机器学习实验二实验报告

# 1 BP神经网络

BP神经网络(Back-Propagation Neural Network)是最传统的神经网络，是我们理解许多复杂网络的基础，其也被称为多层感知机MLP(Multi-Layer Percepton)。本实验使用python中的scikit-learn工具包对神经网络进行简单实现。

from sklearn.neural\_network import MLPClassifier

clf = MLPClassifier(solver='adam', alpha=1e-5,

hidden\_layer\_sizes = hidden\_layer\_sizes, random\_state=1)

def BPtraining(clf, name, X\_train, y\_train, X\_test, y\_test):

clf.fit(X\_train, y\_train)

pre = clf.predict(X\_test)

…

size = [50, 100, 150]

layers = [2, 4, 6]

for i in size:

for j in layers:

hidden\_layer\_sizes = (i,) \* j

clf = MLPClassifier(solver='adam', alpha=1e-5,

hidden\_layer\_sizes=hidden\_layer\_sizes, random\_state=1)

name = str(hidden\_layer\_sizes)

info = BPtraining(clf, name, X\_train, y\_train, X\_test, y\_test)

针对给定的数据集，随机分30%为test集，用于验证模型精度，表1即为所得结果。中使用含有50神经元的6层隐含层的BP神经网络取得最高精度，总体而言精度相差不大。

表1 参数选择

|  |  |
| --- | --- |
| 参数 | 精度 |
| 隐含层2，50神经元 | 96.76% |
| 隐含层2，100神经元 | 96.76% |
| 隐含层2，150神经元 | 96.98% |
| 隐含层4，50神经元 | 96.76% |
| 隐含层4，100神经元 | 95.68% |
| 隐含层4，150神经元 | 96.98% |
| 隐含层6，50神经元 | 97.08% |
| 隐含层6，100神经元 | 96.98% |
| 隐含层6，150神经元 | 96.66% |

使用默认的随机森林模型与支持向量机模型对同样的使用数据做对比得出表2。总体而言，随机森林的精度最高，而支持向量机回归的精度较低。BP神经网络则处于中间。

表2 模型对比

|  |  |
| --- | --- |
| 模型 | 精度 |
| BP神经网络 | 96.76% |
| 支持向量机 | 96.01% |
| 随机森林 | 96.97% |

# 2 卷积神经网络的实现——手写数字识别

使用MNIST手写数字数据集对LeNet-5进行测试。要求更改CNN结构，第一卷积层padding=1，程序不可以继续运行，这是因为改变第一层卷积的padding后使得数据的大小在进入全连接层发生了变化，数组匹配不上，可以进行如下的改法。

self.linear1 = nn.Linear(16 \* 4 \* 4, 120)

out = out.view(-1, 16 \* 4 \* 4)

之后测试5次和10次的迭代精度，代码如下。

for iterTime in [5, 10]:

model = LeNet(num\_classes)

criterion = nn.CrossEntropyLoss()

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

total\_step = len(train\_loader)

for epoch in range(iterTime):

…

如图1所示，约在迭代1500次左右精度收敛。

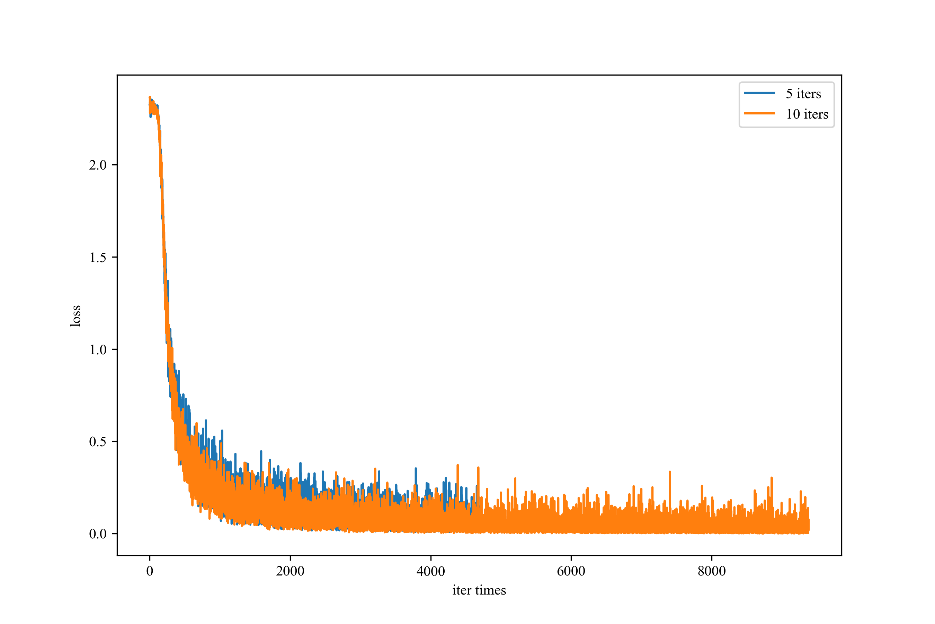


图1 精度曲线