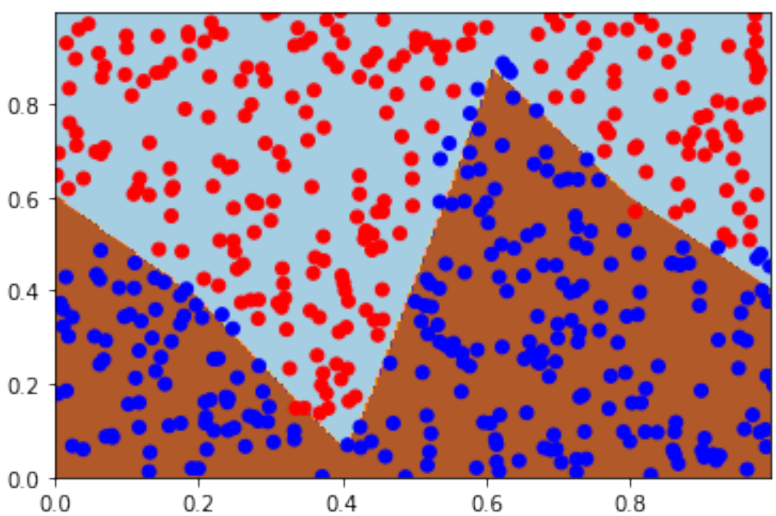

```
Epoch: [161/200][0/4]   LR: 0.01        Loss 0.2286 (0.2286)    Train Acc 92.5
 * Train Acc 91.375
Test: [0/2]     Loss 0.1664 (0.1664)    Prec@1 95.703 (95.703)
 * Test Acc 95.667
-----------------------------------------------
Epoch: [162/200][0/4]   LR: 0.01        Loss 0.2144 (0.2144)    Train Acc 94.5
 * Train Acc 92.250
Test: [0/2]     Loss 0.1343 (0.1343)    Prec@1 94.922 (94.922)
 * Test Acc 94.000
-----------------------------------------------
Epoch: [163/200][0/4]   LR: 0.01        Loss 0.1520 (0.1520)    Train Acc 94.5
 * Train Acc 92.500
Test: [0/2]     Loss 0.1548 (0.1548)    Prec@1 94.922 (94.922)
 * Test Acc 95.000
-----------------------------------------------
Epoch: [164/200][0/4]   LR: 0.01        Loss 0.1652 (0.1652)    Train Acc 95.3
 * Train Acc 92.750
Test: [0/2]     Loss 0.1618 (0.1618)    Prec@1 93.750 (93.750)
 * Test Acc 94.333
-----------------------------------------------
Epoch: [165/200][0/4]   LR: 0.01        Loss 0.2085 (0.2085)    Train Acc 92.1
 * Train Acc 92.875
Test: [0/2]     Loss 0.1450 (0.1450)    Prec@1 93.750 (93.750)
 * Test Acc 93.333
-----------------------------------------------
Epoch: [166/200][0/4]   LR: 0.01        Loss 0.1920 (0.1920)    Train Acc 93.3
 * Train Acc 93.625
Test: [0/2]     Loss 0.1707 (0.1707)    Prec@1 94.922 (94.922)
 * Test Acc 95.000
-----------------------------------------------
Epoch: [167/200][0/4]   LR: 0.01        Loss 0.2224 (0.2224)    Train Acc 91.0
 * Train Acc 93.500
Test: [0/2]     Loss 0.1444 (0.1444)    Prec@1 95.703 (95.703)
 * Test Acc 94.667
-----------------------------------------------
Epoch: [168/200][0/4]   LR: 0.01        Loss 0.1706 (0.1706)    Train Acc 96.0
 * Train Acc 94.125
Test: [0/2]     Loss 0.1509 (0.1509)    Prec@1 95.312 (95.312)
 * Test Acc 95.667
-----------------------------------------------
Epoch: [169/200][0/4]   LR: 0.01        Loss 0.2175 (0.2175)    Train Acc 92.9
 * Train Acc 92.500
Test: [0/2]     Loss 0.1699 (0.1699)    Prec@1 94.922 (94.922)
```

```
 * Test Acc 95.667
----------------------------------------------
Epoch: [170/200][0/4]   LR: 0.01        Loss 0.2174 (0.2174)    Train Acc 93.3
 * Train Acc 94.750
Test: [0/2]     Loss 0.1414 (0.1414)    Prec@1 94.922 (94.922)
 * Test Acc 94.667
----------------------------------------------
```



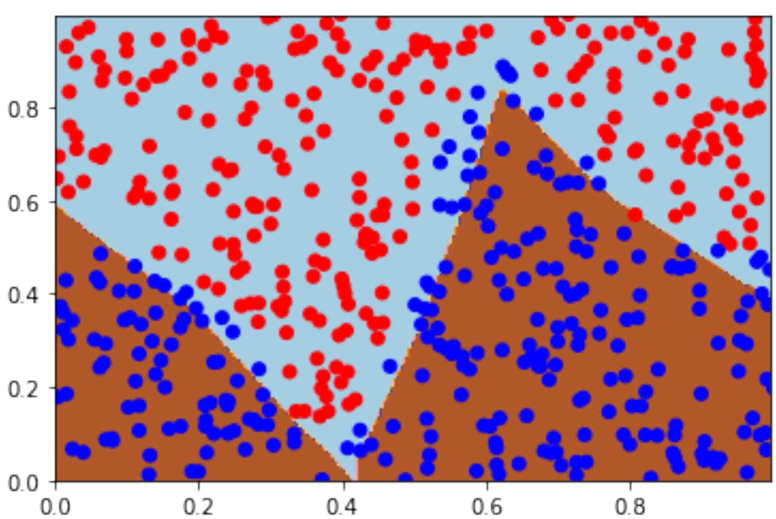

```
Epoch: [171/200][0/4]   LR: 0.01        Loss 0.1552 (0.1552)    Train Acc 95.7
 * Train Acc 94.250
Test: [0/2]     Loss 0.1675 (0.1675)    Prec@1 94.531 (94.531)
 * Test Acc 95.000
----------------------------------------------
Epoch: [172/200][0/4]   LR: 0.01        Loss 0.2422 (0.2422)    Train Acc 91.7
 * Train Acc 93.250
Test: [0/2]     Loss 0.1458 (0.1458)    Prec@1 94.922 (94.922)
 * Test Acc 94.333
----------------------------------------------
Epoch: [173/200][0/4]   LR: 0.01        Loss 0.1571 (0.1571)    Train Acc 95.3
 * Train Acc 94.500
Test: [0/2]     Loss 0.1622 (0.1622)    Prec@1 93.359 (93.359)
 * Test Acc 94.333
----------------------------------------------
Epoch: [174/200][0/4]   LR: 0.01        Loss 0.2177 (0.2177)    Train Acc 93.7
 * Train Acc 94.375
Test: [0/2]     Loss 0.1571 (0.1571)    Prec@1 94.922 (94.922)
 * Test Acc 95.333
----------------------------------------------
Epoch: [175/200][0/4]   LR: 0.01        Loss 0.2299 (0.2299)    Train Acc 93.7
 * Train Acc 94.625
Test: [0/2]     Loss 0.1554 (0.1554)    Prec@1 95.312 (95.312)
 * Test Acc 95.000
----------------------------------------------
Epoch: [176/200][0/4]   LR: 0.01        Loss 0.1691 (0.1691)    Train Acc 94.9
 * Train Acc 93.500
Test: [0/2]     Loss 0.1642 (0.1642)    Prec@1 95.312 (95.312)
```

```
 * Test Acc 95.333
----------------------------------------------
Epoch: [177/200][0/4]   LR: 0.01        Loss 0.1581 (0.1581)    Train Acc 96.(
 * Train Acc 94.750
Test: [0/2]     Loss 0.1382 (0.1382)    Prec@1 95.312 (95.312)
 * Test Acc 94.000
----------------------------------------------
Epoch: [178/200][0/4]   LR: 0.01        Loss 0.2244 (0.2244)    Train Acc 92.9
 * Train Acc 94.375
Test: [0/2]     Loss 0.1600 (0.1600)    Prec@1 94.141 (94.141)
 * Test Acc 94.333
----------------------------------------------
Epoch: [179/200][0/4]   LR: 0.01        Loss 0.1752 (0.1752)    Train Acc 94.9
 * Train Acc 94.000
Test: [0/2]     Loss 0.1501 (0.1501)    Prec@1 94.141 (94.141)
 * Test Acc 94.667
----------------------------------------------
Epoch: [180/200][0/4]   LR: 0.01        Loss 0.1420 (0.1420)    Train Acc 96.4
 * Train Acc 93.750
Test: [0/2]     Loss 0.1357 (0.1357)    Prec@1 93.750 (93.750)
 * Test Acc 94.000
----------------------------------------------
```

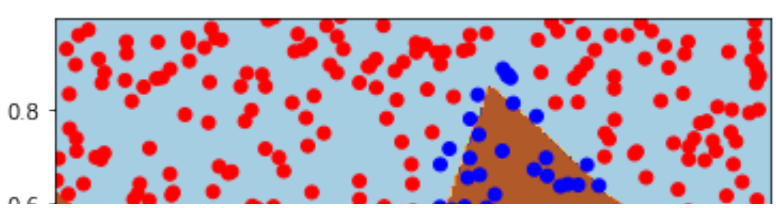


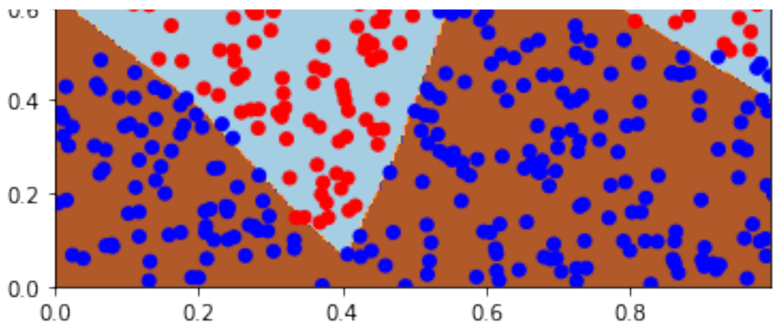

```
Epoch: [181/200][0/4]   LR: 0.01        Loss 0.1961 (0.1961)    Train Acc 94.5
 * Train Acc 94.000
Test: [0/2]     Loss 0.1513 (0.1513)    Prec@1 92.969 (92.969)
 * Test Acc 93.000
----------------------------------------------
Epoch: [182/200][0/4]   LR: 0.01        Loss 0.2542 (0.2542)    Train Acc 90.6
 * Train Acc 93.125
Test: [0/2]     Loss 0.1334 (0.1334)    Prec@1 94.531 (94.531)
 * Test Acc 94.667
----------------------------------------------
Epoch: [183/200][0/4]   LR: 0.01        Loss 0.1895 (0.1895)    Train Acc 93.3
 * Train Acc 94.000
Test: [0/2]     Loss 0.1548 (0.1548)    Prec@1 94.922 (94.922)
 * Test Acc 94.667
```

```
--------------------------------------------------
Epoch: [184/200][0/4]   LR: 0.01        Loss 0.1913 (0.1913)    Train Acc 94.5
 * Train Acc 93.875
Test: [0/2]     Loss 0.1211 (0.1211)    Prec@1 94.922 (94.922)
 * Test Acc 94.000
--------------------------------------------------
Epoch: [185/200][0/4]   LR: 0.01        Loss 0.2075 (0.2075)    Train Acc 92.9
 * Train Acc 93.125
Test: [0/2]     Loss 0.1865 (0.1865)    Prec@1 91.406 (91.406)
 * Test Acc 90.333
--------------------------------------------------
Epoch: [186/200][0/4]   LR: 0.01        Loss 0.2068 (0.2068)    Train Acc 91.7
 * Train Acc 92.125
Test: [0/2]     Loss 0.1512 (0.1512)    Prec@1 93.750 (93.750)
 * Test Acc 93.333
--------------------------------------------------
Epoch: [187/200][0/4]   LR: 0.01        Loss 0.2364 (0.2364)    Train Acc 91.0
 * Train Acc 91.875
Test: [0/2]     Loss 0.2172 (0.2172)    Prec@1 90.234 (90.234)
 * Test Acc 89.667
--------------------------------------------------
Epoch: [188/200][0/4]   LR: 0.01        Loss 0.1660 (0.1660)    Train Acc 94.9
 * Train Acc 91.250
Test: [0/2]     Loss 0.1723 (0.1723)    Prec@1 92.969 (92.969)
 * Test Acc 93.000
--------------------------------------------------
Epoch: [189/200][0/4]   LR: 0.01        Loss 0.2113 (0.2113)    Train Acc 90.6
 * Train Acc 90.750
Test: [0/2]     Loss 0.2103 (0.2103)    Prec@1 90.625 (90.625)
 * Test Acc 90.333
--------------------------------------------------
Epoch: [190/200][0/4]   LR: 0.01        Loss 0.2030 (0.2030)    Train Acc 92.9
 * Train Acc 92.625
Test: [0/2]     Loss 0.1417 (0.1417)    Prec@1 94.531 (94.531)
 * Test Acc 93.667
--------------------------------------------------
```



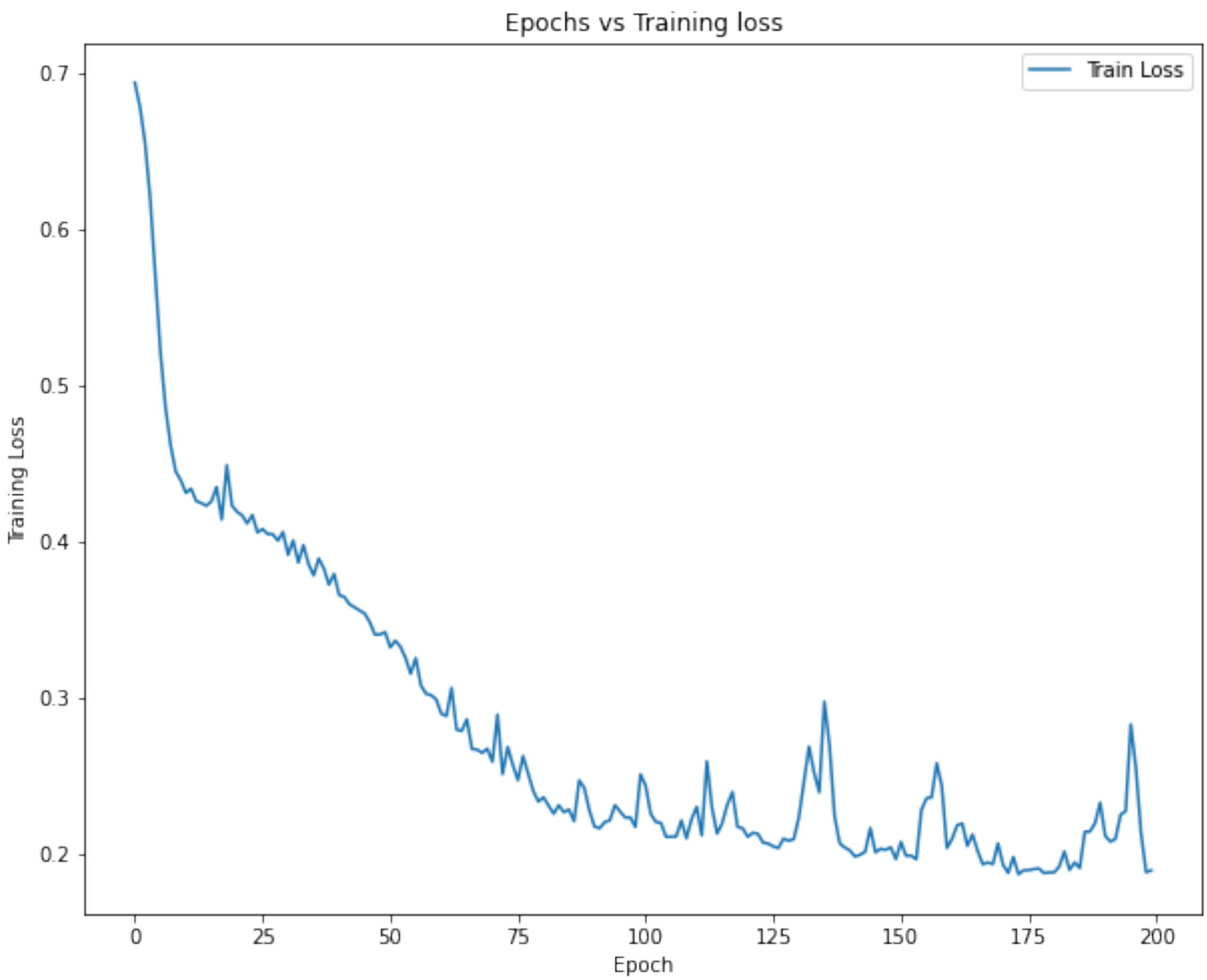
```
Epoch: [191/200][0/4]   LR: 0.01        Loss 0.2216 (0.2216)    Train Acc 91.7
 * Train Acc 92.000
Test: [0/2]     Loss 0.2124 (0.2124)    Prec@1 91.016 (91.016)
 * Test Acc 91.667
----------------------------------------------
Epoch: [192/200][0/4]   LR: 0.01        Loss 0.2258 (0.2258)    Train Acc 92.1
 * Train Acc 92.625
Test: [0/2]     Loss 0.1648 (0.1648)    Prec@1 92.578 (92.578)
 * Test Acc 93.000
----------------------------------------------
Epoch: [193/200][0/4]   LR: 0.01        Loss 0.2372 (0.2372)    Train Acc 90.2
 * Train Acc 91.125
Test: [0/2]     Loss 0.1942 (0.1942)    Prec@1 92.188 (92.188)
 * Test Acc 91.667
----------------------------------------------
Epoch: [194/200][0/4]   LR: 0.01        Loss 0.2520 (0.2520)    Train Acc 89.8
 * Train Acc 91.500
Test: [0/2]     Loss 0.1792 (0.1792)    Prec@1 92.578 (92.578)
 * Test Acc 93.000
----------------------------------------------
Epoch: [195/200][0/4]   LR: 0.01        Loss 0.1848 (0.1848)    Train Acc 92.5
 * Train Acc 88.750
Test: [0/2]     Loss 0.1971 (0.1971)    Prec@1 92.578 (92.578)
 * Test Acc 93.000
----------------------------------------------
Epoch: [196/200][0/4]   LR: 0.01        Loss 0.2719 (0.2719)    Train Acc 91.4
 * Train Acc 89.750
Test: [0/2]     Loss 0.1479 (0.1479)    Prec@1 94.141 (94.141)
 * Test Acc 94.667
----------------------------------------------
Epoch: [197/200][0/4]   LR: 0.01        Loss 0.1864 (0.1864)    Train Acc 93.3
 * Train Acc 92.000
Test: [0/2]     Loss 0.1382 (0.1382)    Prec@1 94.531 (94.531)
 * Test Acc 95.000
----------------------------------------------
Epoch: [198/200][0/4]   LR: 0.01        Loss 0.1945 (0.1945)    Train Acc 94.5
 * Train Acc 94.250
Test: [0/2]     Loss 0.1652 (0.1652)    Prec@1 94.531 (94.531)
 * Test Acc 94.333
----------------------------------------------
Epoch: [199/200][0/4]   LR: 0.01        Loss 0.2113 (0.2113)    Train Acc 93.3
 * Train Acc 94.375
Test: [0/2]     Loss 0.1478 (0.1478)    Prec@1 92.578 (92.578)
 * Test Acc 93.333
----------------------------------------------
```



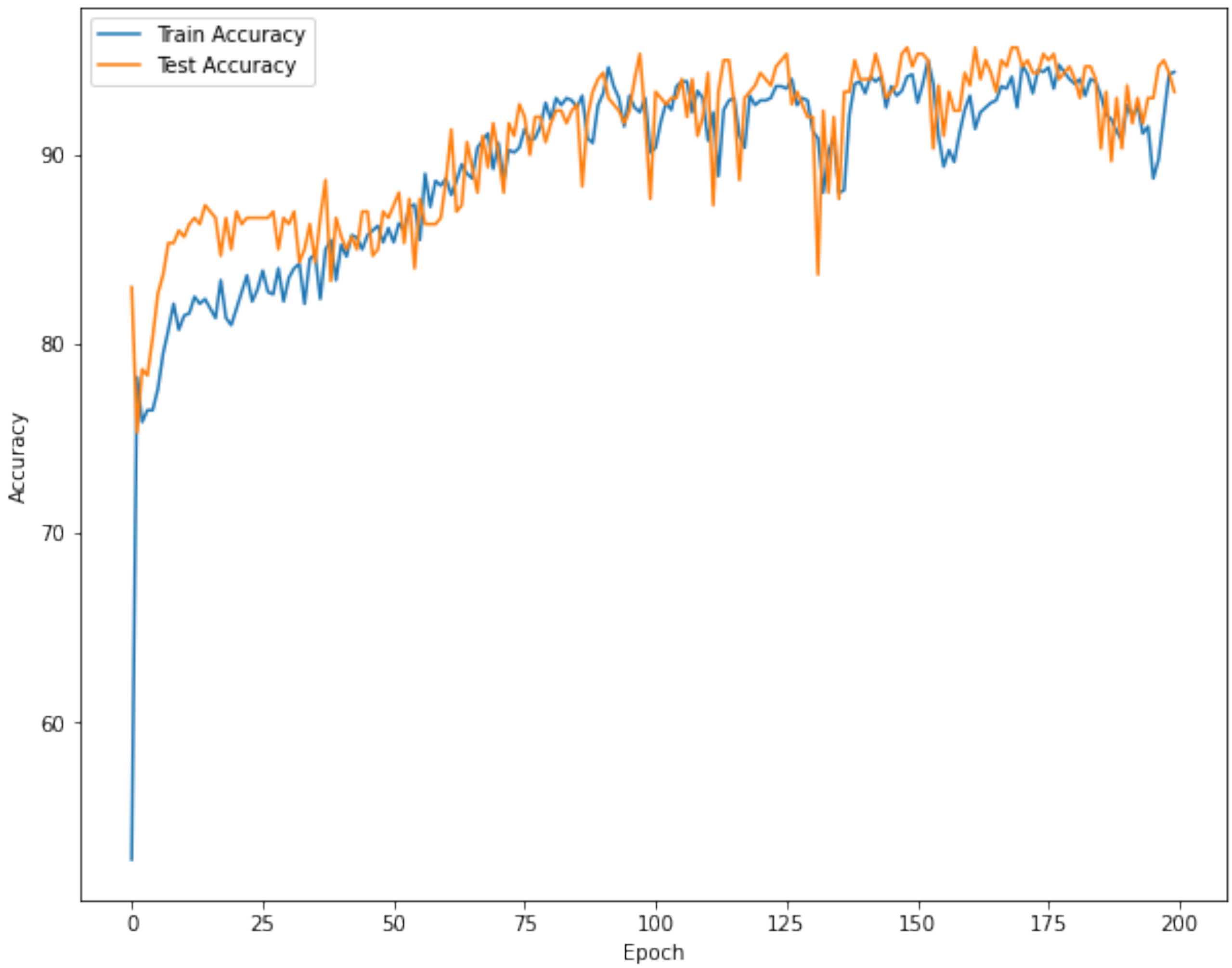


a)

```
rcParams['figure.figsize'] = 10, 8  # adjust the size of the plot
rcParams['xtick.labelsize'] = 10   # adjust font size for x-axis ticks
rcParams['ytick.labelsize'] = 10   # adjust font size for y-axis ticks
rcParams['legend.fontsize'] = 10   # adjust font size for legend

plt.plot(train_losses, label='Train Loss')
plt.xlabel('Epoch')
plt.ylabel('Training Loss')
plt.title('Epochs vs Training loss')
plt.legend()
plt.show()
```



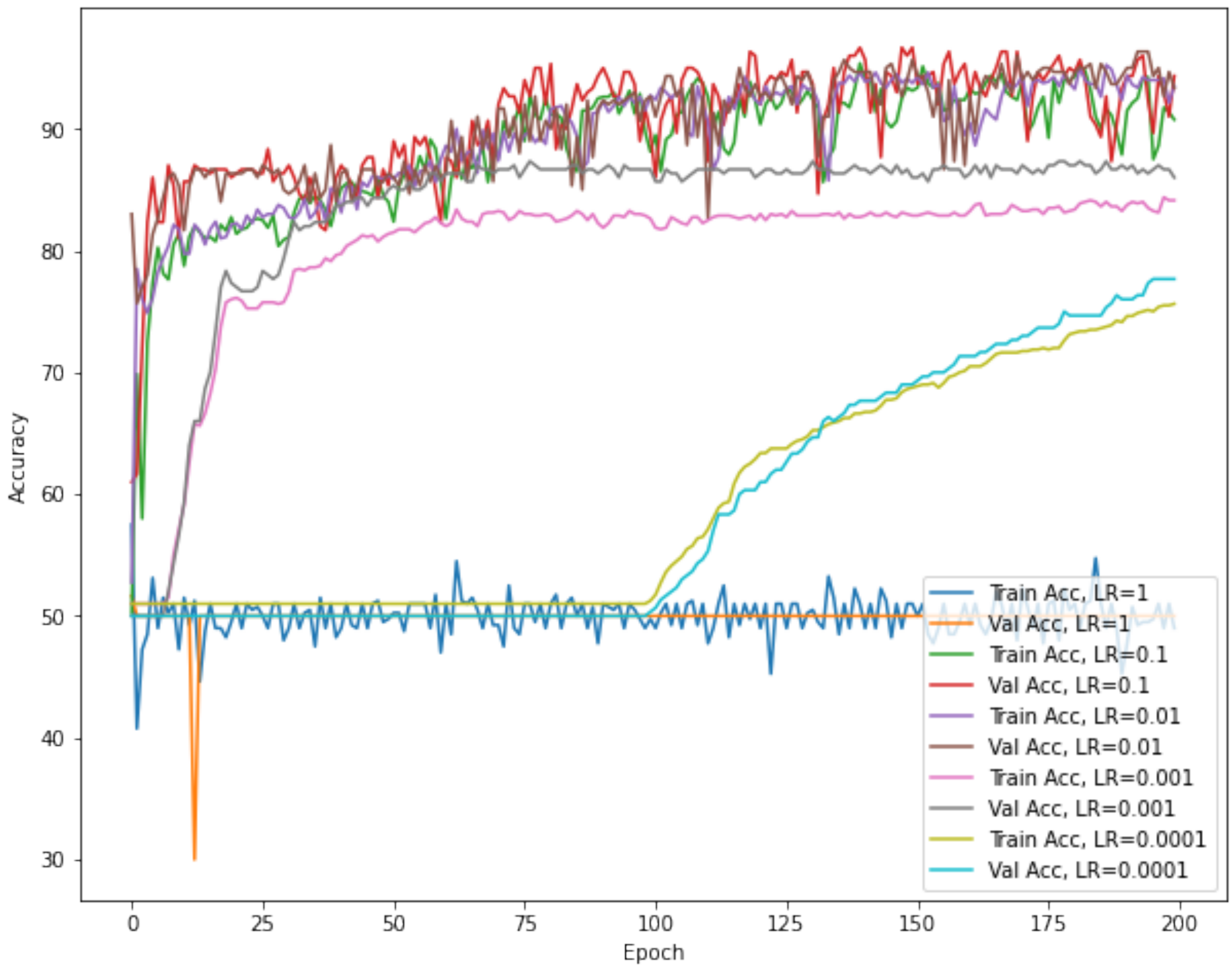
```
rcParams['figure.figsize'] = 10, 8  # adjust the size of the plot
rcParams['xtick.labelsize'] = 10   # adjust font size for x-axis ticks
rcParams['ytick.labelsize'] = 10   # adjust font size for y-axis ticks
rcParams['legend.fontsize'] = 10   # adjust font size for legend

plt.plot(train_accuracies, label='Train Accuracy')
plt.plot(test_accuracies, label='Test Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



b)

```
lrs = [1, 0.1, 0.01, 0.001, 0.0001]

train_losses = []
train_accuracies = []
val_accuracies = []

best_val_acc = 0
best_lr = 0

for lr in lrs:
    # create new model and optimizer with the current learning rate
    torch.manual_seed(999)
    model = linear_nn(num_neurons, activations).to(device)
    criterion = nn.CrossEntropyLoss().to(device)
    optimizer = torch.optim.Adam(model.parameters(), lr=lr, weight_decay=weight_dec

    # train the model for num_epochs and record the accuracy
    train_acc, val_acc = [], []
    for epoch in range(num_epochs):
        train_loss, train_acc_epoch = train(train_loader, model, criterion, optimiz
        val_acc_epoch = validate(val_loader, model, criterion)
        train_acc.append(train_acc_epoch)
        val_acc.append(val_acc_epoch)

        # update best validation accuracy and learning rate
        if val_acc_epoch > best_val_acc:
            best_val_acc = val_acc_epoch
            best_lr = lr

    # store the accuracy for the current learning rate
    train_losses.append(train_loss)
    train_accuracies.append(train_acc)
    val_accuracies.append(val_acc)
```

```
# plot the training and validation accuracy curves for each learning rate
for i, lr in enumerate(lrs):
    plt.plot(train_accuracies[i], label='Train Acc, LR={}'.format(lr))
    plt.plot(val_accuracies[i], label='Val Acc, LR={}'.format(lr))

plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

print("Best learning rate is:", best_lr)
```




```
Best learning rate is: 0.1
```

Based on the validation accuracy, we determined that the best learning rate is 0.1. It is possible that different learning rates may perform better or worse depending on the specific problem and dataset being used, but in this case, based on the given code and data, 0.1 appears to be the best choice.

c)

```
def linear_nn_one_layer():
    model = LinearNN(num_neurons=[2, 100, 2], activations=['relu'])
    return model

# Create new model
model_new = linear_nn_one_layer()

# Define optimizer for new model
optimizer_new = torch.optim.Adam(model_new.parameters(), lr=lr, weight_decay=weight



# Train new model
train_losses_new = []
train_accuracies_new = []
test_accuracies_new = []

best_prec1_new = 0
for epoch in range(num_epochs):
    if epoch in lr_step:
        for param_group in optimizer_new.param_groups:
            param_group['lr'] *= 0.1

    # train for one epoch
    train_loss, train_acc = train(train_loader, model_new, criterion, optimizer_new
    train_losses_new.append(train_loss)
    train_accuracies_new.append(train_acc)

    # evaluate on validation set
    prec1_new = validate(val_loader, model_new, criterion)
    test_accuracies_new.append(prec1_new)

    # remember best prec@1 and save checkpoint
    is_best_new = prec1_new > best_prec1_new
```

```
    best_prec1_new = max(prec1_new, best_prec1_new)

# Print final training and testing accuracies and number of parameters
print('Model with 3 hidden layers:')
print('Train Accuracy: {:.2f}%'.format(train_accuracies[-1]))
print('Test Accuracy: {:.2f}%'.format(test_accuracies[-1]))
print('Number of parameters: {}'.format(sum(p.numel() for p in model.parameters()))

print('Model with 1 hidden layer:')
print('Train Accuracy: {:.2f}%'.format(train_accuracies_new[-1]))
print('Test Accuracy: {:.2f}%'.format(test_accuracies_new[-1]))
print('Number of parameters: {}'.format(sum(p.numel() for p in model_new.parameters
```

```
 * Train Acc 81.250
Test: [0/2]     Loss 0.4958 (0.4958)    Prec@1 86.328 (86.328)
 * Test Acc 86.667
Epoch: [188/200][0/4]   LR: 0.0001      Loss 0.4951 (0.4951)    Train Acc 83.9
 * Train Acc 81.250
Test: [0/2]     Loss 0.4970 (0.4970)    Prec@1 85.547 (85.547)
 * Test Acc 86.667
Epoch: [189/200][0/4]   LR: 0.0001      Loss 0.5143 (0.5143)    Train Acc 83.5
 * Train Acc 81.500
Test: [0/2]     Loss 0.4929 (0.4929)    Prec@1 86.719 (86.719)
 * Test Acc 86.667
Epoch: [190/200][0/4]   LR: 0.0001      Loss 0.5164 (0.5164)    Train Acc 79.6
 * Train Acc 81.625
Test: [0/2]     Loss 0.4909 (0.4909)    Prec@1 86.719 (86.719)
 * Test Acc 86.667
Epoch: [191/200][0/4]   LR: 0.0001      Loss 0.5194 (0.5194)    Train Acc 79.2
 * Train Acc 81.750
Test: [0/2]     Loss 0.4893 (0.4893)    Prec@1 87.500 (87.500)
 * Test Acc 86.667
Epoch: [192/200][0/4]   LR: 0.0001      Loss 0.5144 (0.5144)    Train Acc 83.5
 * Train Acc 81.750
Test: [0/2]     Loss 0.4790 (0.4790)    Prec@1 88.672 (88.672)
 * Test Acc 86.667
Epoch: [193/200][0/4]   LR: 0.0001      Loss 0.4839 (0.4839)    Train Acc 86.7
 * Train Acc 81.750
Test: [0/2]     Loss 0.4877 (0.4877)    Prec@1 85.547 (85.547)
 * Test Acc 86.667
Epoch: [194/200][0/4]   LR: 0.0001      Loss 0.5207 (0.5207)    Train Acc 81.6
 * Train Acc 81.750
Test: [0/2]     Loss 0.4751 (0.4751)    Prec@1 87.500 (87.500)
 * Test Acc 86.667
Epoch: [195/200][0/4]   LR: 0.0001      Loss 0.5154 (0.5154)    Train Acc 81.6
 * Train Acc 81.625
Test: [0/2]     Loss 0.4777 (0.4777)    Prec@1 88.672 (88.672)
```

```
 * Test Acc 86.667
Epoch: [196/200][0/4]   LR: 0.0001      Loss 0.4885 (0.4885)    Train Acc 84.
 * Train Acc 81.625
Test: [0/2]     Loss 0.4816 (0.4816)    Prec@1 86.719 (86.719)
 * Test Acc 86.667
Epoch: [197/200][0/4]   LR: 0.0001      Loss 0.5235 (0.5235)    Train Acc 80.
 * Train Acc 81.625
Test: [0/2]     Loss 0.4784 (0.4784)    Prec@1 87.891 (87.891)
 * Test Acc 86.667
Epoch: [198/200][0/4]   LR: 0.0001      Loss 0.5057 (0.5057)    Train Acc 82.
 * Train Acc 81.750
Test: [0/2]     Loss 0.4802 (0.4802)    Prec@1 87.891 (87.891)
 * Test Acc 86.333
Epoch: [199/200][0/4]   LR: 0.0001      Loss 0.4861 (0.4861)    Train Acc 84.
 * Train Acc 81.750
Test: [0/2]     Loss 0.4826 (0.4826)    Prec@1 86.328 (86.328)
 * Test Acc 86.000
Model with 3 hidden layers:
Train Accuracy: 78.00%
Test Accuracy: 80.33%
Number of parameters: 402
Model with 1 hidden layer:
Train Accuracy: 81.75%
Test Accuracy: 86.00%
Number of parameters: 502
```

The model with 1 hidden layer containing 100 neurons performs better than the model with 3 hidden layers. The test accuracy of the model with 1 hidden layer is higher than that of the model with 3 hidden layers. Additionally, the model with 1 hidden layer has more parameters than the model with 3 hidden layers, indicating that it is more expressive and can capture more complex patterns in the data. Therefore, the shallow model with 1 hidden layer is better than the deep model with 3 hidden layers in this case.

✓ 4s completed at 10:56 PM