神经网络概述

0. 前置知识 1：向量

在人工智能这个领域中，向量是一串表示特定信息的数字，一个特定的向量代表着一个独一无二的信息。

比如，身份证号码就可以被视为某种向量，这一串号码包含了你的地址、生日等各种信息，而这串数字只属于你。

数学里面学过，一个二维平面中的向量可以表示成(x, y)，画在图上就是一个从原点指向点（x,y）的箭头。这叫做“二维向量”
那么推广到三维的空间呢？
三维空间的点会有(x, y, z)三个坐标，表示这个点在这个三维空间中的位置。那么向量(x, y, z)就是从原点指向点(x, y, z)的一个箭头，所以这种(x, y, z)拥有 3 个值的向量我们叫做“三维向量”。

三维是人类大脑处理的极限，但是并不是向量的极限
向量还可以是四维的，可以用(x1, x2, x3, x4)来表示（其他字母也行，这里只是图方便用 x1 x2 之类的），这意味着，五维、六维等等的向量都是可以有的。

由此得出，向量可以是 n 维的！我们可以用(x1, x2, ……, xn)来表示一个 n 维的向量！！！

另外，表示相似信息的向量在空间中会指向相近的方向，就像是一个住址和你很近、生日和你差不多的人身份证号码会和你比较像，但不会完全相同。

前置知识 2：模型的训练——监督训练
AI 模型不是我们写完代码就可以用的，它需要“学习”。我们需要收集很多数据喂给它。比如，我想做视觉识别模型，识别图片中的物品，那么在写完代码之后，我需要给模型“看”很多图片，并且这些图片都有配套的“正确答案”，比如图中展示的图片是猫，那么这个图片正确答案就是“猫”，模型就会学习到“猫是长成这个样子的”。这里，这个正确答案叫做“标签（label）”，当然我们不会只给它看一张猫的图片，还要喂给它各种各样猫的图片，标签都是“猫”。就像是根据例题学习，你只看一道例题，可能只会死记硬背，但是你看很多例题，那么你这个种类的题目都会做了，在考试的时候就不虚。这种根据输入和其对应标签让 AI 学习的方式，叫做“监督学习（Supervised Learning）”。模型在学习完之后，才能投入实际使用，这个就是它考试的时候了，遇到没有标签的猫的图片，一个训练得好的模型就能认出这是“猫”。

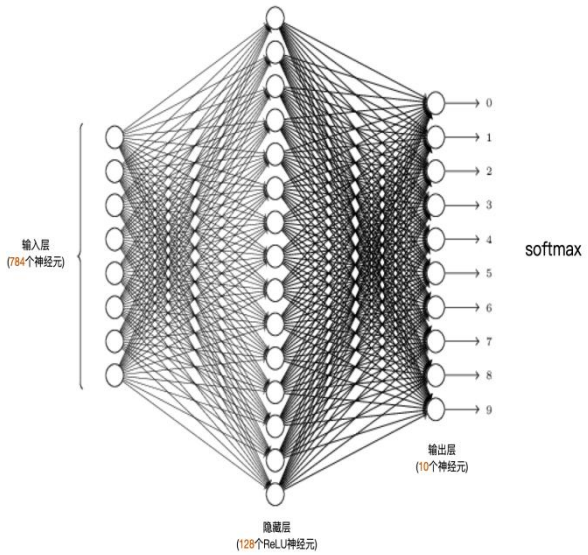
1. 定义
神经网络(Neural Network)是一种计算模型，模拟生物神经系统的结构和功能，由大量称为神经元的简单处理单元组成。这些神经元通过连接（权重）相互作用，能够从数据中学习和识别模式。

神经网络由以下主要部分组成：

输入层：接收外部数据。
隐藏层：位于输入层和输出层之间，负责特征提取和转换。是“思考”的部分
输出层：输出思考后的最终结果

通俗点讲，就是你对着神经网络输入一个什么数据，然后神经网络通过“思考”，给你想要的结果。
神经元：神经网络上的“节点”，每个神经元都会对输入的数据进行某些变换和计算。

2. 结构解析

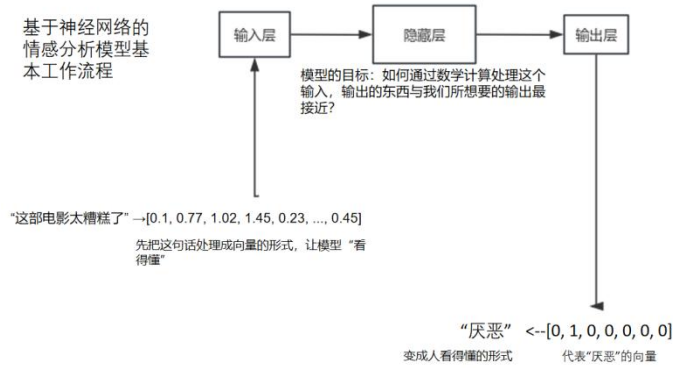


这是一个简单的神经网络，具体来讲，这种结构叫做“前馈神经网络”，它的输入层有 784 个神经元，负责接受输入的数据。隐藏层有 128 个神经元，负责处理输入的数据，“捕捉”输入的一些特征（例如，你输入的是一只猫的图片，那么隐藏层就会通过某些方式去捕捉图片中猫的特征，比如耳朵、尾巴、毛茸茸的质感）。输出层有 10 个神经元，负责输出你想要的结果。

一个有趣但是有点难懂的事实：隐藏层的神经元越多，越能够捕捉输入的各种特征。但是太多的神经元可能会在运行的时候干爆你的显卡。

这样也许会很难理解。我们可以先尝试不从“神经元”的层面理解，
神经网络的隐藏层会通过一系列数学计算来处理我们的输入，然后把算出来的数据传到输出层，得到我们想要的输出，仅此而已。

3. 原理解析
以“神经网络情感分析”为例，我们的工作流程是：



4. 数据流动过程（这个过程叫做前向传播）
当输入的向量进入输入层，来到隐藏层后，会经过各种数学变换
“隐藏层”很多时候又是很多“小层”组合而成的
我们两个参数：权重 w 和偏置 b
然后所有的数值都通过一个"过滤器"（通过一个函数 f，暂时先记住就行，这种函数叫做“激活函数”，在这里我们使用的激活函数叫做“ReLU”）
我们先计算：

$$y = wx + b$$

这个看起来像是一次函数，它叫做线性变换，而完成这个线性变换的“小层”叫做“线性层”
再计算：

$$f(y)$$

其中 f 是 ReLU
把上一个线性层的输出 y 通过激活函数，完成这个操作的小层叫做“激活层”
隐藏层有很多小层时，上一小层的输出会被作为下一小层的输入
这里，权重 w 和偏置 b 在实际情况下，模型会自己调整
模型会不断试错来自动调整 w 和 b，这个过程就属于“机器学习”（Machine Learning）
模型的每个线性层都有属于自己的 w 和 b，在学习过程中，自动调整所有的 w 和 b
详细推导见《神经网络数学推导》

最终，经过这些隐藏层处理的数据会通过一个叫做 softmax 的函数，它是一个激活函数，这个函数会生成一个概率分布。在我们这个模型中，一共有 7 个情感，那么 softmax 就会输出我们输入的句子进行数学变换之后这句话每个情感的可能性是多少，呈现的形式是一个 7 维向量，7 个情感的概率之和为 1。

最终呈现：
假设输出结果是[0.01, 0.89, 0.03...]，并且向量的第二个维度代表“厌恶”，说明：
“厌恶”的可能性最高
神经网络判断这句话的情感是“厌恶”

这里的举例只是针对这个“情感分析模型”的，你在实际构建神经网络解决各种问题的时候，你可以根据任务来构建这些层，比如使用其他的激活函数、在隐藏层堆叠更多或者更少的“小层”

5. 神经网络如何学习（反向传播）

上文提到，所有层的 w 和 b 都是在学习过程中自己调整的，以下是解释模型自己调整的原理
学习三要素：
▫ 损失函数：考试评分标准（预测值与真实值的差距）
▫ 优化器：智能辅导老师（根据错误调整参数）
▫ 反向传播：错题分析系统（逐层追溯错误来源）

举个现实案例：
当网络把"厌恶"误判为"平静"时：
1. 计算误差：正确答案“平静”对应的概率应该是 0.9，但是实际只有 0.1
2. 逆向检查：从输出层→隐藏层→输入层逐层追溯，看看是哪里的参数调整的不好
3. 参数调整：降低导致误判的神经连接权重，加强正确特征的权重 w，同时也调整“主观判断”偏置 b，最终提高我们模型的准确度
详细内容见《神经网络数学推导》

6. 激活函数：神经网络的"开关"
为什么需要？
想象你喝水的过程：只有水量超过喉咙承受能力时才会咽下去。激活函数就是神经网络的"吞咽阈值"，决定信息是否传递。

常见类型：
- ReLU（修正线性单元）：最简单的开关
  输入<0 → 关闭（输出 0）`
  输入≥0 → 直接通过`
  → 就像水龙头：水量不够不出水，够量就直接放行

- Sigmoid：温柔过渡
  把任意数值压缩到 0-1 之间，适合概率判断
  → 类似考试成绩换算：卷面分 100 → 转换成优秀等级 0.95

- Softmax：终极裁判
  专门用于输出层，把 10 个输出值变成概率总和为 1
  → 像选秀打分：10 位评委给分后，自动换算成选手得票比例
详细见《神经网络数学推导》

7. 损失函数：考试的评分标准
核心作用：告诉神经网络"错得多离谱"
监督学习中，就是看看，模型在刷题时，自己做出来的答案和正确答案差了多远
常见类型：
- MSE（均方误差）：适合预测房价等连续值
  (预测价格-真实价格)² → 就像报价偏差越大扣分越狠

- 交叉熵损失：适合分类（比如识别猫狗）
  如果真实答案是猫：

-log(模型认为猫的概率)→ 猜得越准扣分越少

假设考试满分 100 分：
- 把"9"认成"4" → 扣 30 分
- 把"9"认成"7" → 扣 20 分
- 正确认出"9" → 扣 0 分

神经网络的目标就是不断刷题，让自己考试扣分越来越少
详细内容见《神经网络数学推导》

8. 优化器：智能导航系统
作用：告诉神经网络"该怎么调整权重才能更快进步"
经典算法：
- SGD（随机梯度下降）：基础版导航
  → 像蒙眼下山：用脚试探坡度，找下降最快的方向

- Adam：智能升级版
  → 像老司机开车：根据历史经验自动调节油门和刹车
  （实际会根据梯度大小自动调整学习步长）

学习率：
→ 相当于"学习时的迈步大小"
- 步子太大（学习率高）：容易错过正确答案
- 步子太小（学习率低）：学习速度太慢

详细内容见《神经网络数学推导》

9. 拟合 vs 欠拟合
欠拟合：
→ 小学生做高考题，根本学不会（模型太简单）
表现：模型蠢得要死，什么都不会干

过拟合：
→ 学霸死记硬背答案，遇到新题就懵（模型太复杂）
表现：模型在训练学习的时候表现很好，但是实际应用的时候表现很糟糕

应对武器：
- Dropout：随机让部分神经元"失忆"
  → 像考试时随机抽走几本笔记，防止死记硬背
- L2 正则化：限制权重不能太大
  → 像防沉迷系统，防止过度专注某些特征

详细内容见《神经网络数学推导》

10. Epoch vs Batch
- Batch（批次）：

→ 一次练习做多少题
比如有 60000 张训练图，设置 batch_size=100 → 分 600 次完成一轮练习

- Epoch（轮次）：
→ 整套试卷做多少遍
比如 epoch=10 → 把 60000 张图反复学习 10 遍

为什么分批次？
→ 人类也无法一次性记住所有知识，分批次学习更高效

11. 归一化（Normalization）
为什么需要？
假设判断一个人是否健康：
- 身高 1.7 米 vs 体重 70000 克 → 数值量级差异导致模型混乱

做法：
把输入数据压缩到统一范围（比如 0-1 之间）
→ 类似把身高换算成"米"，体重换算成"千克"

12. 模型评估指标
- 准确率：考试得分（识别 100 张图，正确 90 张 → 90%）
- 混淆矩阵：详细错题集（显示把"3"认成"8"多少次）

13. 实际应用时的神奇现象
- 特征自动提取：
当网络深度足够时，浅层学边缘/颜色，深层学耳朵/车轮等抽象特征
→ 像人类先认笔画，再认偏旁，最后认整个字

总结流程图：
输入图片 → 归一化 → 前向传播（层层提取特征）
↘ ↗
反向传播（根据错误调整权重）
↗ ↘
损失计算 ← 与正确答案对比