模式识别与计算机视觉：第一次作业

人工智能学院 221300066 季千焜

2025 年 5 月 21 日

1 习题一

不妨设公式：

$$f(a)=\sqrt[3]{a+\frac{a+1}{3}\sqrt{\frac{8a-1}{3}}}+\sqrt[3]{a-\frac{a+1}{3}\sqrt{\frac{8a-1}{3}}}$$

(a)

由于只考虑实数的情况，虚部为零，因此有

$$\frac{8a-1}{3}\quad\geq 0$$

则可推出对输入的要求:

$$a\geq\frac{1}{8}$$

(b)

当 $a=\frac{1}{8}$ 时，带入公式有

$$f\left(\frac{1}{8}\right)=\sqrt[3]{\frac{1}{8}}+\sqrt[3]{\frac{1}{8}}=\frac{1}{2}+\frac{1}{2}=1$$

(c)

带入方便计算的特殊样例 $a=\frac{1}{2}$ 可得

$$f\left(\frac{1}{2}\right)=\sqrt[3]{\frac{1}{2}}+\sqrt[3]{\frac{1}{2}}=\frac{1}{\sqrt[3]{2}}+\frac{1}{\sqrt[3]{2}}=1$$

同理带入方便计算的特殊样例 $a = \frac{13}{8}$ 可得

$$f\left(\frac{13}{8}\right) = \frac{3}{2} + \frac{\sqrt[3]{-1}}{2} = 1$$

我们发现两者结果均为 1。

(d)

这条命令的返回值为 1.2182 + 0.1260i。

(e)

由于 $(.)^{(1/3)}$ 在 MATLAB 中等价于 `power( , 1/3)`，该函数是在复数域内计算，最终计算结果的误差会累计增大，得到一个错误的结果，我们应该使用在实数域计算的函数 `nthroot( , n)`，即使用

```
a = 3 / 4
f = nthroot(a + (a + 1)/3 * sqrt((8*a-1)/3),3) + ...
nthroot(a + (a + 1)/3 * sqrt((8*a-1)/3),3)
```

可以算出结果为 1。

给 $a \geq 0.125$ 带入不同的值，依然等于这个结果。

(f)

由于 $a \geq \frac{1}{8}$，我们不妨令 $a = \frac{3x^2}{8} + \frac{1}{8}$，其中 $x \geq 0$，则有

$$f\left(\frac{3x^2}{8} + \frac{1}{8}\right) = \sqrt[3]{a + \frac{a+1}{3}\sqrt{\frac{8a-1}{3}}} + \sqrt[3]{a - \frac{a+1}{3}\sqrt{\frac{8a-1}{3}}}$$

$$= \sqrt[3]{\frac{3x^2 + 1 + (x^2+3)\sqrt{\frac{3x^2-1}{3}}}{8}} + \sqrt[3]{\frac{3x^2 + 1 - (x^2+3)\sqrt{\frac{3x^2-1}{3}}}{8}}$$

$$= \frac{\sqrt[3]{-(x-1)^3}}{2} + \frac{\sqrt[3]{(x+1)^3}}{2}$$

$$= \frac{1-x}{2} + \frac{1+x}{2} = 1$$

可见当 $a \geq \frac{1}{8}$ 时有 $f(a) = 1$。

(g)

令 $a = 2$，则有

$$f(2) = \sqrt[3]{2 + \frac{2+1}{3}\sqrt{\frac{16}{3} - 1}} + \sqrt[3]{2 - \frac{2+1}{3}\sqrt{\frac{16}{3} - 1}}$$

$$= \sqrt[3]{2+\sqrt{5}} + \sqrt[3]{2-\sqrt{5}}$$

$$= 1$$

(h)

查阅资料后，得知 Cardano 证明了三次方程

$$z^3 + pz + q = 0$$

其中 p, q 是实数，且 $\Delta = \frac{q^2}{4} + \frac{p^3}{27} > 0$ 时，方程有实根

$$\sqrt[3]{-\frac{q}{2} + \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}} + \sqrt[3]{-\frac{q}{2} - \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}}$$

因此我们推测式子

$$f(a) = \sqrt[3]{a + \frac{a+1}{3}\sqrt{\frac{8a-1}{3}}} + \sqrt[3]{a - \frac{a+1}{3}\sqrt{\frac{8a-1}{3}}}$$

是某个三次方程的根。

由于 $f(a) = 1$ 在 $a > 0.125$ 时恒成立，我们可以猜测存在该三次方程存在一个根 $z = 1$。

使用待定系数法，可得

$$(z-1)(z^2 + bz + c) = z^3 + (b-1)z^2 + (c-b)z - c$$

令 $b - 1 = 0$，$-c = q$，则有

$$(z-1)(z^2 + z - q) = z^3 + (-q-1)z + q$$

再观察求根公式与 $f(a)$ 的差异，我们可以令 $a=-\frac{q}{2}$，则有

$$\begin{cases} p=2a-1 \\ q=-2a \end{cases}$$

则我们可以知道，$f(a)$ 是三次方程

$$z^3+(2a-1)z-2a=0$$

的一个根，在 $a>0.125$ 时恒等于 1。

并且经过检验，该结论成立。

2 习题二

(a)

已知 $X\sim N(0,1)$，则有

$$P(X\geq\epsilon)=\int_{\epsilon}^{+\infty}\frac{1}{\sqrt{2\pi}}e^{-t^2/2}\,dt$$

$$=\int_{0}^{+\infty}\frac{1}{\sqrt{2\pi}}e^{-(x+\epsilon)^2/2}\,dx$$

$$\leq e^{-\epsilon^2/2}\int_{0}^{+\infty}\frac{1}{\sqrt{2\pi}}e^{-x^2/2}\,dx$$

$$=\frac{e^{-\epsilon^2/2}}{2}$$

(b)

已知 X 的概率密度函数为 $f(x)=\frac{1}{\sqrt{2\pi}}e^{-x^2/2}$ 求导得 $f'(x)=-xf(x)$ 则有

$$P(|X|\geq\epsilon)=2\int_{\epsilon}^{+\infty}f(x)\,dx$$

$$=2\int_{\epsilon}^{+\infty}xf'(x)\,dx$$

$$\leq 2\int_{\epsilon}^{+\infty}xf(\epsilon)\,dx$$

$$= \sqrt{\frac{2}{\pi}} \frac{e^{-\epsilon^2/2}}{\epsilon}$$

因此我们有

$$P(|X| \geq \epsilon) \leq \min \left\{ 1, \sqrt{\frac{2}{\pi}} \frac{e^{-\epsilon^2/2}}{\epsilon} \right\}$$

3 习题三

(a) FC 层与 BN 层的数学定义

- FC 层：全连接层（Fully Connected Layer）的数学含义是将输入数据通过一个线性变换（即矩阵乘法）和一个偏置项相加，得到输出。其数学表达式为：

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b}$$

其中，$\mathbf{W}$ 是权重矩阵，$\mathbf{x}$ 是输入向量，$\mathbf{b}$ 是偏置向量，$\mathbf{y}$ 是输出向量。
- BN 层：Batch Normalization 层的数学含义是对输入数据进行归一化处理，使其具有零均值和单位方差，然后再通过可学习的参数进行线性变换。具体来说，对于输入数据 $\mathbf{x}$，BN 层的计算过程如下：
首先计算该 batch 内所有数据的均值 μ 和方差 σ^2：

$$\mu = \frac{1}{m} \sum_{i=1}^{m} x_i$$

$$\sigma^2 = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu)^2$$

其中，m 是 batch 大小。然后对每个数据进行归一化：

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

其中，ϵ 是一个极小的常数，用于防止除零操作。最后，通过可学习的参数 γ 和 β 进行线性变换：

$$y_i = \gamma \hat{x}_i + \beta$$

这样，BN 层的输出既具有归一化的特性，又可以通过学习参数来恢复原始数据的分布。

(b) 证明两个 FC 层可以合并为一个

假设第一个 FC 层的权重矩阵为 $\mathbf{W}_1$，偏置向量为 $\mathbf{b}_1$，第二个 FC 层的权重矩阵为 $\mathbf{W}_2$，偏置向量为 $\mathbf{b}_2$。则前向计算过程为：
首先，第一个 FC 层的输出为：

$$\mathbf{y}_1 = \mathbf{W}_1\mathbf{x} + \mathbf{b}_1$$

然后，第二个 FC 层的输入是 $\mathbf{y}_1$，输出为：

$$\mathbf{y}_2 = \mathbf{W}_2\mathbf{y}_1 + \mathbf{b}_2$$

将 $\mathbf{y}_1$ 代入上式，得到：

$$\mathbf{y}_2 = \mathbf{W}_2(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2 = (\mathbf{W}_2\mathbf{W}_1)\mathbf{x} + (\mathbf{W}_2\mathbf{b}_1 + \mathbf{b}_2)$$

这表明，这两个 FC 层的组合可以等价于一个权重矩阵为 $\mathbf{W}_2\mathbf{W}_1$，偏置向量为 $\mathbf{W}_2\mathbf{b}_1 + \mathbf{b}_2$ 的单个 FC 层。

(c) 合并 FC 层的优势情况及优势

当两个连续的 FC 层之间没有非线性激活函数时，合并它们可以带来以下优势：
- 减少参数数量：合并后的 FC 层参数数量为 $(n_1 \times n_2 + n_2) + (n_2 \times n_3 + n_3) = n_1 \times n_3 + n_3$（假设输入维度为 n_1，中间维度为 n_2，输出维度为 n_3），而分开时参数数量为 $n_1 \times n_2 + n_2 + n_2 \times n_3 + n_3$，合并后参数数量更少，有助于减少模型的复杂度和存储需求。
- 提高计算效率：合并后的矩阵乘法操作可以一次性完成，减少了计算步骤，从而在推理阶段提高运行速度，尤其是在硬件资源有限的情况下，这种优化更为明显。

(d) 合并不总是带来优势的情况

当第一个 FC 层后面有非线性激活函数时，合并两个 FC 层可能带来劣势。例如，在第一个 FC 层后使用 ReLU 激活函数，此时合并后的 FC 层无法再简单地用矩阵乘法和偏置相加来表示，因为 ReLU 的非线性特性会破坏这种线性组合的性质。这种情况下，合并会导致模型无法正确表达原有的非线性映射关系，从而影响模型的性能和准确性。

(e) 证明 FC+BN 层可以替换为一个单独的 FC 层及优势情况

在推理阶段，BN 层的参数（均值、方差、γ、β）已经固定。此时，对于输入数据 $\mathbf{x}$，FC 层的输出为 $\mathbf{y}_{fc} = \mathbf{W}_{fc}\mathbf{x} + \mathbf{b}_{fc}$，BN 层的处理可以表示为：

$$\mathbf{y}_{bn} = \gamma \cdot \frac{\mathbf{y}_{fc} - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta$$

将 $\mathbf{y}_{fc}$ 代入上式，得到：

$$\mathbf{y}_{bn} = \gamma \cdot \frac{\mathbf{W}_{fc}\mathbf{x} + \mathbf{b}_{fc} - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta$$

这可以重新整理为：

$$\mathbf{y}_{bn} = \left(\frac{\gamma}{\sqrt{\sigma^2 + \epsilon}} \mathbf{W}_{fc} \right) \mathbf{x} + \left(\frac{\gamma(\mathbf{b}_{fc} - \mu)}{\sqrt{\sigma^2 + \epsilon}} + \beta \right)$$

这表明，FC+BN 层的组合可以等价于一个权重矩阵为 $\frac{\gamma}{\sqrt{\sigma^2+\epsilon}}\mathbf{W}_{fc}$，偏置向量为 $\frac{\gamma(\mathbf{b}_{fc}-\mu)}{\sqrt{\sigma^2+\epsilon}} + \beta$ 的单个 FC 层。

这种替换在以下情况下能够带来优势：

- 减少计算步骤：在推理阶段，将 FC 层和 BN 层合并为一个 FC 层，可以减少一层的计算开销，从而提高模型的运行效率，尤其是在需要快速推理的应用场景中，如实时图像处理等。
- 简化模型结构：合并后的模型结构更加简洁，便于部署和维护，尤其是在硬件资源受限的设备上，这种简化有助于降低实现复杂度。

(f)

可以将卷积层的权重和偏置与 BN 层的参数结合，计算出等价的卷积核和偏置项。然后通过实际测试模型的运行速度，观察这种替换是否对模型加速有作用，以及作用的大小。这种实践可以帮助我更深入地理解理论知识在实际中的应用效果，同时也为模型优化提供实践经验。

4 习题四

(a) 预处理方式

1. 最近邻插值: 将拍摄图像中的 $(4i+1, 4j+1)$ 的像素点 $f(4i+1, 4j+1)$ 像素值作为最近邻插值, 插值成为存储图像的 (i,j) 像素点的像素值.

2. 双线性插值: 将拍摄图像中的均值 $[f(4i+1,4j+1)+f(4i+2,4j+1)+f(4i+1,4j+2)+f(4i+2,4j+2)]/4$ 的像素值作为双线性插值, 插值成为存储图像的 (i,j) 像素点的像素值.
3. 均值插值: 将拍摄图像中的 4 x 4 像素点, 类似双线性插值一般取取均值, 插值成为存储图像的 (i,j) 像素点的像素值.

(b) 降低存储开销

将 2×2 块压缩为 1 个值（如取均值进行插值），存储开销降为原来的 25%（降低 75%）。

(c) 准确率计算

- 训练集准确率：99%（9900/10000）。
- 测试集准确率：50%（5000/10000）。

(d) Micro 与 Macro 区别

对于在一个二分类混淆矩阵上综合考察查准率、查全率以及准确率等指标的情况，我们有两种不同的方法。

第一种是 micro 方法，将自身类作为正类，其他所有类作为反类，先计算每一类正例和反例的样本数，其中

$$\text{micro-Acc} = \frac{\sum_{i\in\{A,B\}} TP_i}{\sum_{i\in\{A,B\}}(TP_i+FP_i+TN_i+FN_i)}$$

得到 micro 准确率为:

$$\text{micro-Acc} = \frac{9900+100}{9900+0+100} \times 100\% = 99\%$$

第二种是 macro 方法，先对各类别求出准确率，得到

$$\text{Acc}_i = \frac{TP_i}{TP_i+FP_i}$$

再取平均值计算出 macro 准确率:

$$\text{macro-Acc} = \frac{1}{N}\sum_{i=1}^{N}\text{Acc}_i = \frac{1}{2}\left(\frac{9900}{9900+0}\times 100\% + \frac{100}{100+0}\times 100\%\right) = 50\%$$

因此我们在 (c) 中采用的是 micro 方法。

或者我们通过 F1 来分析，其中 micro-F1 为先对混淆矩阵的对应元素进行平均，再进行计算：

$$\text{micro-P} = \frac{\bar{TP}}{\bar{TP} + \bar{FP}}$$

$$\text{micro-R} = \frac{\bar{TP}}{\bar{TP} + \bar{FN}}$$

$$\text{micro-F1} = \frac{2 \times \text{micro-P} \times \text{micro-R}}{\text{micro-P} + \text{micro-R}}$$

其中 macro-F1 为先在各个混淆矩阵上算出查准率和查全率，再算平均值：

$$\text{macro-P} = \frac{1}{N} \sum_{i=1}^{N} P_i$$

$$\text{macro-R} = \frac{1}{N} \sum_{i=1}^{N} R_i$$

$$\text{macro-F1} = \frac{2 \times \text{macro-P} \times \text{macro-R}}{\text{macro-P} + \text{macro-R}}$$

经过计算我们可以算出，在多分类问题下，准确率 (accuracy)、查准率 (precision)、查全率 (recall) 以及 F1 的值都是相同的。这也可以说明我们在 (c) 中采用的是 micro 方法。

(e) 长尾问题解决方法

我们应该采用 macro 方法来评估准确率。因为 macro-F1 是计算每一类的 F1 score，然后再求算术平均，如果模型在小样本上表现不好，小样本的 F1 会极大程度上拉低 macro-F1，这样就能对长尾识别问题中类别不平衡问题进行一定的改善。

并且我们知道，按照 macro 方法，(c) 中训练集的准确率结果为

$$\text{macro-Acc} = \frac{1}{2}\left(\frac{9900}{9900} \times 100\% + \frac{100}{100} \times 100\%\right) = 50\%$$

可以看出是通过对各个类别的准确率都赋予了相同的权重，避免了类别不平衡导致的问题。

为了长尾识别问题中的类别不平衡问题，我们可以采用以下方法：

1. **重采样**：对样本少的类别进行有放回的随机采样，并加入训练集中。例如，此时类别 A 有 9900 个样本，类别 B 有 100 个样本，我们就可以随机在类别 B 上重采样 9800 个样本，来平衡不同类别的样例。

2. **欠采样**：在样本多的类别中取出与样本少的类别数目相同的样本用于训练。

3. **代价敏感矩阵**：给不同类别的样本赋予不同的权重，以增加样本少的类别对结果的影响。

5 习题五

(a)

- $z_1 = (0, -2)$ 的最近邻分类结果为 $x_3 = 0, -1$ 对应的类别 A。
- $z_2 = (8, 2)$ 的最近邻分类结果为 $x_7 = (8, 1)$ 对应的类别 A。

(b)

- $z_1 = (0, -2)$ 的 k-近邻分类结果为 k-近邻 x_1, x_3, x_4 投票得到的类别 A。
- $z_2 = (8, 2)$ 的 k-近邻分类结果为 k-近邻 x_6, x_7, x_8 投票得到的类别 B。

(c)

z_1 附近都是类别 A 的样本，因此仍然是分类为类别 A 不变，但是 z_2 附近只是偶然有一个类别 A 的样本 x_7，但是还有更多的类别为 B 的临近样本 x_6, x_8，因此被分类成类别 B。

(d)

x_7 可能属于类别 B，可能是在采集数据的时候数据不小心打错了标签。因此，k-NN 相比于 1-NN 的一个很大的优势就是容错率高，不容易被偶然的错误样本影响到分类结果。