作业4：神经网络

助教邮箱:wanghc17@mails.tsinghua.edu.cn

1 神经网络的学习能力

本问题我们将探索在分类问题中，两层及三层神经网络的学习能力。两层及三层神经网络的示意图如图1所示。假设样本$\mathbf{x}$是二维bool向量，即$\mathbf{x} = (x_1, x_2)^T$，$x_i \in \{0, 1\}$；我们需要设计神经网络学习分类函数，对样本进行分类。这里，网络激活函数$f(a)$可以表示为：

$$f(a) = \begin{cases} 1 \ if \ a > 0 \\ 0 \ if \ a < 0 \end{cases}$$

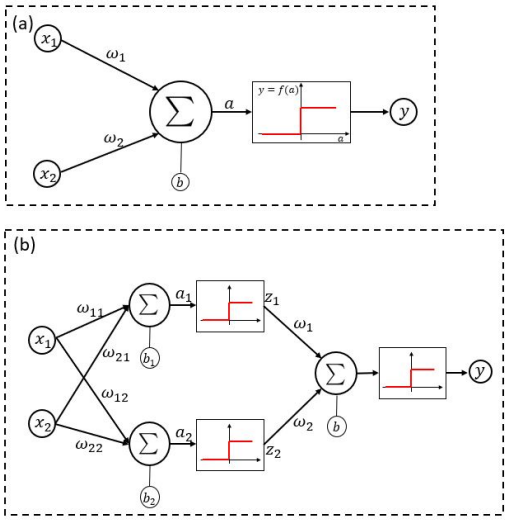

图 1: (a)两层神经网络示例，包括两个输入节点和一个输出节点 (b)三层神经网络示例，包括两个输入节点，两个隐藏节点以及一个输出节点

1. 假设使用两层神经网络完成分类任务，此时神经网络可以用一线性函数$y = f(\boldsymbol{\omega}^T x + b)$表示。请设计权值$\boldsymbol{\omega}$与偏置$b$，如图2所示，使神经网络可完成AND(当且仅当$x = (1, 1)^T$时， $y = 1$)与OR型的分类任务。

2.第一问中的两类样本排布(AND型与OR型）都可以被两层神经网络分类，请思考：什么样的样本排布无法被上述神经网络分类；请提出一种可能情景，并说明原因。

3.设想在三层（单隐层）神经网络的情形下，即$y = f(\boldsymbol{\omega}^T \boldsymbol{z} + b)$,$z_j = f(\boldsymbol{\omega_j}^T \boldsymbol{x} + b_j)$，建立一个可以区分上述样本的神经网络。请设计合理的权值与偏置。

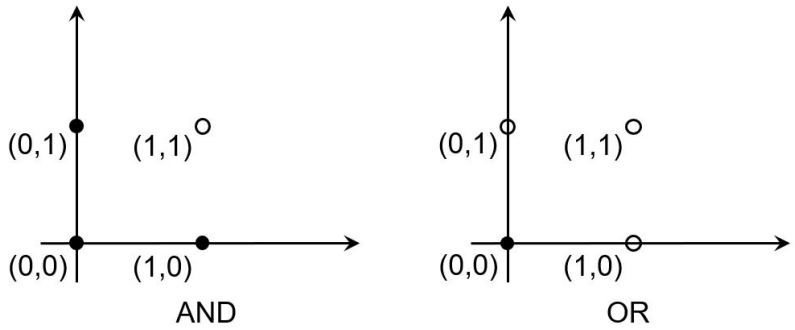

图 2: AND，OR

2 单隐层全连接神经网络梯度计算

考虑一个单隐层全连接神经网络如图3所示：

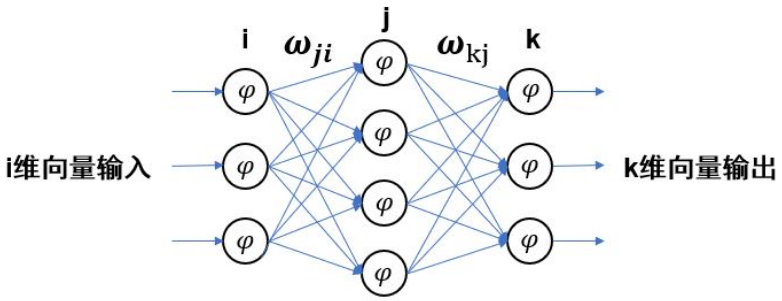

图 3: 单隐层全连接神经网络

该网络包括输入层i（共q个节点），隐含层j（共p个节点），以及输出层k（共c个节点）；n表示第n次迭代，我们设:

$$\begin{cases} E(n) = \frac{1}{2}\sum_{k=1}^{c} e_k^2(n) \\ e_k(n) = d_k(n) - y_k(n) \\ y_k(n) = \varphi(z_k(n)) \\ z_k(n) = \sum_{j=1}^{p} \omega_{kj} y_j(n) \end{cases}$$

其中，$E(n)$代表第n次迭代输出的均方误差；$e_k(n)$表示第n次迭代第k个输出节点的误差；$d_k(n)$表示第k个节点的训练标签；$y_k(n)$表示第k个节点的实际输出；φ表示激活函数；$z_k(n)$表示第k个节点经过激活函数前的输出；$\omega_{kj}(n)$表示第k个节点与第j个节点之间的权值；

在BP算法中，需要根据链式法则，计算权值梯度，并更新权值。因此，我们需要完成如下任务：

1. 利用链式求导法则，求输出层与隐层节点间权值梯度$\frac{\partial E(n)}{\partial \omega_{kj}}$。
2. 利用链式求导法则，求隐层节点与输入层间权值梯度$\frac{\partial E(n)}{\partial \omega_{ji}}$。

3 学习深度学习框架

请同学们至少学习一种主流的深度学习框架，包括但不限于keras，tensorflow，pytorch，Theano等；希

望同学们完成以下工作：

- 将该框架安装到电脑上并通过样例测试，保证框架正确安装；
- 了解全连接网络、CNN网络、RNN网络等各类神经网络的搭建方式，熟悉卷积层、池化层、全连接层等各类网络层的参数设置；
- 了解优化器和损失函数的参数设置；

4 多层神经网络在MNIST上数据集上的应用

MINST是著名的手写体数字图片集合，如图4所示。该数据集包含0∼9共十类手写体图片样本，每张图片由$28 \times 28 = 784$个像素组成。利用神经网络的方法可以从这些手写体数组图片中识别出相应的数字。文件夹中的MNIST.py文件提供了MNIST数据集的载入方式。载入后得到四个矩阵X-train(训练图片)，y-train(训练图片标签)，X-test(测试图片)，y-test(测试图片标签)。对于X-train(60000,28,28)，X-test(10000,28,28)，各个维度分别代表(样本，像素行，像素列)，即对应60000个与10000个28×28的图片，每个像素取值为0到255的整数灰度值；对于y-train(60000,10)，y-test(10000,10)，代表每个样本的标签。

图 4: MNIST手写体数据集

1. 使用三层神经网络完成MNIST数据集的分类问题（输出层激活函数选用softmax，其它激活函数选用relu)：

(a)请先查看数据内容，在所有样本中选取出10个数字(0∼ 9)的图片各一张，绘制出类似于图4的图片；(可能用到的函数:matplotlib.pyplot.imshow)；

(b)预处理。为将图片输入到神经网络中，需要将每一副图片的所有像素都转换成一维向量，例如，对于训练集X-train，需要转换成60000×784的矩阵(可能用到的函数:reshape)；

(c)通常需要将对神经网络的输入数据进行归一化，y的灰度值范围是0-255，所以各维特征除以255；并将数据转换为float格式；

(d)搭建三层全连接神经网络，设置优化器、损失函数；(epochs设置为20，损失函数选用categorical-crossentropy，优化函数任选)；

(e)在训练集合(X-train,y-train)上分割训练集、验证集、测试集(70%训练，15%验证，15%测试)，将训练集送入神经网络训练，并监测验证集误差，当验证集误差持续上升5个epoch后停止训练，以防止过拟合；

(f)保存训练集误差曲线与验证集误差曲线，并在测试集合(X-test,y-test)上进行测试，保存混淆矩阵，分析错误率和收敛速度与隐层节点数目(5,10,20,50,100个隐层节点）的关系，并解释可能的原因。(注意：这里我们使用的是测试集合(X-test,y-test)，而不是训练集分割出来的测试集。在Kaggle比赛中，参赛者会在训练时分出一部分测试集来评测模型效果，而真正的测试集合(X-test,y-test)中的y-test是对于参赛者是未知的。不过，这里我们充当主办方，直接用测试集合(X-test,y-test)对模型进行评价)。

说明：混淆矩阵的含义。如图5所示，横坐标代表目标类别，纵坐标代表训练预测结果类别，各坐标点(m,n)方格中的整数代表将目标类别m预测成了n的样本个数，其中的百分数代表该种情况占所有样本的比例。对角线上的数值为预测正确的情况，其余为预测错误的情况。该矩阵下面额外一行代表的是对应目标类别的分类正

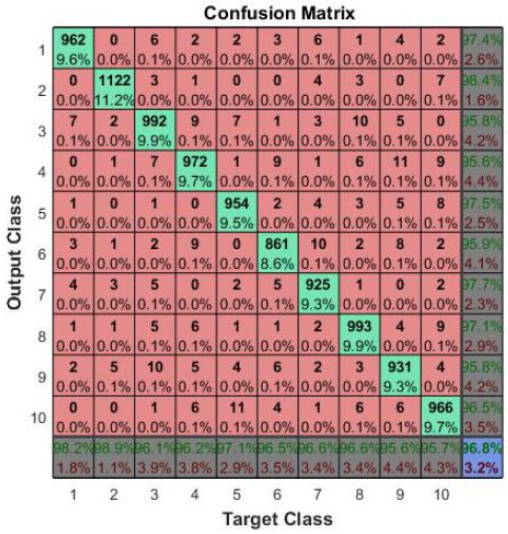

图 5: 混淆矩阵示例

确率和错误率，而矩阵右边的额外一列代表的是对应输出类别的正确率和错误率，右下角为所有样本的分类正确率和错误率。

要求：请提交所有的混淆矩阵，训练误差曲线。

2. 在1中所保存的混淆矩阵中(仅研究隐藏结点为100的模型)，指出混淆最严重的两个数字(只需要一组)，分别展示分错数字的例子(各三个，不足三个的按照实际个数即可)。

5 卷积神经网络进行cifar10图片分类

Cifar-10 是用于普适物体识别的数据集（https://www.cs.toronto.edu/ kriz/cifar.html），由60000张32*32的RGB 彩色图片构成，共10个分类。50000张图片构成训练集合，10000张图片构成测试集合。我们这里只选取了前三个类别共15000张图片做训练集合，3000张图片做测试集合；同样，我们按照(70%训练，15%验证，15%测试)分割训练集合，并完成如下任务：

Input	(32,32,3)
Conv	3 * 3卷积, 64卷积核
激活函数	relu
Max-pooling	2 * 2池化
Dropout	0.25
Flatten	
全连接	64 nodes
激活函数	relu
Dropout	0.5
全连接	Nodes num = 类别个数
激活函数	Softmax

图 6: 卷积神经网络结构

1. 对比0,1,2,3个隐层的全连接神经网络（512个节点，中间层使用relu作为激活函数，输出层选用softmax作为激活函数)与如图6形式的卷积神经网络在3000张测试集合中的正确率；并绘制在训练过程中验证集正确率以及验证集loss随训练代数(epochs)的变化曲线。观察前20个epochs的曲线变化规律，并做简要分析。

2. 将1中的神经网络的中间层激活函数变化为sigmoid，观察验证集正确率及loss变化曲线，以及测试集合中的正确率，并与1中的原始网络作比较，并做简要分析。

3. 据称，dropout层可以一定程度上避免过拟合。请将1中的神经网络的dropout层去除，请观察训练集与验证集的正确率变化曲线，并与1中的原始网络作比较，观察哪一种网络的过拟合现象更严重，并做简单分析。

4. 上述问题仅能初步测试卷积神经网络在图片分类问题中的应用效果，感兴趣的同学可以搜索对cifar10数据集上的各式代码进行测试，观察loss变化曲线及测试集分类,并对各类方法做分析比对。（此问不做要求）

(备注：由于网络结构较为简单，且训练代数较少，我们一般无法看到网络收敛；但通过观察正确率或loss曲线的变化，我们可以观察得到初步的趋势，只需对趋势做简要分析即可）

(推荐把学习率设置为0.1附近)