Lecture 9编程作业

1.

(1) IRIS数据集

给定IRIS数据集，该数据集有三类目标，每个类别有50个样本，每个样本有四维特征。实验时每个类别随机选30个样本进行训练，另外20个样本用于测试。

ovo与softmax算法见 iris.py

(2) ovo多类分类器

测试样本标签与预测标签分别为

```
test_label    = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
predict_label = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 2 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
```

分类正确率：0.966666667

(3) softmax分类器

测试样本标签与预测标签分别为

```
test_label    = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
predict_label = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 2 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
```

分类正确率：0.983333333

2.MNIST_SoftMax

(1) MNIST数据集

给定MNIST数据集，该数据集每个样本为28*28大小的灰度图像，有0到9共10个类别的手写体数字，其中训练样本60000，测试样本10000，编写Softmax算法对该数据集实现分类，权向量初始值由均值为0、标准差为0.01的正态分布产生的随机数得到

(2) softmax核心算法

算法见mnist_softmax.py

```
    def train(self,epoch,alpha):
        t=0
        ac=np.zeros(epoch)
        L=np.zeros(epoch)
        test_ac=np.zeros(epoch)
        while t<epoch:
            count=0
            for x, y in self.train_iter:    #读取每个batch中的数据
                cnt=0
```

```
                y_hat = self.Model(X)
                L[t]+=cross_entropy(y_hat, y).sum()
                ac[t]+=self.accuracy(y_hat,y)
                one_hot = torch.zeros(len(y), 10)
                #转换为独热编码
                for i in one_hot:
                    i[y[cnt]] = 1
                    cnt += 1
                #进行权系数更新
                b_diff=1/X.shape[0]*torch.mm(torch.ones(1,X.shape[0]),(y_hat-
one_hot))
                w_diff=1/X.shape[0]*torch.mm(X.view((-1, self.num_inputs)).T,
(y_hat-one_hot))
                w_t=self.w-alpha*w_diff
                b_t=self.b-alpha*b_diff
                #计算权系数变化量
                deltaw=torch.norm(w_t-self.w)
                deltab=torch.norm(b_t-self.b)
                #计算错分样本数
                misindex = torch.nonzero(torch.argmax(y_hat, axis = 1)!=y)
                misnum = len(misindex)
                #如果错分数为0或者权系数不再改变，则停止迭代
                if misnum==0 or (deltaw+deltab<1e-5):
                    break
                # 否则，继续迭代
                else:
                    self.w = w_t
                    self.b = b_t
                count+=1

            L[t]=L[t]/count
            ac[t]=ac[t]/count
            test_ac[t]=self.test()
            t+=1
        epoch_plot(epoch,L,ac,test_ac)    #画出迭代函数曲线
        print("mnist_train accuracy:",ac[epoch-1])
        print("mnist_test accuracy:",test_ac[epoch-1])
```

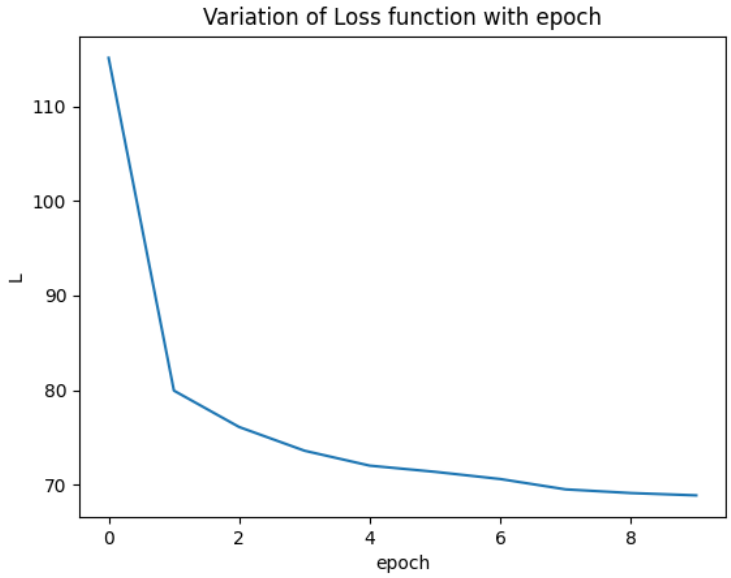
(3) 测试集的分类精度

$$Accuracy_{(test)} = 0.9217890625$$

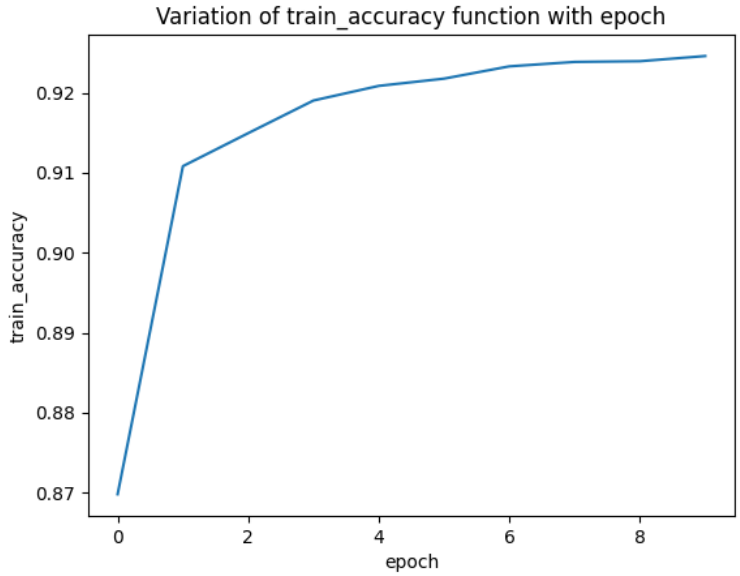
(4) 画出训练时的损失函数、训练集上的分类精度和测试集上的分类精度随epoch增加的变化曲线

训练时的batch size为256，一共训练10遍epoch

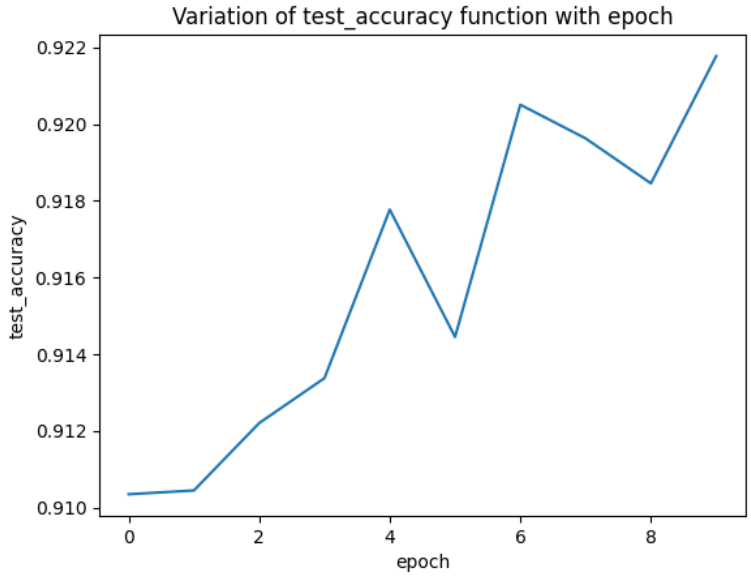
损失函数随迭代次数的变化如图：



训练集分类精度随迭代次数的变化如图：



测试集分类精度随迭代次数的变化如图：



(5) 测试集上随机抽取10个样本，观察分类结果

```
测试集标签:tensor([7., 2., 1., 0., 4., 1., 4., 9., 5., 9.])
预测标签  :tensor([7., 2., 1., 0., 4., 1., 4., 9., 6., 9.])
```