第5章 MNIST数据集及神经网络

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | 人工智能基础与应用 | | 章名 | | | MNIST数据集及神经网络 | |
| 教学内容 | MNIST数据集及神经网络1 | | | | 课时 | 2 | |
| 项目性质 | □演示性 □验证性 □设计性 √综合性 | | | | | | |
| 授课班级 |  | 授课日期 | |  | | 授课地点 |  |
| 教学目标 | 了解MNIST数据集  熟悉神经元常用函数  掌握深度神经网络 | | | | | | |
| 教学内容 | 1. MNIST数据集 2. 神经元常用函数 3. 深度神经网络 | | | | | | |
| 教学重点 | 神经元常用函数、深度神经网络 | | | | | | |
| 教学难点 | 神经元常用函数、深度神经网络 | | | | | | |
| 教学准备 | 装有Python的计算机  教学课件PPT  教材：《人工智能基础与应用（微课版）》 | | | | | | |
| 作业设计 |  | | | | | | |

教学过程

|  |  |
| --- | --- |
| **教学环节** | **教学内容与过程**  **（教学内容、教学方法、组织形式、教学手段）** |
| **课前组织** | 做好上课前的各项准备工作（打开计算机、打开课件、打开软件、打开授课计划、教案等），吸引学生注意力。 |
| **课程说明** | 【课前说明】  从数据集的基本概念引入本节课学习内容。  【目的】  使学生从了解本节课的学习目标、学习重点、考评方式等方面明确课程学习的要求和目标。 |
| **课程内容描述** | 5.1 MNIST数据集简介  数据集（Dataset）是一类数据的集合。传统的数据集通常表现为表格或者文档形式，每个数值被称为数据资料。不同的数据集，形式是不同的，如图像的数据集可能是文件的形式，在文件中可能是十六进制数值的形式，或者是一系列照片的形式，再或者是视频的形式。  经典的数据集如表所示。  经典的数据集   |  |  | | --- | --- | | 数据集名称 | 功能 | | Iris Flower数据集 | 由罗纳德·费希尔（Ronald Fisher）引入的多变量数据集 | | MNIST数据集 | 通常用于测试分类、聚类和图像处理算法的手写数字图像 | | 分类数据分析数据集 | 一个统计程序清单，可用于分类数据的分析 | | 时间序列数据集 | 在时间上顺序索引的一系列数据 |   MNIST数据集是一个含有手写数字的大型数据集，包含0～9共10个数字，通常用于训练图像处理系统。该数据集还广泛用于机器学习领域的训练和测试。  MNIST数据集包含60000个训练图像和10000个测试图像，其中，训练集的一半和测试集的一半来自NIST的训练数据集，训练集的另一半和测试集的另一半来自NIST的测试数据集。  MNIST数据集共有4个文件，分别是训练集数据、训练集标签以及测试集数据、测试集标签。MNIST数据集的图像以字节的形式进行存储，每幅图像都为单通道图像，由28×28个像素点构成。  MNIST数据集的测试集样本图像如图所示。    MNIST数据集的测试集样本图像  不同分类器使用MNIST数据集的错误率如表所示。  不同分类器使用MNIST数据集的错误率   |  |  | | --- | --- | | 类型 | 错误率（%） | | 非线性分类器 | 3.3 | | [线性分类器](https://en.wikipedia.org/wiki/Linear_classifier) | 7.6 | | 2层深度[神经网络](https://en.wikipedia.org/wiki/Neural_network)（DNN） | 1.6 | | [支持向量机](https://en.wikipedia.org/wiki/Support-vector_machine)（SVM） | 0.56 | | 6层深度[神经网络](https://en.wikipedia.org/wiki/Neural_network)（DNN） | 0.35 | | 5层[卷积神经网络](https://en.wikipedia.org/wiki/Convolutional_neural_network)（CNN） | 0.21 |   5.2 神经元常用函数  神经网络中常用的函数有激活函数、池化函数和损失函数等。 5.2.1 激活函数 在神经网络中，激活函数的存在使得网络加入了非线性因素，从而弥补了线性模型处理非线性问题时的局限性，使神经网络能够更好地解决语音、图像等非线性问题。激活函数的输入是一个数字，然后对该输入进行某种数学运算或操作。  1．sigmoid()函数  在之前的一段时间内，sigmoid()函数是非常常用的，因为它对神经元的激活有很好的解释，且它本身为单调连续，非常适合作为输出层。但它的缺陷也是非常明显的。第一，当输入稍微远离了坐标原点，函数的梯度就变得很小了，几乎为0，这会导致在反向传播过程中，梯度很小的时候接近0，神经网络无法更新参数，这个问题称为梯度饱和，也可以称为梯度弥散。第二，sigmoid()函数的输出不是零中心的，这会影响梯度下降的运作。  2．tanh()函数  tanh()函数的数学公式为，tanh()函数图像如图所示。  5-2  tanh()函数图像  tanh()函数的定义域是，值域是(-1,1)。其输入如果是很大的负数，其值就会无限接近于-1；输入如果是很大的正数，其值就会无限接近于1。和sigmoid()函数一样，tanh()也存在梯度饱和问题，但是tanh()的输出是零中心的，所以实际使用更多一些。  3．ReLU函数  ReLU函数的数学公式为，ReLU函数图像如图所示。  5-3  ReLU函数图像  ReLU函数的定义域是，值域是。这个激活函数可以理解为一个关于0的阈值。ReLU在近几年非常流行。相对于sigmoid()函数和tanh()函数，ReLU函数对于随机梯度下降的收敛有很大的加速作用，克里泽夫斯基（Alex Krizhevsky）等人的论文指出这个加速作用强度是其他两种函数的6倍之多。同时，ReLU函数做的是阈值计算，不是指数运算，更加节省计算资源。但是它的缺点也显而易见，ReLU单元容易“死掉”，当一个很大的梯度流过ReLU的神经元时，神经元将无法被其他任何数据点再次激活，流过这个神经元的梯度将都变成0，而且是不可逆转的。通过合理地设置学习率，出现这种现象的概率会降低。  除了这些激活函数外，还有LeakyReLU函数、Maxout()函数等。需要注意的是：在同一网络中，很少会混合使用不同类型的激活函数。 5.2.2 池化函数 池化是用一个矩阵窗口在张量上进行扫描，通过取最大值或者平均值等方式来减少参数数量的方法。最大值池化和平均值池化的过程如图所示。  5-4  最大值池化和平均值池化的过程  上图中，池化的核为2×2，步长为2，即每次卷积核向右或者向下移动两格，所以在最大值池化中，池化结果左上角的3是由输入数据左上角2×2的[1,1,2,3]中的最大值决定的，平均值池化中池化结果的7/4是由输入数据左上角2×2的[1,1,2,3]的平均值决定的。  为了减少CNN的计算负载、内存使用以及参数量等，同时也为了降低CNN过拟合的风险，在网络中添加池化函数。池化层主要对输入的图像进行降采样（Subsample），池化并不改变深度或维度，只改变大小。池化函数有平均值池化函数和最大值池化函数等。  1．平均值池化函数  TensorFlow函数：tf.nn.avg\_pool(value,ksize,strides,padding,name=None)，该函数计算的是池化区域的平均值。  value：需要池化的输入，一般池化层接在卷积层后面，这是一个四维的张量，4个维度为[batch,height,width,channels]。  ksize：池化窗口的大小，是一个四维向量，一般是[1,height,width,1]。  strides：和卷积类似，是窗口在每一个维度上滑动的步长，一般是[1,stride,stride,1]。  padding：和卷积类似，可以取“VALID”或“SAME”。  name：该池化的名称。  2．最大值池化函数  TensorFlow函数：tf.nn.max\_pool(value,ksize,strides,padding,name=None)，该函数计算的是池化区域的最大值。  在池化时一般选择最大值池化。 5.2.3 损失函数 在第4章的线性回归中，直接使用L2距离作为损失函数，解决了回归问题；在逻辑回归中，使用作为损失函数，解决了二分类问题。  1．hinge损失函数  hinge损失函数源自支持向量机，在支持向量机中，最终的目的是最大化分类间隔，减少错误分类的样本数目。损失函数公式为，其中，，是一个超参数，设置为1.0时，在绝大多数情况下都是安全的。  2．square损失函数  损失函数公式为，大多用在线性回归中。  3．log损失函数  在使用似然函数最大化时，其形式是进行连乘，但是为了便于处理，一般会加上log，这样便可以将连乘转换为求和。由于log函数是单调递增函数，因此不会改变优化结果。log类型的损失函数也是一种常见的损失函数，例如，在第4章中，使用逻辑回归解决二分类问题时就使用了该损失函数。  4．交叉熵  在神经网络中解决多分类问题一般需要设计*n*个输出节点，如在MNIST数据集中，有0～9共10个结果，所以需要10个输出节点。可以使用一个*n*维的数组作为输出，如图像识别结果为5，那么数组中下角标为5的结果要和其余的不一样，理想的输出结果为[0,0,0,0,0,1,0,0,0,0]。当然，一般不可能有这么理想的结果产生。  如何判断输出的结果与期望结果有多接近呢？交叉熵（Cross Entropy）是常用的方法之一。交叉熵的公式为，是分类问题的真实分布概率，是分类问题的预测分布概率，也就是说，交叉熵的输入是概率，范围是[0,1]。交叉熵得到的值越小，真实分布概率和预测分布概率越接近，预测的结果就越真实。  由于网络的输出是任意的，如ReLU函数的值域为，所以在进行交叉熵计算之前还需要将输出结果转换为概率分布。softmax()函数就是一个常用的方法，它的公式为，该函数所有输出值的和为1，输入的负数会变成正数，然后外面嵌套函数，就可以得到优化结果。  举一个例子就非常容易理解这个过程了。如果识别0、1、2这3个值，输出只有3个分类，现在有一个真实值为0的图像需要进行识别，假如神经网络一的输出结果为[3,0.9,-2]，那么使用softmax()函数之后约等于[0.886,0.108,0.006]，[0.886,0.108,0.006]经过交叉熵计算后得到的结果约等于0.12。如果现在有一个新的神经网络二，使用softmax()函数输出为[0.5,0.3,0.2]，经过交叉熵计算后得到的结果约等于0.69。由于神经网络一中softmax()的结果为[0.886,0.108,0.006]，相对于神经网络二中softmax()的结果[0.5,0.3,0.2]，对于两个结果中下角标为0的值，0.886比0.5更接近于1，所以神经网络一的效果更好。但是在实际运算中，由于难度较大，所以不会直接比较softmax()的结果，使用交叉熵的值作为优化的标准。在交叉熵的计算结果中，0.69是大于0.12的，所以根据计算结果可知第一个网络更好。  5.3 深度神经网络  深度神经网络（Deep Neural Networks，DNN）可以理解为有很多隐藏层的神经网络。  第4章的线性分类方程为*y*=***W****x*+***b***。其中，***W***是权重矩阵；***b***是偏置矩阵；*x*是输入，它包含数据集的所有信息，如数据集是图像，输入的是全部像素信息。  具有一层隐藏层的神经网络如图所示。  5-5  具有一层隐藏层的神经网络  神经网络某个节点的计算公式是，*n*为网络的第*n*层，是权重矩阵，是第一层的偏置矩阵，*f*()是激活函数，它会作用到每个元素。经过这些激活函数的转换后，由于每个节点不再是线性变换的，所以整个神经网络就成为非线性的了。为了便于理解，此处将矩阵简单地表示为常数，在图5-5中，若，，，，，，，，，，，，，，，选取ReLU函数作为激活函数，隐藏层节点推导公式为：  （5.1）  （5.2）  （5.3）  输出层推导公式为：  （5.4）  在该网络中，如果没有非线性这一步，计算、、和*Y*的先后两次矩阵运算将会合并为一次，所以非线性过程是非常重要的。  参数***W***和***b***可以通过梯度下降的方法在反向传播的过程中通过链式求导法则计算得到。  反向传播过程：如果有表达式，现在需要求出，先将这个复合函数的表达式分解为和，根据链式求导法则可知，所以需要分别求出和，可以求得，，故得到。前向传播与反向传播过程如图所示。  5-6  图5-6 前向传播与反向传播过程  在上图中，假设，，，每个圆里面是变量名，在圆的上方是前向传播的结果，圆的下方是反向传播的结果，连接线上是运算符。 |
| **总结评价** | 本节课主要介绍了MNIST数据集，从数据集开始引入了神经网络的结构。 |