Lecture 10-2 Lossless and Lossy Compression

Yuyao Zhang, Xiran Cai PhD

zhangyy8@shanghaitech.edu.cn caixr@shanghaitech.edu.cn

SIST Building 2 302-F/302-C

Course piazza link： piazza.com/shanghaitech.edu.cn/spring2021/cs270spring2021

上海科技大学 ShanghaiTech University

Outline

- Lossless compression is important for
  - Medical diagnosis
  - Legal reasons
  - Satellite imagery (expensive)
- Lossy compression is common for image and video
  - JPEG
  - MPEG

Evaluation

– To measure performance, we can use the compression ratio:

$$R = \frac{\#\, Bits\ before}{\#\, Bits\ after}$$

– For real images, lossless compression ratios range in the 2-10. The format .BMP (Bitmap) refers to no compression at all. Format .gif is highly compressed.

Basic idea for lossless image compression

- Recall the image histogram, thinking about it as a PMF (Probability Mass Function):

$$P_i = P(I(x,y) = i) = \frac{\# \, I(x,y) = i}{\# \, Pixels}$$

- The average bits per pixel:

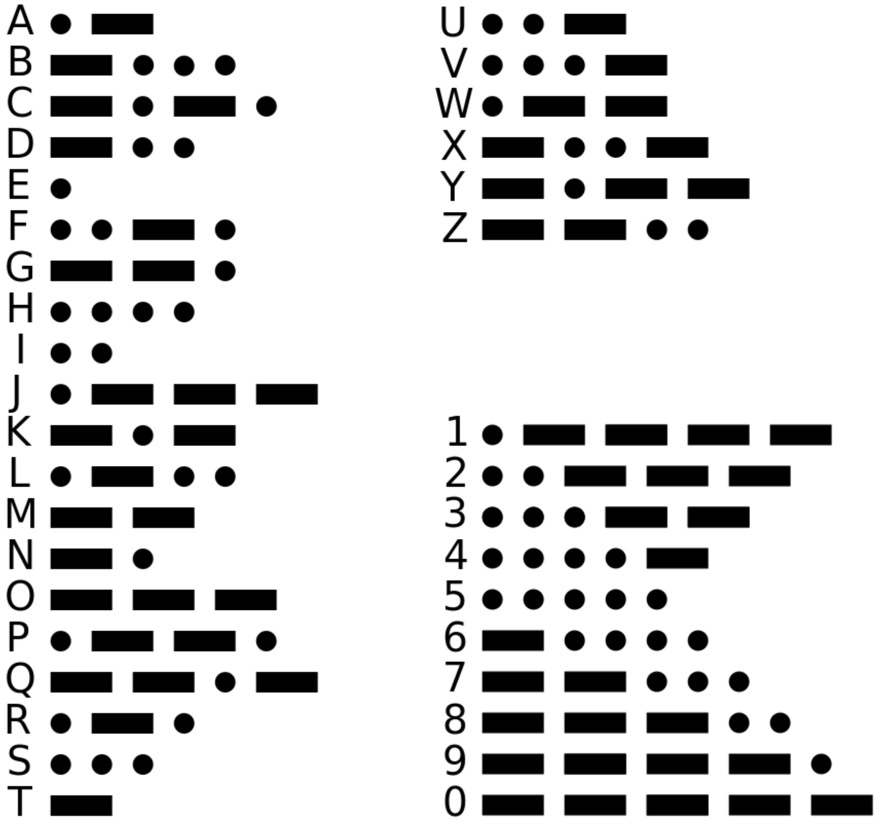
$$E(b) = \sum_{i=1}^{N} P_i \cdot b_i$$

- Uniform coding, $b_i = \mathrm{b}$ (e.g. 8 bits).

Morse code

- Idea： use fewer bits to describe more frequency symbols.

  Example: Morse code.



- E use the shortest symbol.

- For image compression we should

use the same idea.

A coding example

Gray level	Probability p_i	Uniform coding	Variable-level coding
0	0.19	000	00
1	0.25	001	11
2	0.21	010	01
3	0.16	011	101
4	0.08	100	1001
5	0.06	101	10001
6	0.03	110	100001
7	0.02	111	1000001
Avg length		3	2.7

- Consider an 8-level image as in the table on the left.
- Compression ratio = 3/2.7=1.11.
- 11% better than uniform coding.

Measuring Image Information （信息量）

- **Information Unit:**

$$I(E) = \log\frac{1}{P(E)} = -\log P(E)$$

Where $P(E)$ is the probability of a random event E.

- **Entropy (熵)**

$$H = -\sum_{j=1}^{J} P(a_j)\log P(a_j)$$

Calculate from Histogram

$$\tilde{H} = -\sum_{k=0}^{L-1} p_r(r_k)\log_2 p_r(r_k)$$

A coding example

Gray level	Probability p_i	Uniform coding	Variable-level coding
0	0.19	000	00
1	0.25	001	11
2	0.21	010	01
3	0.16	011	101
4	0.08	100	1001
5	0.06	101	10001
6	0.03	110	100001
7	0.02	111	1000001
Avg length		3	2.7

- Consider an 8-level image as in the table on the left.
- Compression ratio = 3/2.7=1.11.
- 11% better than uniform coding.
- The entropy is the lowest number of average bits per symbol that can be used to code the distribution.

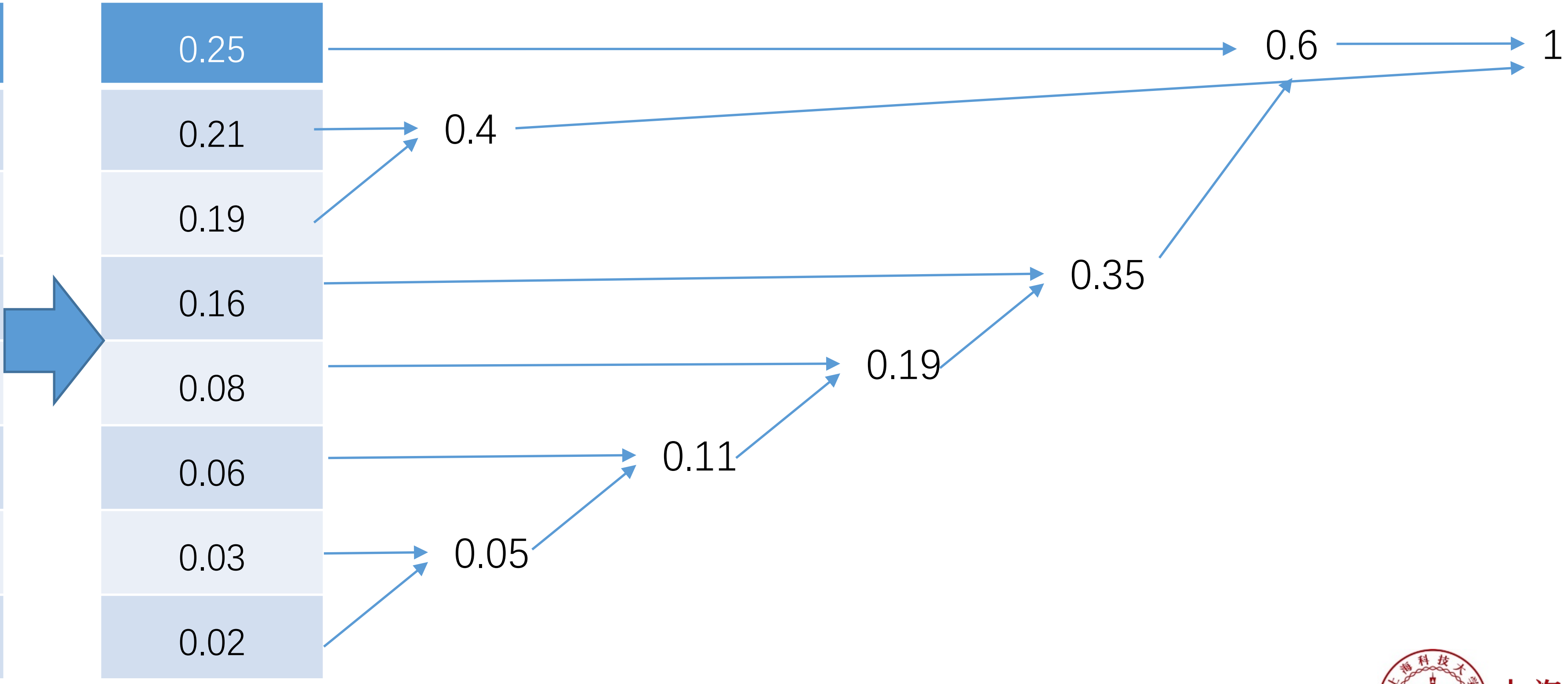
$$H = -\sum_{j=1}^{J} P(a_j)\log P(a_j)$$

上海科技大学 ShanghaiTech University

Huffman coding example

1. **Arrange symbol in decreasing p_i, think of these as node/leaves of a tree.**
2. **Merge the 2 nodes with lowest probability.**
3. **Assign 0/1 to the bottom branch.**
4. **Read code from root to leaf.**

0	0.19
1	0.25
2	0.21
3	0.16
4	0.08
5	0.06
6	0.03
7	0.02



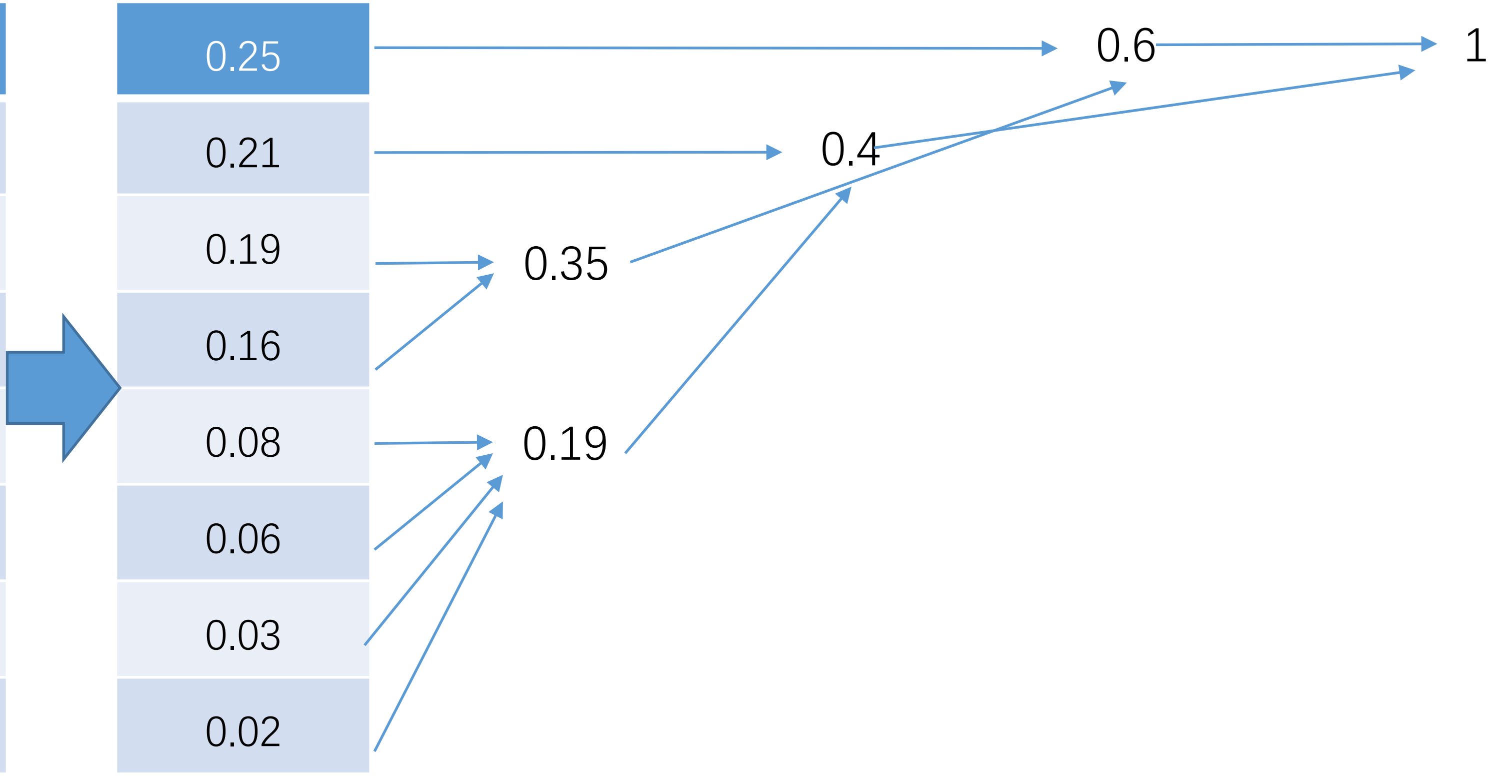
Huffman coding

- Huffman coding is prefix coding.
- Huffman coding is not unique.
- When uniform distributed, the compression ratio is quite limited.
- When probability is close to reciprocal of 2^k, the performance can approximate the entropy limitation.
- For decoding need to know the probability distribution of the input symbol set.
- No error protection.

Truncated Huffman coding

- To avoid extremely long code words for infrequent symbols.
- Huffman code the most probable K symbols in the source.
- Replace the result with a prefix + A fixed length codes.

0	0.19
1	0.25
2	0.21
3	0.16
4	0.08
5	0.06
6	0.03
7	0.02



Lempel-ziv-welch (LZW) coding

- Basis for Gif, Tiff, PNG, ZIP, PDF.
- Idea: Parse source string sequentially into "phrases": strings that haven't appeared before.
- Code each phrase by giving location of the prefix plus last bit.
- No knowledge of PMF for symbols is needed.
- Exploits how symbols occur together.
- Really pays off for very long strings.

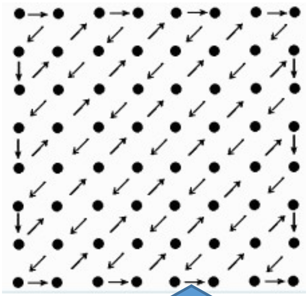
Lempel-ziv-welch (LZW) coding

- AABBBAABAAABBBABBABBABBABABBAAABBBAAABBBAAABBBB

Position	1	2	3	4	5	6	7	8	9
Sequence									
Numerical presentation									

The most widely-used lossy compression: JPEG

Input image → Break into nxn blocks e.g. 8x8,16x16 → Forwards nxn transform (each block) → Quantizer (lossy) → Symbol Encoder e.g. huffman → Compressed image

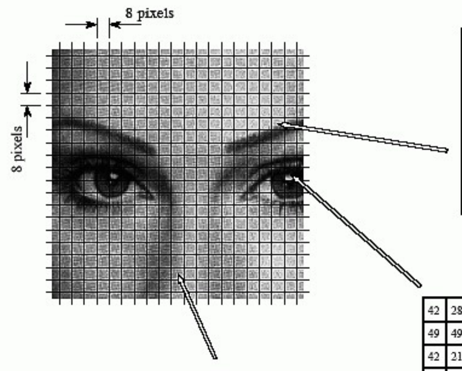


a. Low compression

1	1	1	1	1	2	2	4
1	1	1	1	1	2	2	4
1	1	1	1	2	2	2	4
1	1	1	1	2	2	4	8
1	1	2	2	2	2	4	8
2	2	2	2	2	4	8	8
2	2	2	4	4	8	8	16
4	4	4	4	8	8	16	16

b. High compression

1	2	4	8	16	32	64	128
2	4	4	8	16	32	64	128
4	4	8	16	32	64	128	128
8	8	16	32	64	128	128	256
16	16	32	64	128	128	256	256
32	32	64	128	128	256	256	256
64	64	128	128	256	256	256	256
128	128	128	256	256	256	256	256



231	224	224	217	217	203	189	196
210	217	203	189	203	224	217	224
196	217	210	224	203	203	196	189
210	203	196	203	182	203	182	189
203	224	203	217	196	175	154	140
182	189	168	161	154	126	119	112
175	154	126	105	140	105	119	84
154	98	105	98	105	63	112	84

42	28	35	28	42	49	35	42
49	49	35	28	35	35	35	42
42	21	21	28	42	35	42	28
21	35	35	42	42	28	28	14
56	70	77	84	91	28	28	21
70	126	133	147	161	91	35	14
126	203	189	182	175	175	35	21
49	189	245	210	182	84	21	35

154	154	175	182	189	168	217	175
154	147	168	154	168	168	196	175
175	154	203	175	189	182	196	182
175	168	168	168	140	175	168	203
133	168	154	196	175	189	203	154
168	161	161	168	154	154	189	189
147	161	175	182	189	175	217	175
175	175	203	175	189	175	175	182

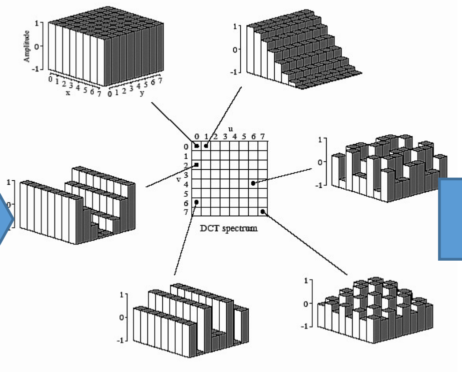


FIGURE 27-10
The DCT basis functions. The DCT spectrum consists of an 8×8 array, with each element in the array being an amplitude of one of the 64 basis functions. Six of these basis functions are shown here, referenced to where the corresponding amplitude resides.

$$
\begin{bmatrix}
34 & 34 & 34 & 33 & 34 & 28 & 35 & 32 \\
34 & 34 & 34 & 33 & 34 & 28 & 35 & 32 \\
34 & 34 & 34 & 33 & 34 & 28 & 35 & 32 \\
34 & 34 & 34 & 33 & 34 & 28 & 35 & 32 \\
34 & 34 & 34 & 33 & 34 & 28 & 35 & 32 \\
36 & 36 & 29 & 27 & 33 & 31 & 30 & 31 \\
32 & 32 & 35 & 30 & 32 & 33 & 31 & 27 \\
30 & 30 & 27 & 28 & 30 & 30 & 28 & 29
\end{bmatrix}
\overset{\text{DCT}}{\Longrightarrow}
\begin{bmatrix}
257.1 & 6.4 & 2.5 & -0.3 & 0.4 & 0.1 & -6.0 & 6.9 \\
8.4 & 0.0 & 0.5 & -5.0 & 1.9 & 3.4 & -4.2 & 3.3 \\
-5.3 & -1.0 & -1.4 & 1.3 & -0.7 & -0.5 & 2.1 & -1.7 \\
2.4 & 1.7 & 1.5 & 1.5 & -0.6 & -1.5 & 0.2 & 0.4 \\
-1.1 & -1.6 & -0.2 & -1.8 & 1.6 & 1.2 & -1.4 & -0.1 \\
1.4 & 0.9 & -1.9 & -0.1 & -2.0 & 0.9 & 1.5 & 0.7 \\
-2.0 & -0.1 & 3.1 & 2.0 & 1.8 & -2.7 & -0.9 & -1.3 \\
1.5 & -0.2 & -2.3 & -1.9 & -1.0 & 2.3 & 0.3 & 1.1
\end{bmatrix}
$$

How JPEG really works

- Specify a normalized matrix for $T(u,v)$. This implicitly specifies how many "levels " for each coefficient.

a. Low compression

1	1	1	1	1	2	2	4
1	1	1	1	1	2	2	4
1	1	1	1	2	2	2	4
1	1	1	1	2	2	4	8
1	1	2	2	2	2	4	8
2	2	2	2	2	4	8	8
2	2	2	4	4	8	8	16
4	4	4	4	8	8	16	16

b. High compression

1	2	4	8	16	32	64	128
2	4	4	8	16	32	64	128
4	4	8	16	32	64	128	128
8	8	16	32	64	128	128	256
16	16	32	64	128	128	256	256
32	32	64	128	128	256	256	256
64	64	128	128	256	256	256	256
128	128	128	256	256	256	256	256

$$\hat{T}(u,v) = round\frac{T(u,v)}{N(u,v)}$$

$$\dot{T}(u,v) = \hat{T}(u,v)N(u,v)$$

- The smaller the value in the normalization matrix, the more accurate the reconstruction coefficients. One can use either the default normalization matrix, or specify an arbiter matrix.
- To adjust the "JPEG quality": the $N(u,v)$ matrix can be multiplied by a

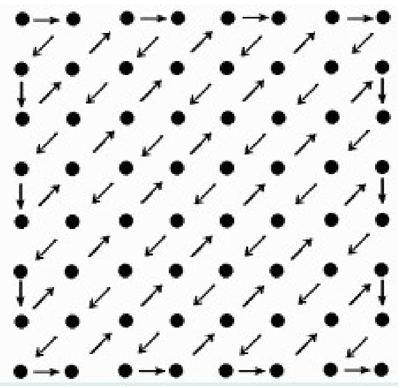
  Number less than 1: for higher quality

  Or greater than 1: for higher compression ratio.

Further details about JPEG

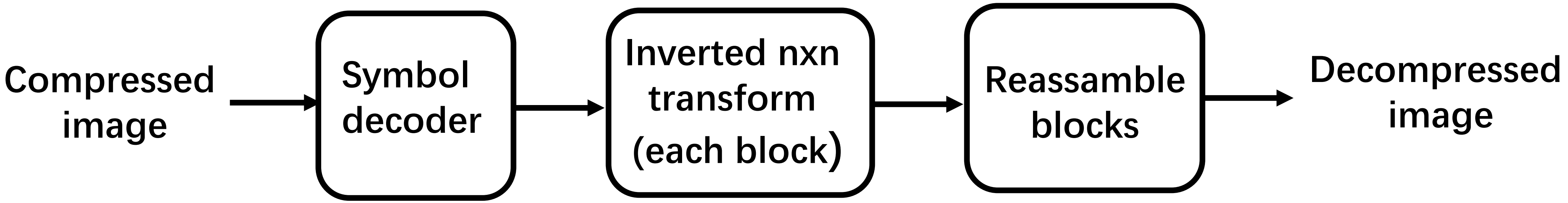
- After quantization, JPEG reorders the coefficients in a ZIG-ZAG pattern, then applies a lossless coder (e.g. Huffman/RLE).
- DC coefficient is coded separately. W.r.t the DC for the previous sub-blocks.
- For color, convert to a luminance/chromiance colorspace – intensity + 2 color channels (e.g. LAB). Chroma/ color channels are coded with lower bitrate.

ZIG-ZAG



The most widely-used lossy compression: JPEG

- JPEG: Joint Photographic Experts Group
- Block transform coding:



- What 2D transform to use?
- DFT, DCT (JPEC), HADAMARD, HAAR, WAVELET (JEPG2000)

Fidelity Criteria (保真度准则)

Objective Fidelity Criteria (客观保真度准则)

- **Root Mean Square Error (均方根误差)**

$$e_{\mathrm{rms}} = \left\{ \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y-0}^{N-1} \left[\hat{f}(x,y) - f(x,y) \right]^2 \right\}^{1/2}$$

Where $f(x, y)$ is the original image, and $\hat{f}(x, y)$ is an approximation.

- **Mean-square Signal-to-noise ratio (均方信噪比)**

$$\mathrm{SNR}_{\mathrm{ms}} = \frac{\sum_{x=0}^{M-1} \sum_{y-0}^{N-1} [f(x,y)]^2}{\sum_{x=0}^{M-1} \sum_{y-0}^{N-1} \left[\hat{f}(x,y) - f(x,y) \right]^2}$$

上海科技大学
ShanghaiTech University

Take home message

- Lossless compression is part of the information theory.
- The basic foundation for lossy image compression is unitary/frequency domain transform.
- Lossless compression can be component for lossy compression.