**基于神经网络的逆向云模型生成器实验报告**

齐智鹏 2020110855

计算机学院（国家示范性软件学院）

**摘要：**

本次的大作业我选择的题目是基于神经网络的逆向云模型生成器，逆向云模型即从已有的云模型云滴反向预测出云模型的三个参数——均值（Ex）、熵（En）和超熵（He）。此次的实验作业，我使用机器学习中BP神经网络的方法对固定参数，即均值、熵、超熵分别为（0，1，0.25），的云模型进行了学习和预测，并记录下不同网络结构和不同特征划分情况下的结果。

**关键词：**

BP神经网络；云模型；机器学习；

1. **云模型数字特征**

云模型概念的整体特征可以用云的数字特征来反映，这是定性概念的整体定量特性，对理解定性概念的内涵和外延有着重要意义。

云模型用期望Ex（Expected Value）、熵En（Entropy）和超熵He（Hyper Entropy）三个数字特征来整体表征一个概念。

期望Ex：定性概念的基本确定性的度量，是云滴在论域空间分布中的数学期望。通俗的说，就是最能够代表定性概念的点，或是这个概念量化的最基本的样本。

熵En：定性概念的不确定性度量，由概念的随机性和模糊性共同决定。一方面，熵是定性概念随机性的度量，反映了能够代表这个定性概念的云滴的离散程度；另一方面，又是隶属于这个定性概念的度量，决定了论域空间中可被概念接受的云滴的确定度。

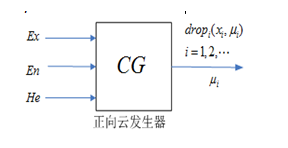
超熵He：熵的熵，是熵的不确定性度量，也可以称为二阶熵。对于一个常识性概念，被普遍接受的程度越高，超熵越小；对于一个在一定范围内能够被接受的概念，超熵较小；对于还难以形成共识的概念，则超熵较大。

1. **正向高斯云算法**

高斯分布是概率论中最重要的分布，通常用均值和方差两个数字特征表示随机变量的整体特征。高斯云作为最常用的云模型，是基于高斯分布但不同于高斯分布的云模型。

正向高斯云算法步骤：

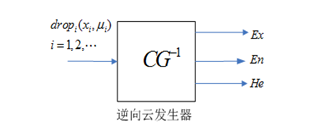
1、生成以En为期望，He为标准差的正态分布随机数En’

1. 生成以Ex为期望，|En‘|为标准差的正态分布随机数x
2. *u*=*e*−2(*En*′)2(*x*−*Ex*)2​

图表 1正向云发生器示意图

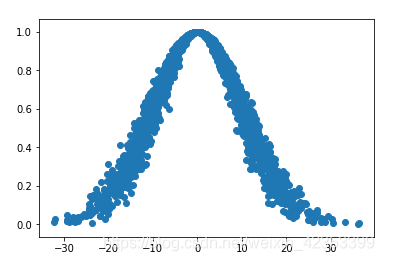
1. **逆向高斯云**

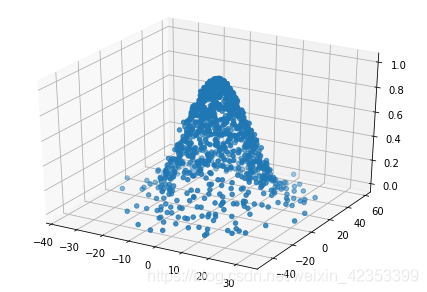
逆向高斯云算法实现从样本数据集合到表示定性概念的数字特征(Ex, En , He ）的变换，是正向高斯云算法的逆运算。逆向云算法本质熵是基于统计的参数估计方法。在课堂所学的逆向云算法抓哟是基于矩估计的方法。



图表 2逆向云发生器示意图

1. **实验目的**

本次的实验目的是能够通过BP神经网络学习云滴特征的方式对数字特征为（0，1，0.25）的云模型进行学习，并能够根据未知云滴特征预测出其原本的云模型数字特征，即通过神经网络的方式

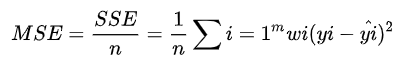
图表 3云模型云滴分布（二维）

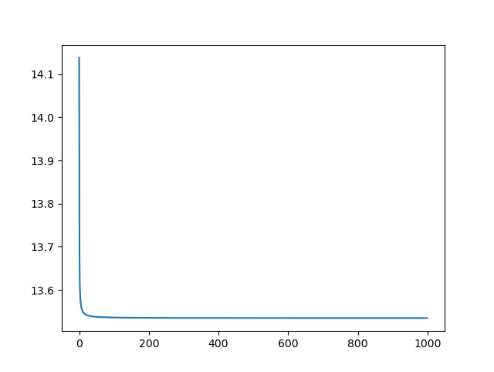
图表 4云模型云滴分布（三维）

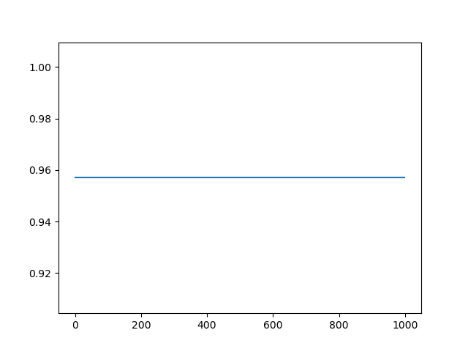
1. **实验步骤**
2. 创建训练数据集。生成1000组正向云模型数据，每组数据1000个点。1000组数据中，900组以（0，1，0.25）为数字特征，作为学习集，另有100组干扰数据集（数字特征为随机生成），用于和标准训练集区分。每组数据有对应的预期标准输出（label），即为该云模型的三个数据特征，均值、熵、超熵。
3. 创建测试数据集。生成20组正向云模型数据，每组数据1000个云滴，作为对训练出的模型的测试数据集，其预测结果用于计算准确率和损失。
4. 提取特征。对于这些数据，根据横坐标（En-3\*He，En+3\*He）范围划分为100个区域，把每个云模型云滴坐落在这100个区域中的个数作为特征，即神经网络的输入为100维向量。
5. 设计并构建神经网络结构。建立BP神经网络，包含三个隐藏层，一个输入层和一个输出层。本文中，设输入层为第0层，从第一隐藏层到输入层分别为第1、2、3、4层。
6. 输入层：每组的输入数据为100维向量
7. 隐藏层：每个隐藏层都包含201个神经元（根据经验，第一个隐藏层神经元个数应为输入层的2\*n+1个，其后的隐藏层设计为与第一隐藏层相等）
8. 输出层：输出层包含三个神经元，分别代表云模型的三个数字特征。、
9. 激活函数：初步设计每一层的激活函数均为带偏置的sigmoid函数。
10. 模型训练。将1中创建的训练数据集用3中的方法处理过后输入神经网络中，并用预期标准输出（label）作为预期输出进行训练，并保存训练好的模型。
11. 输出实验结果，进行结果分析。
12. **实验结果**

在此，实验结果用两张图表来进行表示，一张是训练过程中的loss（损失）变化曲线，另一张是训练过程中对应的accuracy（准确率）变化曲线。两张图表的横坐标为使用训练集训练的次数（从1到1000）。

在本次的实验中，损失函数使用的是**mse**（均方误差）。计算公式如下：





图表 5最初实验条件下的loss（损失）变化曲线

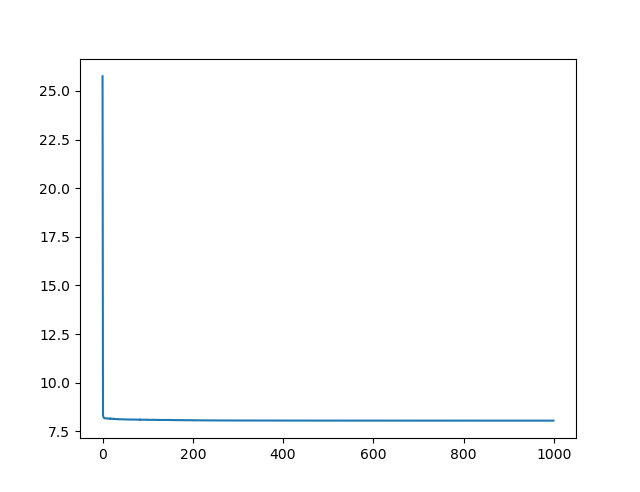
图表 6最初实验条件下的accuracy（准确率）变化曲线

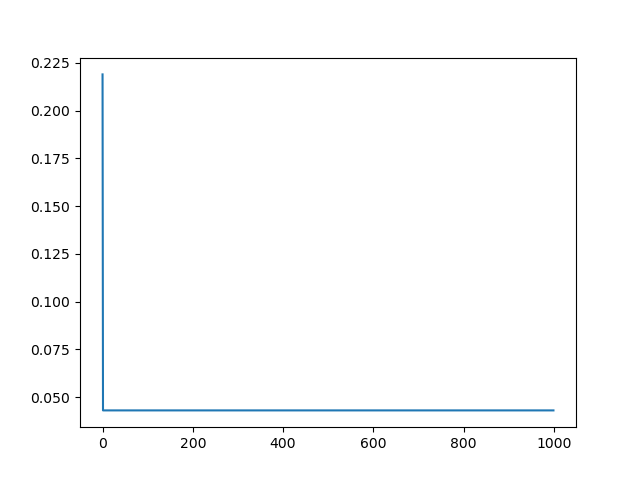
1. **实验结果分析与改进**

由以上两个图表可以看出，在最初实验条件：四层均用**sigmoid**作为激活函数，使用切片区域云滴个数作为输入特征 下，实验的损失十分高，而且准确率并没有显著改变。

由此，针对修改神经网络激活函数和修改网络的特征输入格式对网络进行优化。

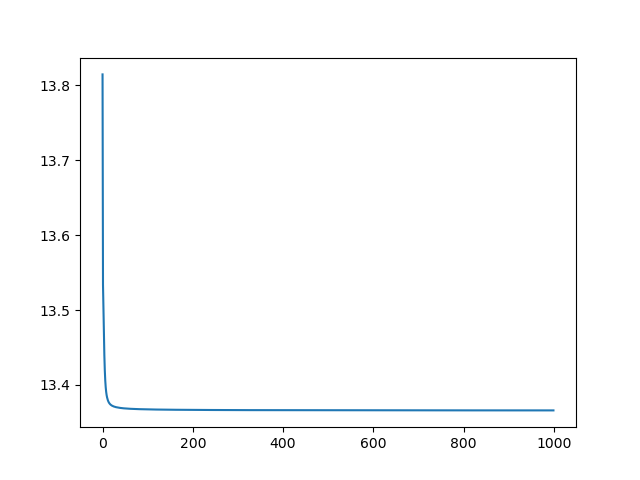
1. 修改神经网络的激活函数
   1. 将激活函数全部改为**ReLu**（线性整流函数）：

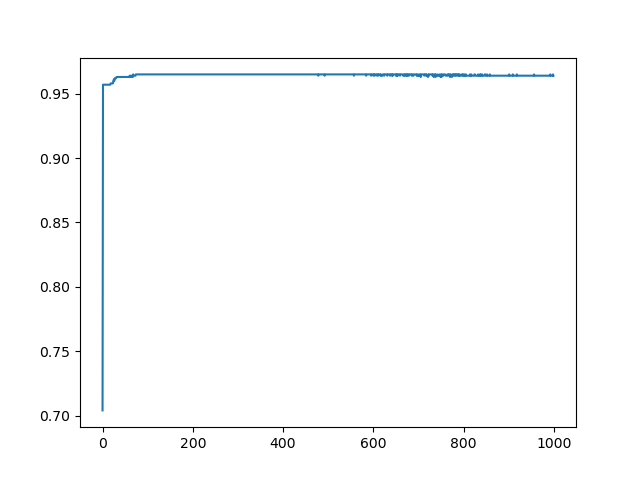


图表 7激活函数全部为**ReLu**函数时的loss（损失）变化曲线

图表 8激活函数全部为**ReLu**函数时的accuracy（准确率）变化曲线

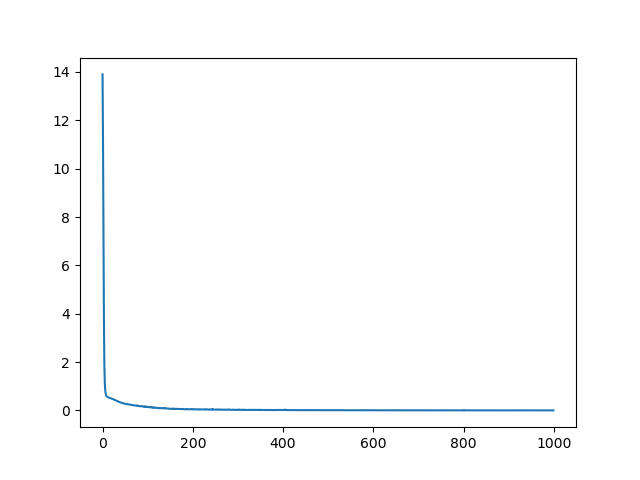
从以上两个图表和图表5、6的比较可知，将各层的激活函数均改为**ReLu**并没有起到优化逆向生成结果的效果，损失和准确率的指标反而变差，由此可见这种改动方法不可取。

* 1. 将第1/3层改为**ReLu**函数，第2/4层仍为**sigmoid**函数：

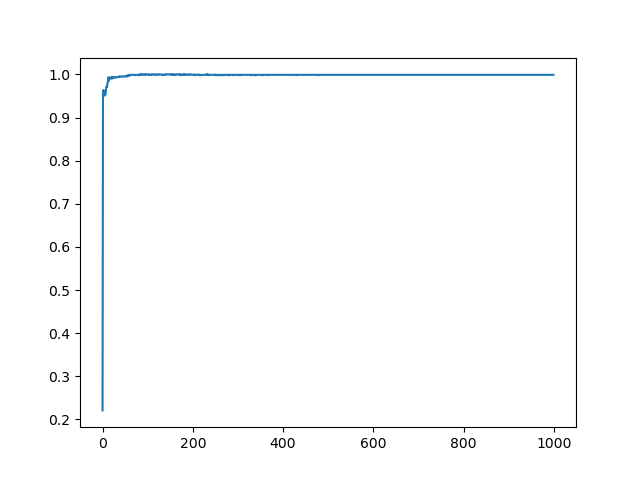
图表 9将第1/3层激活函数改为ReLu函数时的loss（损失）变化曲线

图表 10将第1/3层激活函数改为**ReLu**函数时的accuracy（损失）变化曲线

从以上两个图表和图表5、6的比较可知，将第1、3层的激活函数改为**ReLu**函数的情况下，训练的损失和准确率相比初始条件并没有显著的优化提升。

* 1. 将第2/4层改为**ReLu**函数，第1/3层仍为**sigmoid**函数

图表 11将第2/4层激活函数改为**ReLu**函数时的loss（损失）变化曲线



图表 12将第2/4层激活函数改为**ReLu**函数时的accuracy（准确率）变化曲线

从图表和程序运行数据可以看出，在此情况下，神经网络的训练效果有了显著的提升，损失曲线在训练到第五次时就将损失降到了“1”一下，而最后训练结果的损失在“0.0044”左右；算法的准确率相应的也有显著的提升，最终稳定在 99.9% 左右。

由此可见，将第2、4层的激活函数改为**ReLu**的优化效果显著。

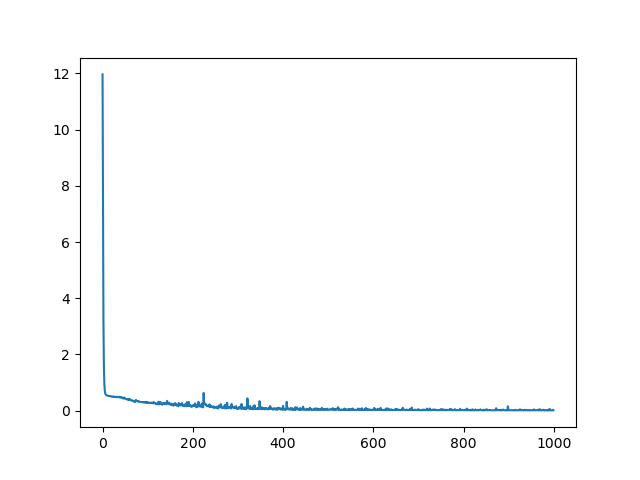
1. 修改神经网络输入特征

在原来的实验中，我根据横坐标（En-3\*He，En+3\*He）范围划分为100个区域，把每个云模型云滴坐落在这100个区域中的个数作为特征。效果并不是很好，因此，我考虑改进输入方法如下：

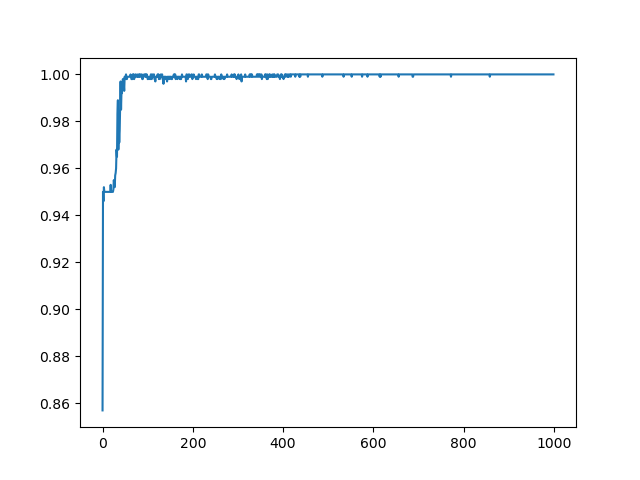
对每组数据生成100个云滴，将这100个云滴的横坐标作为神经网络的输入（选择生成100个点是为了保持原来的神经网络结构不变，而100个点作为输入和1000个点的结果相差并不大）。

除此之外，在此次的实验过程中，我将神经网络中各层的激活函数按照1）中③的方式，即将第2、4层激活函数改为**ReLu**函数，因为这种情况下的预测效果是最好的，有实验参考价值。

修改神经网络输入特征的实验结果如下：



图表 13将神经网络输入改为云滴横坐标后的loss（损失）变化曲线



图表 14将神经网络输入改为云滴横坐标后的accuracy（准确率）变化曲线

由上述实验结果图表和实验输出数据可知，在将第2/4层改为**ReLu**函数，第1/3层仍为**sigmoid**函数，且输入改为所有云滴横坐标的情况下，训练的损失和准确率指标达到最优，损失可以降低到0.0159，而准确率则可以近似达到1。

综上所述：

在将第2/4层改为**ReLu**函数，第1/3层仍为**sigmoid**函数，且输入改为所有云滴横坐标的情况下，训练和预测的效果达到最优。