图标

描述已自动生成

**计算机视觉**

**实验一报告**

|  |  |  |  |
| --- | --- | --- | --- |
| **专业** | **：** | **计算机** |  |
| **班级** | **：** | **CS2002** |  |
| **学号** | **：** | **I201920024** |  |
| **姓名** | **：** | **木林** |  |
| **电话** | **：** | **15623029026** |  |
| **邮箱** | **：** | **792967028@qq.com** |  |

|  |  |
| --- | --- |
| 分数 |  |
| 教师签名 |  |

## 1 任务要求

设计一个前馈神经网络，对一组数据实现分类任务。下载“dataset.csv”数据集，其中包含四类二维高斯数据和它们的标签。设计至少含有一层隐藏层的前馈神经网络来预测二维高斯样本所属的分类。这个数据集需要先进行随机排序，然后选取90%用于训练，剩下的10%用于测试。

## 2 任务设计

前馈神经网络 (FNN) 是最简单的人工神经网络类型之一。 在该网络中，信息仅沿一个方向移动——向前——从输入节点，通过隐藏节点（如果有），到达输出节点。 网络中不存在循环或环路。

前馈神经网络的架构由三种类型的层组成：输入层、隐藏层和输出层。 每层都由称为神经元的单元组成，各层通过权重互连。

输入层：该层由接收输入并将其传递到下一层的神经元组成。 输入层中神经元的数量由输入数据的维度决定。

隐藏层：这些层不暴露于输入或输出，可以被视为神经网络的计算引擎。 每个隐藏层的神经元获取前一层输出的加权和，应用激活函数，并将结果传递到下一层。 该网络可以有零个或多个隐藏层。

输出层：为给定输入生成输出的最后一层。 输出层中神经元的数量取决于网络设计产生的可能输出的数量。

前馈神经网络的工作涉及两个阶段：前馈阶段和反向传播阶段。

前馈阶段：在此阶段，输入数据被馈送到网络中，并通过网络向前传播。 在每个隐藏层，计算输入的加权和并通过激活函数，这将非线性引入模型。 这个过程一直持续到到达输出层并做出预测。

反向传播阶段：一旦做出预测，就会计算误差（预测输出与实际输出之间的差异）。 然后，该误差通过网络传播回来，并调整权重以最小化该误差。 调整权重的过程通常使用梯度下降优化算法来完成。

激活函数在前馈神经网络中起着至关重要的作用。 它们向网络引入非线性属性，使模型能够学习更复杂的模式。 常见的激活函数包括 sigmoid、tanh 和 ReLU（修正线性单元）。

神经网络架构：

X1

X2

O

O

O

O

a

a

a

a

O

O

O

O

a

a

a

a

Yp

Relu

Softmax

图2.1 神经网络架构

初始化（\_\_init\_\_ 方法）：这是定义神经网络层的地方。

self.func1 = nn.Linear(2, 32)：这是一个线性（也称为全连接）层，其输入大小为 2，输出大小为 32。这是网络的第一层。

self.relufunc = nn.ReLU()：这是 ReLU（修正线性单元）激活函数。 它用于将非线性引入模型，使其能够学习更复杂的模式。

self.func2 = nn.Linear(32, 4)：这是另一个线性层，其输入大小为 32（前一层的输出），输出大小为 4。这是网络的最后一层。

前向传播（前向方法）：这是实际计算发生的地方。 输入数据依次通过每一层。

x = self.func1(x)：输入数据x通过第一个线性层。

x = self.relufunc(x)：第一层的输出然后通过ReLU激活函数。

x = self.func2(x)：激活函数的输出通过第二个线性层。

x = F.softmax(x, dim=0)：最后，对第二层的输出应用softmax函数。 这会将原始输出值压缩为总和为 1 的概率，这对于多类分类问题非常有用。

总之，该模型采用大小为 2 的输入，将其传递到两个线性层，中间有一个 ReLU 激活函数，最后对输出应用 softmax 函数。 模型的输出是 4 维概率向量。

## 3 实验结果与分析

训练过程的损失函数和准确率如下图3.1：

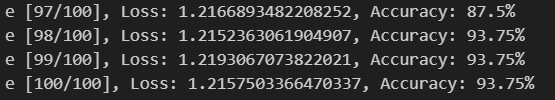


图3.1训练过程的损失函数和准确率

测试集的损失函数和准确率如下图3.2：



图3.2测试集的损失函数和准确率

实验分析：

如下图3.3，不同时期训练准确率的波动可能是由于以下几个原因造成的：

数据的可变性：每个时期的数据可能会有很大差异，特别是如果数据集在分成批次之前没有进行洗牌的话。 如果第 23 Epoch碰巧包含更困难或以前未见过的示例，则模型的准确性可能会下降。

过度拟合：如果模型与训练数据过度拟合，它可能在某些时期表现良好，但在其他时期表现不佳。 如果模型很复杂并且训练样本数量很少，这种情况尤其可能发生。

学习率：如果学习率太高，模型的性能可能会振荡甚至发散，而不是稳步提高。

随机机会：神经网络涉及很多随机性，从初始权重到数据的洗牌。 有时，性能下降只是运气不好。

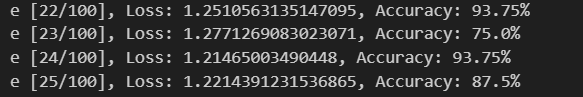


图3.3准确率的波动

## 4 结论

神经元的数量、隐藏层和激活函数都会对神经网络的性能产生影响。 模型的学习效率和收敛速度都会随着神经元和隐藏层数量的增加而提高。 但是，如果模型过于复杂，就会出现过度拟合。 因此，在创建模型时，选择正确的激活函数、神经网络层数以及每层神经元的数量与决定正确的学习率和 B 大小同样重要。