

深度学习入门

基于Python的理论与实现

Stephen CUI 

March 14, 2023

Chapter 1

神经网络的学习

本章的主题是神经网络的学习。这里所说的“学习”是指从训练数据中自动获取最优权重参数的过程。

1.1 从数据中学习

神经网络的特征就是可以从数据中学习。所谓“从数据中学习”，是指可以由数据自动决定权重参数的值。

1.1.1 数据驱动

如果让我们自己来设计一个能将5正确分类的程序，就会意外地发现这是一个很难的问题。人可以简单地识别出5，但却很难明确说出是基于何种规律而识别出了5。

深度学习有时也称为端到端机器学习（end-to-end machine learning）。这里所说的端到端是指从一端到另一端的意思，也就是从原始数据（输入）中获得目标结果（输出）的意思。

1.1.2 训练数据和测试数据

机器学习中，一般将数据分为训练数据和测试数据两部分来进行学习和实验等。首先，使用训练数据进行学习，寻找最优的参数；然后，使用测试数据评价训练得到的模型的实际能力。为什么需要

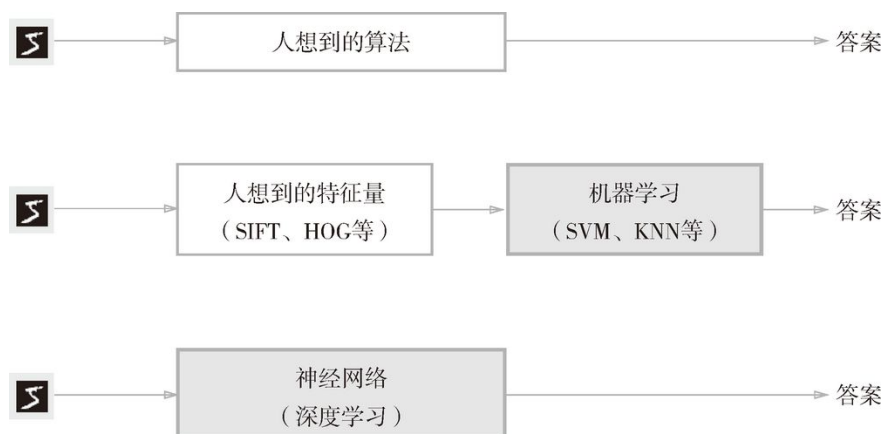


Figure 1.1: Moving from human designed rules to machine learning from data

将数据分为训练数据和测试数据呢？因为我们追求的是模型的泛化能力。为了正确评价模型的泛化能力，就必须划分训练数据和测试数据。另外，训练数据也可以称为监督数据。

泛化能力是指处理未被观察过的数据（不包含在训练数据中的数据）的能力。获得泛化能力是机器学习的最终目标。

1.2 损失函数

神经网络的学习通过某个指标表示现在的状态。然后，以这个指标为基准，寻找最优权重参数。神经网络的学习中所用的指标称为损失函数（loss function）。这个损失函数可以使用任意函数，但一般用均方误差和交叉熵误差等。

损失函数是表示神经网络性能的“恶劣程度”的指标，即当前的神经网络对监督数据在多大程度上不拟合，在多大程度上不一致。以“性能的恶劣程度”为指标可能会使人感到不太自然，但是如果给损失函数乘上一个负值，就可以解释为“在多大程度上不坏”，即“性能有多好”。并且，“使性能的恶劣程度达到最小”和“使性能的优良程度达到最大”是等价的，不管是用“恶劣程度”还是“优良程度”，做的事情本质上都是一样的。

1.2.1 均方误差

可以用作损失函数的函数有很多，其中最有名的是均方误差（mean squared error）。均方误差如下式所示。

$$E = \frac{1}{2} \sum_k (y_k - t_k)^2 \quad (1.1)$$

这里， y_k 是表示神经网络的输出， t_k 表示监督数据， k 表示数据的维数。

1.2.2 交叉熵误差

除了均方误差之外，交叉熵误差（cross entropy error）也经常被用作损失函数。交叉熵误差如下式所示：

$$E = - \sum_k t_k \log y_k \quad (1.2)$$

1.2.3 mini-batch学习

机器学习使用训练数据进行学习。使用训练数据进行学习，严格来说，就是针对训练数据计算损失函数的值，找出使该值尽可能小的参数。因此，计算损失函数时必须将所有的训练数据作为对象。

前面介绍的损失函数的例子中考虑的都是针对单个数据的损失函数。如果要求所有训练数据的损失函数的总和，以交叉熵误差为例，Equation 1.2改写为：

$$E = - \frac{1}{N} \sum_n \sum_k t_{nk} \log y_{nk} \quad (1.3)$$

式中，假设数据有 N 个， t_{nk} 表示第 n 个数据的第 k 个元素的值（ y_{nk} 是神经网络的输出， t_{nk} 是监督数据）。

如果遇到大数据，数据量会有几百万、几千万之多，这种情况下以全部数据为对象计算损失函数是不现实的。因此，我们从全部数据中选出一部分，作为全部数据的“近似”。神经网络的学习也是从训练数据中选出一批数据（称为mini-batch,小批量），然后对每个mini-batch进行学习。这种学习方式称为mini-batch学习。

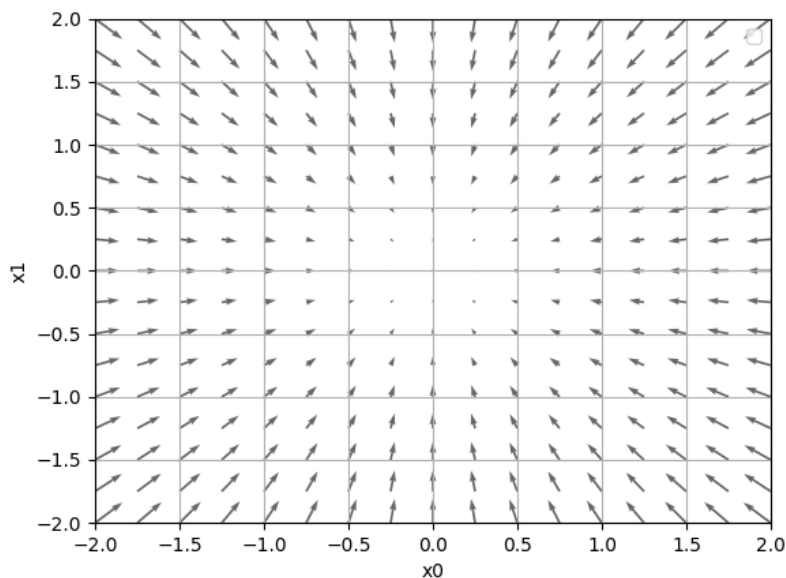


Figure 1.2: 2D Gradient Schematic

计算电视收视率时，并不会统计所有家庭的电视机，而是仅以那些被选中的家庭为统计对象。比如，通过从关东地区随机选择1000个家庭计算收视率，可以近似地求得关东地区整体的收视率。这1000个家庭的收视率，虽然严格上不等于整体的收视率，但可以作为整体的一个近似值。和收视率一样，mini-batch的损失函数也是利用一部分样本数据来近似地计算整体。也就是说，用随机选择的小批量数据（mini-batch）作为全体训练数据的近似值。

1.2.4 mini-batch版交叉熵误差的实现

1.2.5 为何要设定损失函数

在进行神经网络的学习时，不能将识别精度作为指标。因为如果以识别精度为指标，则参数的导数在绝大多数地方都会变为0。

其实我觉得是数学层面的妥协，因为没有办法给出精度的准备数学方程，因此很多求解方法都无法应用。

1.3 数值微分

1.4 梯度

像 $\left(\frac{\partial f}{\partial x_0}, \frac{\partial f}{\partial x_1}\right)$ 这样的由全部变量的偏导数汇总而成的向量称为梯度（gradient）。

实际上，梯度会指向各点处的函数值降低的方向。更严格地讲，梯度指示的方向是各点处的函数值减小最多的方向¹。这是一个非常重要的性质。

¹ $Directional derivative = \cos(\theta) \times grad$ （ θ 是方向导数的方向与梯度方向的夹角）。因此，所有的下降方向中，梯度方向下降最多。

1.4.1 梯度法

机器学习的主要任务是在学习时寻找最优参数。同样地，神经网络也必须在学习时找到最优参数（权重和偏置）。这里所说的最优参数是指损失函数取最小值时的参数。但是，一般而言，损失函数很复杂，参数空间庞大，我们不知道它在何处能取得最小值。而通过巧妙地使用梯度来寻找函数最小值（或者尽可能小的值）的方法就是梯度法。

这里需要注意的是，梯度表示的是各点处的函数值减小最多的方向。因此，无法保证梯度所指的方向就是函数的最小值或者真正应该前进的方向。实际上，在复杂的函数中，梯度指示的方向基本上都不是函数值最小处。

函数的极小值、最小值以及被称为鞍点（saddle point）的地方，梯度为0。极小值是局部最小值，也就是限定在某个范围内的最小值。鞍点是从某个方向上看是极大值，从另一个方向上看则是极小值的点。虽然梯度法是要寻找梯度为0的地方，但是那个地方不一定就是最小值（也有可能是极小值或者鞍点）。此外，当函数很复杂且呈扁平状时，学习可能会进入一个（几乎）平坦的地区，陷入被称为“学习高原”的无法前进的停滞期。

虽然梯度的方向并不一定指向最小值，但沿着它的方向能够最大限度地减小函数的值。因此，在寻找函数的最小值（或者尽可能小的值）的位置的任务中，要以梯度的信息为线索，决定前进的方向。

在梯度法中，函数的取值从当前位置沿着梯度方向前进一定距离，然后在新的地方重新求梯度，再沿着新梯度方向前进，如此反复，不断地沿梯度方向前进。像这样，通过不断地沿梯度方向前进，逐渐减小函数值的过程就是梯度法（gradient method）。梯度法是解决机器学习中最优化问题的常用方法，特别是在神经网络的学习中经常被使用。

根据目的是寻找最小值还是最大值，梯度法的叫法有所不同。严格地讲，寻找最小值的梯度法称为梯度下降法（gradient descent method），寻找最大值的梯度法称为梯度上升法（gradient ascent method）。但是通过反转损失函数的符号，求最小值的问题和求最大值的问题会变成相同的问题，因此“下降”还是“上升”的差异本质上并不重要。一般来说，神经网络（深度学习）中，梯度法主要是指梯度下降法。

用数学式来表示梯度法：

$$\begin{aligned}x_0 &= x_0 - \eta \frac{\partial f}{\partial x_0} \\x_1 &= x_1 - \eta \frac{\partial f}{\partial x_1}\end{aligned}\tag{1.4}$$

式中 η 表示更新量，在神经网络的学习中，称为学习率（learning rate）。学习率决定在一次学习中，应该学习多少，以及在多大程度上更新参数。学习率需要事先确定为某个值，比如0.01或0.001。一般而言，这个值过大或过小，都无法抵达一个“好的位置”。在神经网络的学习中，一般会一边改变学习率的值，一边确认学习是否正确进行了。

像学习率这样的参数称为超参数。这是一种和神经网络的参数（权重和偏置）性质不同的参数。相对于神经网络的权重参数是通过训练数据和学习算法自动获得的，学习率这样的超参数则是

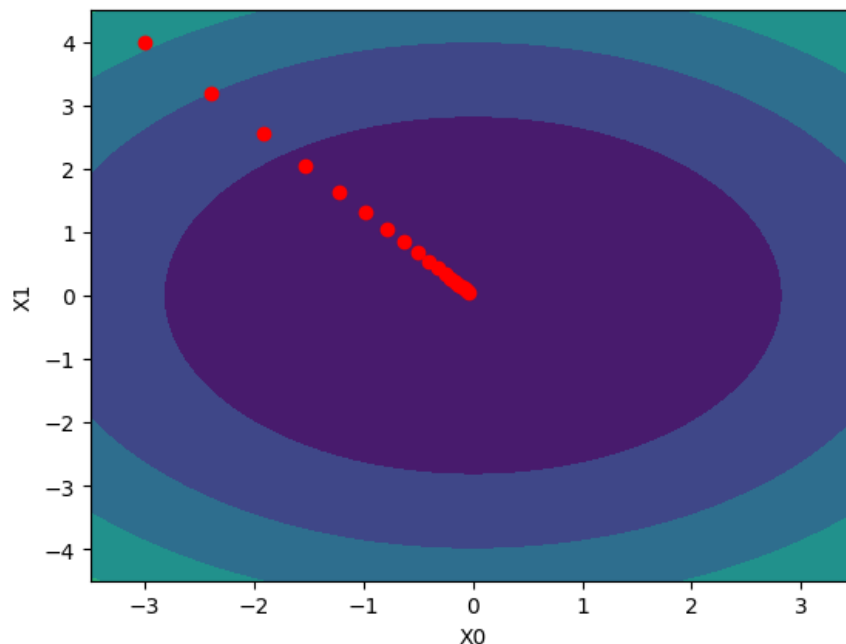


Figure 1.3: The update process of the gradient method

人工设定的。一般来说，超参数需要尝试多个值，以便找到一种可以使学习顺利进行的设定。

1.4.2 神经网络的梯度

1.5 学习算法的实现

神经网络的学习步骤如下所示：

- 前提：神经网络存在合适的权重和偏置，调整权重和偏置以便拟合训练数据的过程称为“学习”。
- 步骤1（mini-batch）从训练数据中随机选出一部分数据，这部分数据称为mini-batch。我们的目标是减小mini-batch的损失函数的值。
- 步骤2（计算梯度）为了减小mini-batch的损失函数的值，需要求出各个权重参数的梯度。梯度表示损失函数的值减小最多的方向。
- 步骤3（更新参数）将权重参数沿梯度方向进行微小更新。
- 步骤4（重复）重复步骤1、步骤2、步骤3。