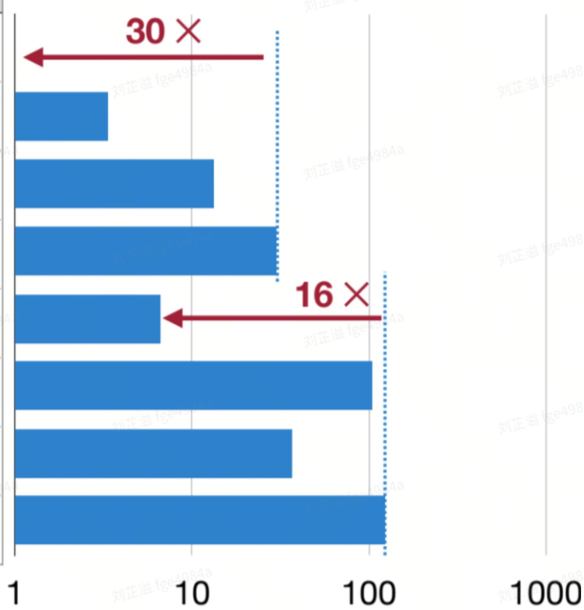
Quantization

- 减小存放权重的位数
- 剪枝+量化可以极大地减小**权重矩阵W的内存占用**
- 更少的比特宽度有利于减小计算资源的消耗

Operation	Energy [pJ]
8 bit int ADD	0.03
32 bit int ADD	0.1
16 bit float ADD	0.4
32 bit float ADD	0.9
8 bit int MULT	0.2
32 bit int MULT	3.1
16 bit float MULT	1.1
32 bit float MULT	3.7

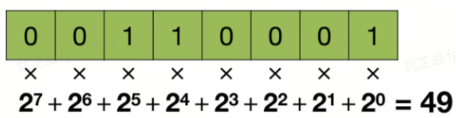


Rough Energy Cost For Various Operations in 45nm 0.9V

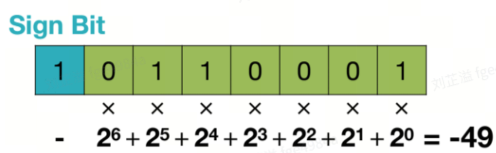
Numeric Data Types

- Integer

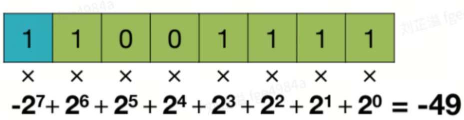
- Unsigned Integer
  - n-bit Range: $[0,\ 2^n - 1]$



- Signed Integer
  - Sign-Magnitude Representation
    - n-bit Range: $[-2^{n-1} - 1,\ 2^{n-1} - 1]$
    - Both 000…00 and 100…00 represent 0



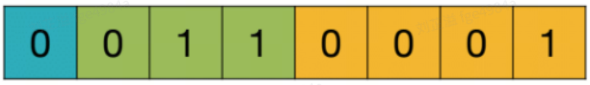
  - Two's Complement Representation
    - n-bit Range: $[-2^{n-1},\ 2^{n-1} - 1]$
    - 000…00 represents 0
    - 100…00 represents -2^{n-1}



- Fixed-Point Number

$$\times \quad \times \quad \times \quad \times \quad \times \quad \times \quad \times \quad \times$$

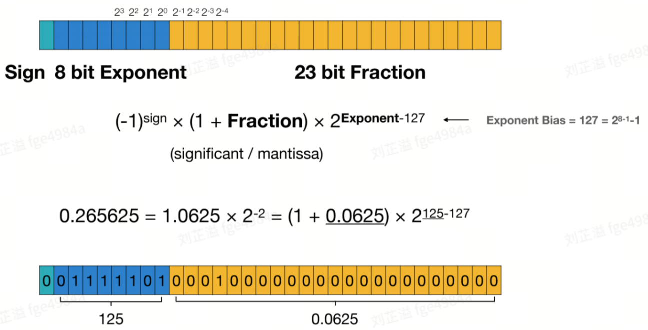
$$-2^3 + 2^2 + 2^1 + 2^0 + 2^{-1} + 2^{-2} + 2^{-3} + 2^{-4} \;\; = 3.0625$$

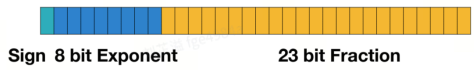


$$\times \quad \times \quad \times \quad \times \quad \times \quad \times \quad \times \quad \times$$

$$(\;-2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 \;) \times 2^{-4} = 49 \times 0.0625 = 3.0625$$

- Floating-Point Number: IEEE754 Standard





Exponent	Fraction=0	Fraction≠0	Equation
$00_H = 0$	±0	subnormal	$(-1)^{sign} \times \textbf{Fraction} \times 2^{1-127}$
$01_H \ldots FE_H = 1 \ldots 254$	normal		$(-1)^{sign} \times (1 + \textbf{Fraction}) \times 2^{\textbf{Exponent}-127}$
$FF_H = 255$	±INF	NaN	

Exponent Width → Range; Fraction Width → Precision

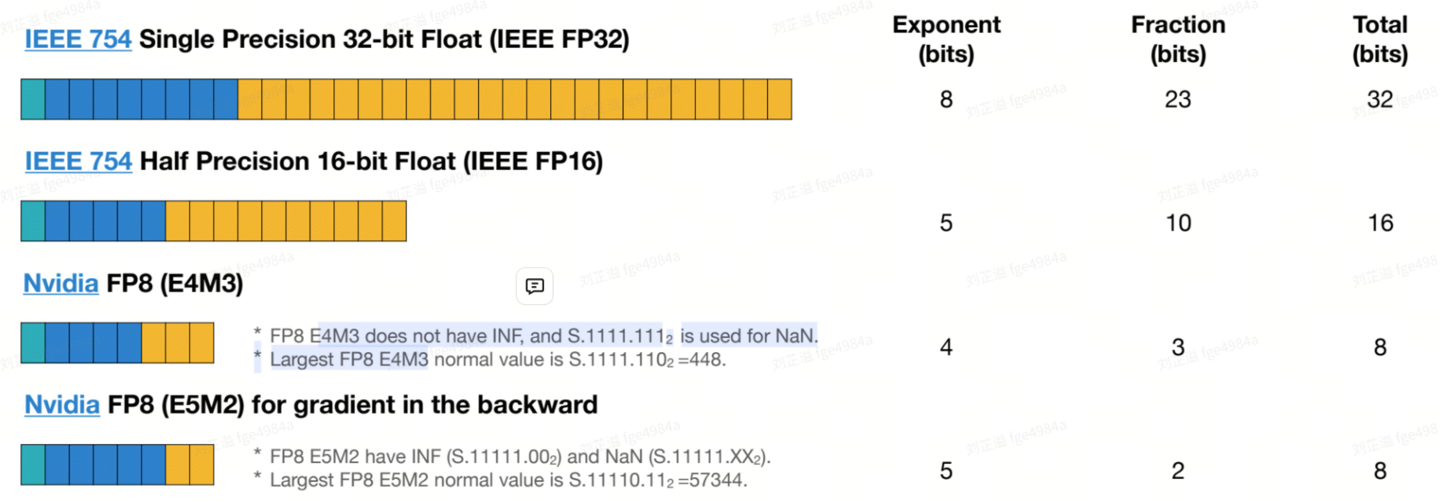
[IEEE 754] Single Precision 32-bit Float (IEEE FP32)

[IEEE 754] Half Precision 16-bit Float (IEEE FP16)

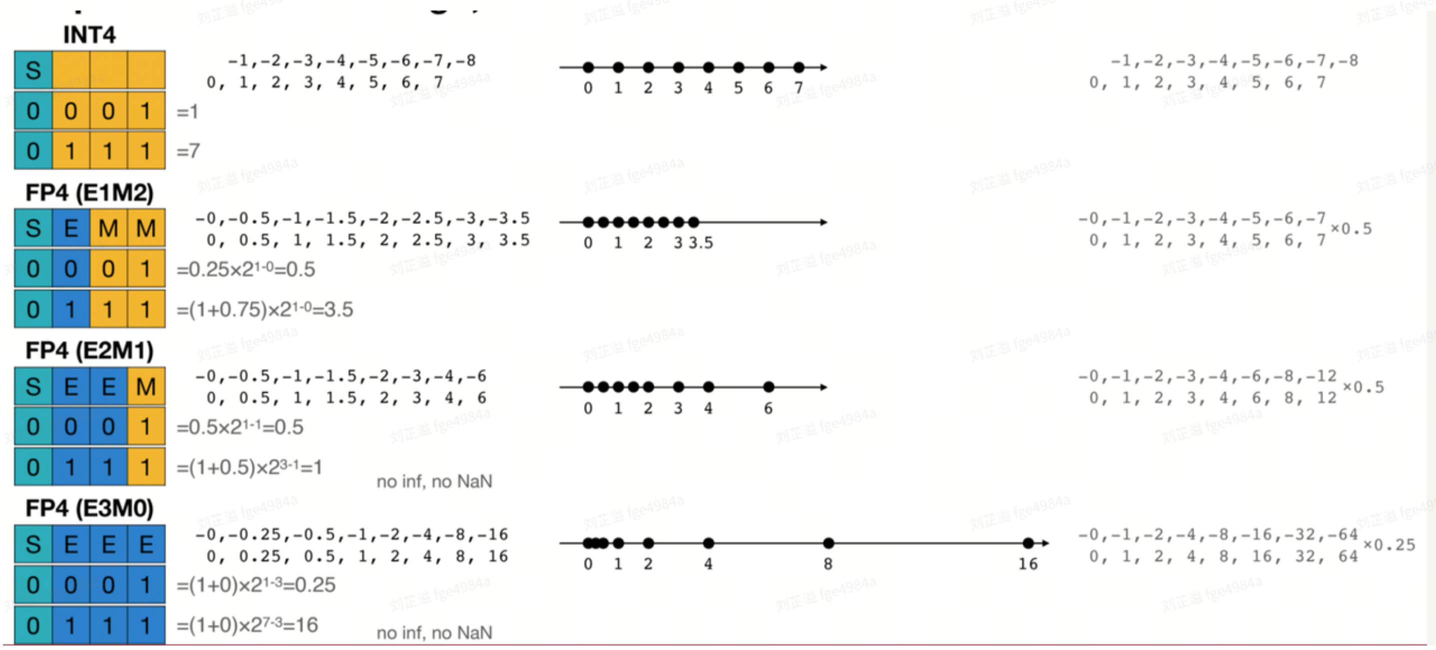
[Google] Brain Float (BF16)

	Exponent (bits)	Fraction (bits)	Total (bits)
IEEE FP32	8	23	32
IEEE FP16	5	10	16
BF16	8	7	16

- FP8 (E4M3)通常在forward使用；FP8 (E5M2)通常在backward使用，因为可表示的范围更大

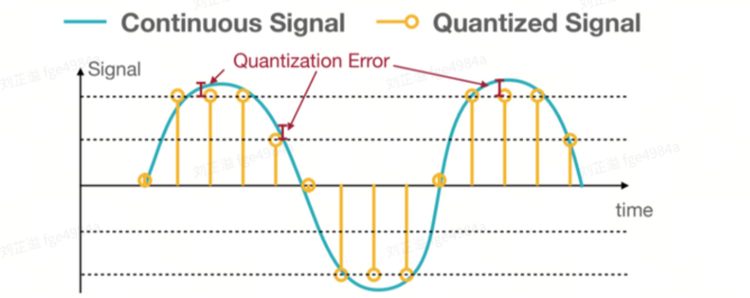


- FP4 and INT4



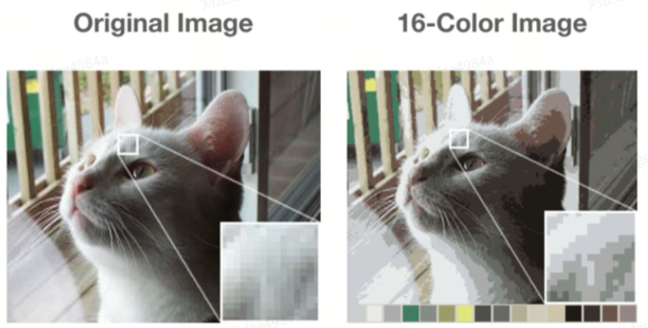
什么是量化

- 量化是将输入**从连续或大型集的值限制成离散集的过程**



The difference between an input value and its quantized value is referred to as quantization error.

Quantization [Wikipedia]



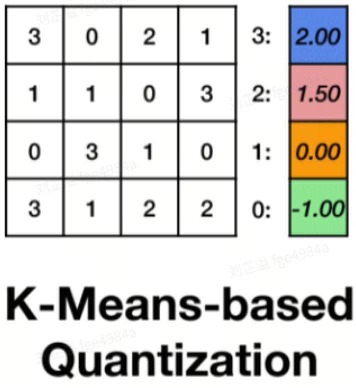
Images are in the public domain.

"Palettization"

K-Means-based Weight Quantization

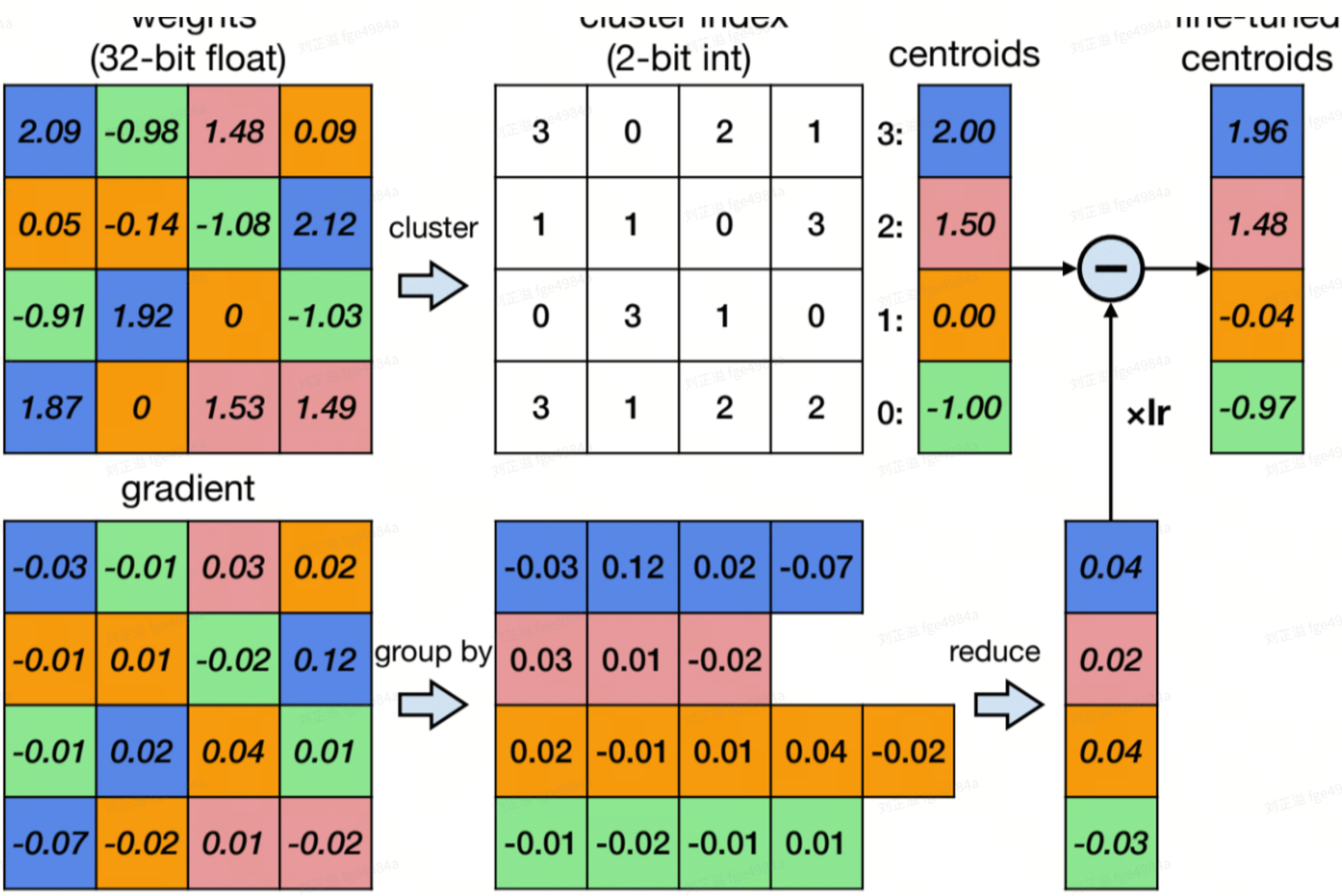
- Storage: Integer Weights; Floating-Point Codebook

- Computation: Floating-Point Arithmetic



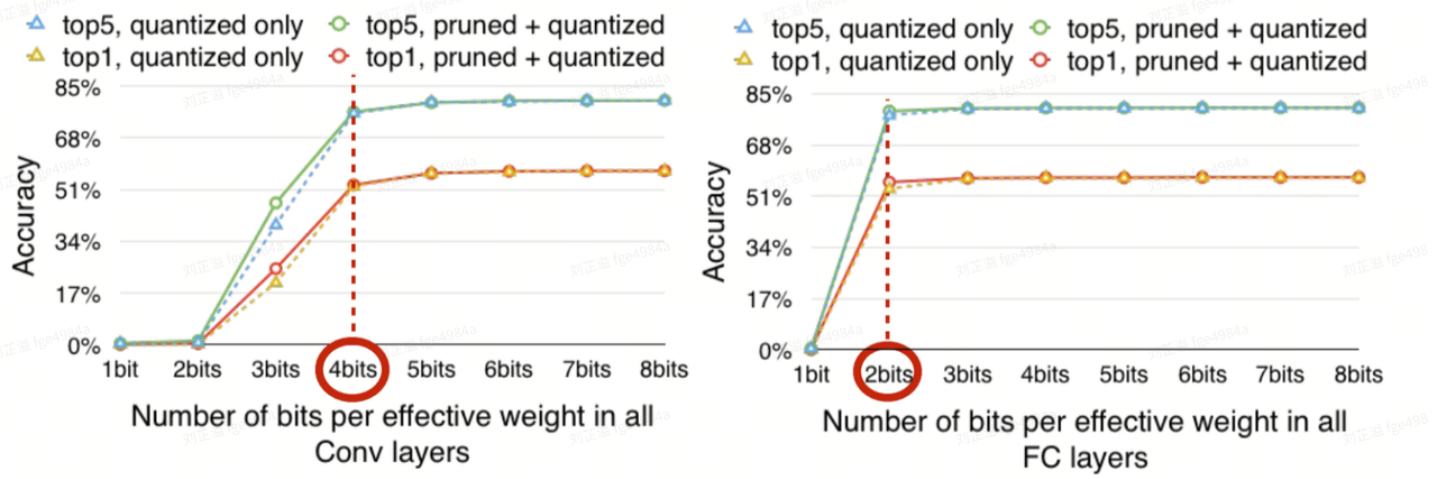
Fine-tuning Quantized Weights

- 按照Codebook进行分组相加后得到Codebook的gradient

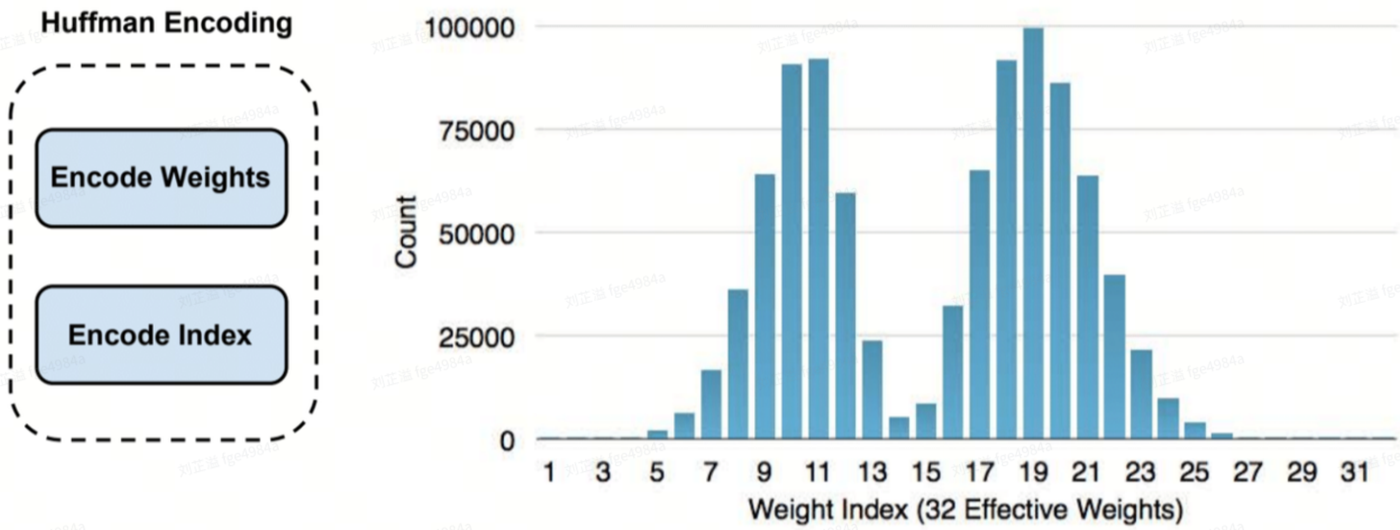


不同层需要的bit数

- 卷积层4bits；FC层2bits



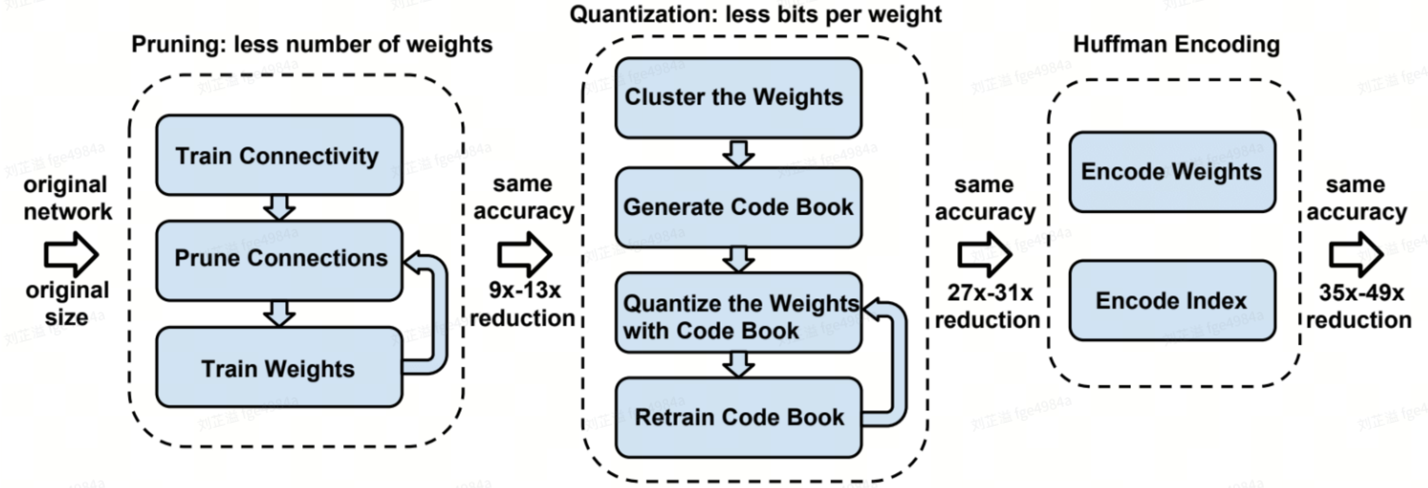
霍夫曼编码



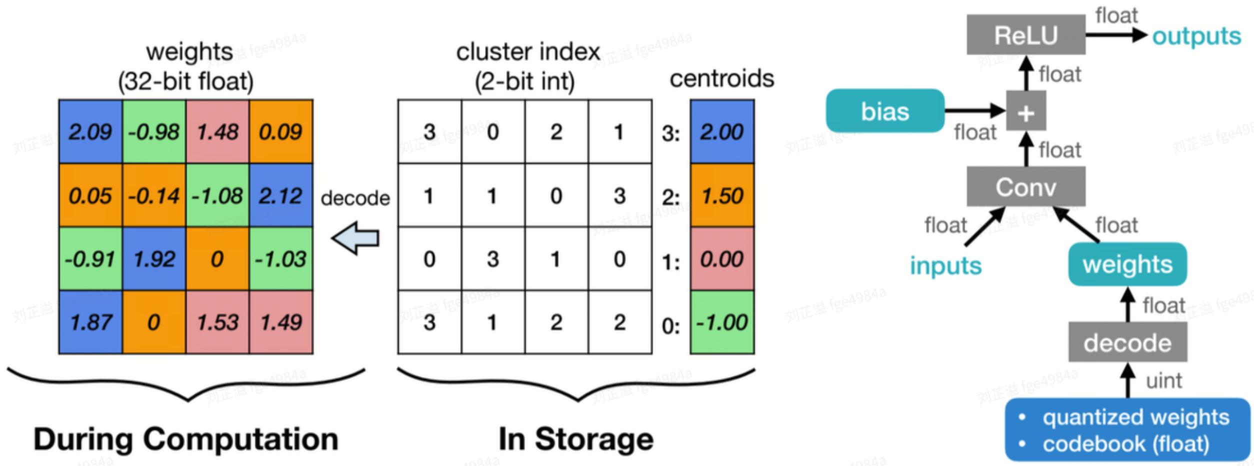
- In-frequent weights: use more bits to represent
- Frequent weights: use less bits to represent

Deep Compression

- 使用剪枝+量化+编码（不常使用）



- 只有存储是使用量化，而**计算和内存访问仍然是floating-point**



- The weights are decompressed using a lookup table (*i.e.*, codebook) during runtime inference.
- K-Means-based Weight Quantization only saves storage cost of a neural network model.
  - All the computation and memory access are still floating-point.

Linear Quantization