神经网络基础作业答案

April 2025

1　第一次作业

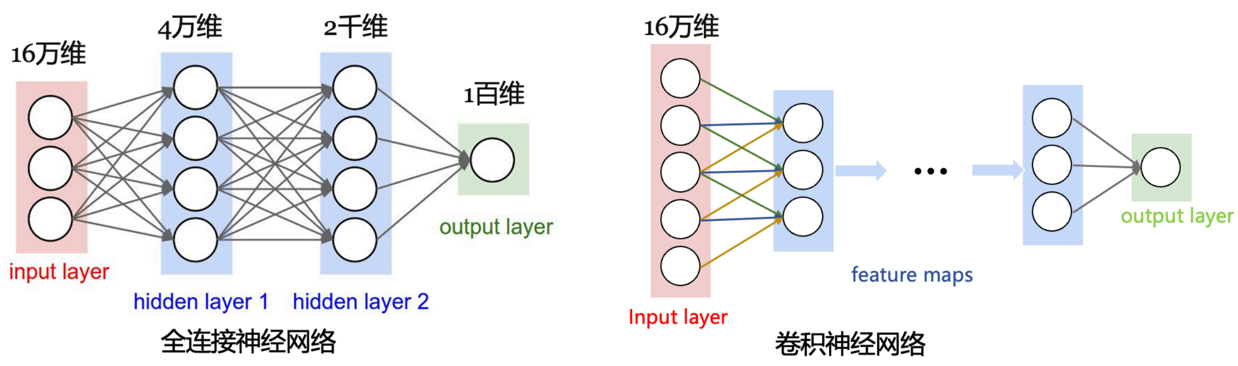

图 1: 神经网络示意图

1.1

在图??左图所示的四层全连接神经网络中，输入层维度为 160,000，第一个隐藏层维度为 40,000，第二个隐藏层维度为 2,000，输出层维度为 100。请计算该神经网络的参数总量。
答：

每个全连接层的参数量由输入大小、输出大小和偏置项决定。神经网络的参数总量为各层之间的连接参数之和：

- 输入层 → 隐藏层 1：$(160,000+1)\times 40,000 = 6,400,040,000$
- 隐藏层 1 → 隐藏层 2：$(40,000+1)\times 2,000 = 80,002,000$
- 隐藏层 2 → 输出层：$(2000+1)\times 100 = 200,100$

综上，总参数量为 $6,400,040,000 + 80,002,000 + 200,100 = 6,480,242,100 \approx 6.4\times 10^9$.

1.2

在图??右图所示的卷积神经网络中，输入层维度为 160,000，第一个卷积层使用 3 个 5*5 的卷积核，使用 max pooling 将特征图尺寸降为 40,000，第二个卷积层使用 5 个 4*4*3 的卷积核，紧接着使用 max pooling 将特征图尺寸降为 2,000，第三个卷积层使用 5 个 4*4*5 的卷积核，使用 max pooling 将特征图尺寸降为 400，在卷积过程中始终使用 padding 保持输入和输出维度一致，最后一层使用全连接层将维度展平后降为 100。请计算该神经网络的参数总量。

答:

卷积层参数计算公式：参数量 = (卷积核大小 × 前一层特征图的通道数) × 当前层卷积核数量

因此各层参数计算如下:

- 卷积层 1 参数（输入通道数为 1，输出通道数为 3，卷积核大小为 5*5+1）：$3 \times (5 \times 5 + 1) = 78$
- 卷积层 2 参数（输入通道数为 3，输出通道数为 5，卷积核大小为 4*4*3+1）：$5 \times (4 \times 4 \times 3 + 1) = 245$
- 卷积层 3 参数（输入通道数为 5，输出通道数为 5，卷积核大小为 4*4*5+1）：$5 \times (4 \times 4 \times 5 + 1) = 405$
- 全连接层：(输入维度为 400，输出维度为 100)：$(400 + 1) \times 100 = 40,100$

综上，总参数量为 $78 + 245 + 405 + 40,100 = 40,828$.

2 第二次作业

2.1

在图??神经网络，假设激活函数为 ReLU，用平方损失 $\frac{1}{2}(y-\hat{y})^2$ 计算误差，请用 BP 算法更新一次所有参数（学习率为 1），给出更新后的参数值，并计算给定输入值 $x = (0.2, 0.3)$ 时初始时和更新后的输出值，检查参数更新是否降低了平方损失值。

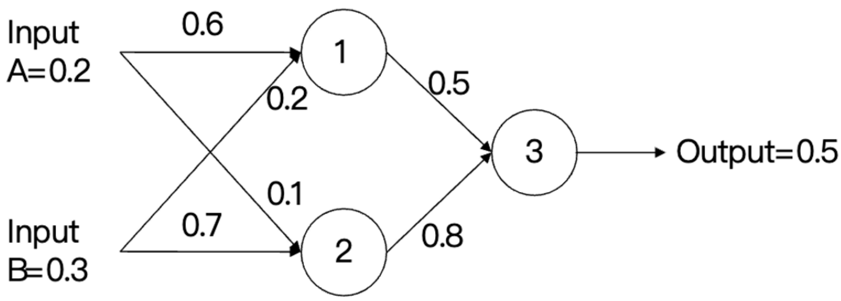

图 2: 神经网络示意图

答:

参考课程讲义的解法，题目忽略偏置的计算。首先对符号进行表示:

- 激活函数 ReLU 表示为: $f(x) = \max\{0, x\}$
- 输入层第 $i(i = 1, 2)$ 个输入为 x_i.
- 输入层第 i 个神经元到隐藏层第 $h(h = 1, 2)$ 个神经元之间的连接权重 v_{ih}.
- 隐藏层第 h 个神经元的输入表示为 $\alpha_h = \sum_{i=1}^{2} v_{ih} x_i$，输出表示为 b_h.
- 隐藏层第 h 个神经元到输出层第 $j(j = 1)$ 个神经元之间的连接权重 $w_{hj}(j = 1)$.

- 输出层输入表示为 $\beta_j = \sum_{h=1}^{2} w_{hj} b_h$, 输出层的输出表示为 $y_j = f(\beta_j)$.
- 损失函数表示为 $E = \frac{1}{2}(y - \hat{y})^2$, 期望输出表示为 $\hat{y} = 0.5$.

接下来对前向传播过程和反向传播过程分别进行计算:
前向传播过程:

- 隐藏层输入: $\alpha_1 = \sum_{i=1}^{2} v_{i1} x_i = 0.6 \times 0.2 + 0.2 \times 0.3 = 0.18$, $\alpha_2 = \sum_{i=1}^{2} v_{i2} x_i = 0.1 \times 0.2 + 0.7 \times 0.3 = 0.23$
- 隐藏层输出: $b_1 = f(_1) = b_1 = 0.18, b_2 = f(_2) = b_2 = 0.23$
- 输出层输入: $y = y_1 = f(\beta_1) = f(\sum_{i=1}^{2} w_{h1} b_h) = f(0.5 \times 0.18 + 0.8 \times 0.23) = f(0.274) = 0.274$

损失计算:

$$E = \frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}(0.274 - 0.5)^2 = 0.025538$$

.

后向传播过程:
首先根据链式法则计算 Δw_{hj}.

$$\begin{aligned}\Delta w_{hj} &= -\eta \frac{\partial E}{\partial w_{hj}} \\ &= -\eta \frac{\partial E}{\partial y_j} \cdot \frac{\partial y_j}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial w_{hj}}.\end{aligned} \tag{1}$$

$$\begin{aligned}g_j &= -\frac{\partial E}{\partial y_j} \cdot \frac{\partial y_j}{\partial \beta_j} \\ &= -(y - \hat{y}) \cdot \frac{\partial y_j}{\partial \beta_j} \\ &= -(y - \hat{y}) \cdot f'(\beta_j).\end{aligned} \tag{2}$$

根据 ReLU 函数 f 的性质，当 $\beta_j \leq 0$ 时, $f'(\beta_j) = 0$, 反之，$f'(\beta_j) = 1$.
因为 $\beta_j > 0$, 所以 $g_j = -(y - \hat{y}) = 0.226$.
再将 $\frac{\partial \beta_j}{\partial w_{hj}} = b_h$, $\eta = 1$ 代入，得到 $w_{hj} = \eta g_j b_h = 0.226 b_h$.
因此得到

$$w_{11}^{+} = w_{11} + \Delta w_{11} = 0.5 + 0.226 \times 0.18 = 0.54068$$

$$w_{21}^{+} = w_{21} + \Delta w_{21} = 0.8 + 0.226 \times 0.23 = 0.85198$$

.

同理，计算 Δv_{ih}. 已知 $\eta = 1$

$$
\begin{aligned}
\Delta v_{ih} &= -\eta \frac{\partial E}{\partial v_{ih}} \\
&= -\frac{\partial E}{\partial b_h} \cdot \frac{\partial b_h}{\partial \alpha_h} \cdot \frac{\partial \alpha_h}{\partial v_{ih}} \\
&= -\frac{\partial E}{\partial b_h} \cdot \frac{\partial b_h}{\partial \alpha_h} x_i \\
&= -\sum_{i=1}^{2} \frac{\partial E}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial b_h} f'(\alpha_h) x_i \\
&= \sum_{j=1}^{1} w_{hj} g_j f'(\alpha_h) x_i
\end{aligned} \tag{3}
$$

通过 ReLU 函数的性质，易得 $f'(\alpha_1) = 1, f'(\alpha_2) = 1$

代入得

$$
v_{11}^{+} = v_{11} + \Delta v_{11} = v_{11} + \eta w_{11} g_1 x_1 = 0.6 + 1 \times 0.5 \times 0.226 \times 0.2 = 0.6226
$$

同理，$v_{12}^{+} = 0.13616, v_{21}^{+} = 0.2339, v_{22}^{+} = 0.75424$

计算权重更新后的前向传播:

过程省略,输出结果 $y' = (0.6226*0.2+0.2339*0.3)*0.54068+(0.13616*0.2+0.75424*0.3)*0.85198 = 0.3212453271 \approx 0.3212$

则新的误差为 $E^{+} = \frac{1}{2}(0.3212 - 0.5)^2 = 0.01598 < E = 0.02554.$

综上可知，参数更新后损失值降低。

2.2

计算 $\sigma(x) = \frac{1}{1+exp\ (-x)}$ 的一阶和二阶导数、$\log \mathrm{softmax}(x)_{[i]} = \log \frac{\exp(x_i)}{\sum_{j=1}^{C} \exp(x_j)}$ 的梯度

答:

$\sigma(x) = \frac{1}{1+exp\ (-x)}$ 一阶导数：$\sigma'(x) = \frac{exp(-x)}{(1+exp(-x))^2} = \sigma(1-\sigma)$

$\sigma(x) = \frac{1}{1+exp\ (-x)}$ 二阶导数：$\sigma''(x) = \sigma(1-\sigma)^2 - \sigma^2(1-\sigma) = \sigma(1-\sigma)(1-2\sigma)$

计算 $\log \mathrm{softmax}(x)_{[i]} = \log \frac{\exp(x_i)}{\sum_{i=1}^{C} \exp(x_j)}$ 梯度:

$\log \mathrm{softmax}(x)_{[i]} = \log \frac{\exp(x_i)}{\sum_{i=1}^{C} \exp(x_j)} = x_i - \log \sum_{j=1}^{C} exp(x_j)$

$$
\nabla \log \mathrm{softmax}(x)_{[i]} = \begin{cases} 1 - \dfrac{exp(x_k)}{\sum_{j=1}^{C} exp(x_j)} = 1 - \mathrm{softmax}(x)_{[k]} & k = i \\ -\dfrac{exp(x_k)}{\sum_{j=1}^{C} exp(x_j)} = -\mathrm{softmax}(x)_{[k]} & k \neq i \end{cases} \tag{4}
$$

2.3

2.3.1 每层主要运算

每一层 l（$l = 1, 2, \ldots, L$）执行以下运算:

1. **线性变换**:

$$
z^{(l)} = W^{(l)} a^{(l-1)} + b^{(l)}
$$

其中:

  - $W^{(l)}$：$n \times n$ 权重矩阵。
  - $a^{(l-1)}$：n 维输入向量（前一层的输出）。
  - $b^{(l)}$：n 维偏置向量。

2. **激活函数**：

$$a^{(l)} = \sigma(z^{(l)}), \quad \sigma(z) = \frac{1}{1 + e^{-z}}$$

2.3.2 每层计算复杂度

- **矩阵-向量乘法** $W^{(l)} a^{(l-1)}$：
  - 乘法：$n \times n = n^2$ 次（矩阵 $n \times n$，向量 $n \times 1$）。
  - 加法：$n \times (n-1) \approx n^2$ 次（每个输出元素需 n 次乘法后累加 $n-1$ 次）。
- **偏置加法** $b^{(l)}$：
  - 加法：n 次（每个元素加一次）。
- **Sigmoid 函数** $\sigma(z^{(l)})$：
  - 对 n 个元素逐一计算，复杂度主要为指数运算，暂不计入乘法和加法。

每层操作总数（忽略 Sigmoid 内部复杂度和低阶项）：

- 乘法：$O(n^2)$ 次。
- 加法：$O(n^2 + n) = O(n^2)$ 次。

2.3.3 整个网络前向传播

网络共 L 层，每层乘法约 n^2 次，总计：

$$\text{总乘法次数} = L \times n^2$$

加法总数类似，约为 $L \times n^2$。

每层主要计算

反向传播计算梯度并更新参数，每层 l 涉及：

1. **误差项计算**：
   - **输出层**（$l = L$，均方误差损失）：

$$\delta^{(L)} = (a^{(L)} - y) \odot \sigma'(z^{(L)})$$

   其中，y 是 n 维真实标签，$\odot$ 表示逐元素乘法，Sigmoid 导数：

$$\sigma'(z) = \sigma(z)(1 - \sigma(z))$$

- **隐藏层**（$l = 1, 2, \dots, L-1$）：

$$\delta^{(l)} = \left((W^{(l+1)})^T \delta^{(l+1)}\right) \odot \sigma'(z^{(l)})$$

2. **梯度计算**：

   - 权重梯度：

$$\nabla_{W^{(l)}} = \delta^{(l)} (a^{(l-1)})^T$$

   - 偏置梯度：

$$\nabla_{b^{(l)}} = \delta^{(l)}$$

3. **参数更新**（学习率 α）：

$$W^{(l)} \leftarrow W^{(l)} - \alpha \nabla_{W^{(l)}}$$

$$b^{(l)} \leftarrow b^{(l)} - \alpha \nabla_{b^{(l)}}$$

2.3.4 每层计算复杂度

- **误差项计算**：
  - 输出层：
    * 差值 $(a^{(L)} - y)$：n 次减法。
    * $\sigma'(z^{(L)})$：n 次乘法（每个元素计算 $\sigma(z)(1-\sigma(z))$）。
    * 逐元素乘法：n 次乘法。
  - 隐藏层：
    * 矩阵-向量乘法 $(W^{(l+1)})^T \delta^{(l+1)}$：$n^2$ 次乘法。
    * $\sigma'(z^{(l)})$ 和逐元素乘法：n 次乘法。
  - 总计：$n^2 + n$ 次乘法（隐藏层为主）。
- **梯度计算**：
  - 权重梯度 $\delta^{(l)} (a^{(l-1)})^T$：$n^2$ 次乘法（外积生成 $n \times n$ 矩阵）。
  - 偏置梯度：无需乘法。
- **参数更新**：
  - 权重更新：n^2 次乘法（逐元素乘 α 并减）。
  - 偏置更新：n 次乘法。

每层总乘法（以隐藏层为主，忽略低阶项）：

$$n^2 + n + n^2 + n^2 + n = 3n^2 + 2n \approx 3n^2$$

2.3.5 整个网络反向传播

网络共 L 层，每层乘法约 $3n^2 + 2n$，总计：

$$\text{总乘法次数} = L \times (3n^2 + 2n) = 3Ln^2 + 2Ln$$

2.4

如果用如下卷积替换前馈网络中的线性变换，试计算误差项 $\frac{\partial \mathcal{L}}{z_{ij}^{(l)}}$ 的递推式。

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial z_{ij}^{(l)}} &= \sum_{u=0}^{K-1}\sum_{v=0}^{K-1} \frac{\partial \mathcal{L}}{\partial z_{(i-u)(j-v)}^{(l+1)}} \cdot \frac{\partial z_{(i-u)(j-v)}^{(l+1)}}{\partial a_{ij}^{(l)}} \cdot \frac{\partial a_{ij}^{(l)}}{\partial z_{ij}^{(l)}} \\ &= f'(z_{ij}^{(l)}) \cdot \sum_{u=0}^{K-1}\sum_{v=0}^{K-1} \omega_{uv}^{(l+1)} \cdot \frac{\partial \mathcal{L}}{\partial z_{(i-u)(j-v)}^{(l+1)}}\end{aligned} \tag{5}$$

2.5

如果有池化层，这一层的误差如何反向传播?

设池化窗口大小为 $K' \times K'$，步长为 S.

假如是平均池化:

$a_{ij}^{(l)} = f(z_{ij}^{(l)})$ 将被改写为 $a_{ij}^{(l)} = f(\max_{u=0}^{K'-1} \max_{u=0}^{K'-1} z_{Si+u,Sj+v}^{(l)})$

反向传播时，梯度仅传递到前向传播中最大值的位置，其他位置梯度为零。

假如是最大池化:

$a_{ij}^{(l)} = f(z_{ij}^{(l)})$ 将被改写为 $a_{ij}^{(l)} = f(\frac{1}{K'^2} \sum_{u=0}^{K'-1} \sum_{u=0}^{K'-1} z_{Si+u,Sj+v}^{(l)})$

反向传播时，梯度平均分配到前向传播中对应的位置。

3 第三次作业

3.1 Xavier 初始化，tanh 激活函数推导

考虑权重矩阵 $W^{(l)}$，其中元素 $W_{ij}^{(l)}$ 假设期望是 0，所以 $\mathbb{E}W_{ij}^{(l)} = 0$。令 $z_i^{(l)}$ 是第 l 层第 i 个神经元的输入: $z_i^{(l)} = \sum_{j=1}^{M_{l-1}} W_{ij}^{(l)} a_j^{(l-1)} + b_i^{(l)}$

当我们计算 $z_i^{(l)}$ 的方差时: $\mathrm{Var}(z_i^{(l)}) = \sum_{j=1}^{M_{l-1}} \mathrm{Var}(W_{ij}^{(l)} a_j^{(l-1)})$

假设 $W_{ij}^{(l)}$ 和 $a_j^{(l-1)}$ 相互独立，则: $\mathrm{Var}(W_{ij}^{(l)} a_j^{(l-1)}) = \mathrm{Var}(W_{ij}^{(l)}) \cdot \mathrm{Var}(a_j^{(l-1)}) + \mathrm{Var}(W_{ij}^{(l)}) \cdot \mathbb{E}[a_j^{(l-1)}]^2 + \mathrm{Var}(a_j^{(l-1)}) \cdot \mathbb{E}[W_{ij}^{(l)}]^2$。

因为 $\mathbb{E}[W_{ij}^{(l)}] = 0$，且假设输入的均值为零，所以: $\mathrm{Var}(W_{ij}^{(l)} a_j^{(l-1)}) = \mathrm{Var}(W_{ij}^{(l)}) \cdot \mathrm{Var}(a_j^{(l-1)})$

前向传播中稳定（每层输入方差一致）：我们希望: $\mathrm{Var}(z_i^{(l)}) = \mathrm{Var}(a_j^{(l-1)})$，即: $\sum_{j=1}^{M_{l-1}} \mathrm{Var}(W_{ij}^{(l)}) \cdot \mathrm{Var}(a_j^{(l-1)}) = \mathrm{Var}(a_j^{(l-1)})$

因为所有的 $a_j^{(l-1)}$ 有独立同分布,所以有相同的方差,所以: $M_{l-1} \cdot \mathrm{Var}(W_{ij}^{(l)}) \cdot \mathrm{Var}(a_j^{(l-1)}) = \mathrm{Var}(a_j^{(l-1)})$。因此: $\mathrm{Var}(W_{ij}^{(l)}) = \frac{1}{M_{l-1}}$。

反向传播梯度一致: 对于误差 $\delta^{(l)} = \frac{\partial L}{\partial z^{(l)}}$，我们有: $\frac{\partial L}{\partial W_{ij}^{(l)}} = \delta_i^{(l)} \cdot a_j^{(l-1)}$

对于反向传播: $\delta^{(l-1)} = (W^{(l)})^T \delta^{(l)} \cdot f'(z^{(l-1)})$

经过类似推导，为了保持反向传播中梯度的方差稳定，我们得到: $\mathrm{Var}(W_{ij}^{(l)}) = \frac{1}{M_l}$

求解: 结合前向和反向传播的方差要求，取调和平均数: $\mathrm{Var}(W_{ij}^{(l)}) = \frac{2}{M_l + M_{l-1}}$。

3.1.1 均匀分布初始化

对于均匀分布 $U(-r,r)$，其方差为: $\mathrm{Var}(W_{ij}^{(l)})=\frac{r^2}{3}$。令 $\frac{r^2}{3}=\frac{2}{M_l+M_{l-1}}$ 解得

$$r=\sqrt{\frac{6}{M_{l-1}+M_l}}$$

3.1.2 高斯分布初始化

直接令方差等于 $\mathrm{Var}(W_{ij}^{(l)})=\sigma^2=\frac{2}{M_l+M_{l-1}}$ ，求解即可。

3.2 PRelu 作为激活函数的初始化

PReLU (Parametric Rectified Linear Unit) 激活函数定义.

$$f(x)=\begin{cases} x, & \text{if } x>0 \\ \alpha x, & \text{if } x\leq 0 \end{cases}$$

其中 α 是一个可学习的参数，通常初始化为一个小的正数（如 0.01 或 0.25）。

推导前置推导与 ReLU 保持一致，参考神经网络 PPT46，47，48 页。替换激活函数

$$\mathbb{E}\left[\left(a^{(l-1)}\right)^2\right]=\mathbb{E}\left[\left(\mathrm{PReLU}\left(z^{(l-1)}\right)\right)^2\right]=\int_{-\infty}^{+\infty}\left(\mathrm{PReLU}\left(z^{(l-1)}\right)\right)^2 p\left(z^{(l-1)}\right)\,dz^{(l-1)}$$

,

由于 PReLU 是分段函数，上式可替换

$$\mathbb{E}\left[\left(a^{(l-1)}\right)^2\right]=\int_{-\infty}^{0}\left(\alpha z^{(l-1)}\right)^2 p\left(z^{(l-1)}\right)\,dz^{(l-1)}+\int_{0}^{+\infty}\left(z^{(l-1)}\right)^2 p\left(z^{(l-1)}\right)\,dz^{(l-1)}$$

$$=\alpha^2\int_{-\infty}^{0}\left(z^{(l-1)}\right)^2 p\left(z^{(l-1)}\right)\,dz^{(l-1)}+\int_{0}^{+\infty}\left(z^{(l-1)}\right)^2 p\left(z^{(l-1)}\right)\,dz^{(l-1)}$$

$$=\frac{1+\alpha^2}{2}\mathbb{E}(z^{(l-1)})^2$$

后续同 ReLU 推导一致，参考 PPT48，

$$\mathrm{var}\left[z^{(l)}\right]=M_{l-1}\mathrm{var}\left[w^{(l)}\right]\mathbb{E}\left[\left(a^{(l-1)}\right)^2\right]$$

3.3

按照 60 页面的方式计算下 ResNet 的参数量和激活的存储开销。

以 ResNet-18 为例，ResNet-18 的网络结构如图：

计算过程如下：

```
INPUT: [224×224×3]    memory: 224×224×3 = 150 K      params: 0
CONV7−64: [112×112×64]    memory: 112×112×64 = 802 K     params: (7×7×3)×64 = 9 408
POOL: [56×56×64]    memory: 56×56×64 = 200 K       params: 0
# conv2_x, 共 2 个 Basic Block, 每个 Block 包含 2 层 3×3×64 卷积
CONV3−64: [56×56×64]    memory: 56×56×64 = 200 K       params: (3×3×64)×64 = 36 864
```

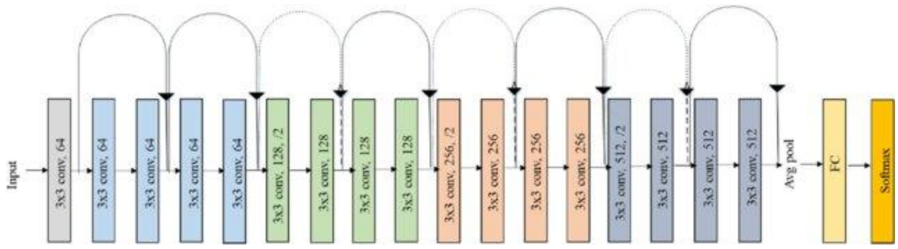

图 3: ResNet-18 网络结构

CONV3–64: [56×56×64]　memory: 56×56×64 = 200 K　params: (3×3×64)×64 = 36 864

(以上为第 1 个 Block; 以下同理)

CONV3–64: [56×56×64]

CONV3–64: [56×56×64]

→ 共 4 层, memory 累计 4×200 K = 800 K; params 累计 4×36 864 = 147 456

conv3_x, 共 2 个 Basic Block, 首层下采样

CONV3–128 (下采样): [28×28×128]

memory: 28×28×128 = 100 K　params: (3×3×64)×128 = 73 728

CONV3–128: [28×28×128]

memory: 28×28×128 = 100 K　params: (3×3×128)×128 = 147 456

(以上为第 1 个 Block; 以下为第 2 个 Block)

CONV3–128: [28×28×128]

memory: 100 K　params: (3×3×128)×128 = 147 456

CONV3–128: [28×28×128]

memory: 100 K　params: (3×3×128)×128 = 147 456

→ 共 4 层, memory 累计 4×100 K = 400 K; params 累计 73 728 + 3×147 456 = 516 096

conv4_x, 共 2 个 Basic Block, 首层下采样

CONV3–256 (下采样): [14×14×256]

memory: 14×14×256 = 50 K　params: (3×3×128)×256 = 294 912

CONV3–256: [14×14×256]

memory: 50 K　params: (3×3×256)×256 = 589 824

(以上为第 1 个 Block; 以下为第 2 个 Block)

CONV3–256: [14×14×256]

memory: 50 K　params: (3×3×256)×256 = 589 824

CONV3–256: [14×14×256]

memory: 50 K　params: (3×3×256)×256 = 589 824

→ 共 4 层, memory 累计 4×50 K = 200 K; params 累计 294 912 + 3×589 824 = 2 064 384

conv5_x, 共 2 个 Basic Block, 首层下采样

CONV3–512 (下采样): [7×7×512]

memory: 7×7×512 = 25 K　params: (3×3×256)×512 = 1 179 648

CONV3–512: [7×7×512]

memory: 25 K　params: (3×3×512)×512 = 2 359 296

(以上为第 1 个 Block；以下为第 2 个 Block)

CONV3–512: $[7 \times 7 \times 512]$

memory: 25 K params: $(3 \times 3 \times 512) \times 512 = 2\ 359\ 296$

CONV3–512: $[7 \times 7 \times 512]$

memory: 25 K params: $(3 \times 3 \times 512) \times 512 = 2\ 359\ 296$

→ 共 4 层, memory 累计 4×25 K = 100 K; params 累计 $1\ 179\ 648 + 3 \times 2\ 359\ 296 = 8\ 257\ 53$

POOL (Global Avg): $[1 \times 1 \times 512]$

memory: $1 \times 1 \times 512 = 0.512$ K params: 0

FC: $[1 \times 1 \times 1000]$ memory: $1 \times 1 \times 1000 = 1$ K params: $512 \times 1000 = 512\ 000$

TOTAL memory:

150 K + 802 K + 200 K + 800 K + 400 K + 200 K + 100 K + 0.512 K + 1 K

≈ 2 653.5 K ≈ 2.6535 M × 4 bytes ≈ 10.6 MB / 图像

TOTAL params:

9 408 + 147 456 + 516 096 + 2 064 384 + 8 257 536 + 512 000

= 11 506 880 ≈ 11.5 M parameters