

图像在现代社会中扮演着重要的角色,它们用于电视、电影、广告等多个领域,然而由于图像数据量庞大,传输和存储成本较高。为了解决这个问题,图像压缩技术应运而生。

### 5.1 数字图像压缩基本概念

数据压缩是以较少的数据量表示信源。数据压缩的目的在于节省存储空间、传输时间、信号频带或发送能量等,数据压缩系统组成如图 5.1 所示。

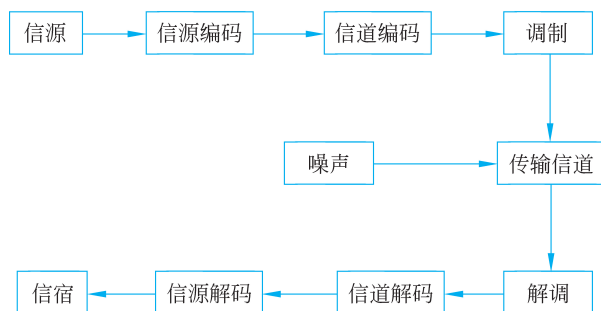


图 5.1 数据压缩系统组成

图像是信源的一种形式,且大部分图像存在冗余信息,为了减少图像占用存储空间的容量,降低传输带宽,提高传输速率,有必要对图像进行压缩编码研究。

图像之所以能压缩,是因为一般图像中存在着以下数据冗余因素。

(1) 编码冗余。一般情况下,灰度图像每像素占用 8 位二进制,彩色图像每像素占用 24 位二进制,都为等长编码,等长编码就会存在冗余,因此,可以利用非等长编码对图像进行压缩。

(2) 图像中各像素之间、行和帧之间存在着较强的相关性形成的冗余。图 5.2 中,比如房顶、塔或者草地内部各相邻像素间颜色很相似,因此它们之间存在相关性,有冗余信息。

图 5.3 是视频中相邻的两帧图像,如果汽车速度和方向已知,则可以通过前一帧图像推出后面一帧图像,因此相邻帧间存在相关性,即存在冗余信息。

(3) 视觉特性引起的冗余。由于人眼的视觉特性所限,人眼不能完全感受到图像画面的所有细小的变化。人类视觉系统一



图 5.2 图像中各像素间相关性示例

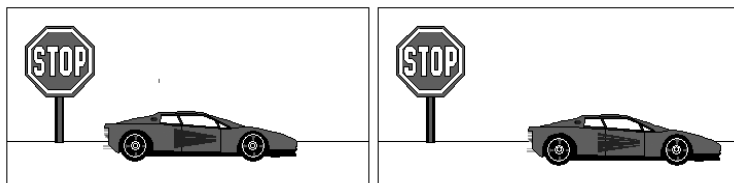


图 5.3 视频中帧间相关性示例

般分辨率大约是  $2^6$  灰度等级,而图像的量化一般是  $2^8$  灰度等级,这样就会产生视觉冗余。

图像压缩后,虽然丢了一些信息,但从人眼的视觉上并未感受到其中的变化,而仍认为图像具有良好的质量。

图像压缩是通过减少图像数据量来减小图像文件的大小。常用的压缩方法包括无损压缩和有损压缩。

无损压缩方法通过消除图像中的冗余信息来实现文件大小的减小,同时保持图像质量不变。无损压缩是指压缩后的数据可以完全还原为原始图像数据,不会引入任何失真或变化。常见的无损压缩算法有行程编码、Huffman 编码、Shannon 编码、算术编码、轮廓编码等,这些算法通常针对图像中的冗余数据进行编码。

有损压缩方法则在一定程度上牺牲图像质量,以达到更高的压缩比。有损压缩则是在保证一定程度的视觉质量下,通过舍弃或近似原始图像数据来减小存储或传输的数据量。常见的有损压缩算法有 JPEG、JPEG2000、预测编码、变换编码、混合编码等。JPEG、JPEG2000 算法通过离散余弦变换(DCT)、小波变换或颜色量化等方法,将图像数据转换为频域或颜色空间的系数,并通过量化、编码和压缩等步骤来减小数据量。

图像压缩中,常用的图像压缩技术指标有熵、平均码长、编码效率、压缩比等。

### 5.1.1 熵

熵代表信源所含的平均信息量。

若信源编码的熵大于信源的实际熵,则信源中的数据一定存在冗余度,冗余数据的去除不会减少信息量。信息量与数据量的关系如式(5.1)所示。

$$I = D - a \quad (5.1)$$

式中,  $I$ 、 $D$ 、 $a$  分别表示信息量、数据量和冗余量。

设数字图像像素灰度级集合为  $(w_1, w_2, \dots, w_k, \dots, w_M)$ , 其对应的概率分别为  $(p_1, p_2, \dots, p_k, \dots, p_M)$ , 则数字图像的熵  $H$  如式(5.2)所示。

$$H = - \sum_{k=1}^M p_k \log_2 p_k \quad (5.2)$$

### 5.1.2 平均码长

设  $\beta_k$  为数字图像第  $k$  个码字  $C_k$  的长度(二进制码的位数),其相应出现的概率为  $p_k$ , 则这幅图像的平均码长  $R$  如式(5.3)所示。

$$R = \sum_{k=1}^M \beta_k p_k \quad (5.3)$$

**例 5.1** 已知  $C_0=00, C_1=01, C_2=1, C_3=101, p_0=0.25, p_1=0.25, p_2=0.3, p_3=0.2$ , 其平均码长是多少?

**【解】**

$$R = \sum_{k=0}^3 C_k p_k = 2 \times 0.25 + 2 \times 0.25 + 1 \times 0.3 + 3 \times 0.2 = 1.9$$

因此平均码长为 1.9。

### 5.1.3 编码效率

编码效率如式(5.4)所示。

$$\eta = \frac{H}{R} (\%) \quad (5.4)$$

式中,  $H$  为信源熵,  $R$  为平均码字长度。

### 5.1.4 压缩比

为了表达对图像压缩的程度, 引入压缩比概念, 如式(5.5)所示。

$$C = \frac{b_1}{b_2} \quad (5.5)$$

式中,  $b_1$  表示压缩前图像每像素的平均比特数,  $b_2$  表示压缩后图像每像素的平均比特数。

### 5.1.5 统计编码

统计编码根据信源的概率分布特性, 分配具有唯一可译性的可变长码字, 降低平均码长, 以提高信息的传输速率, 节省存储空间。

统计编码的基本原理是在信号概率分布情况已知的基础上, 概率大的信号对应的码字短, 概率小的信号对应的码字长, 这样就降低了平均码长。统计编码遵循变长最佳编码定理, 则可获得最佳编码。

**变长最佳编码定理:** 在变长编码中, 对出现概率大的信息符号赋予短码字, 而对于出现概率小的信息符号赋予长码字。如果码字长度严格按照所对应符号出现概率大小逆序排列, 则编码结果平均码长一定小于任何其他排列方式。

下面主要讲解 Huffman 编码、Shannon 编码和算术编码方法。

## 5.2 Huffman 压缩编码

Huffman 编码是根据可变长度最佳编码定理, 应用 Huffman 算法而产生的一种编码方法。它的平均码长在具有相同输入概率集合的前提下, 比其他任何一种唯一可译码都小, 因此, 也常称其为紧凑码。

Huffman 编码步骤如下。

(1) 先将输入灰度级按出现的概率由大到小顺序排列(对概率相同的灰度级可以任意颠倒排列位置)。

(2) 将最小两个概率相加,形成一个新的概率集合。

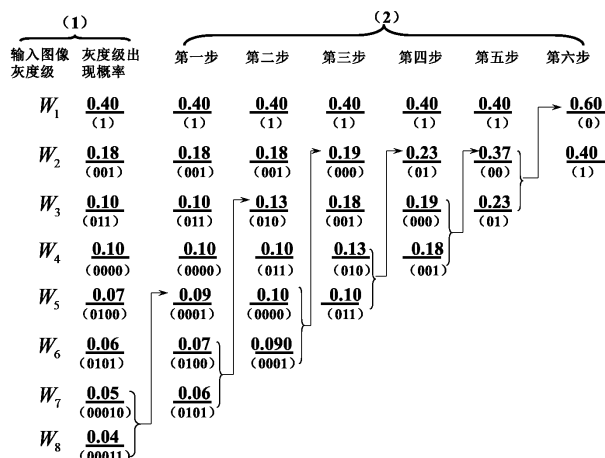
再按第(1)步方法重排(此时概率集合中概率个数已减少一个)。如此重复进行直到只有两个概率为止。

(3) 分配码字。码字分配从最后一步开始反向进行,对最后两个概率一个赋予“0”码,一个赋予“1”码。如概率 0.6 赋予“0”码,0.4 赋予“1”码(也可以将 0.6 赋予“1”码,0.4 赋予“0”码),如此反向进行到开始的概率排列。

在此过程中,若概率不变,仍用原码字。

**例 5.2** 设一幅灰度级为 8(分别用  $W_1$ 、 $W_2$ 、 $W_3$ 、 $W_4$ 、 $W_5$ 、 $W_6$ 、 $W_7$ 、 $W_8$  表示)的图像中,各灰度级所对应的概率分别为 0.40、0.18、0.10、0.10、0.07、0.06、0.05、0.04。现对其进行 Huffman 编码,并计算编码效率。

**【解】** 根据上述编码步骤,对该例题进行压缩编码,编码过程如下:



上述编码过程中,第六步中概率 0.4 到第五步中仍用“1”码。

若概率分裂为两个,其码字前几位码元仍用原来的。码字的最后一位码元一个赋予“0”码元,另一个赋予“1”码元。如第六步中概率 0.6 到第五步中裂为 0.37 和 0.23,则所得码字分别为“00”和“01”。

信源熵:

$$\begin{aligned}
 H &= - \sum_{i=1}^8 P_i \log_2 P_i \\
 &= - \left( 0.4 \log_2 0.4 + 0.18 \log_2 0.18 + 2 \times 0.1 \log_2 0.1 \right. \\
 &\quad \left. + 0.07 \log_2 0.07 + 0.06 \log_2 0.06 + 0.05 \log_2 0.05 \right. \\
 &\quad \left. + 0.04 \log_2 0.04 \right) \\
 &= 2.55
 \end{aligned}$$

平均码长:

$$\begin{aligned}
 R &= \sum_{i=1}^8 \beta_i P_i \\
 &= 0.40 \times 1 + 0.18 \times 3 + 0.10 \times 3 + 0.10 \times 4 + 0.07 \times 4 + \\
 &\quad 0.06 \times 4 + 0.05 \times 5 + 0.04 \times 5 \\
 &= 2.61
 \end{aligned}$$

编码效率:

$$\eta = \frac{H}{R} = \frac{2.55}{2.61} = 97.8\%$$

由此可见, Huffman 编码的编码效率是相当高的, 其冗余度只有 2.2%。

如果采用等长编码, 由于有 8 个灰度级, 则每个灰度级至少需要 3 比特来表示, 对于例 5.2 中的图像而言, 其编码的平均码长为 3, 编码效率为 85%。

对不同概率分布的信源, Huffman 编码的编码效率有所差别。

根据信息论中信源编码理论, 对于二进制编码, 当信源概率为 2 的负幂次方时, Huffman 编码的编码效率可达 100%, 其平均码长也很短; 而当信源概率为均匀分布时, 其编码效果明显降低。Huffman 编码在不同概率分布下编码效果对比如表 5.1 所示。

表 5.1 Huffman 编码在不同概率分布下编码效果对比

| 信源符号 | 概率分布为 2 的负幂次方 |              |              | 概率分布为均匀分布 |            |              |
|------|---------------|--------------|--------------|-----------|------------|--------------|
|      | 出现概率          | Huffman 编码   | 平均码长         | 出现概率      | Huffman 编码 | 平均码长         |
| S0   | $2^{-1}$      | 1            | 1            | 0.125     | 111        | 3            |
| S1   | $2^{-2}$      | 01           | 2            | 0.125     | 110        | 3            |
| S2   | $2^{-3}$      | 001          | 3            | 0.125     | 101        | 3            |
| S3   | $2^{-4}$      | 0001         | 4            | 0.125     | 100        | 3            |
| S4   | $2^{-5}$      | 00001        | 5            | 0.125     | 011        | 3            |
| S5   | $2^{-6}$      | 000001       | 6            | 0.125     | 010        | 3            |
| S6   | $2^{-7}$      | 0000001      | 7            | 0.125     | 001        | 3            |
| S7   | $2^{-1}$      | 0000001      | 7            | 0.125     | 000        | 3            |
| 编码效率 | $H=1.984375$  | $R=1.984375$ | $\eta=100\%$ | $H=3$     | $R=3$      | $\eta=100\%$ |

在表 5.1 中, 显然, 第二种情况的概率分布也服从 2 的负幂次方, 故其编码效率  $\eta$  也可以达到 100%, 但由于它服从均匀分布, 其熵最大, 平均码长很大, 因此从其他指标 (如压缩比  $r$ ) 看, 其编码效果欠佳。也就是说, 在信源概率接近均匀分布时, 一般不使用 Huffman 编码。

通过上述分析, Huffman 编码性能如下。

(1) 实现 Huffman 编码的基础是统计信源数据中各信号的概率分布。

(2) Huffman 编码在无失真的编码方法中效率优于其他编码方法, 是一种最佳变长码, 其平均码长接近熵值。

(3) 当信源数据成分复杂时,庞大的信源集致使 Huffman 码表较大,码表生成的计算量增加,编译码速度相应变慢。

(4) 不等长编码致使硬件译码电路实现困难,上述原因致使 Huffman 编码的实际应用受到限制。

### 5.3 Shannon 压缩编码

Shannon 提出了将信源符号依其概率降序排列,用符号序列累积概率的二进制表示作为对信源的唯一可译编码,即 Shannon 编码。

Shannon 编码应用于图像编码的步骤如下。

(1) 将输入灰度级按其出现的概率从大到小排列。

(2) 按照式(5.6)计算出各概率对应的码字长度  $t_i$ 。

$$-\log_2 p_i \leq t_i < -\log_2 p_i + 1 \quad (5.6)$$

(3) 计算各概率对应的累加概率  $a_i$ ,如式(5.7)所示。

$$\begin{cases} a_1 = 0 \\ a_2 = p_1 \\ a_3 = p_2 + a_2 = p_2 + p_1 \\ a_4 = p_3 + a_3 = p_3 + p_2 + p_1 \\ \vdots \\ a_i = p_{i-1} + p_{i-2} + \cdots + p_1 \end{cases} \quad (5.7)$$

(4) 把各个累加概率由十进制数转换成二进制数,转换位数即步骤(2)中  $t_i$ 。

**例 5.3** 图像概率分布同例 5.2,对其进行 Shannon 编码。

**【解】**

| 输入图像<br>灰度级 | (1)<br>灰度级出<br>现概率 | (2)<br>计算 $t_i$ | (3)<br>计算 $a_i$ | (4)<br>码字 |
|-------------|--------------------|-----------------|-----------------|-----------|
| $W_1$       | 0.40               | 2               | 0               | 00        |
| $W_2$       | 0.18               | 3               | 0.40            | 011       |
| $W_3$       | 0.10               | 4               | 0.58            | 1001      |
| $W_4$       | 0.10               | 4               | 0.68            | 1010      |
| $W_5$       | 0.07               | 4               | 0.78            | 1100      |
| $W_6$       | 0.06               | 5               | 0.85            | 11011     |
| $W_7$       | 0.05               | 5               | 0.91            | 11101     |
| $W_8$       | 0.04               | 5               | 0.96            | 11110     |

### 5.4 算术编码

在信源各符号概率接近的条件下,算术编码是一种优于 Huffman 编码的方法。有关实验数据表明,即使在未知信源概率分布的情况下,算术编码一般也优于 Huffman 编码。

算术编码基本思想如下。

(1) 将要压缩的数据  $\mathbf{X}$  映射到  $[0, 1]$  实数区间中的某一区段上的实数, 该实数的二进制展开式即为原符号串的压缩编码结果。

(2) 算术编码通过对当前的概率区间进行迭代分割来确定实数。

(3) 算术编码是根据信源符号选区间, 即在  $[0, 1]$  区间里确定一个最后的小区间。

(4) 算术编码是一种从整个符号序列出发, 采用递推形式连续编码的方法。它将一个符号序列映射为一个实数。算术编码中, 单个源符号和码字间的一一对应关系并不存在。

(5) 算术编码是具体构造出的用小数表示信息的方法, 因为小数随位数的增加, 它的精度也随之提高, 从信息的角度来说, 它所含有的信息量也随之增加。

(6) 随着符号序列中的符号数量增加, 用来代表它的区间减小而用来表达区间所需的信息单位(如比特)的数量变大。

下面结合一个实例来阐述算术编码方法。

**例 5.4** 已知信源  $\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 1/4 & 3/4 \end{bmatrix}$ , 试对 1011 进行算术编码。

**【解】**

(1) 二进制信源符号只有两个, 即“0”和“1”。

设置小概率:  $Q_c = 1/4$ 。

大概率:  $P_c = 1 - Q_c = 3/4$ 。

(2) 设  $B$  为子区的左端起始位置,  $L$  为子区的长度(等效于符号概率)。

符号“0”的子区为  $[0, 1/4]$ ; 左端  $B = 0$ , 长  $L = 1/4$ 。

符号“1”的子区为  $[1/4, 1]$ ; 左端  $B = 1/4$ , 长  $L = 3/4$ 。

(3) 在编码运算过程中, 随着消息符号的出现, 子区按下列规则缩小。

规则 A: 新子区左端 = 前子区左端 + 当前符号子区左端  $\times$  前子区长度。

规则 B: 新子区长度 = 前子区长度  $\times$  当前符号子区的长度。

(4) 初始子区为  $[0, 1]$ , 即  $0 \leq x \leq 1$ , 编码算法如下:

| 步序  | 符号 | 子区左端                                    | 子区长度                            |
|-----|----|-----------------------------------------|---------------------------------|
| (a) | 1  | $0 + 1/4 \times 1 = 1/4$                | $1 \times 3/4 = 3/4$            |
| (b) | 0  | $1/4 + 0 \times 3/4 = 1/4$              | $3/4 \times 1/4 = 3/16$         |
| (c) | 1  | $1/4 + 1/4 \times 3/16$<br>$= 19/64$    | $3/16 \times 3/4$<br>$= 9/64$   |
| (d) | 1  | $19/64 + 1/4 \times 9/64$<br>$= 85/256$ | $9/64 \times 3/4$<br>$= 27/256$ |

最后的子区左端(起始位置):

$$B = (85/256)_d = (0.01010101)_b$$

最后的子区长度:

$$L = (27/256)_d = (0.00011011)_b$$

最后的子区右端(子区间尾):

$$85/256 + 27/256 = (7/16)d = (0.0111)b$$

编码结果: 在子区头尾之间取值, 其值为 0.011, 可编码为 011, 原来 4 个符号 1011 被压缩为三个符号 011。

考虑到算术编码中任何数据序列的编码都含有“0.”, 所以在编码时, 可以不考虑“0.”。

**例 5.5** 设输入两种符号 A 和 B, 它们出现的概率分别是  $3/4$  和  $1/4$ , 需要编码的信息为 AABA, 试对其进行算术编码。

**【解】**

算术编码过程如下。

从  $[0, 1)$  开始, 符号依次迭代分解区间。

(1) 第一个字符 A, 取  $[0, 1)$  区间的前  $3/4$ , 即区间  $[0, 3/4)$ 。

(2) 第二个字符 A, 取  $[0, 3/4)$  区间的前  $3/4$ , 即区间  $[0, 9/16)$ 。

(3) 第三个字符 B, 取  $[0, 9/16)$  区间的后  $1/4$ , 即区间  $[27/64, 9/16)$ 。

(4) 第四个字符 A, 取  $[27/64, 9/16)$  区间的前  $3/4$ , 即区间  $[27/64, 135/256)$ , 其二进制表示为  $[0.011011, 0.10000111)$ 。

(5) 所余区间中位数最少的小数就是所要的编码输出, 在上述最后区间的二进制表示中, 最少位数的小数是 0.1, 所以 AABA 的编码输出仅需 1 比特编码。

思考题:

设一待编码的数据序列(即信源)为“dacab”, 信源中各符号出现的概率依次为  $P(a) = 0.4$ ,  $P(b) = 0.2$ ,  $P(c) = 0.2$ ,  $P(d) = 0.2$ 。请思考利用上述哪种算术编码方法更好。

## 5.5 预测编码与变换编码

### 1. 预测编码

预测编码的基本思想: 在某种模型的指导下, 根据过去的样本序列推测当前的信号样本值, 然后用实际值与预测值之间的误差值进行编码。

如果模型与实际情况符合得比较好且信号序列的相关性较强, 则误差信号的幅度将远远小于样本信号。

### 2. 变换编码

通过数学变换可以改变信号能量的分布, 从而压缩信息量。比如, 第 3 章中的傅里叶变换、离散沃尔什变换、离散哈达玛变换等。

图像压缩技术在多个领域都有着广泛的应用, 包括图像传输、存储和展示等, 然而当前的图像压缩方法仍然存在一些问题, 比如压缩质量和压缩速度之间的平衡, 以及特定类型图像的压缩效果等。未来的研究方向包括深度学习在图像压缩中的应用、更高效的压缩算法以及对不同类型图像进行自适应压缩等。



## 习 题 五

1. 阐述数据压缩的必要性。
2. 大部分视频压缩方法是有损压缩还是无损压缩? 为什么?
3. Huffman 编码有何优缺点?
4. Huffman 编码是最佳编码, 为什么还要研究算术编码等其他熵编码方法?
5. 设某幅图像共有 8 个灰度级, 各灰度级出现概率分别为:

$$P_1=0.50 \quad P_2=0.01 \quad P_3=0.03 \quad P_4=0.05$$

$$P_5=0.05 \quad P_6=0.07 \quad P_7=0.19 \quad P_8=0.10$$

试对此图像进行 Huffman 编码和 Shannon 编码, 并计算比较两种编码方法的效率。

6. 设有一幅图像, 其尺寸为  $8 \times 8$ , 其灰度级分布如下:

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 4 | 4 | 4 | 4 | 4 | 4 | 4 | 0 |
| 4 | 5 | 5 | 5 | 5 | 5 | 4 | 0 |
| 4 | 5 | 6 | 6 | 6 | 5 | 4 | 0 |
| 4 | 5 | 6 | 7 | 6 | 5 | 4 | 0 |
| 4 | 5 | 6 | 7 | 6 | 5 | 4 | 0 |
| 4 | 5 | 5 | 5 | 5 | 5 | 4 | 0 |
| 4 | 4 | 4 | 4 | 4 | 4 | 4 | 0 |
| 4 | 4 | 4 | 4 | 4 | 4 | 4 | 0 |

试对该图像进行 Huffman 编码, 并计算编码效率。

7. 已知信源  $\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 1/4 & 3/4 \end{bmatrix}$ , 试对 1001 和 10111 进行算术编码。

在图像的研究和应用中,人们往往仅对图像中的某些部分感兴趣,这些部分一般称为目标或前景;为了辨识和分析目标,需要将有关区域分离提取出来,在此基础上对目标区域进行进一步分析和处理,如进行特征提取和测量;图像分割是指把图像分成各具特性的区域,并提取出感兴趣目标的技术过程。

图像分割工程应用案例:回收不同颜色塑料瓶,对塑料瓶图像的分割。该图像分割实例是基于绿色可持续发展理念,旨在实现能源的有效回收和利用,契合国家大力倡导的可持续发展和能源有效利用的社会背景。

6.1 图像分割概述

图像分割算法基于颜色值的相似性或不连续性。根据制定的准则,将图像分割为相似的区域,如阈值处理、区域生长、区域分裂与合并等;不连续性是基于亮度的不连续变化分割图像,如图像的边缘等。

图像分割分类如图 6.1 所示。

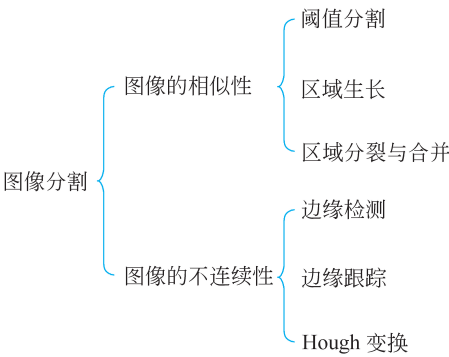


图 6.1 图像分割分类

图 6.2 为一幅汽车场景图像,图像分割将汽车作为目标从背景图像中分割,为后续分析奠定基础。

图像分割结果中同一个子区域内的像素应当是连通的,同一个子区域内的任意两个像素在该子区域内应该是互相连通的。