清华大学电子工程系

媒体与认知 课堂 2

2023-2024 学年春季学期

作业 1

元敬哲　　2022010657　　2024 年 3 月 23 日

理论部分

1　单选题（15 分）

1.1　B

1.2　A

1.3　B

1.4　A

1.5　B

2　计算题（15 分）

2.1　设隐含层为 $\mathbf{z}=\mathbf{W}^T\mathbf{x}+\mathbf{b}$，其中 $\mathbf{x}\in R^{(m\times 1)}$，$\mathbf{z}\in R^{(n\times 1)}$，$\mathbf{W}\in R^{(m\times n)}$，$\mathbf{b}\in R^{(n\times 1)}$ 均为已知，其激活函数如下：

$$\mathbf{y}=\delta(\mathbf{z})=tanh(\mathbf{z})$$

tanh 表示双曲正切函数。若训练过程中的目标函数为 L，且已知 L 对 y 的导数 $\frac{\partial L}{\partial \mathbf{y}}=[\frac{\partial L}{\partial y_1},\frac{\partial L}{\partial y_2},...,\frac{\partial L}{\partial y_n}]^T$ 和 $\mathbf{y}=[y_1,y_2,...,y_n]^T$ 的值。

2.1.1　请使用 y 表示出 $\frac{\partial \mathbf{y}^T}{\partial \mathbf{z}}$，这里的 $\mathbf{y}^T$ 为行向量。

解：依题意，

因为 $\frac{\partial y_i}{\partial z_j}=(1-y_i^2)\delta(i,j)$，其中 $\delta(i,j)=\begin{cases}0, i\neq j\\1, i=j\end{cases}$，故有

$$\frac{\partial y^T}{\partial z}=\begin{bmatrix}\frac{\partial y_1}{\partial z_1} & \frac{\partial y_2}{\partial z_2} & \cdots & \frac{\partial y_n}{\partial z_1}\\ \frac{\partial y_i}{\partial z_2} & \frac{\partial y_2}{\partial z_2} & \cdots & \frac{\partial y_n}{\partial z_2}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\partial y_1}{\partial z_n} & \cdots & \frac{\partial y_n-1}{\partial z_n} & \frac{\partial y_n}{\partial z_n}\end{bmatrix}=\begin{bmatrix}1-y_1^2 & 0 & \cdots & 0\\ 0 & 1-y_2^2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & \cdots & 0 & 1-y_n^2\end{bmatrix}$$

2.1.2 请使用 y 和 $\frac{\partial L}{\partial \mathbf{y}}$ 表示 $\frac{\partial L}{\partial \mathbf{x}}$，$\frac{\partial L}{\partial \mathbf{W}}$，$\frac{\partial L}{\partial \mathbf{b}}$。

提示:$\frac{\partial L}{\partial \mathbf{x}}$，$\frac{\partial L}{\partial \mathbf{W}}$，$\frac{\partial L}{\partial \mathbf{b}}$ 与 x,W,b 具有相同维度。
解：由题，

根据链式求导法则，且 $\frac{\partial L}{\partial \mathbf{x}}$，$\frac{\partial L}{\partial \mathbf{W}}$，$\frac{\partial L}{\partial \mathbf{b}}$ 与 x,W,b 具有相同维度，$\frac{\partial L}{\partial \mathbf{y}}$ 维度为 $(n,1)$,$\frac{\partial \mathbf{y}^T}{\partial \mathbf{z}}$ 维度为 (n,n). 可知：

$$\frac{\partial L}{\partial \mathbf{x}} = W\frac{\partial y^T}{\partial z}\frac{\partial L}{\partial y}$$

$$\frac{\partial L}{\partial W} = \mathbf{x}(\frac{\partial y^T}{\partial z}\frac{\partial L}{\partial y})^T$$

$$\frac{\partial L}{\partial b} = \frac{\partial y^T}{\partial z}\frac{\partial L}{\partial y}$$

编程部分

3 编程作业报告

3.1 初次训练

第一次训练结果如下, 此时 seed 为 2023：

```
Epoch 30: validation accuracy = 69.2%
Epoch 31: loss = 0.144
Epoch 32: loss = 0.137
Epoch 33: loss = 0.159
Epoch 34: loss = 0.124
Epoch 35: loss = 0.113
Epoch 36: loss = 0.101
Epoch 37: loss = 0.092
Epoch 38: loss = 0.136
Epoch 39: loss = 0.091
Epoch 40: loss = 0.138
Epoch 40: validation accuracy = 70.2%
Epoch 41: loss = 0.154
Epoch 42: loss = 0.140
Epoch 43: loss = 0.132
Epoch 44: loss = 0.105
Epoch 45: loss = 0.071
Epoch 46: loss = 0.087
Epoch 47: loss = 0.086
Epoch 48: loss = 0.061
Epoch 49: loss = 0.075
Epoch 50: loss = 0.086
Epoch 50: validation accuracy = 69.8%
Model saved in saved_models/recognition.pth
```

图 1: 第一次训练 Loss

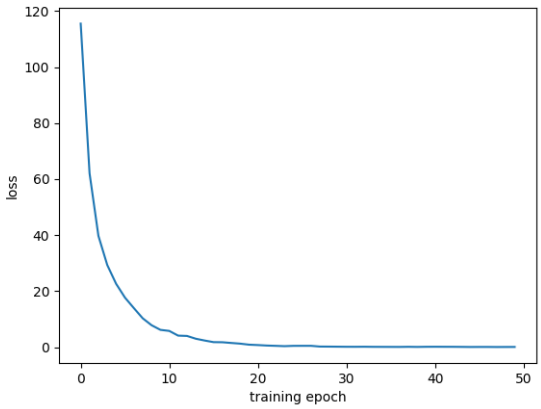

图 2: 第一次训练可视化

可见经过训练,，损失函数有明显的下降，并且几乎收敛于 0。同时多次实验发现，seed 对训练结果的准确度影响并不是特别大，最后都收敛至 68% 左右。

下面是经过 test 的结果：

```
(test) D:\anapy\hw1\hw1\HW1-release>python recognition.py --mode test
[Info] Load model from saved_models/recognition.pth
[Info] Test accuracy = 71.5%
```

图 3: 第一次 test

测试准确性为 71.5%，比较得准确。

3.2 调整参数

改变参数，命令行为 python recognition.py –mode train –hsize 64 –lr 2e-3 –optim_type adam –momentum 0 –weight_decay 0.1，再次训练，得到结果如下：

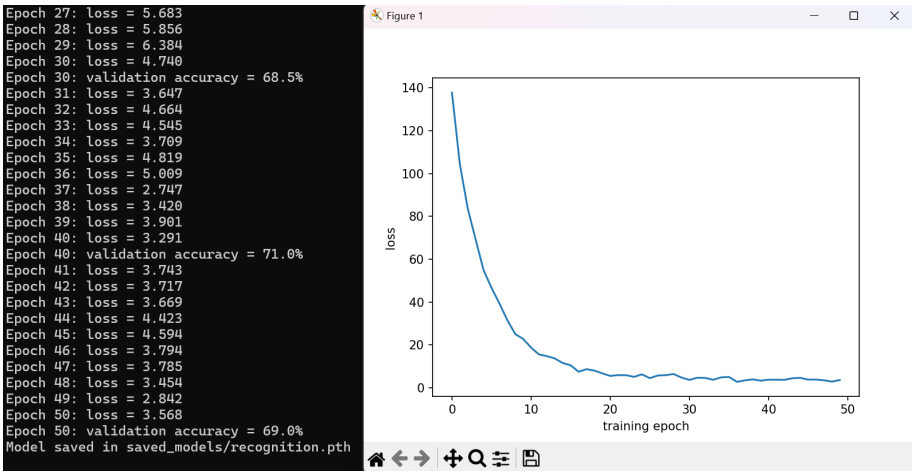

图 4: adam

测试准确性为 68.5%。

经过多次的测试，发现如下命令行参数时为下述时，所得模型的准确性最高：

python recognition.py –mode train –hsize 32 –lr 2e-3 –optim_type adam –momentum 0.9 –weight_decay 0.1

其训练结果如下：

```
Epoch 40: loss = 3.003
Epoch 40: validation accuracy = 74.2%
Epoch 41: loss = 3.742
Epoch 42: loss = 3.455
Epoch 43: loss = 4.224
Epoch 44: loss = 2.966
Epoch 45: loss = 3.339
Epoch 46: loss = 3.383
Epoch 47: loss = 4.338
Epoch 48: loss = 3.485
Epoch 49: loss = 3.299
Epoch 50: loss = 3.261
Epoch 50: validation accuracy = 79.0%
Model saved in saved_models/recognition.pth


(test) D:\anapy\hw1\hw1\HW1-release>python recognition.py --
[Info] Load model from saved_models/recognition.pth
[Info] Test accuracy = 82.2%
```

图 5: 较好模型 Loss 和 validation

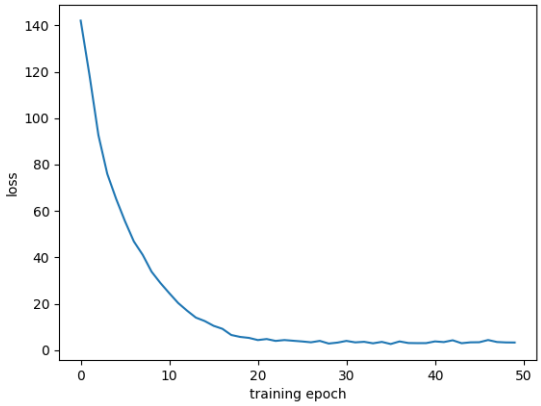

图 6: 较好模型可视化

由图可见，在测试时的准确率可高达 82.2%，性能较好，故用此模型来预测。

3.3 样本预测

下面对提供的样本图片进行预测：如图为图 1、图 2 的图形：

图 7: predict01

图 8: predict02

```
(test) D:\anapy\hw1\hw1\HW1-release>python recognition.py --mode predict --im_path data/character_classification/new_images/predict01.png
[Info] Load model from saved_models/recognition.pth
Prediction: A

(test) D:\anapy\hw1\hw1\HW1-release>python recognition.py --mode predict --im_path data/character_classification/new_images/predict02.png
[Info] Load model from saved_models/recognition.pth
Prediction: B
```

图 9: 对 predict01 和 predict02 分别预测

可见模型很轻松地成功把图片中的字母识别出来。

4 总结与反思

经过我的额外测试发现，本次作业的模型虽然对提供的图片有较好的

识别能力，但是对其他颜色、形状复杂以及相似的字母图片，比如手写字母、彩色字母、O 和 D 等等，识别精确度就大打折扣了。原因可能是算法还比较粗糙或者数据量比较小。希望在以后的学习中可以学到更多的知识来改善这一问题。

本次作业中遇到了不少的问题，比如在程序编写时，对很多 python 自带的函数或操作并未有很好的认识，导致不少本来一行代码可以解决的问题需要花费好几个语句才完成。还有对脚本中的一些注释并未很好地理解到位，用了不少精力思考、查阅资料。在总体写完后，还是发现了不少 bug。比如有些函数是作用在某个数值上的，比如 max() 函数。但实际上输入的是个 vector，这就会导致报错。还有在使用命令行运行时，没有切换到正确的目录下运行。这些都给我造成了比较大的困扰。

但好在我们善良的助教提供了注释和指导，上述困难都能被一一攻克。本次作业建议可以稍微指导一下怎么使用命令行来运行 python。

最后再次感谢老师和助教的教倾情教导与帮助。