前馈神经网络解决昆虫分类问题

陈鸿绪　35　PB21000224

2024 年 4 月 21 日

摘要

对于分类问题有许多经典的分类算法，传统分类算法一般使用统计学习，统计学习的分类算法常见的有贝叶斯分类器、支持向量机、聚类等等。基于统计学习的算法通常需要经过特征工程来提取数据的有用特征。这可能需要大量的手工工作，且对特征的选择和转换具有较高的敏感性，同时基于统计学习的算法在处理复杂的非线性关系时受到模型复杂度的限制，一旦数据集变得庞大繁杂时，就不能很好捕捉到其中的特点。所以为了得到更具有泛化性的分类模型，一般采用构建神经网络来进行分类预测。

本次实验中的昆虫数据集是由三个特征所组成，我们将采用前馈神经网络进行训练测试，我们调整优化训练时的参数得到在测试集上准确率的最好结果高达 96%。同时，我们在实验中也会结合准确率、损失曲线图像，通过变量消融实验来探究不同网络结构、不同激活函数、不同学习率以及不同 batch size 对模型性能带来的影响。实验代码位于 src 文件夹中，代码使用说明参见 README.md。

一、前言（问题的提出）

1.1 问题背景及描述

生物学家对某昆虫进行研究，发现该昆虫具有 3 个不同的属性，即可分为 3 类，依据的资料是体长和翼长。所给数据集 1 是无误差的噪声的分类结果，数据集 2 是存在噪声的数据集（测量昆虫体征时带有部分误差）。我们需要构建分类模型，在模型经过数据集 1、2 的训练集的训练后，可以在数据集 1、2 的测试集上准确地预测分类结果。

1.2 基于统计学习的分类算法

统计学习算法建立在扎实的数学和统计学基础上，模型决策过程相对容易理解和解释。这使得它们在一些需要高度可解释性和可靠性的场景中具有一定优势。但是一旦数据具有复杂的非线性关系，传统统计学习算法可能就会受到本身复杂度的限制。鉴于神经网络的强大泛化性能，我们采用神经网络构建分类模型。

二、相关工作

"The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain" [1]，在这篇论文中，罗森布拉特首次提出了多层感知器（即前馈神经网络）的概念，这是首个用算法精确定义神经网络的数学模型，是后来很多神经网络模型的始祖。他详细描述了感知器模型如何通过训练学习来识别并分类不同的模式。这一模型展示了前馈神经网络在处理分类问题上的潜力，为后续的研究者提供了重要的启示。

三、问题分析

3.1 前馈神经网络的优越性

前馈神经网络具有很强的非线性映射能力，可以处理复杂的非线性关系。传统的分类模型如线性回归或逻辑回归往往只能处理线性关系，而现实生活中的许多问题都涉及到复杂的非线性关系。前馈神经网络通过其特殊的网络结构和激活函数，能够逼近任意的非线性函数，从而在处理复杂分类问题时表现出色。我们观察到本次数据量较大，所以我们合理采用前馈神经网络作为分类模型。

3.2 避免过拟合

由于前馈神经网络强大的拟合性能，所以在训练轮数过大很可能在训练集上过分关注训练集噪声，以至于在训练集上表现得过于完美，然而测试集上性能表现较差。我们观察训练数据的分布情况，数据集 1

的训练集没有任何噪声，数据集 2 训练集存在一定噪声，如果训练两者的轮数过大，模型很可能过分关注到数据集 2 的噪声，所以实验中需要采取早停、正则化或者随机 drop 的策略。本次实验中采取较为简单的早停，即选择测试集上表现最佳的训练结果即可。

四、建模的假设

4.1 假设 1: 网络的泛化能力

神经网络假设其结构和学习算法能够使其从训练数据中学习到一般性的规律，并对未见过的数据进行准确的分类。这要求网络具有足够的复杂性来捕捉数据的内在模式，同时又要避免过拟合，以保持对新数据的泛化能力。

4.2 假设 2: 数据的分布特性

虽然神经网络不需要对数据分布做出严格的假设，但数据的某些特性（如是否服从某种分布、是否存在噪声等）可能会影响网络的训练效果和分类性能。因此，在一定程度上，我们假设输入数据具有一定的结构和规律，这有助于网络从中学习到有用的信息。（实际上该假设大概率是合理成立的，可以由数据集可视化分布观察出）

4.3 假设 3: 非线性映射的存在

对于许多复杂的分类问题，数据之间的关系可能是非线性的。神经网络假设其能够通过其内部的非线性激活函数和多层结构，学习到这些非线性关系，并准确地进行分类。

五、符号说明

表 1: 符号说明

符号	说明
D	D 为数据的维数，在实验中数据维数为 2（不包括预测属性）
K	数据类型总共有 K 个类别，对于该数据集有 3 个类别
L	隐藏层层数
n_l	n_l 是第 l 层隐藏层的神经元个数
w_{lij}	w_{lij} 是连接第 $l-1$ 层第 j 个神经元和第 l 层第 i 个神经元的权重
b_{li}	b_{li} 是第 l 层第 i 个神经元的偏置
w_{ki}	w_{ki} 是连接最后一个隐藏层第 i 个神经元和输出层第 k 个神经元的权重
b_k	b_k 是输出层第 k 个神经元的偏置
$f(.)$	f(.) 表示具体的激活函数

六、数学模型建立

前馈神经网络（Feedforward Neural Network，FNN）是一种基本的神经网络结构，它由输入层、一个或多个隐藏层以及输出层组成。每一层由多个神经元组成，每个神经元都与下一层的所有神经元相连接。信息在前馈神经网络中是单向传递的，从输入层到输出层，没有环。下面我们构建对于该多分类问题的神经网络模型。

6.1 输入层

输入层接收外部数据，每个神经元代表数据的一个特征。假设输入层有 D 个神经元，输入数据可以表示为一个 D-维向量 $x = [x_1, x_2, ..., x_D]^T$。实际上在真实训练过程中，为了加速训练进程，多条训练数据将会打包成一个 batch 送进输入层。

6.2 隐藏层

假设有 L 个隐藏层，第 l 个隐藏层有 n_l 个神经元。每个隐藏层神经元的输出通过激活函数转换，常用的激活函数有 ReLU、Tanh、LeakyReLU 等。在实验中，我们也将使用前述激活函数进行对比实验。

- 线性组合：第 l 个隐藏层的第 i 个神经元的输入是上一层所有神经元的输出的加权和：

$$z_{li} = \sum_{j=1}^{n_{l-1}} w_{lij} y_{l-1,j} + b_{li}$$

  其中，w_{lij} 是连接第 $l-1$ 层第 j 个神经元和第 l 层第 i 个神经元的权重，b_{li} 是第 l 层第 i 个神经元的偏置，$y_{l-1,j}$ 是第 $l-1$ 层第 j 个神经元的输出。

- 激活函数：z_{li} 通过激活函数 $f(\cdot)$ 得到第 i 个神经元的输出：

$$y_{li} = f(z_{li})$$

6.3 输出层

输出层的每个神经元代表一个类别的得分或预测值。假设输出层有 K 个神经元，对于多分类问题，通常使用 Softmax 函数作为输出层的激活函数。

- 线性组合：输出层的第 k 个神经元的输入是最后一个隐藏层所有神经元的输出的加权和：

$$z_k = \sum_{i=1}^{n_L} w_{ki} y_{Li} + b_k$$

  其中，w_{ki} 是连接最后一个隐藏层第 i 个神经元和输出层第 k 个神经元的权重，b_k 是输出层第 k 个神经元的偏置，y_{Li} 是最后一个隐藏层第 i 个神经元的输出。

- Softmax 激活函数: 输出层的每个神经元的输出通过 Softmax 函数转换为概率分布：

$$\hat{y}_k = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}$$

  其中，$\hat{y}_k$ 是模型预测输入样本属于类别 k 的概率。实际上在使用 torch 时，我们使用交叉熵函数作为损失的操作会自动为模型最后添加上 Softmax 操作，所以在实际代码中并没有在最后输出层加上 Softmax 激活函数。

6.4 损失函数的选择

使用交叉熵损失函数来衡量模型预测的概率分布与真实标签的概率分布之间的差异：

$$L(y, \hat{y}) = -\sum_{k=1}^{K} y_k \log(\hat{y}_k)$$

其中，y_k 是真实标签的独热编码（0 或 1），$\hat{y}_k$ 是模型预测的概率。

6.5 训练过程

通过反向传播算法和梯度下降优化算法来更新网络的权重和偏置，以最小化交叉熵损失函数。为了更加稳定地梯度下降，我们代码中将采用 Adam 优化器。

七、结果（与消融实验对比）

7.1 最优调参结果展示

表 2: 网络超参数及结果

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	16	1000	ReLU	4	1024	96.2962%

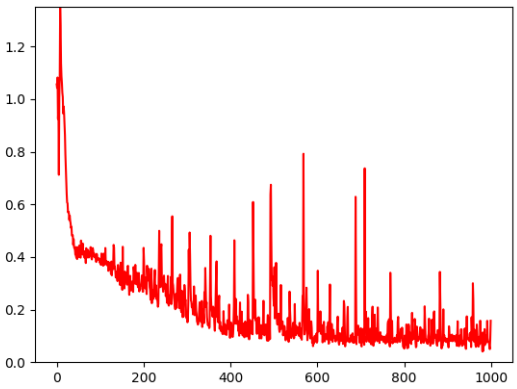

图 1: 训练集上损失函数的变化

7.2 结果对比

- **网络深度对模型性能影响：**

表 3: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	16	1000	ReLU	1	1024	93.9815%

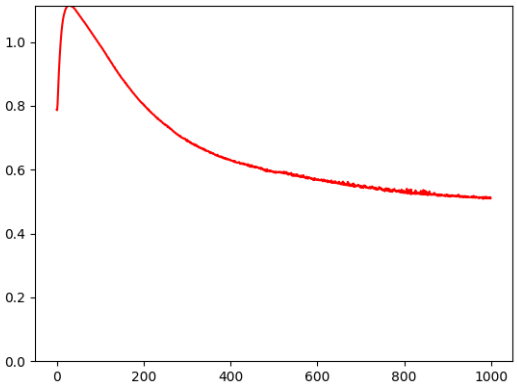

图 2: 训练集上损失函数的变化，层数为 1

表 4: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	16	1000	ReLU	2	1024	95.3704%

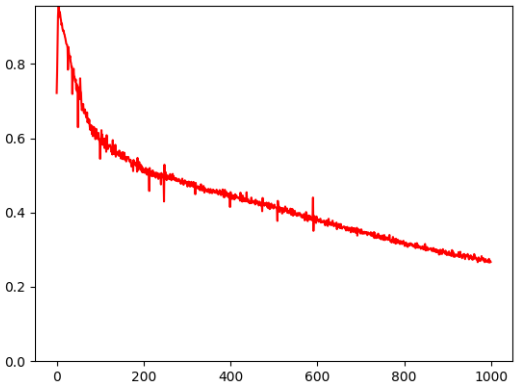

图 3: 训练集上损失函数的变化，层数为 2

表 5: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	16	1000	ReLU	8	1024	96.0648%

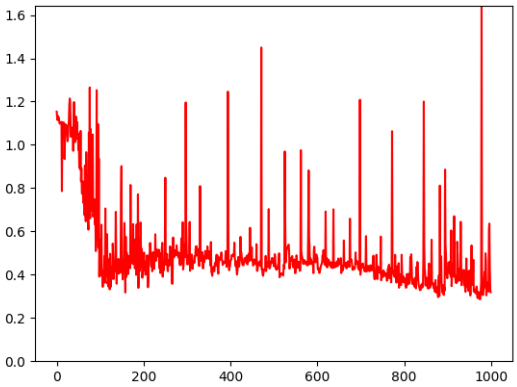

图 4: 训练集上损失函数的变化，层数为 8

- **网络宽度对模型性能影响：**

表 6: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	16	1000	ReLU	4	128	94.6759%

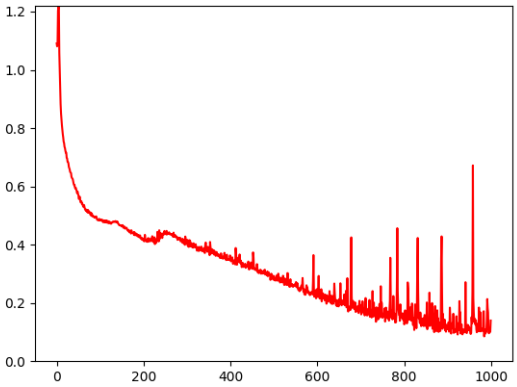

图 5: 训练集上损失函数的变化，宽度为 128

表 7: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	16	1000	ReLU	4	256	95.8333%

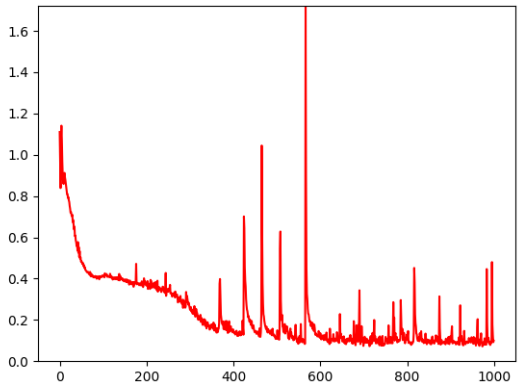

图 6: 训练集上损失函数的变化，宽度为 256

表 8: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	16	1000	ReLU	4	512	96.0648%

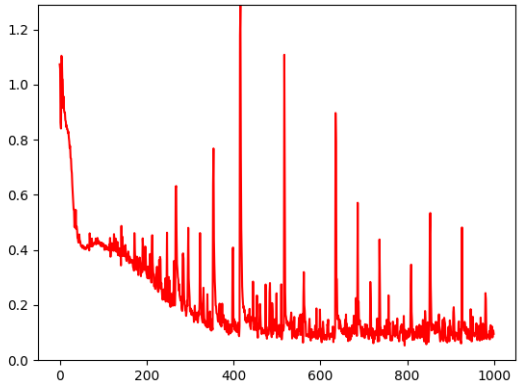

图 7: 训练集上损失函数的变化，宽度为 512

- **学习率对模型性能影响：**

表 9: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.3	16	1000	ReLU	4	1024	32.4074%

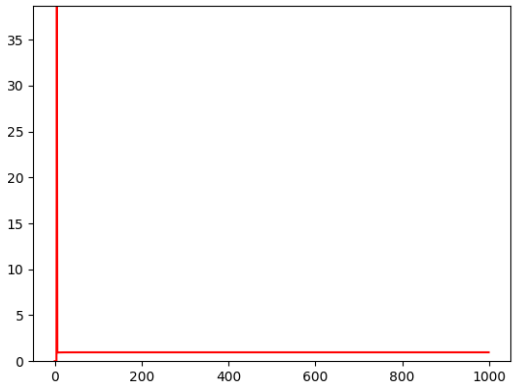

图 8: 训练集上损失函数的变化，学习率为 0.3

表 10: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.03	16	1000	ReLU	4	1024	87.7314%

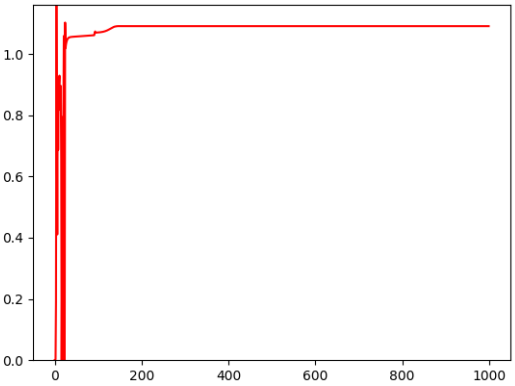

图 9: 训练集上损失函数的变化，学习率为 0.03

表 11: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.000003	16	1000	ReLU	4	1024	93.5185%

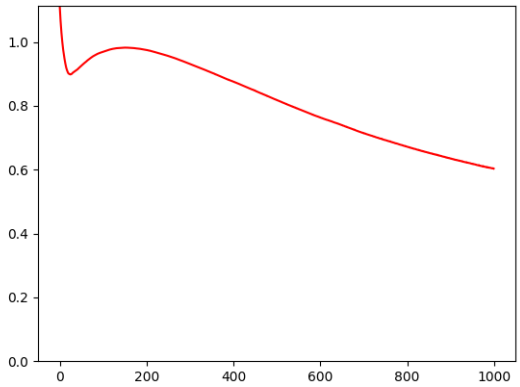

图 10: 训练集上损失函数的变化，学习率为 0.000003

- **激活函数对模型性能影响：**

表 12: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	16	1000	Tanh	4	1024	95.8333%

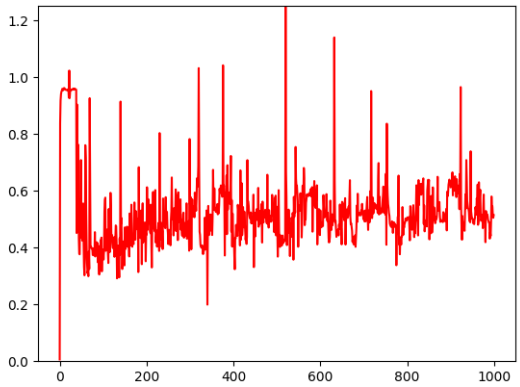

图 11: 训练集上损失函数的变化，激活函数为 Tanh

表 13: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	16	1000	LeakyReLU	4	1024	96.0648%

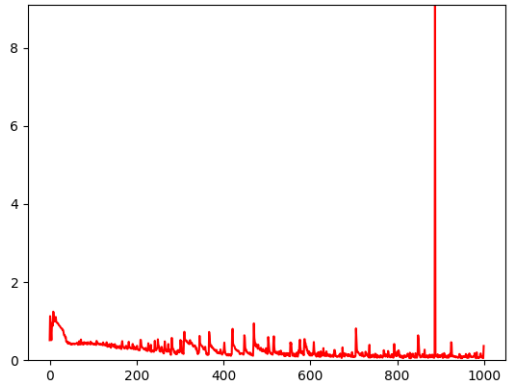

图 12: 训练集上损失函数的变化，激活函数为 LeakyReLU

- **batch size 对模型性能影响：**

表 14: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	64	1000	ReLU	4	1024	92.63392%

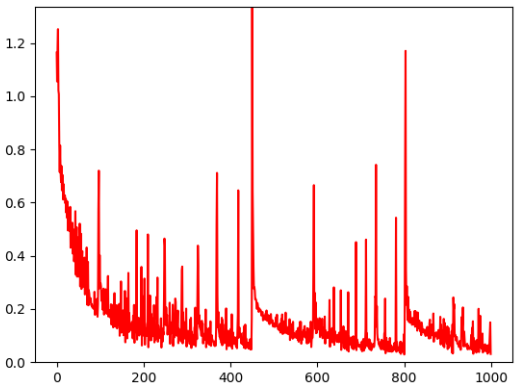

图 13: 训练集上损失函数的变化，batch size 为 64

表 15: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	256	1000	ReLU	4	1024	79.2968%

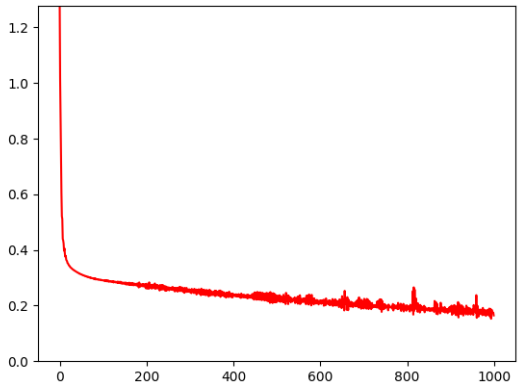

图 14: 训练集上损失函数的变化，batch size 为 256

表 16: **网络超参数及结果**

学习率	batch size	最大轮数	激活函数	隐藏层层数	单个隐藏层神经元个数	最优预测准确率
0.0003	1024	1000	ReLU	4	1024	40.1367%

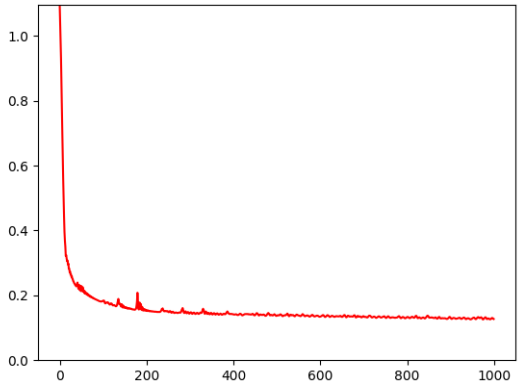

图 15: 训练集上损失函数的变化，batch size 为 1024

八、结论

8.1 网络结构影响分析

首先关注网络深度的影响：观察图 1 至图 4，我们发现在深度为 4 的时候为最优准确率，隐藏层深度为 8 时的准确率略低。深度从 1 到 4 增大时，准确率也随着增大，说明前馈神经网络的拟合能力在逐步提升，然而到 8 却有准确率略微下降，这是因为前馈神经网络的参数量增大，所需要的训练数据同时就会增多，不足的训练数据会一程度上导致网络训练欠拟合，所以这就解释了为什么参数量过大的网络对于偏小的数据集效果欠佳的原因。同时我们观察到对于层数越大的网络，loss 下降也更加不稳定，这实际上也是网络处在欠拟合状态的证明。其次关注网络宽度的影响：观察图 1 以及图 5 到图 7，相似的也

是需要从参数量角度入手，只不过模型并没有发生欠拟合现象，所以随着隐藏层神经元个数的增多，模型预测准确率也在稳步提升。

8.2 学习率影响分析

从理论上分析，过大的学习率会导致模型无法有效收敛，甚至会导致激活函数 ReLU 的" 死亡" 现象；过小的学习率又会导致模型无法快捷有效更新。选择一个有效的学习率需要不断地调整参数对比才可以得到。观察图 1 以及图 8 到图 10，可以发现与理论分析一致，在学习率较大时（如图 8，图 9 学习率为 0.3、0.03），模型训练最后的损失值始终为恒定较大的值，这就说明 ReLU 到了" 死亡 ReLU" 状态，所有梯度更新无法有效进行，所以模型最优准确率始终处于较低状态。在学习率非常低（如图 10 学习率为 0.000003），可以观察到损失曲线仍然有下降的趋势，且最优预测准确率大概比全局最优准确率小 2.5%，说明模型处于欠拟合状态，模型参数更新迭代过慢。

8.3 损失函数影响分析

根据实验结果，我们对图 1、图 11、图 12 进行对比，其实三者都可以达到一个非常高的准确率，虽然 ReLU>LeakyReLU>Tanh，但是三者准确率相差并不是很大。但是可以发现三者的损失曲线特征区别非常明显。LeakyReLU、Tanh 两者的损失曲线过于不稳定，在实践过程中，我们当然希望在几乎持平的准确率情况下，损失曲线越平稳越好，所以综合起来可以认为 ReLU 是明显优于其余两个激活函数的。而且再加上 ReLU 简单良好的数学性质，其梯度下降的时间也会相应少于其余两者。正是因为上述这些优势才使得 ReLU 激活函数是当今深度学习领域应用最多的激活函数。

8.4 batch size 影响分析

根据实验结果，对图 1、图 13 到图 15 进行分析对比，首先关注到四者损失曲线的平滑程度是随着 batch size 的增大而更加平滑的。其实从理论上分析，更大的 batch size 会使得梯度下降更少依赖于样本的随机性，所以表现在损失曲线上就是更加光滑，然而发现光滑的曲线并没有带来准确率的提升，这是因为在相同的学习率下，理论上越大的 batch size 会带来更小的梯度更新，所以就出现了在 batch size 为 1024 的时候，甚至最优准确率只有 40% 左右。实际上为了有效率地更新，我们需要相应增大学习率。

九、问题

首先，对于调参而言，由于笔者调整参数组个数有限，并不能保证表 2 中超参数在全局参数空间取到最优预测准确率。其次对于前馈神经网络而言，是否存在一个更优的网络结构可以达到更高的准确率？这些问题都值得被思考解决。

参考文献

[1] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.