数据拟合实验报告

江鹏飞　PB20030896

2023 年 4 月 30 日

摘要

本次实验实验使用神经网络实现昆虫分类问题，使用的框架为 pytorch。首先，我们搭建全连接神经网络，该网络较常用于分类问题且分类效果较好。然后，对于数据集，我们从 insects-training 中按 4:1 随机划分训练集与验证集，以 insects-testing 中的数据作为测试集，在训练集上训练，验证集上调参，训练结束后，记录该组参数下的损失和准确率。最后选取在验证集上最好的一组参数作用在测试集上，并对结果可视化。为防止模型过拟合，我们引入早停策略（earlystopping），若验证集上的准确率连续下降若干个 epoch，提前结束训练。使用神经网络进行实验的一般步骤为：定义网络，确定损失函数，优化。

一、前言

所谓神经网络，是由大量的神经元互相连接而成，不同神经元之间的连接被赋予了不同权重，每个权重代表了一个节点对另一个节点的影响大小，从而具有并行分布结构。每个节点代表一种特定函数，来自其他节点的信息经过相应权重计算，输入到激活函数，得到活性值，代表兴奋或者抑制。人工神经网络学习机制包括无监督学习方法和有监督学习方法。前者的更新规则是：若 x_i 与 y_j 同时被激活，即 x_i 与 y_j 同时为正，那么 w_{ij} 将增大; 若 x_i 被激活，y_j 被抑制，即 x_i 为正 y_j 为负，那么 w_{ij} 将变小。后者的更新规则为：若预测值更大，降低激活输入的权重；若预测值更小，增加激活输入的权重。本次实验的分类任务属于有监督学习。

二、问题分析

分类问题的目标就是基于已知的若干数据点 (训练集) 拟合一个映射

$$\mathbf{F} : D \subset R^2 \to \{1, 2, \ldots, N\}$$

我们希望在训练集上训练好的模型，也可以对验证集和测试集中的数据做出较为准确的预测, 则说明我们的模型在陌生数据上也有良好的表现。即对所有的 $(X, Y) \in S$, 用我们训练好的模型来预测一个结果 $\mathbf{F}(X)$ 。当且仅当 $\mathbf{F}(X) = Y$ 时模型是准确的, 由此得到衡量准确性的指标

$$\mathbf{ACC} = \frac{1}{|S|} \left|\{(X, Y) \in S : \mathbf{F}(X) = Y\}\right|,$$

为了达到这一目标，我们定义损失函数, 这个函数的值越小, 说明此时模型和数据的拟合程度越高，在多分类问题中一般使用交叉熵损失函数。同时，为防止模型过拟合，我们使用早停策略，验证集上的准确率连续下降时，提前结束训练。

三、符号说明

表 1: 符号说明

符号	含义
L	神经网络的层数
M_l	第 l 层神经元的个数
$f_l(\cdot)$	第 l 层神经元的激活函数
$W^{(l)} \in R^{M_l \times M_{l-1}}$	第 $l-1$ 层到第 l 层的权重矩阵
$b^{(l)} \in R^{M_l}$	第 $l-1$ 层到第 l 层的偏置
$z^{(l)} \in R^{M_l}$	第 l 层神经元的净输入（净活性值）
$a^{(l)} \in R^{M_l}$	第 l 层神经元的输出 (活性值)

四、数学模型-神经网络

4.1 全连接神经网络

本次实验中，我们使用的是一种简单的全连接神经网络，其大致结构如下：

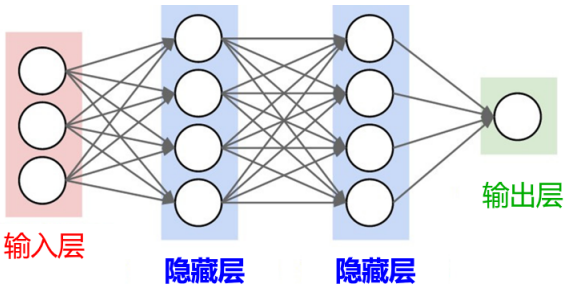


图 1: 全连接神经网络结构

由上图可知，各神经元分别属于不同的层，层内无连接；相邻两层之间的神经元全部两两连接，从传入数据 $X = (x_0, x_1)$ 开始，神经网络通过如下的公式进行信息传播：

仿射变换： $z^{(l)} = W^{(l)} a^{(l-1)} + b^{(l)} \quad \Rightarrow \quad a^{(l)} = f_l\left(W^{(l)} a^{(l-1)} + b^{(l)}\right)$

非线性变换： $a^{(l)} = f_l\left(\mathbf{z}^{(l)}\right) \quad \Rightarrow \quad z^{(l)} = W^{(l)} f_l\left(\mathbf{z}^{(l-1)}\right) + b^{(l)}$

前馈计算： $x = a^{(0)} \to \mathbf{z}^{(1)} \to a^{(1)} \to \mathbf{z}^{(2)} \to a^{(2)} \to \cdots \to a^{(L-1)} \to \mathbf{z}^{(L)} \to a^{(L)}$

对于多分类问题，输出层输出的 $y = (y_1, y_2, y_3)$ 即为输入值在三个类别上的得分，X 在某一个类别上获得的得分最高，我们就认为 X 属于这个类别，从而获得了 X 对应的标签值，即

$$Y = argmax(y_i)$$

4.2 损失函数

采用交叉熵损失函数，方便起见，这里用 one-hot 向量 y 表示类别，对于样本 (x, y)，其损失函数为

$$\mathcal{L}(y, \hat{y}) = -y^\top \log \hat{y} = -\log \frac{\exp z_c^{(L)}}{\sum_k \exp z_k^{(L)}} = -z_c^{(L)} + \log \sum_k \exp z_k^{(L)}$$

4.3 优化

使用基于随机梯度下降（SGD）方法的 Adam 优化算法来完成优化问题。

五、实验结果

实验中使用的数据包括无噪声昆虫数据和有噪声昆虫数据，调节的参数包括：网络深度（宽度），学习率，激活函数，所得实验结果如下

5.4 无噪声数据集

5.4.1 网络深度（宽度）

固定其它参数，调节 neurous 为以下值

```
neurous = [2, 300, 3] # depth = 1
neurous = [2, 50, 30, 3] # depth = 2
neurous = [2, 20, 25, 40, 3] # depth = 3
neurous = [2, *[20]*5, 3] # depth = 5
neurons = [2, *[15]*8, 3] # depth = 8
```

上述 neurous 参数中，第一个和最后一个值分别表示输入层和输出层，其值代表输入或输出的数据维度。计算网络深度时不考虑输入输出层。按上述方式设置网络，可以使网络参数保持在 1800 左右，一定程度排除其它因素的干扰，对宽度的调节同理。我们记录每个参数下的训练集损失，验证集损失，训练集准确率，验证集准确率，所得结果如下：

neurous	**loss_tr**	**loss_val**	**acc_val**
depth=1	0.156536	0.164086	0.950000
depth=2	0.143817	0.158794	0.955556
depth=3	0.150082	0.148267	0.955556
depth=5	0.235314	0.227164	0.911111
depth=8	0.187961	0.176948	0.933333

可以看到，验证集准确率随网络深度变化不明显，主要原因可能是因为实验用到的数据集规模较小且较为规整，使用浅层的神经网络就能达到较好的分类效果，网络的加深带来的提升并不明显。

5.4.2 学习率

固定其它参数，调整学习率分别为 0.1，0.05，0.01，0.005，0.001，0.0001 实验结果如下：

学习率 lr	loss_tr	loss_val	acc_val
lr=0.1	8.284439	5.737251	0.444444
lr=0.05	0.65514	0.887197	0.677778
lr=0.01	0.322650	0.309355	0.877778
lr=0.005	0.259032	0.268895	0.888889
lr=0.001	0.143887	0.145378	0.966667
lr=0.0001	0.574125	0.626222	0.900000

可以看到，当学习率较大时，更新的步长较大，会导致误差值较大；随着学习率减小，参数更新的精度提高，误差随之减小；但当学习率过小时，参数更新的步长很小，容易在局部取得最优，另外，过小的学习率导致每轮的训练时间也会较长。

5.4.3 激活函数

固定其它参数, 调整激活函数为以下值：

```
activation = {'relu', 'tanh', 'sigmoid', 'elu', 'leakyrelu', 'softplus'}
```

激活函数	loss_tr	loss_val	acc_val
relu	0.188271	0.187376	0.933333
tanh	0.138666	0.136651	0.966667
sigmoid	0.247157	0.249018	0.871111
leakyrelu	0.150508	0.168356	0.944444
elu	0.203667	0.185098	0.911111
softplus	0.337626	0.383057	0.866667

整体来看，sigmoid 函数和 softplus 函数作为激活函数效果较差，这可能是因为 sigmoid 在远离零点的地方，梯度变得非常微小，所以求导以后很难指导我们进行反向传播和梯度下降，要求的计算精度也较高。relu 家族整体效果都较好。

5.4.4 测试集效果

使用在验证集上表现最好的一组参数作用在测试集上，可视化结果如下

```
# 最优参数
neurous = [2, 50, 30, 3] # depth = 2
lr = 0.001
activation = 'relu'
```

所得测试集的损失值，前 60 个数据的准确率，后 150 个陌生数据的准确率，平均准确率如下：

```
Test loss: 0.136734 |average acc: 0.971429 |acc_60: 0.950000 |acc_150: 0.980000
```

绘制该参数下的损失曲线及准确率曲线，原数据集的分布及经过模型预测后的分类结果

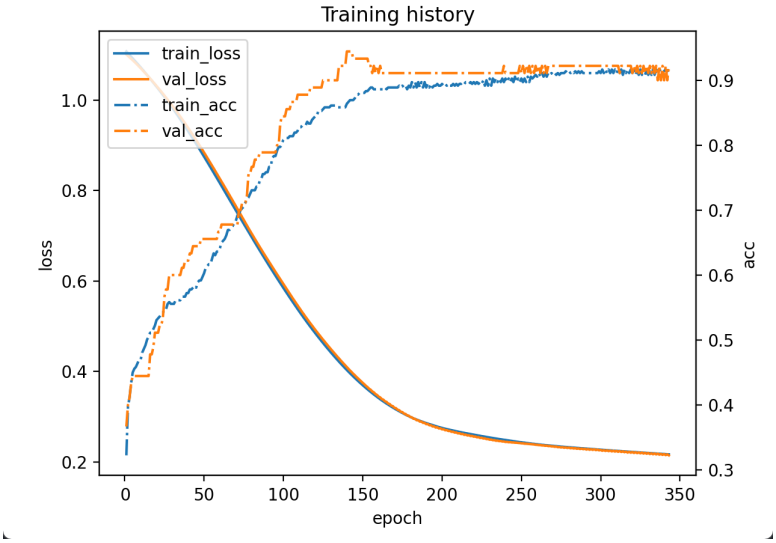
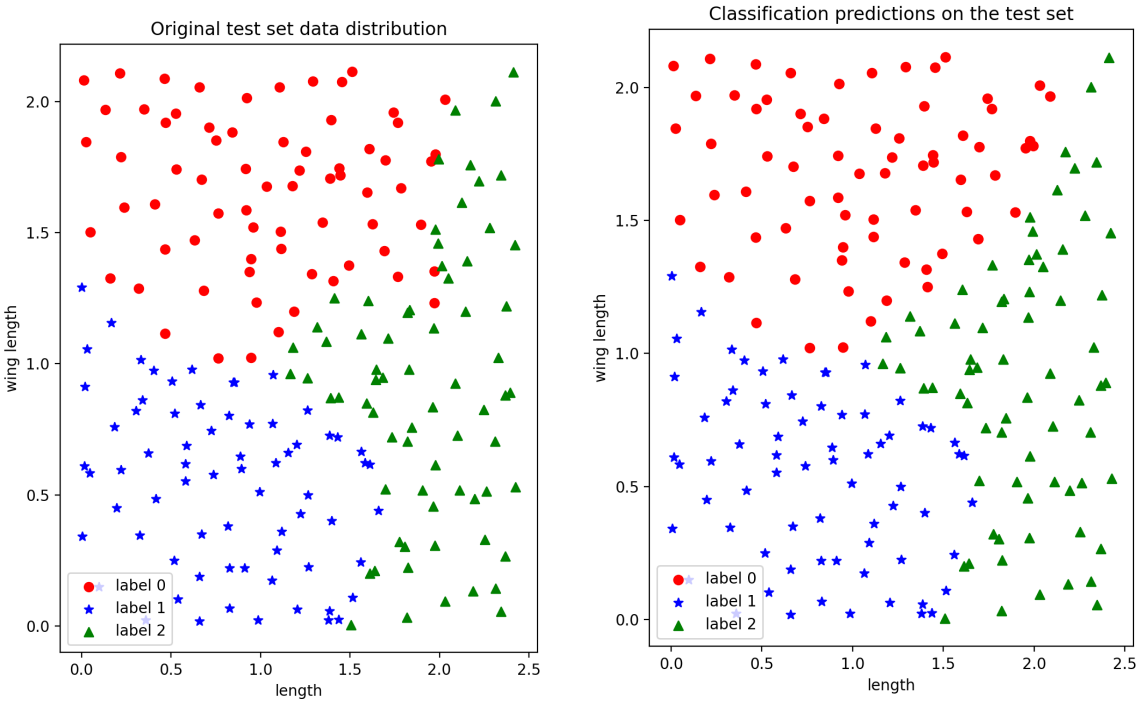


图 2: Training History



(a) 原数据集分布

(b) 模型分类结果

可以看到模型在测试集上的的分类结果还是很好的。此外，考虑将预测向量中的每个值转为 one-hot 向量，这样我们就得到了一个由 0 和 1 组成的标签矩阵 L，其中 1 的位置表明了它的类别，0 表示其他类别。1 对应二分类问题中的正类，0 对应负类；此外，由于神经网络输出值是 x 在三个标签的预测概率值，我们记这个概率矩阵为 P 将标签矩阵 L 和概率矩阵 P 分别按行展开，然后转置形成两列，这就得到了一个二分类结果，根据结果直接画出 ROC 曲线。便得到了多分类下的 ROC 曲线：

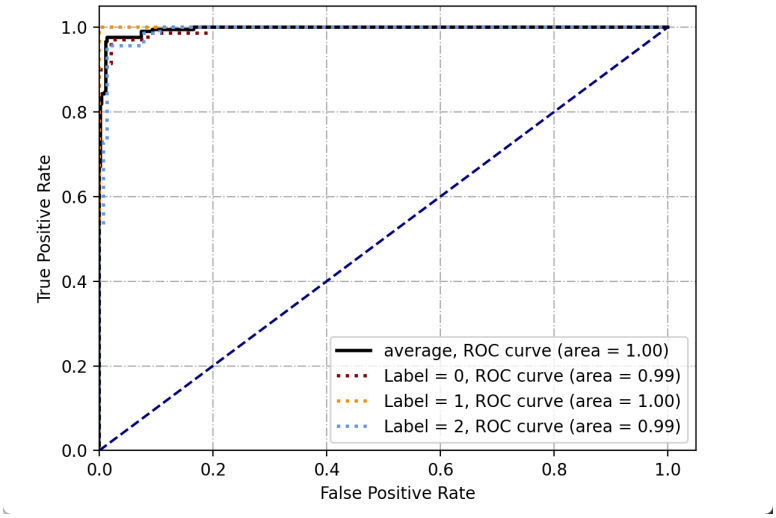


图 3: 多分类 ROC 曲线

5.5 有噪声数据集

参照无噪声数据的处理，固定其它参数，调节 neurous 的值，所得实验结果如下：

neurous	**loss_tr**	**loss_val**	**acc_val**
depth=1	0.256331	0.264086	0.888889
depth=2	0.311442	0.315025	0.844444
depth=3	0.284993	0.293221	0.855556
depth=5	0.270563	0.304413	0.866667
depth=8	0.309429	0.355860	0.833333

更深的网络应至少有和浅层网络一样好的性能，但在噪声数据集上深层网络的预测准确率却下降，可能的原因是更深层的网络更难优化，更容易陷入局部极小值。

5.5.1 学习率

固定其它参数，调整学习率分别为 0.1，0.05，0.01，0.005，0.001，0.0001 实验结果如下：

学习率 lr	**loss_tr**	**loss_val**	**acc_val**
lr=0.1	10.005360	4.310062	0.588889
lr=0.05	1.188950	1.044683	0.644444
lr=0.01	0.359985	0.368948	0.822222
lr=0.005	0.342291	0.328495	0.844444
lr=0.001	0.282446	0.287483	0.833333
lr=0.0001	0.538372	0.560063	0.822222

5.5.2 激活函数

固定其它参数, 调整激活函数

激活函数	loss_tr	loss_val	acc_val
relu	0.269495	0.264018	0.888889
tanh	0.276556	0.279722	0.855556
sigmoid	0.316370	0.313095	0.855556
leakyrelu	0.273746	0.285499	0.822222
elu	0.282867	0.276962	0.844444
softplus	0.363301	0.366869	0.844444

5.5.3 测试集效果

使用在验证集上表现最好的一组参数作用在测试集上，可视化结果如下

```
# 最优参数
neurous = [2, 20, 25, 40, 3] # depth = 2
lr = 0.001
activation = 'elu'
```

所得测试集的损失值，前 60 个数据的准确率，后 150 个陌生数据的准确率，平均准确率如下：

```
Test loss: 0.251400 |total acc: 0.900000 |acc_60: 0.966667 |acc_150: 0.873333
```

参照无噪声数据的处理，绘制可视化曲线

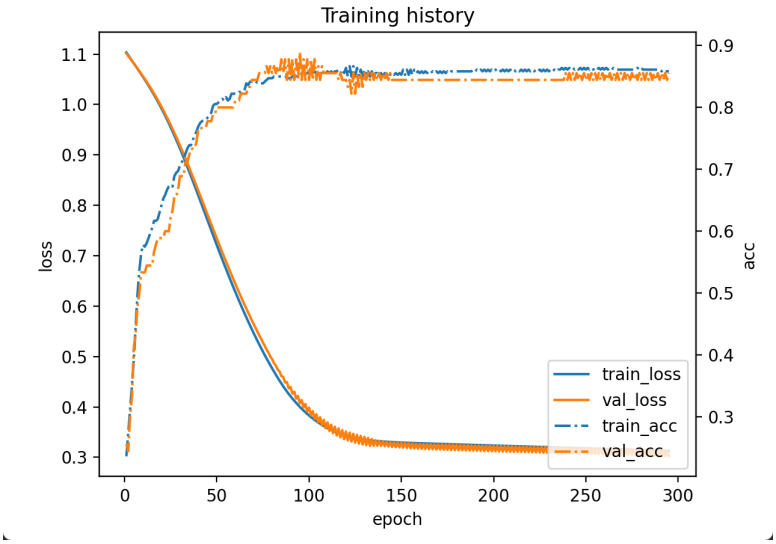
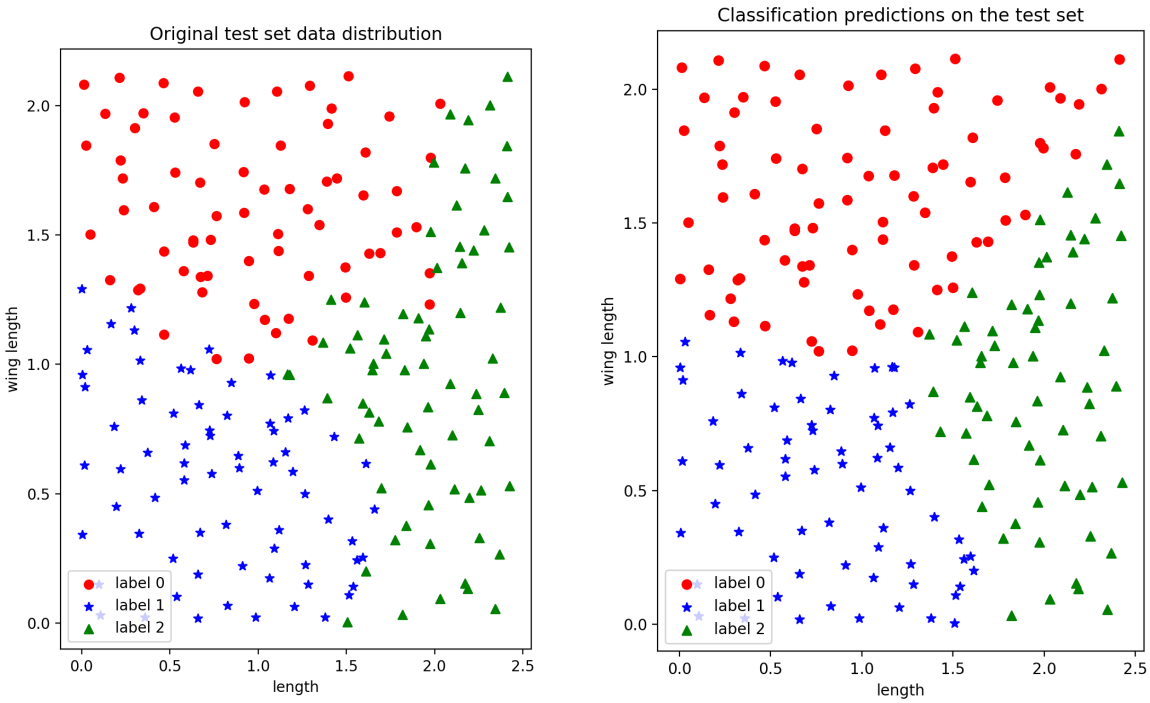


图 4: Training History in Noise data



(a) 噪声数据集分布　　(b) 模型分类结果

同样的，绘制该数据集下的 ROC 曲线：

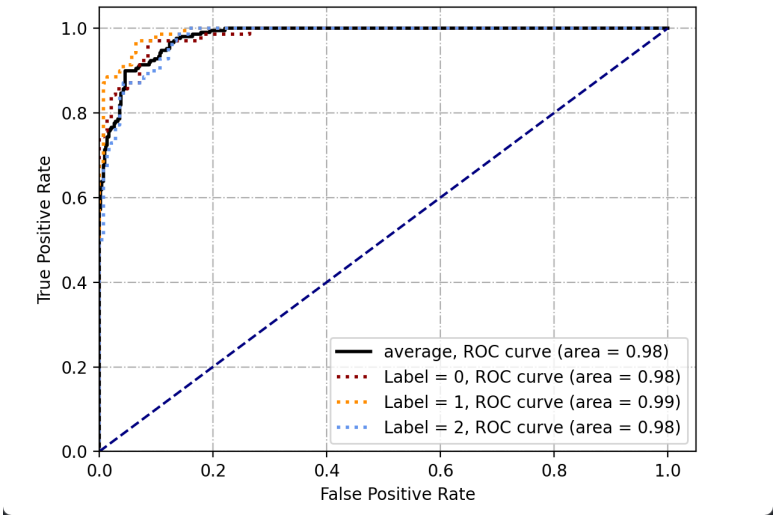


图 5: 噪声数据集 ROC 曲线

六、结论

本次实验我们用全连接的神经网络实现了有噪声和无噪声的昆虫数据分类问题。可以看出神经网络模型对于这种点集分类问题可以起到很好的拟合作用。

实验过程中，我们可以发现神经网络在训练（优化）过程中，损失函数总体上是不断降低的，并且在最开始的几次迭代下降得最快，迅速降低到某一个比较平稳的值。在参数调节的过程中，我们还发现了以下规律：一般更深的网络能更快收敛，但其也更难优化，易陷入局部极小值；学习率设置的过大或过小都会导致拟合效果不理想，一般在训练网络时，应该动态调节学习率。初始学习率可以略大来加快收敛速度，在接近极小值时，可以减小学习率，以实现收敛；激活函数一般选取 relu 或其家族函数，效果相对较好。

本次的实验还是有一些可以改进的地方的，如我们的模型在有噪声数据集上的预测准确率不是很高，可以考虑添加 dropout 层或者改善网络结构以提高性能。