学习算法报告

刘锦坤
2022013352

2024 年 5 月 25 日

目录

1 问题 1

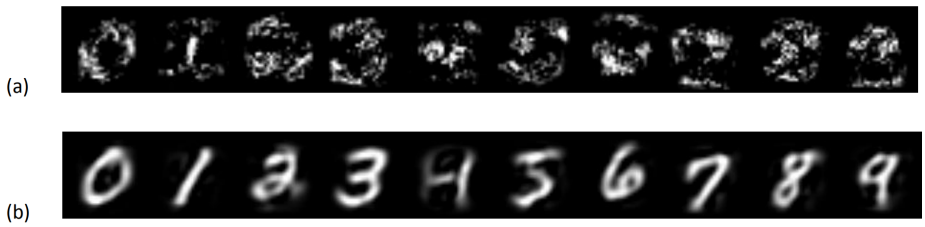

图 1: Question1

图 2: weights.png

根据生成的 weights.png（如图 2）可知，Question1 中，图 1中的图片 (a) 更能够代表感知机所学习到的权重矩阵。

可以看到，各个数字分类的权重向量在可视化后都呈现出对应的数字的轮廓的形状。这是因为对于每个输入数字，要能够将其通过 Softmax Regression 的方法将其正确分类，就需要其在输出层对应数字位置的值最大。在训练过程中，感知机根据输入的数字的特征，调整权重矩阵，使得输入

数字的特征与其对应的权重矩阵的乘积最大，从而使得输出层对应数字位置的值最大。在这个问题中，每个输入的特征就是其各个像素点的值，因此在对应数字越可能出现的位置，权重中对应的值就越大，从而呈现出对应数字的轮廓的形状。

但是因为即使是对应同一个数字的各个输入中，手写的数字的形状也是有所不同的，因此权重矩阵在可视化后不会表现成一个完全锐利的数字轮廓，而是一个较为模糊的数字轮廓，其反映的事实上是对应数字的各个输入中总体出现的较多的像素点的那些位置，这也就导致对于某些输入，当他们的写法偏离训练的输入写法较大时，不经过或较少经过这些权重矩阵集中分布的像素点，这种情况下，感知机就可能会产生错误的分类判断。

2 算法分析

2.1 K-Means 算法

K-Means 算法是一种基于距离的无监督学习算法，其目的是将数据集中的数据点划分为 K 个簇，使得每个数据点都属于离其最近的簇。

在 Question1 中，我们实现了 K-Means 算法，算法主要分为以下几个步骤：

1. 初始化 K 个簇的中心点，可以随机选择数据集中的 K 个点作为初始中心点。
2. 对于数据集中的每个点，计算其与 K 个簇中心点的距离，将其划分到距离最近的簇中。
3. 对于每个簇，重新计算其中心点，即将簇中所有点的坐标取平均值作为新的中心点。
4. 重复 2、3 步骤，直到达到最大迭代次数。
5. 输出 K 个簇的中心点。

可以看到，最后 K-Means 算法成功的完成了对数据集的聚类，将数据集中的数据点划分为 K 个簇，使得每个数据点都属于离其最近的簇，但是也可以看出 K-Means 算法有以下不足。

1. K 值的选择：K-Means 算法需要事先确定 K 值，本次是根据数据集的情况选择的 K 值，实际 K 值的选择可能要根据聚类相似性关于 K 值的变化情况来确定（a sharp drop at the optimal number of cluster）。
2. 对初始中心点的敏感性：K-Means 算法对初始中心点的选择是敏感的，本题中指定了随机化种子，但是不同的初始中心点很可能会导致不同的聚类结果。
3. 对异常值的敏感性：K-Means 算法对异常值比较敏感，异常值可能会导致聚类结果的不准确，可能需要数据预处理获得更好的聚类效果。

2.2 KNN 算法

KNN 算法是一种基于距离的有监督学习算法，其目的是对于一个新的数据点，找到与其最近的 K 个数据点，根据这 K 个数据点的标签，对新的数据点进行分类。

在 Question2 中，我们实现了 KNN 算法，算法主要分为以下几个步骤：

1. 计算新的数据点与数据集中所有数据点的距离。
2. 找到与新的数据点距离最近的 K 个数据点。

3. 根据这 K 个数据点的标签，对新的数据点进行分类。

4. 输出新的数据点的分类结果。

可以看到，最后 KNN 算法成功的完成了对新的数据点的分类，并且达到了 91% 准确率，但是也可以看出 KNN 算法有以下不足。

1. 计算量大：KNN 算法需要计算新的数据点与数据集中所有数据点的距离，计算量较大，当数据集较大时，计算时间会较长。

2. 对 K 值的选择敏感：KNN 算法中对超参数 K 值的选择是敏感的，K 值的选择可能会影响分类结果，需要根据数据集的情况选择合适的 K 值。

3. 依赖特征选择：由于 KNN 算法是基于特征空间中的距离进行分类的，因此对特征选择是敏感的，需要选择合适的特征来进行分类。

2.3 Softmax Regression 算法

Softmax Regression 算法是一种多分类的有监督学习算法，其目的是对于一个新的数据点，找到其在输出层对应数字位置的值最大的位置，从而对新的数据点进行分类。采用 Softmax 的方法，将输出层的值转化为概率，使得输出层的值和为 1 且能够进行反向传播计算梯度从而利用梯度下降法计算。

在 Question3 中，我们实现了 Softmax Regression 算法，并且在算法中加入了 L2 正则化项，以防止过拟合。最终达到 87% 的准确率。关于 Softmax Regression 算法，有如下总结：

1. Softmax Regression 算法超参数，例如学习率、学习轮次、正则化系数、批次大小，这些都会影响性能，需要根据数据集的情况进行调整。

2. Softmax Regression 算法同样依赖于特征选择，不难看出，如果特征空间中各个分类之间是线性可分的，那么 Softmax Regression 算法的性能会更好。

3. Softmax Regression 算法对于数据集的分布是敏感的，如果数据集的分布不均匀，那么 Softmax Regression 算法的性能会受到影响，例如数据集中如果数字 1 出现得更多，最后的分类结果也会更多的向数字 1 倾向。

2.4 SVM 算法

SVM 算法是一种有监督学习算法，其目的是找到一个最优的超平面，使得数据点能够被最大间隔分开。SVM 算法的优化目标是最大化间隔，同时使得数据点能够被正确分类。

在 Question4 中，我们通过调用 sklearn 库中的 SVM 算法，实现了对数据集的分类，最终的准确率为 93%，关于 SVM 算法，有如下总结：

1. SVM 算法本质上是一种线性分类器，但是通过核函数的选择，可以将 SVM 算法扩展到非线性分类器，例如多项式核函数、高斯核函数等，这一问中采用的是高斯核函数。

2. SVM 算法对于超参数的选择是敏感的，例如核函数的选择、正则化系数的选择、损失函数的选择，这些都会影响 SVM 算法的性能，都需要根据数据集的情况进行调整。

3. SVM 算法可以认为是一种最优的 Perceptron 算法，通过最大化间隔，使得数据点能够被最大间隔分开，从而使得分类效果更好。

2.5 Better Classifier

在这一问中，我们通过 pytorch 库，搭建了一个深度神经网络，实现了更好的数据集分类，最终的准确率约为 98%，网络结构设置如图 3所示：

1. 经过两层卷积层，每层卷积核的大小为 3，步长为 1，padding 为 1，激活函数为 ReLU。每层卷积层后接一个最大池化层，池化核的大小为 2，步长为 2，并在第二层卷积层后添加了 dropout 防止过拟合。
2. 然后将卷积层的输出展平，经过三层全连接层，前两层激活函数为 ReLU，经过第一层后添加 dropout，最后一层全连接层的激活函数为 Softmax 分类器。

在训练过程中，如果检测到设备的 GPU，就会使用 GPU 进行训练，否则使用 CPU 进行训练。在训练过程中，采用了交叉熵损失函数，采用了 SGD 优化器，超参数设置如下：

1. 学习率：0.1
2. 动量：0.5
3. 批次大小：200
4. 学习轮次：200

可以看到，最终的准确率达到了 98%，相比于之前的分类器，深度神经网络的分类效果更好，这是因为深度神经网络能够学习到更多的特征，从而使得分类效果更好。

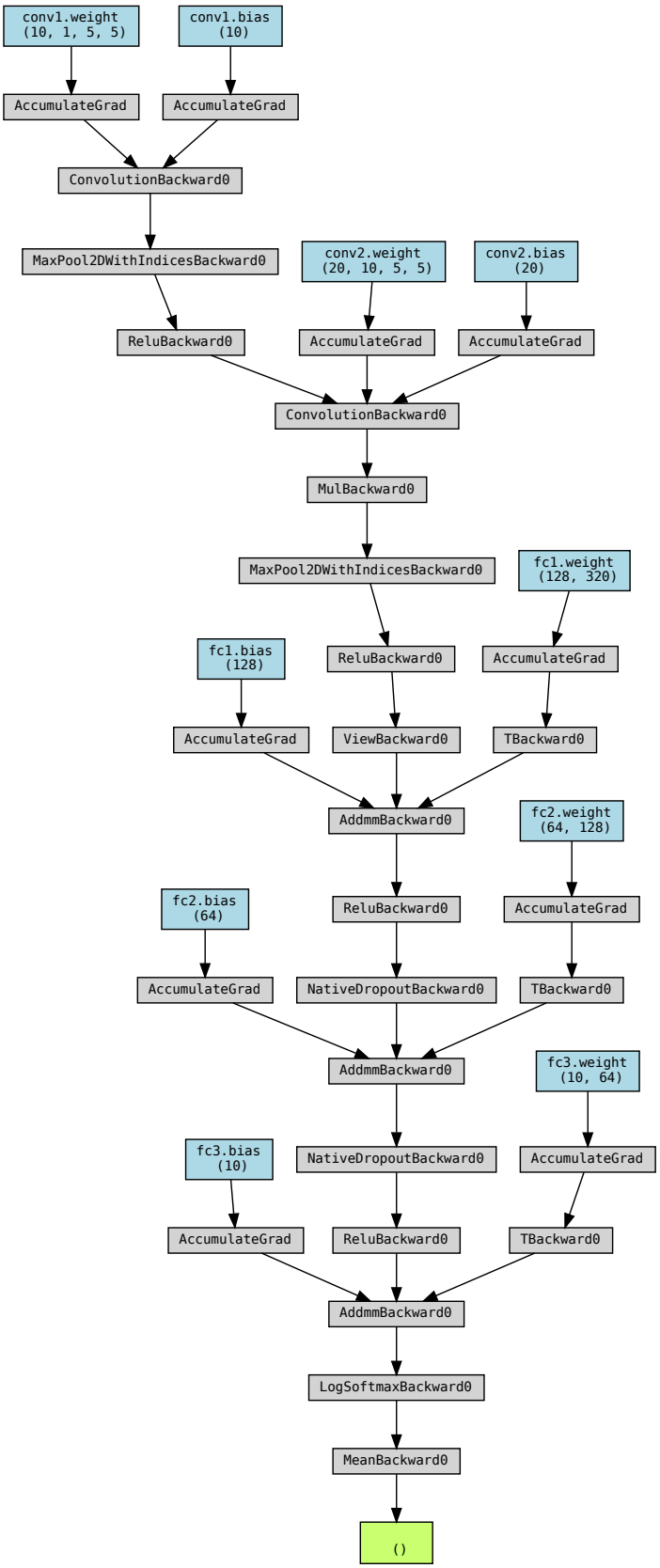
conv1.weight
(10, 1, 5, 5)
conv1.bias
(10)
AccumulateGrad
AccumulateGrad
ConvolutionBackward0
MaxPool2DWithIndicesBackward0
conv2.weight
(20, 10, 5, 5)
conv2.bias
(20)
ReluBackward0
AccumulateGrad
AccumulateGrad
ConvolutionBackward0
MulBackward0
MaxPool2DWithIndicesBackward0
fc1.weight
(128, 320)
fc1.bias
(128)
ReluBackward0
AccumulateGrad
AccumulateGrad
ViewBackward0
TBackward0
AddmmBackward0
fc2.weight
(64, 128)
fc2.bias
(64)
ReluBackward0
AccumulateGrad
AccumulateGrad
NativeDropoutBackward0
TBackward0
AddmmBackward0
fc3.weight
(10, 64)
fc3.bias
(10)
NativeDropoutBackward0
AccumulateGrad
AccumulateGrad
ReluBackward0
TBackward0
AddmmBackward0
LogSoftmaxBackward0
MeanBackward0
()

图 3: 模型示意图