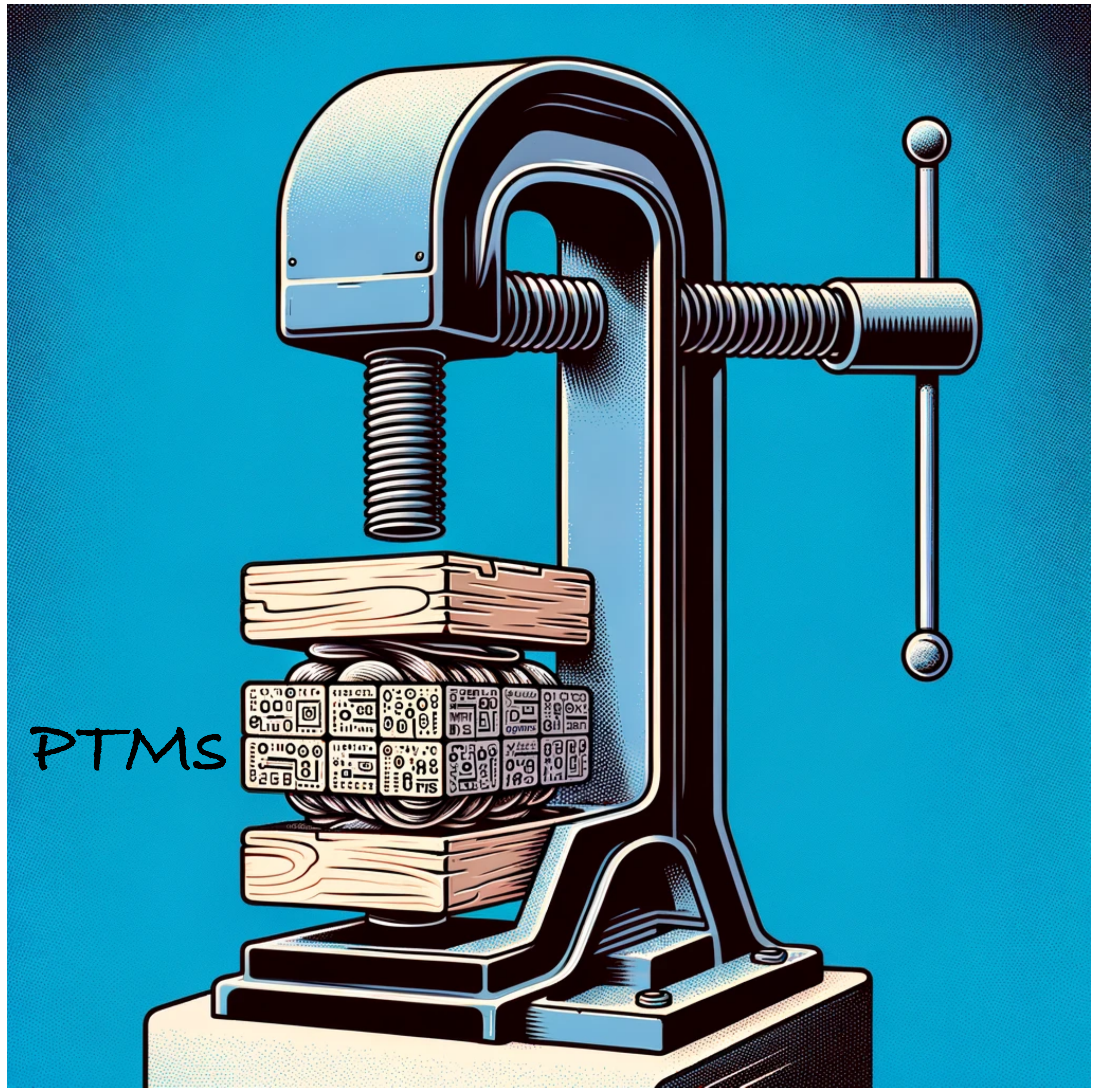
Pre-Trained Model Compression

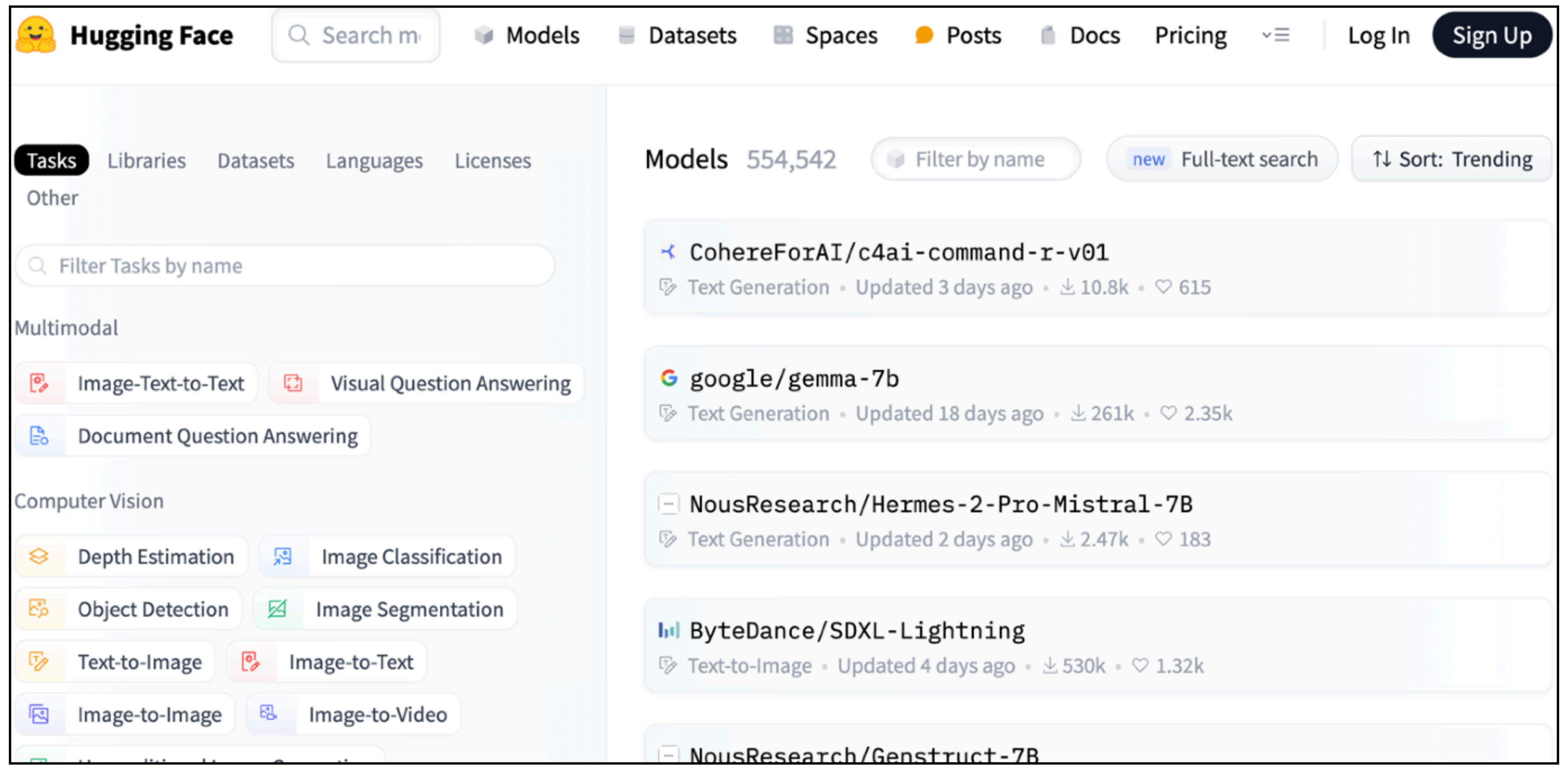


Zhaoyuan Su
Computer Science, University of Virginia
acf7ea@virginia.edu

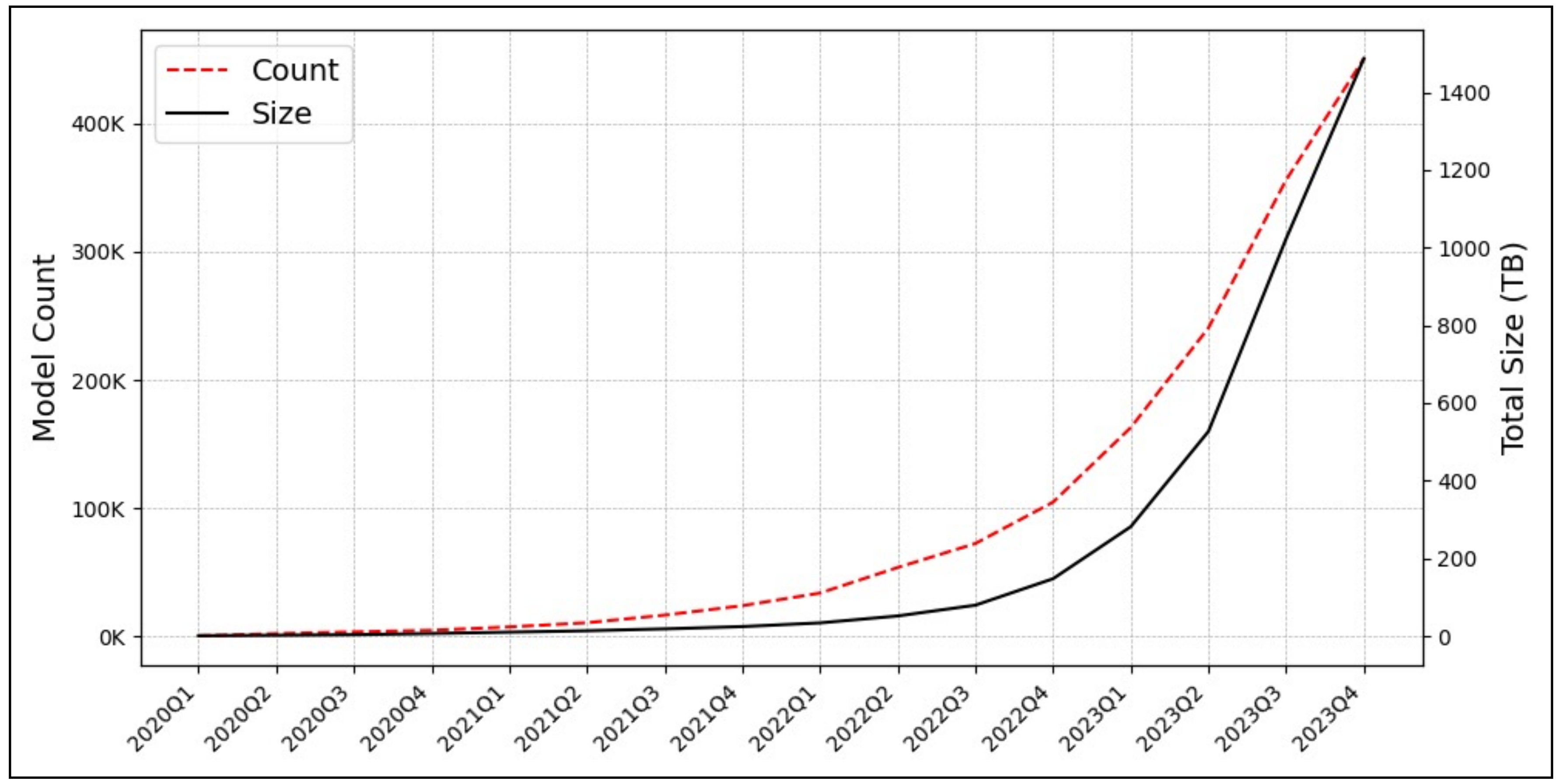
Learning Objectives

- Know basic concepts of model pruning and model quantization
- Understand the insight that motivates ELF and how the ELF compression & decompression algorithm work

Pre-Trained Models



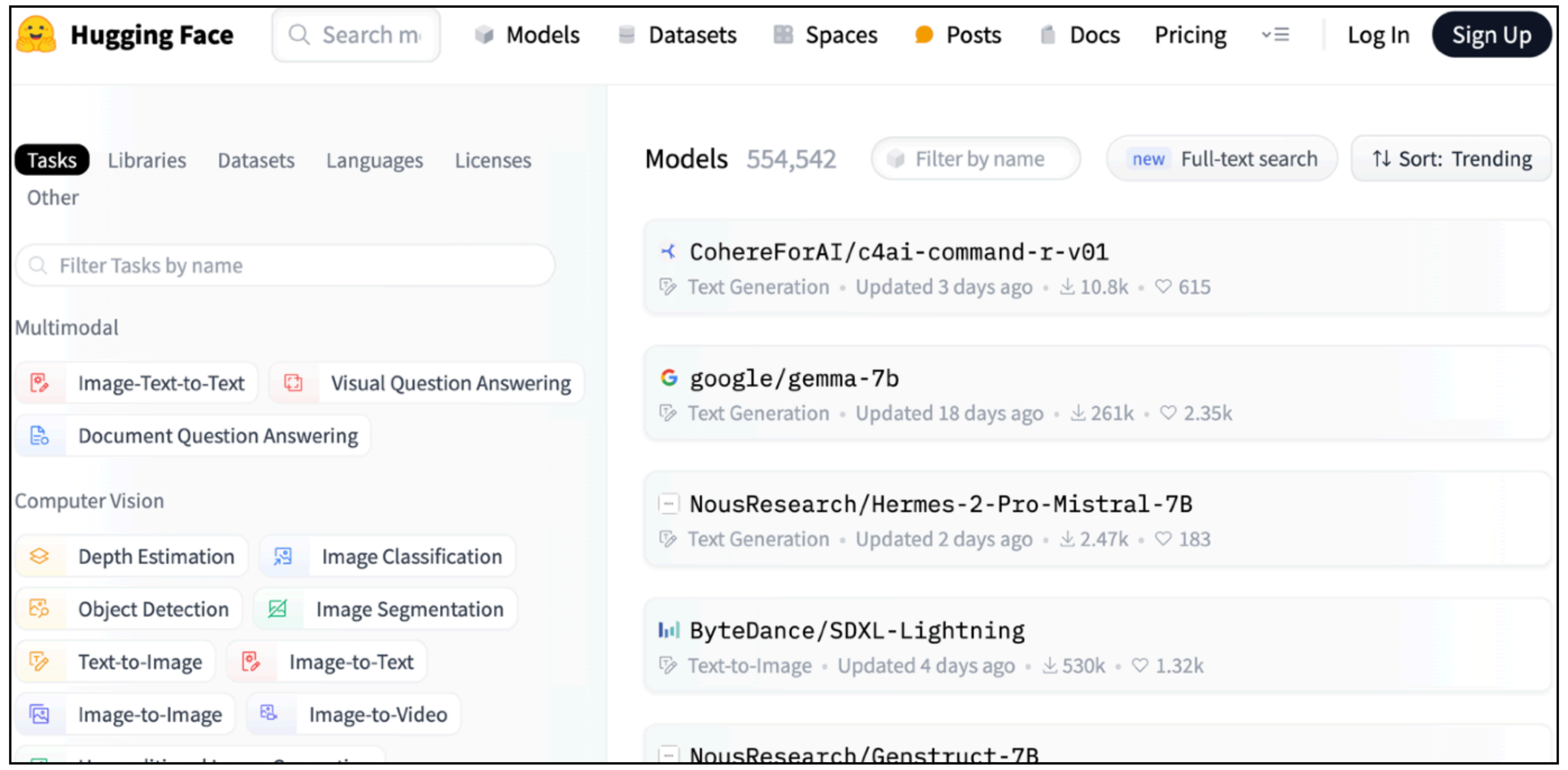
Searching PTMs on HuggingFace



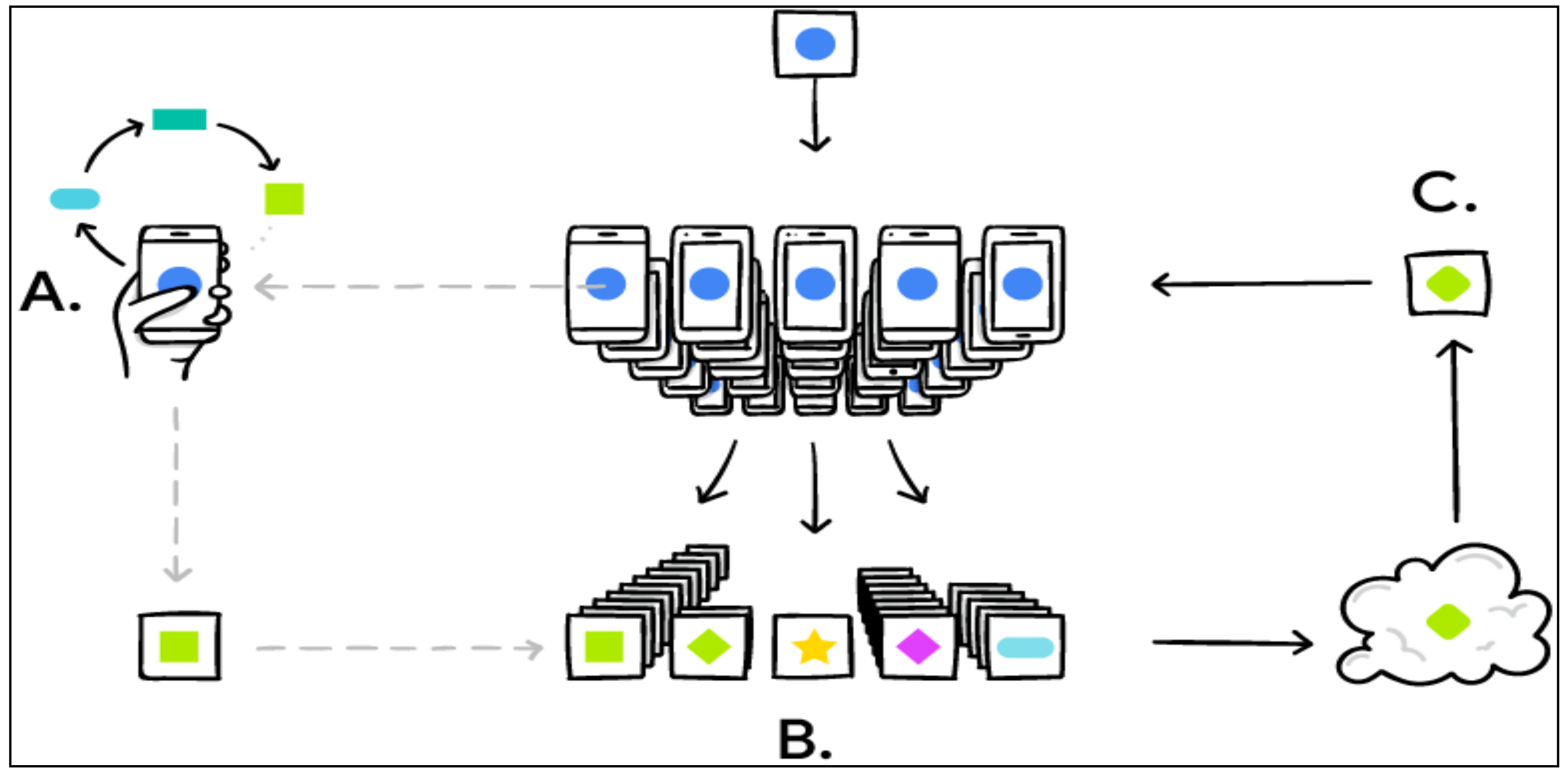
The PTM storage burden for HuggingFace

Model Storage

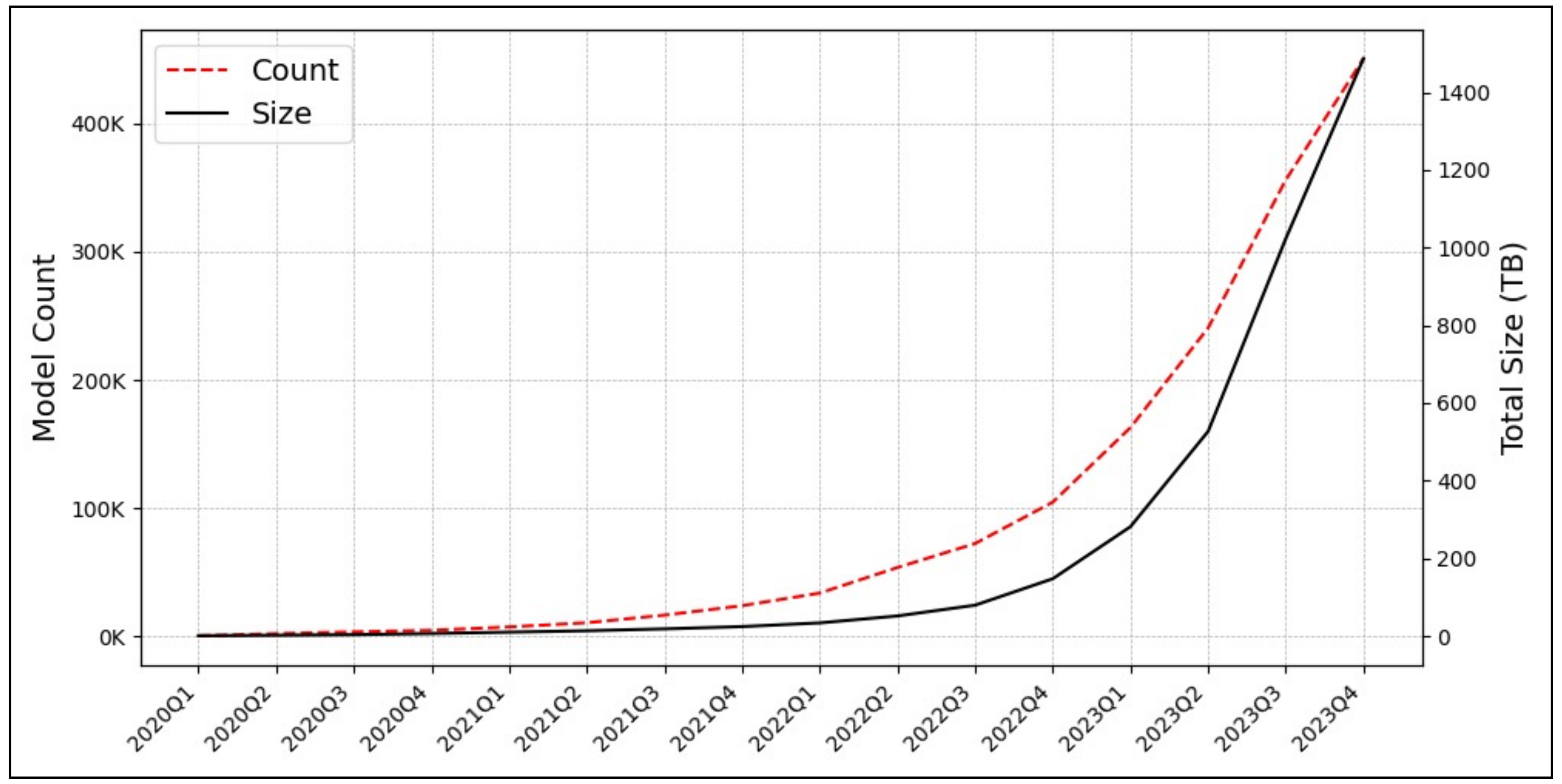
Pre-Trained Models



Searching PTMs on HuggingFace



Model transferring during FL training



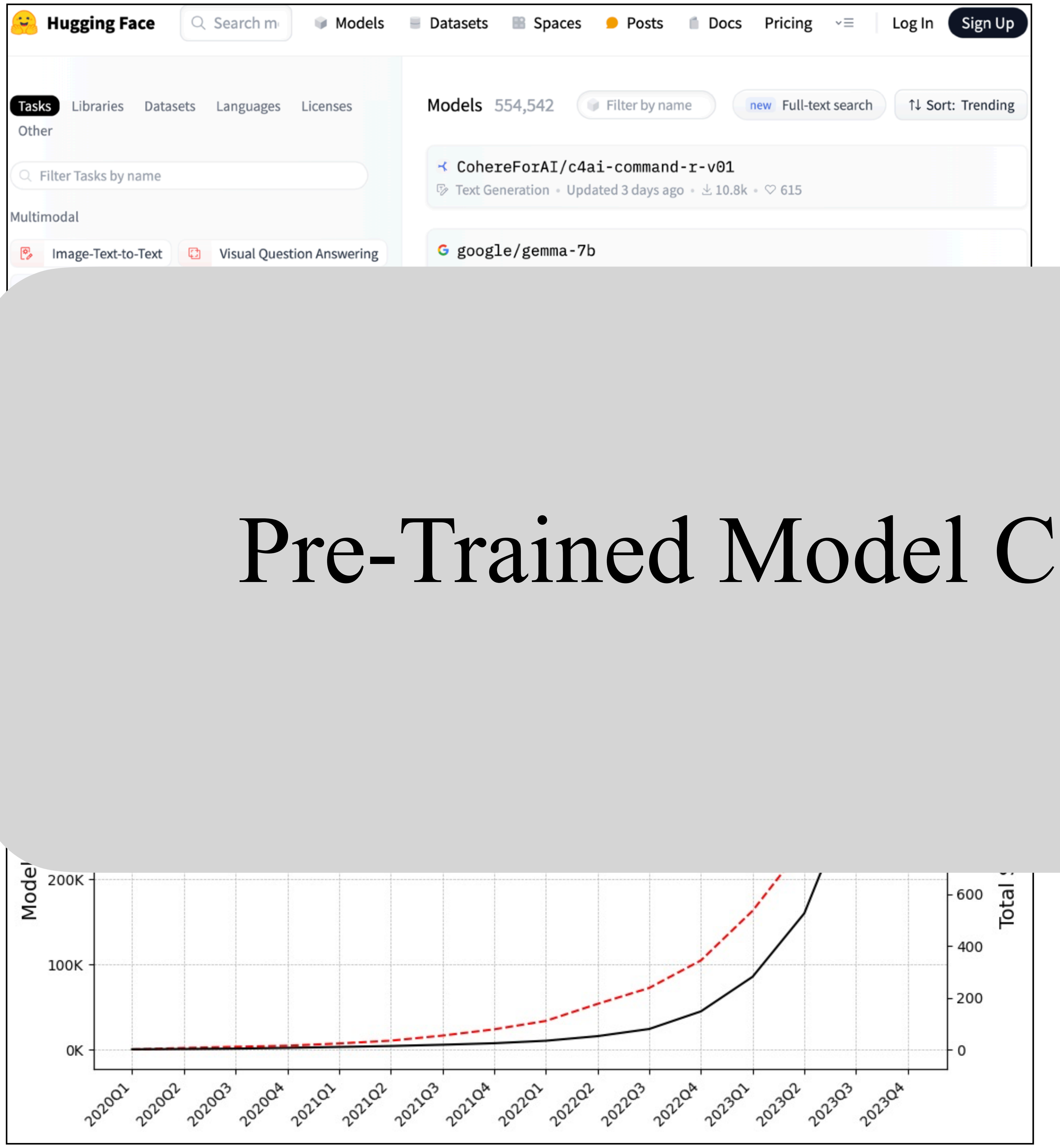
The PTM storage burden for HuggingFace

Model downloading for autopilot

Model Storage

Model Transfer

Pre-Trained Models



Pre-Trained Model Compression Matters!

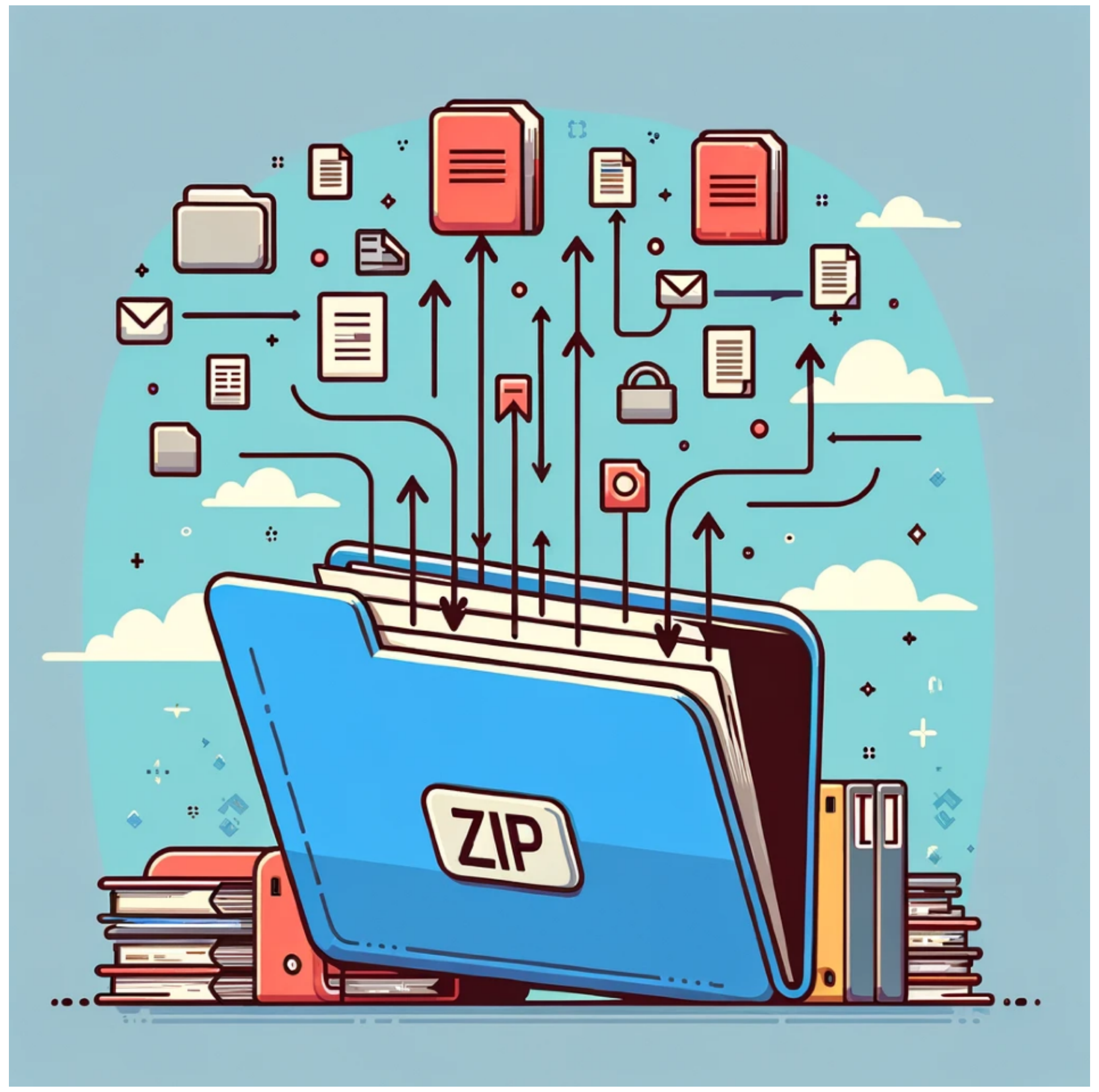
The PTM storage burden for HuggingFace

Model downloading for autopilot

Model Storage

Model Transfer

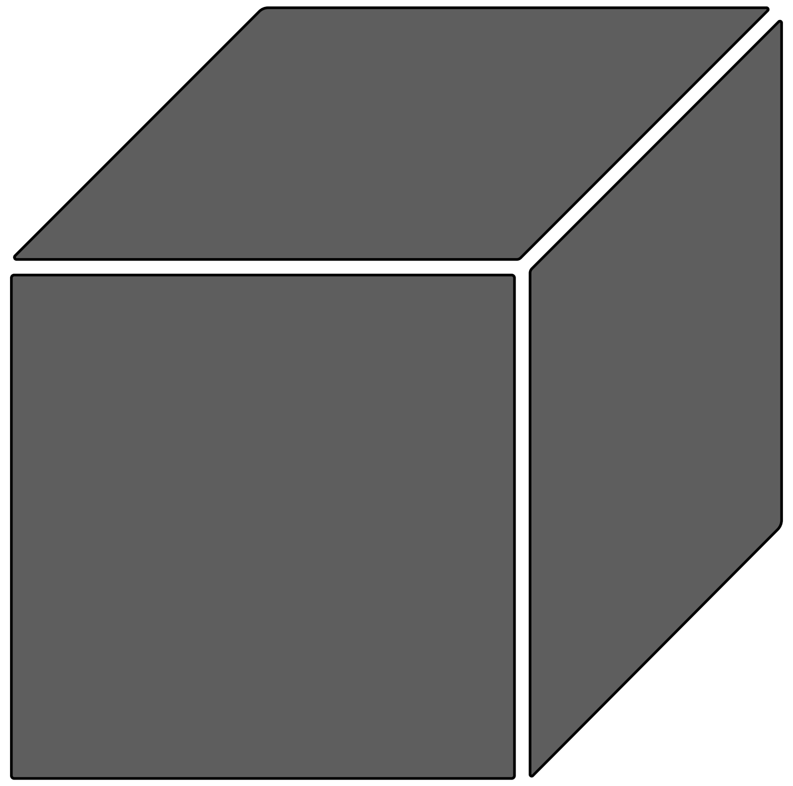
How about just ZIP pre-trained models?



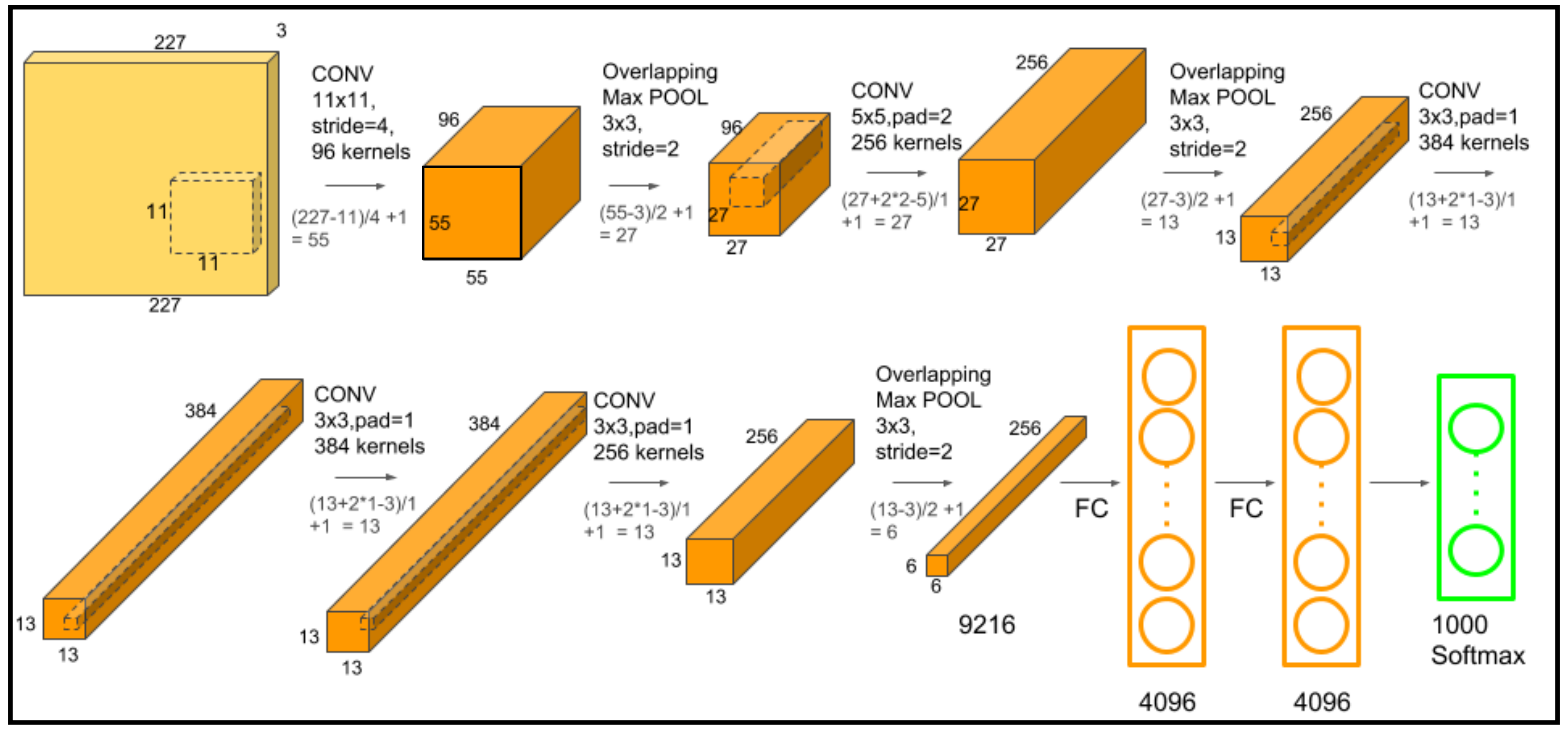
Answer: It may not be a good strategy.

PTMs typically achieve only about a 7% footprint reduction by general-purpose compressors like gzip, zstandard, etc.

What is inside a pre-trained model?



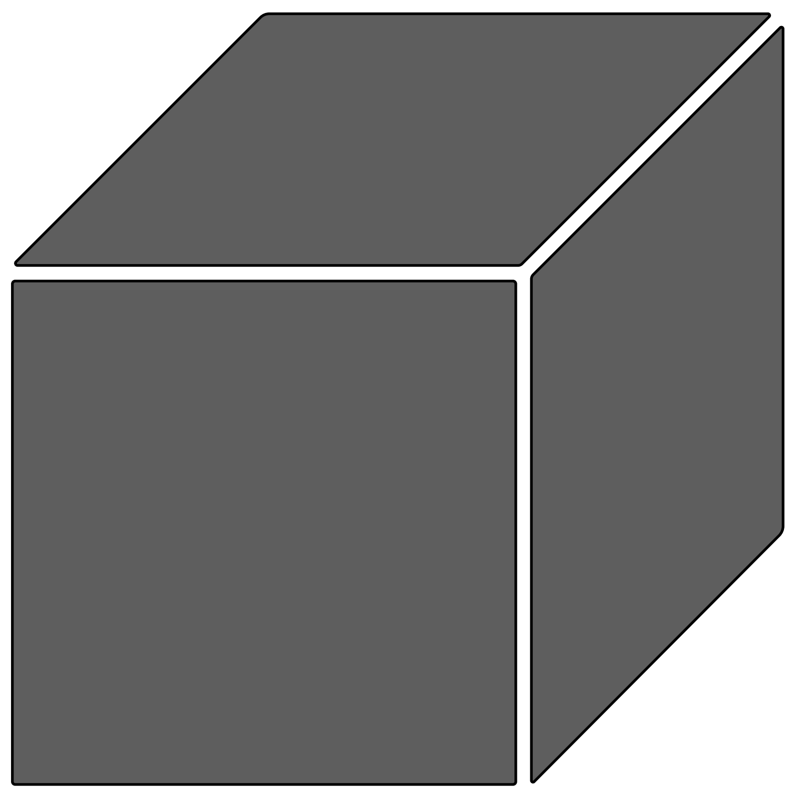
A pre-trained model



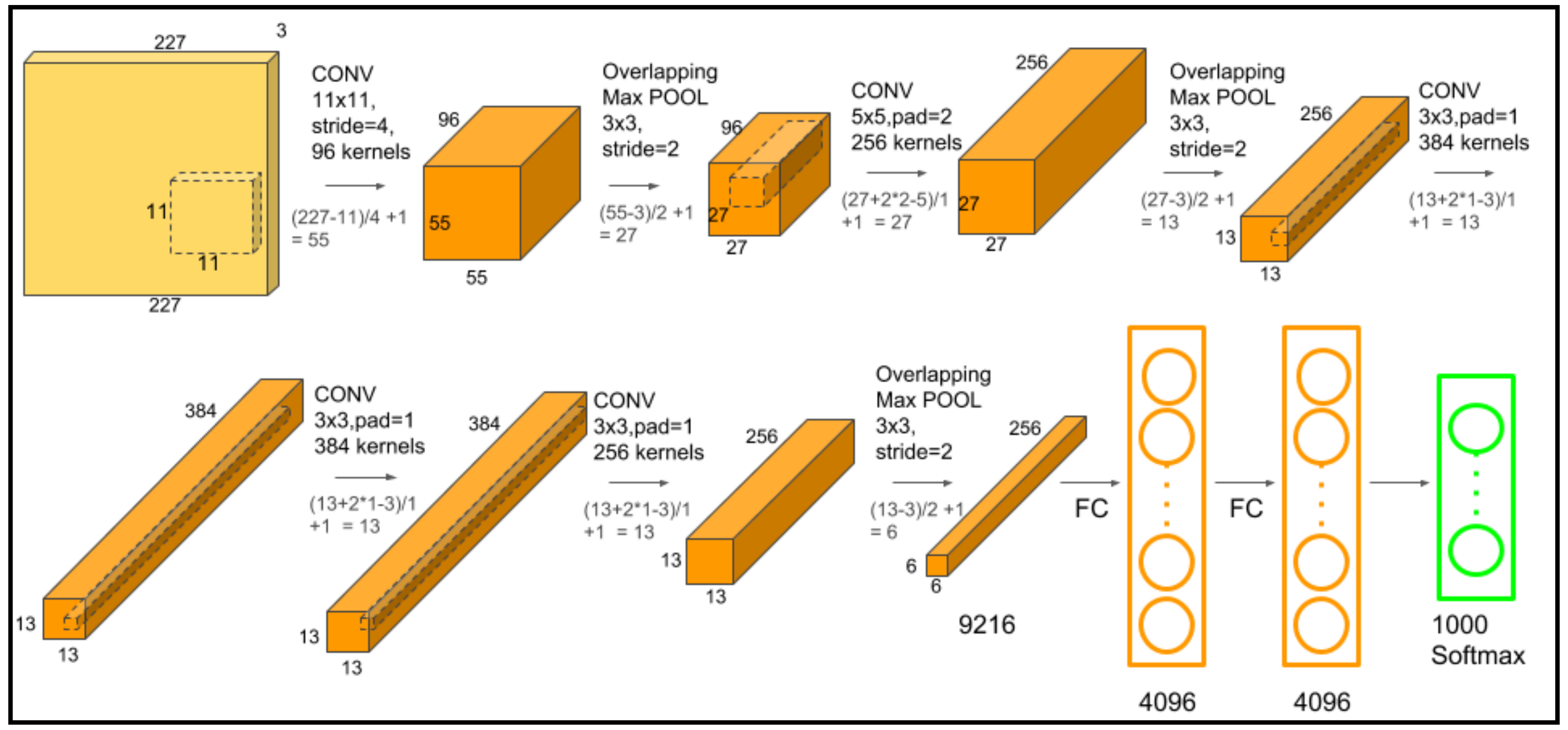
Model architecture (AlexNet as an example)

Abjieet Pujara, Concept of AlexNet:- Convolutional Neural Network, https://medium.com/analytics-vidhya/concept-of-alexnet-convolutional-neural-network-6e73b4f9ee30

What is inside a pre-trained model?



A pre-trained model



Model architecture (AlexNet as an example)

[parameter 1, paremeter 2, …
…
…, parameter i-1, parameter i],

…

,[parameter j, paremeter j+1, …
…
…, parameter n-1, parameter n]

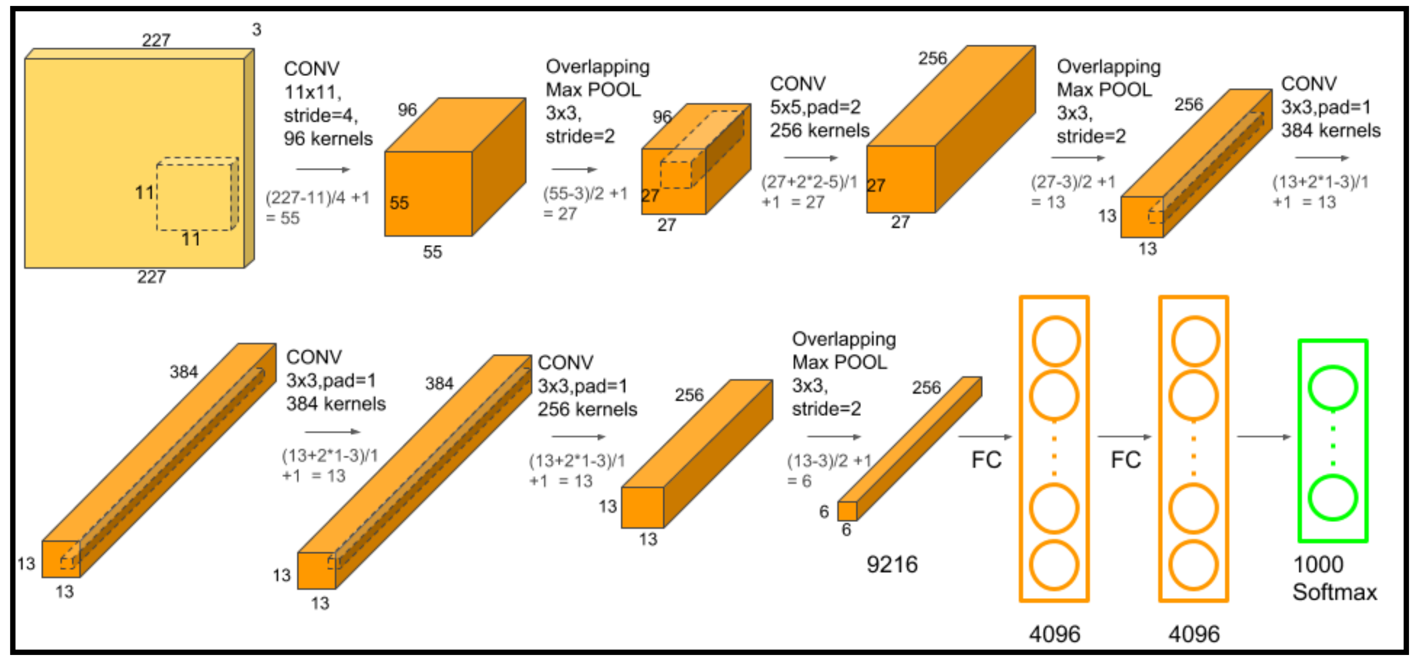
The 2D layer view

[0.15625, 0.3141152, …, -0.523787, 0.06256103]

Floating-point numbers

Insights from PTM dataset analysis

1. Layer Level: Not much layers duplicated.



Model architecture (AlexNet as an example)

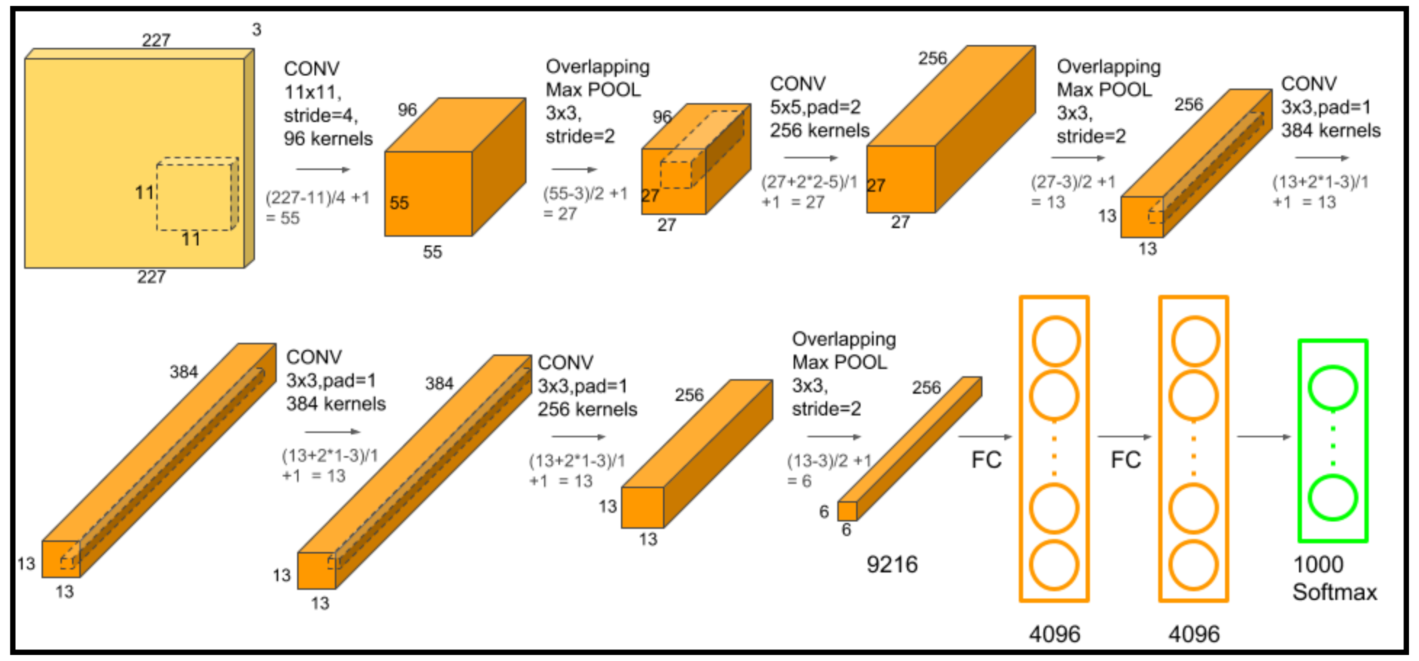
Layer Type	Count	Dup %	Total Sz in GB	Dup Sz in GB (%)
float32	240,966	8.35%	557.84	30.14 (5.40%)
float16	4,018	3.61%	14.51	0.14 (0.96%)
float64	199	0%	0.81	0 (0%)
uint8	1,597	99.81%	1.75	1.74 (99.43%)
int64	1,765	96.77%	0.97	0.94 (96.91%)
Overall	248,545	9.48%	575.88	32.96 (5.72%)

Model layer duplication statistics based on data types

"Everything You Always Wanted to Know About Storage Compressibility of Pre-Trained ML Models but Were Afraid to Ask." [Su et al., *arXiv preprint arXiv:2402.13429* (2024)].

Insights from PTM dataset analysis

1. Layer Level: Not much layers duplicated.
2. Chunk Level: Ineffective chunk deduplication.



Model Architecture (AlexNet as an example)

Data Type	Total Sz (GB)	Size of Duplicates in GB (%)		
		4 KB (FSC)	512 B (FSC)	CDC
float32	557.84	40.35 (7.23%)	42.92 (7.69%)	44.50 (8.16%)
float16	14.51	0.14 (0.96%)	0.14 (0.96%)	0.15 (1.03%)
float64	0.81	0 (0%)	0 (0%)	0 (0%)
uint8	1.75	1.74 (99.43%)	1.74 (99.43%)	1.74 (99.43%)
int64	0.97	0.94 (96.91%)	0.96 (98.97%)	0.96 (98.97%)
Overall	575.88	43.17 (7.50%)	45.76 (7.95%)	47.35 (8.22%)

Model chunk duplication statistics based on data types

"Everything You Always Wanted to Know About Storage Compressibility of Pre-Trained ML Models but Were Afraid to Ask." [Su et al., *arXiv preprint arXiv:2402.13429* (2024)].

Insights from PTM dataset analysis

1. Layer Level: Not much layers duplicated.
2. Chunk Level: Ineffective chunk deduplication.
3. Parameter Level: Float32 numbers dominate model parameters,

[parameter 1, paremeter 2, …
…
…, parameter i-1, parameter i],

…

,[parameter j, paremeter j+1, …
…
…, parameter n-1, parameter n]

The 2D layer view

Layer Type	Count (%)	Total Sz in GB (%)	Avg Para #	Avg Sz in MB
float32	240,966 (96.95%)	557.84 (96.87%)	621,421	2.37
float16	4,018 (1.62%)	14.51 (2.52%)	1,939,421	3.70
others	3,561 (1.43%)	3.53 (0.61%)	595,181	1.02
Overall	248,545 (100%)	575.88 (100%)	642,352	2.37

Model layer data type distribution

"Everything You Always Wanted to Know About Storage Compressibility of Pre-Trained ML Models but Were Afraid to Ask." [Su et al., *arXiv preprint arXiv:2402.13429* (2024)].

Insights from PTM dataset analysis

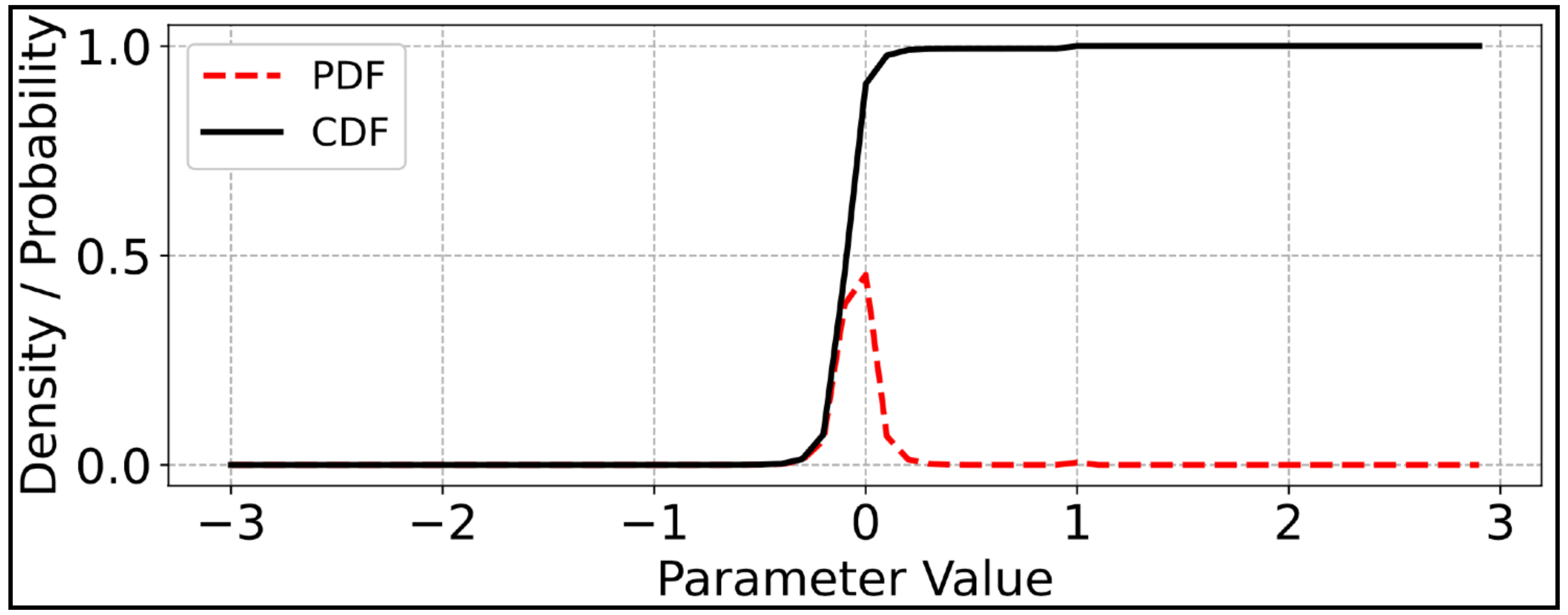
1. Layer Level: Not much layers duplicated.
2. Chunk Level: Ineffective chunk deduplication.
3. Parameter Level: Float32 numbers dominate model parameters, and almost all of them are within the range (-1, 1).

[parameter 1, paremeter 2, …
…
…, parameter i-1, parameter i],

…

,[parameter j, paremeter j+1, …
…
…, parameter n-1, parameter n]

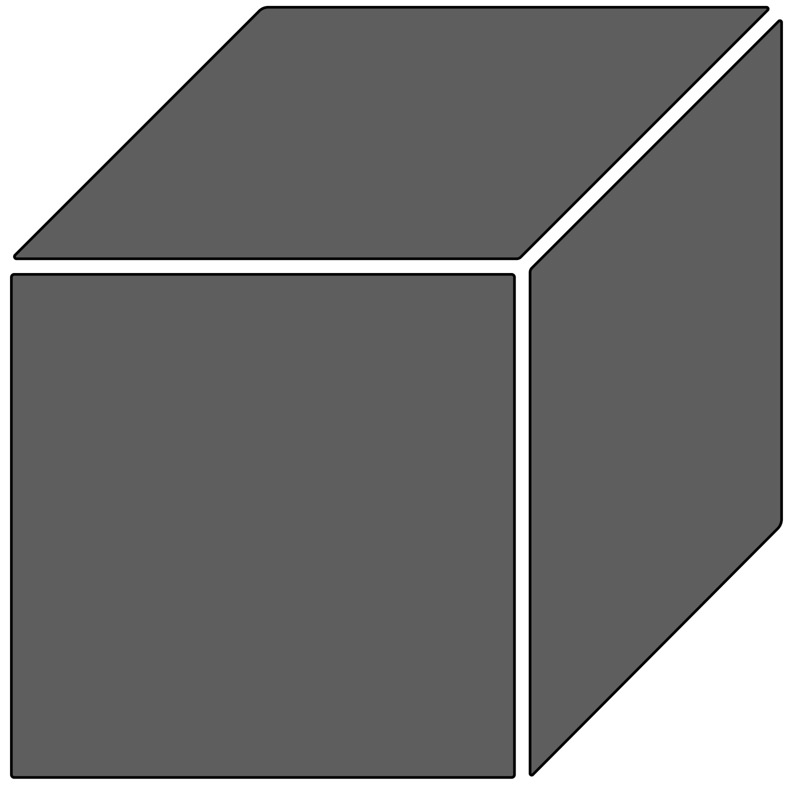
The 2D layer view



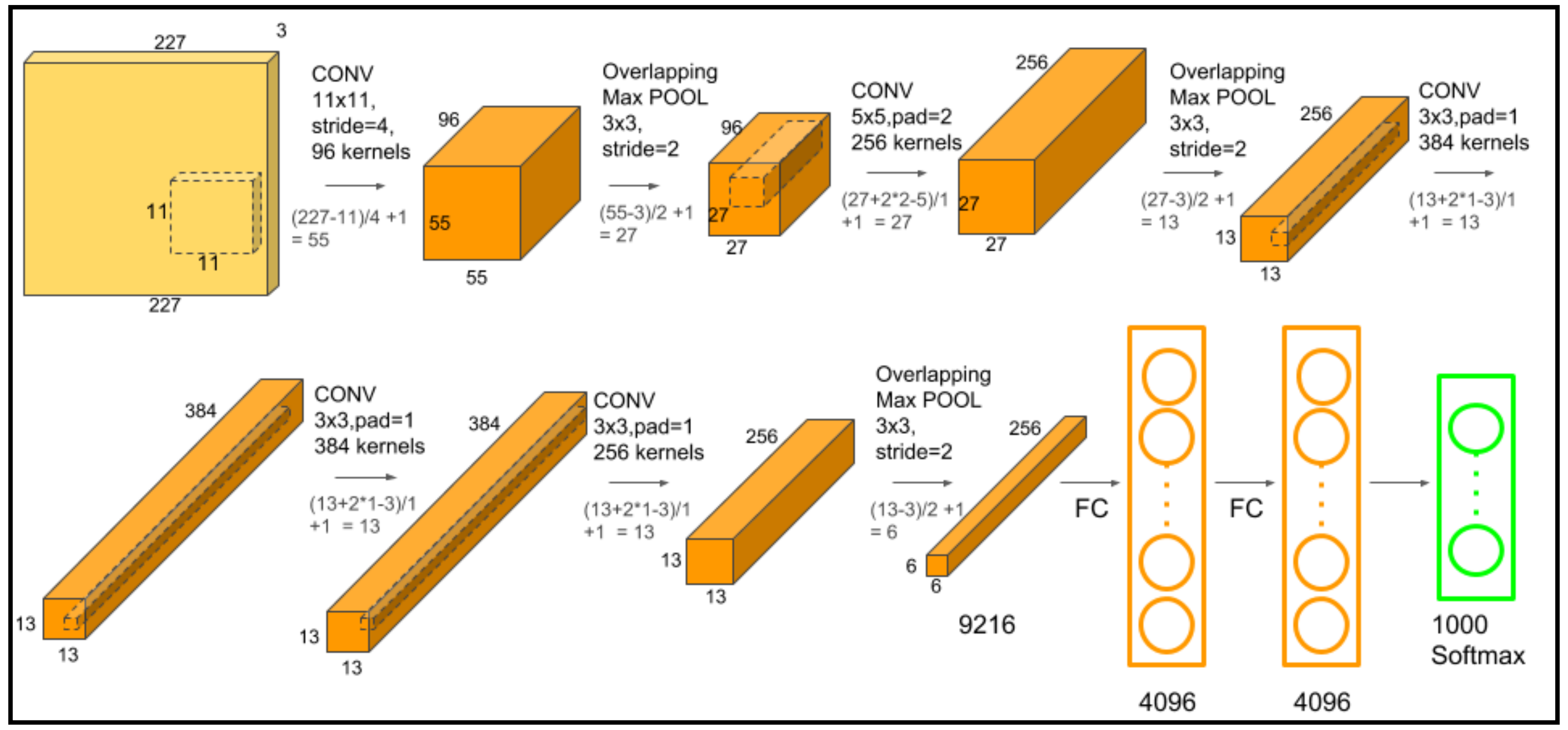
Model parameter distribution

"Everything You Always Wanted to Know About Storage Compressibility of Pre-Trained ML Models but Were Afraid to Ask." [Su et al., *arXiv preprint arXiv:2402.13429* (2024)].

What is inside a pre-trained model?



A pre-trained model



Model architecture (AlexNet as an example)

[parameter 1, paremeter 2, …
…
…, parameter i-1, parameter i],

…

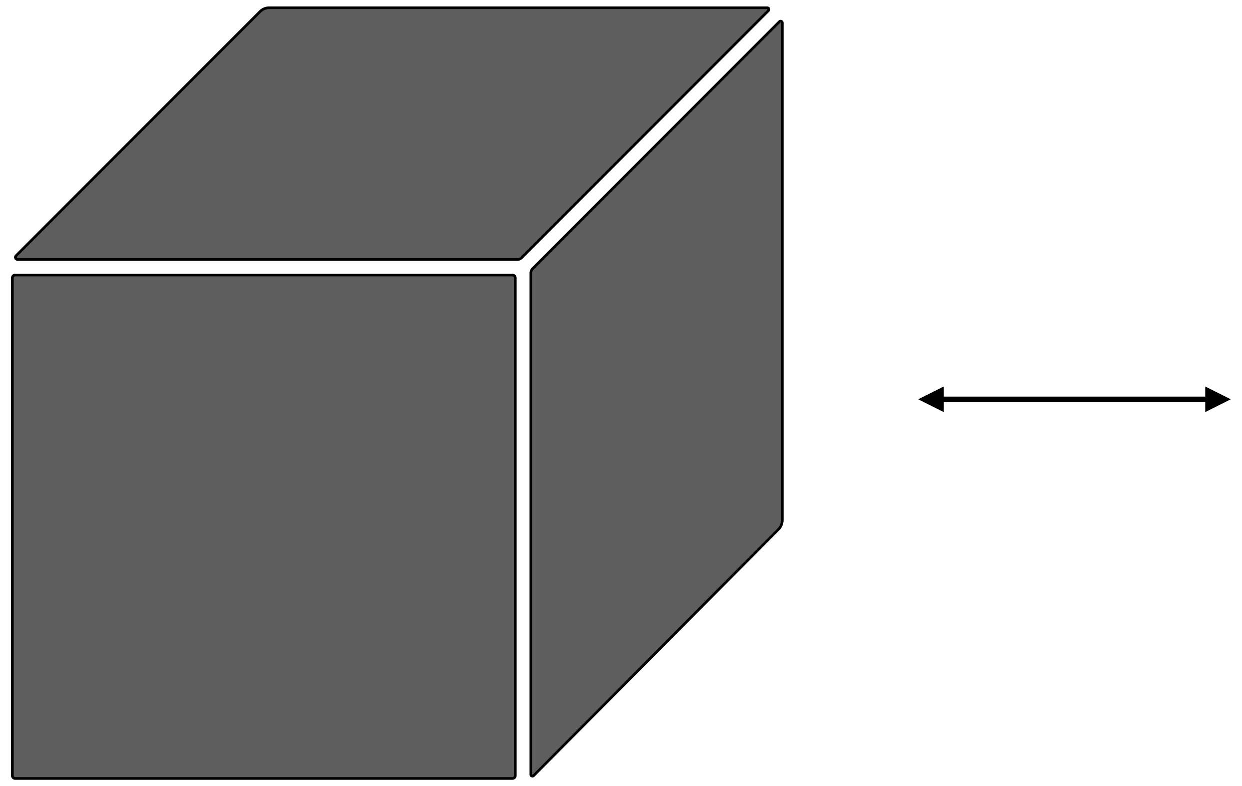
,[parameter j, paremeter j+1, …
…
…, parameter n-1, parameter n]

The 2D layer view

[0.15625, 0.3141152, …, -0.523787, 0.06256103]

Floating-point numbers

Pre-trained model size reduction

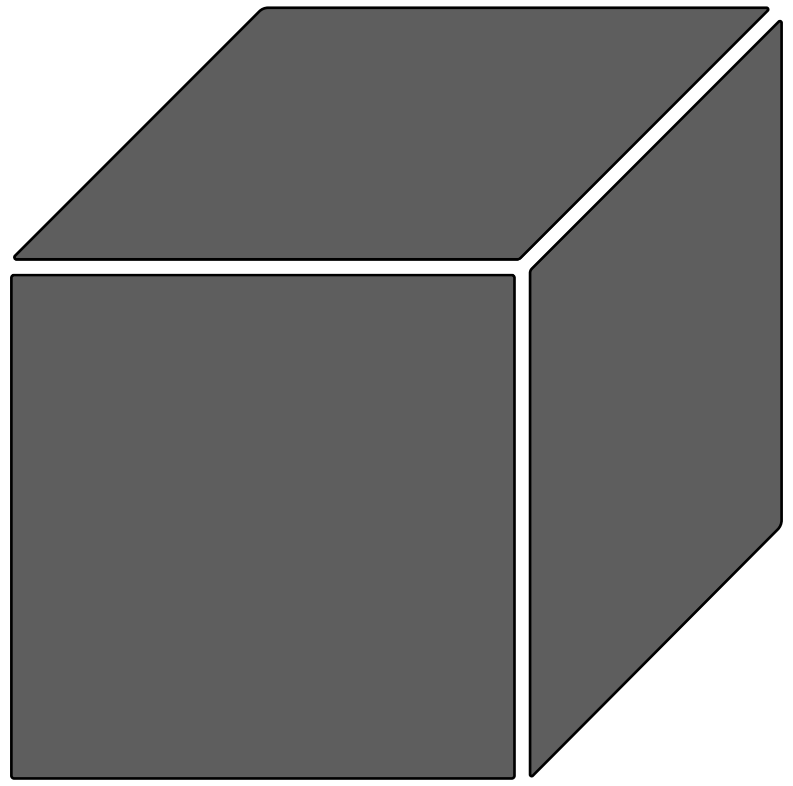


[0.15625, 0.3141152, …, -0.523787, 0.06256103]

A pre-trained model

Floating-point numbers

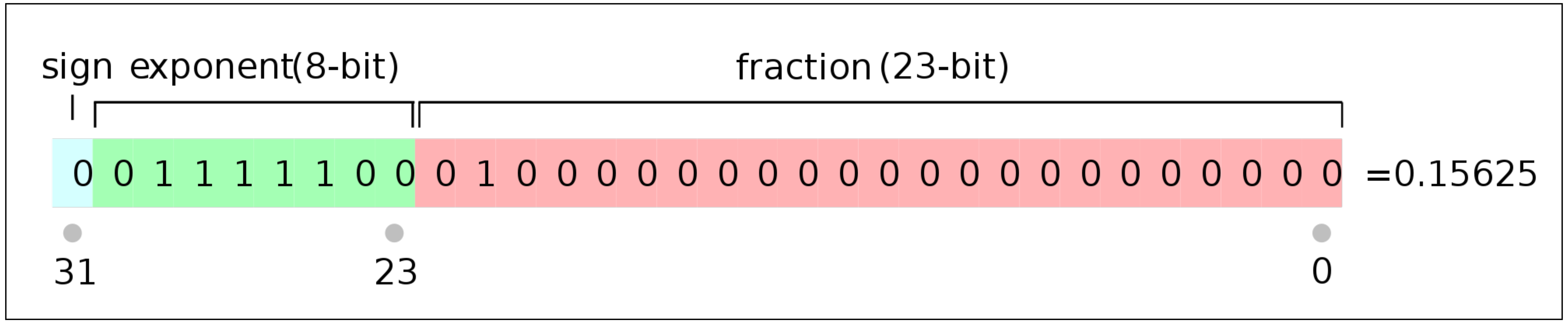
Pre-trained model size reduction



[**0.15625**, 0.3141152, …, -0.523787, 0.06256103]

A pre-trained model

Floating-point numbers



Binary presentation for floating-32 points in IEEE 754

Credit: Codekaizen, wikipedia.org

Pre-trained model size reduction

Question: How to calculate a model size?

$\boxed{\text{Model Size} = \#\text{Parameters} * \text{Bit\_width}}$

For example:

AlexNet - 61M parameters - float32

Size_AlexNet = 61M * 4 Bytes (32 bits) = 224M Bytes ≈ 214 MB

Pre-trained model size reduction

Question: How to reduce a model size?

Model Size = #Parameters * Bit_width

Pruning: Reduces the total **number** of parameters in a model;

Quantization: Decreases the **bit width** used to represent parameters;

Model pruning for PTM storage

Question: What weights should be pruned?

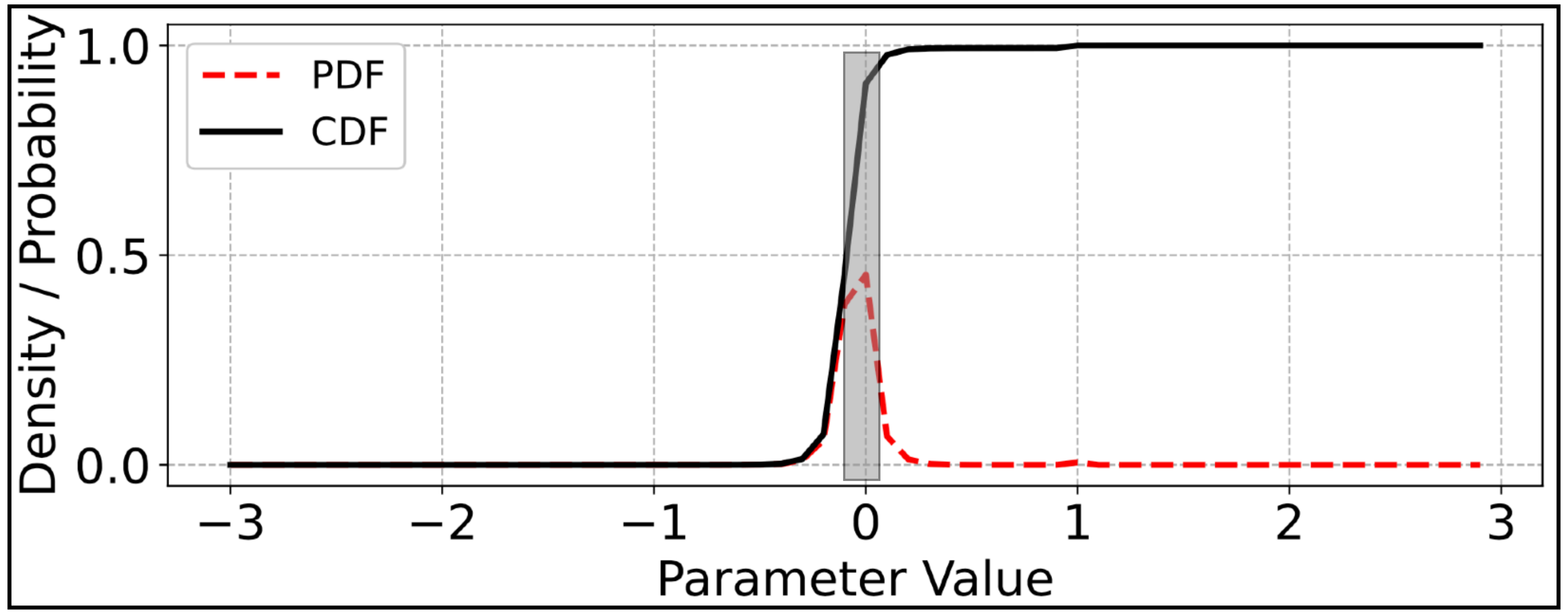
For example:

$[x_1,\ x_2,\ x_3] * [0.4278, -0.8491, 0.0249]^T = 0.4278x_1 + (-0.8491)x_2 + 0.0249x_3$

Input   Weights

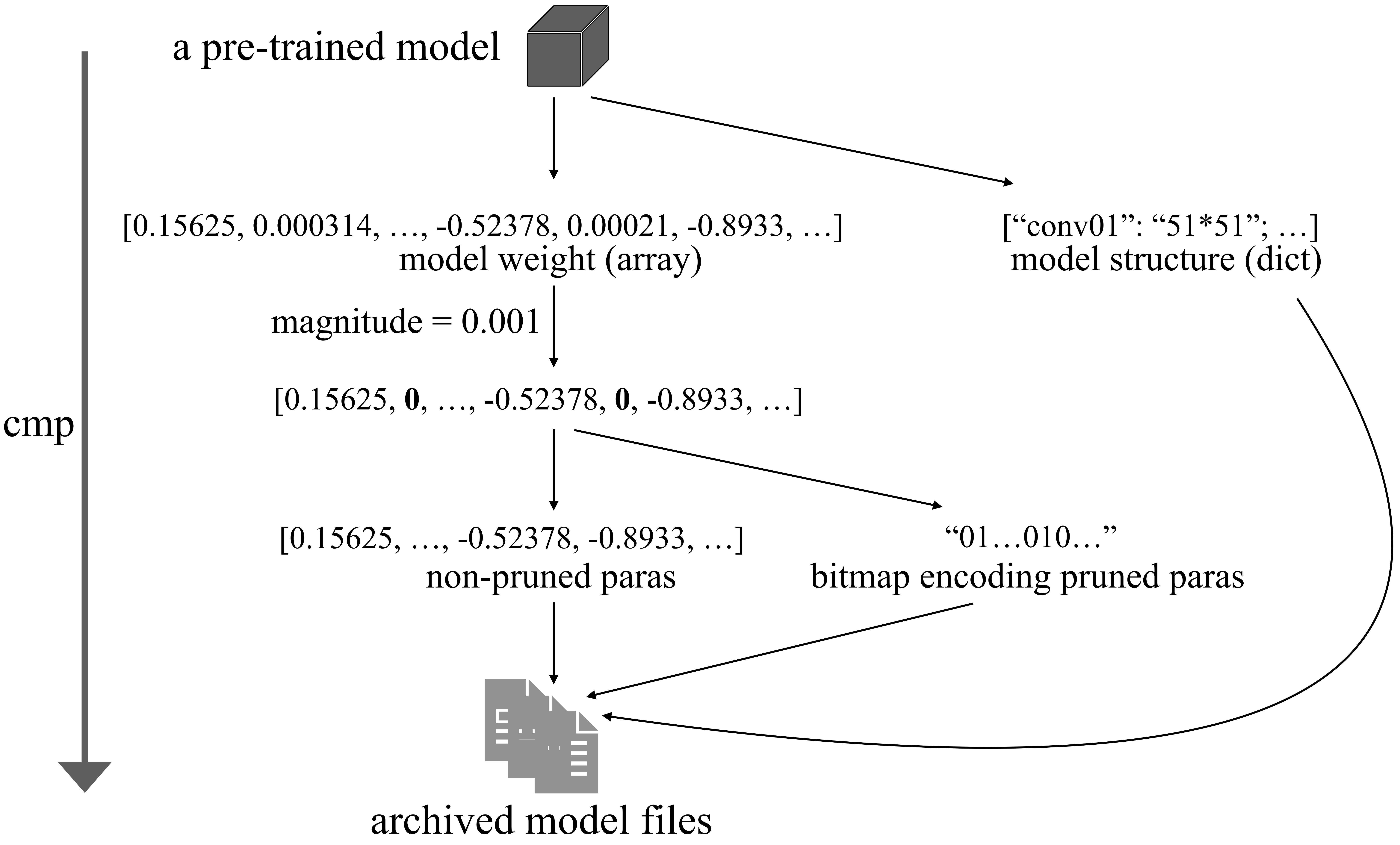
Which weight should be removed? Why?

The smaller the **magnitude,** the less **important** a parameter is.

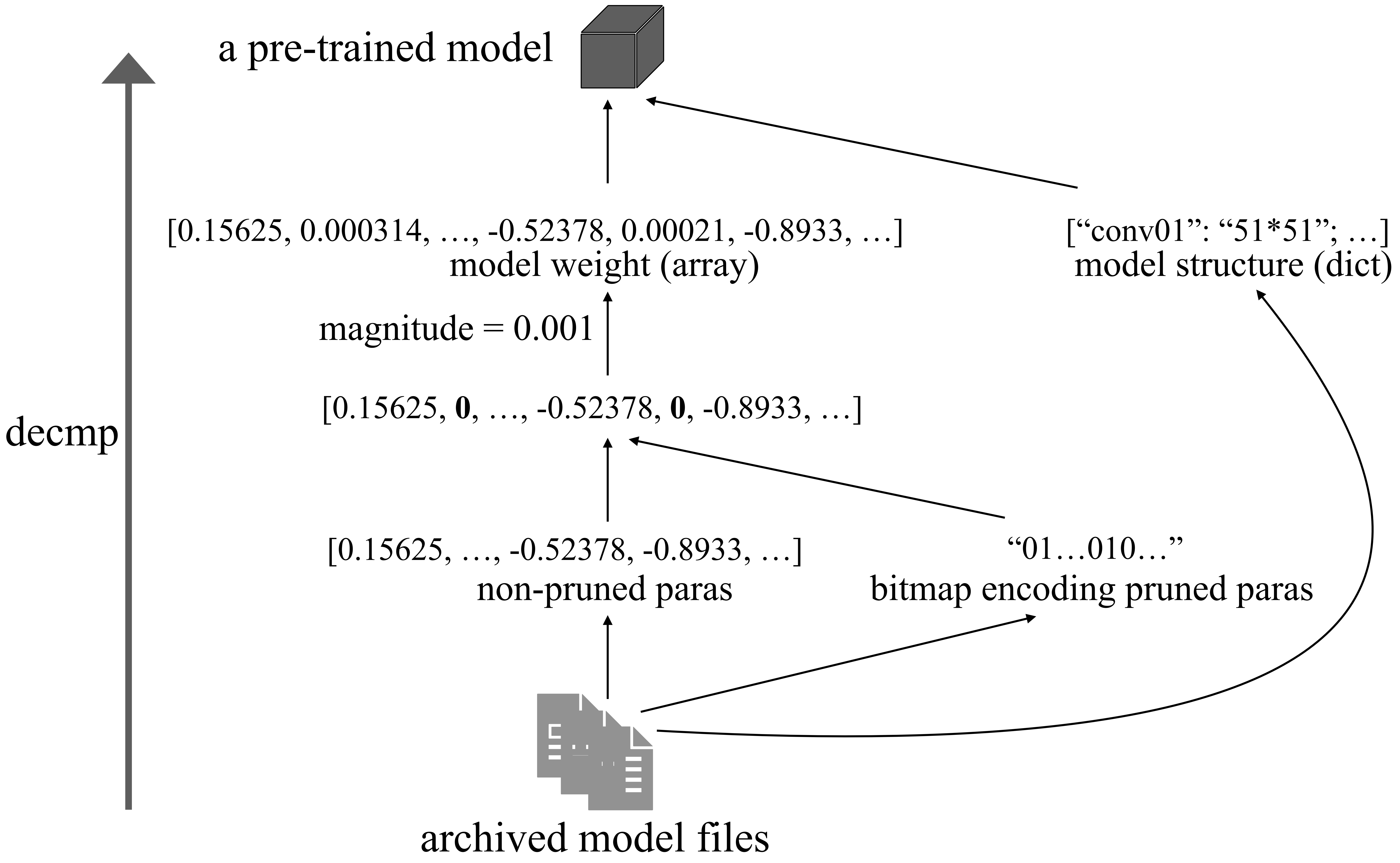


Model parameter distribution

Global magnitude pruning for PTM storage



Global magnitude pruning for PTM storage



Global magnitude pruning for PTM storage

Question: How to calculate the compression ratio (CR) with a magnitude?

Compression Ratio = Size_original / Size_compressed

AlexNet - 61M parameters - float32.
Size_original = 61M * 4 Bytes (32 bits) = 244M Bytes

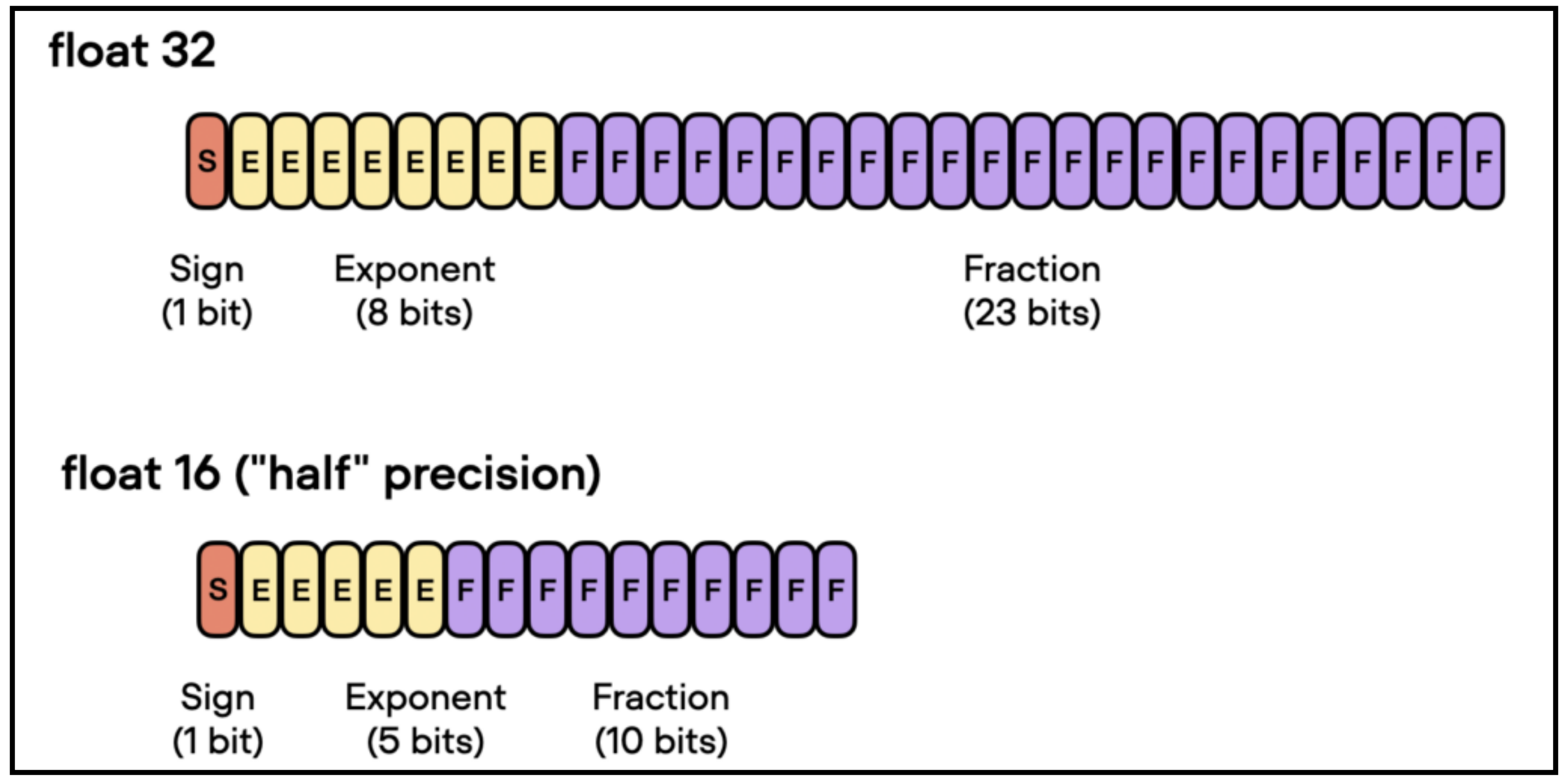
After pruning 15% of parameters:
Size_compressed = 61M * (1-15%) * 32 bits + 61M * 1 bit = 215M Bytes

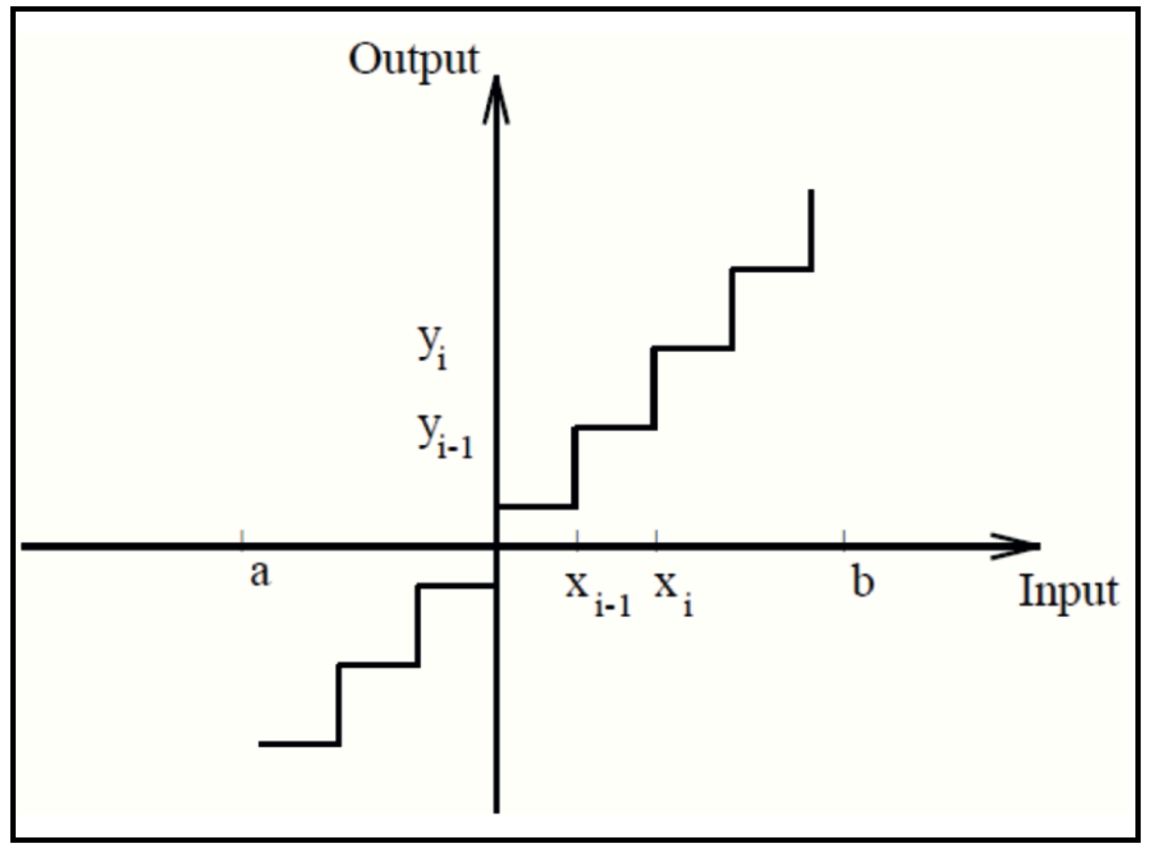
CR = Size_original / Size_compressed = 244M / 215M = 1.135

Magnitude Threshold vs. Compression Ratio vs. Accuracy Degradation

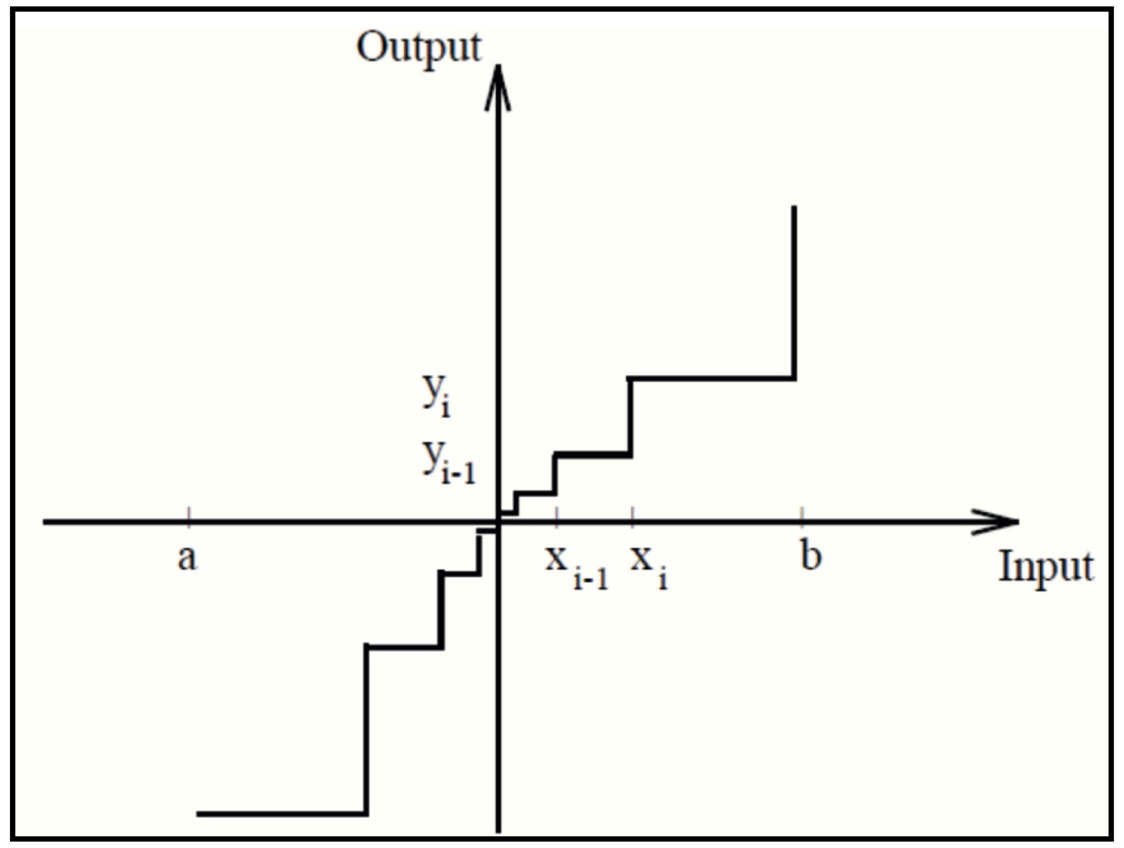
Model quantization for PTM storage

Question: How to reduce the bit length for represent a parameter?





Uniform quantization



Non-uniform quantization

Fajardo Carlos, Oscar M Reyes, and Ana Ramirez. "Seismic data compression using 2D lifting-wavelet algorithms." *Ingeniería y Ciencia* 11.21 (2015): 221-238.
Sebastian Raschka, Accelerating Large Language Models with Mixed-Precision Techniques, https://lightning.ai/pages/community/tutorial/accelerating-large-language-models-with-mixed-

Model quantization for PTM storage

Non-uniform quantization

Float 32 [2.09, -0.98, 1.48, 0.09, -0.05, 1.92, -1.03, 1.53, 0, 2.12]

↓ Clustering

Float 32 [2.09, -0.98, 1.48, 0.09, -0.05, 1.92, -1.03, 1.53, 0, 2.12]

↓ Approximating

Float 32 [2.0, -1.0, 1.5, 0, 0, 2.0, -1.0, 1.5, 0, 2.0]

↓ Encoding

Unsigned int [3, 0, 2, 1, 1, 3, 0, 2, 1, 3]

&

Codebook [0: -1; 1: 0; 2: 1.5; 3: 2.0]

Model quantization for PTM storage

Non-uniform quantization

Float 32 [2.09, -0.98, 1.48, 0.09, -0.05, 1.92, -1.03, 1.53, 0, 2.12]

↓ Clustering

Float 32 [2.09, -0.98, 1.48, 0.09, -0.05, 1.92, -1.03, 1.53, 0, 2.12]

↓ Approximating

Float 32 [2.0, -1.0, 1.5, 0, 0, 2.0, -1.0, 1.5, 0, 2.0]

↓ Encoding

2-bit rep [11, 00, 10, 01, 01, 11, 00, 10, 10, 11]

&

2 bits + 32 bits [00: -1; 01: 0; 10: 1.5; 11: 2.0]

Model quantization for PTM storage

Non-uniform quantization

Float 32	[2.09, -0.98, 1.48, 0.09, -0.05, 1.92, -1.03, 1.53, 0, 2.12]
	Quantization ↓
2-bit rep	[11, 00, 10, 01, 01, 11, 00, 10, 10, 11]
	&
2 bits + 32 bits	[00: -1; 01: 0; 10: 1.5; 11: 2.0]

Question: How to calculate the CR for non-uniform quantization?

CR = 10 * 32 / (10*2 + 4 * (2+32)) = 2.05

Model quantization for PTM storage

Uniform quantization

Float 32 [2.09, -0.98, 1.48, 0.09, -0.05, 1.92, -1.03, 1.53, 0, 2.12]

↓

Float 32 [2.08, -1.04, 1.04, 0, 0, 2.08, -1.04, 1.04, 0, 2.08]

||

([1, -2, 0, -1, -1, 1, -2, 0, -1, 1] - (-1)) * 1.04

Quantized 2-bit parameters Zero_point Scaling_factor

Model quantization for PTM storage

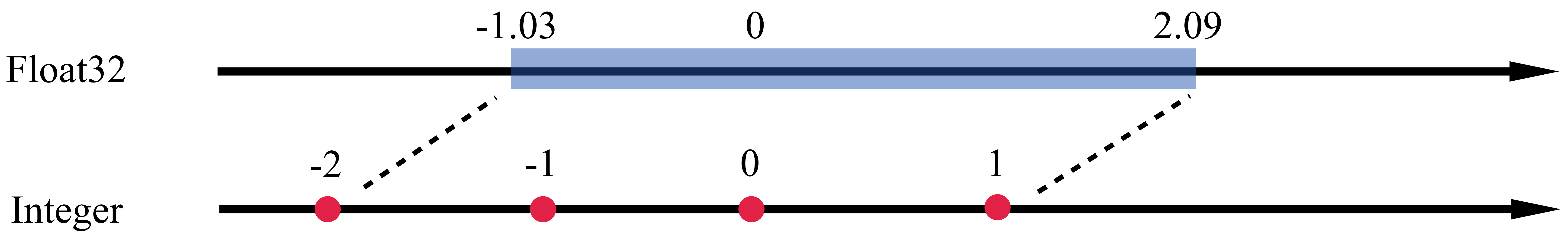
Uniform quantization

Float 32 [2.09, -0.98, 1.48, 0.09, -0.05, 1.92, -1.03, 1.53, 0, 2.12]

Quantization ↓

([1, -2, 0, -1, -1, 1, -2, 0, -1, 1] - (-1)) * 1.04

Quantized 2-bit parameters Zero_bias Scaling_factor



Bitwidth	Qmin	Qmax
2	-2	1
3	-4	3
4	-8	7
N	-2^{n-1}	2^{n-1}

Model quantization for PTM storage

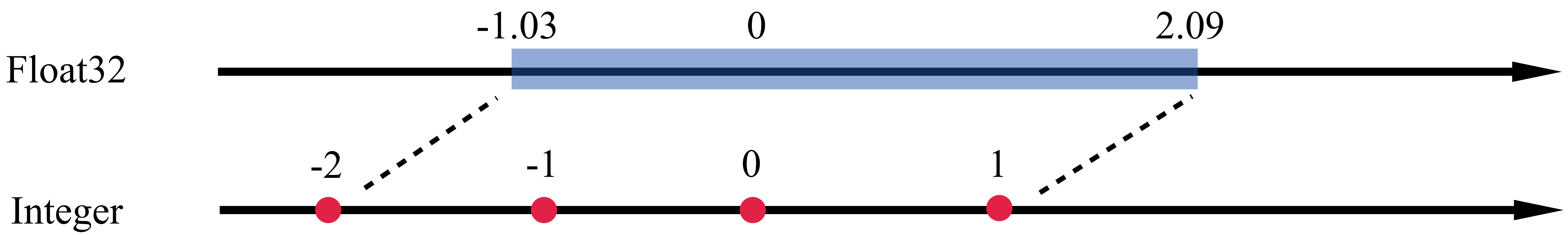
Uniform quantization

Float 32 [2.09, -0.98, 1.48, 0.09, -0.05, 1.92, -1.03, 1.53, 0, 2.12]

Quantization ↓

([1, -2, 0, -1, -1, 1, -2, 0, -1, 1] - (-1)) * 1.04

Quantized 2-bit parameters Zero_bias Scaling_factor



Rmax = (Qmax - Z) * Scaling
Rmin = (Qmin - Z) * Scaling

Scaling = (Rmax-Rmin) / (Qmax - Qmin)
Z = Round_int(Qmax - (Rmax/Scaling))

Bitwidth	**Qmin**	**Qmax**
2	-2	1
3	-4	3
4	-8	7
N	-2^{n-1}	2^{n-1}

Model quantization for PTM storage

Uniform quantization

Float 32 [2.09, -0.98, 1.48, 0.09, -0.05, 1.92, -1.03, 1.53, 0, 2.12]

$$\Big\downarrow \text{Quantization}$$

$$(\underbrace{[\ 1,\ -2,\ 0,\ -1,\ -1,\ 1,\ -2,\ 0,\ -1,\ 1\ ]}_{\text{Quantized 2-bit parameters}} - \underbrace{(-1)}_{\text{Zero\_bias}}) * \underbrace{1.04}_{\text{Scaling\_factor}}$$

Question: How to calculate the CR for uniform quantization?

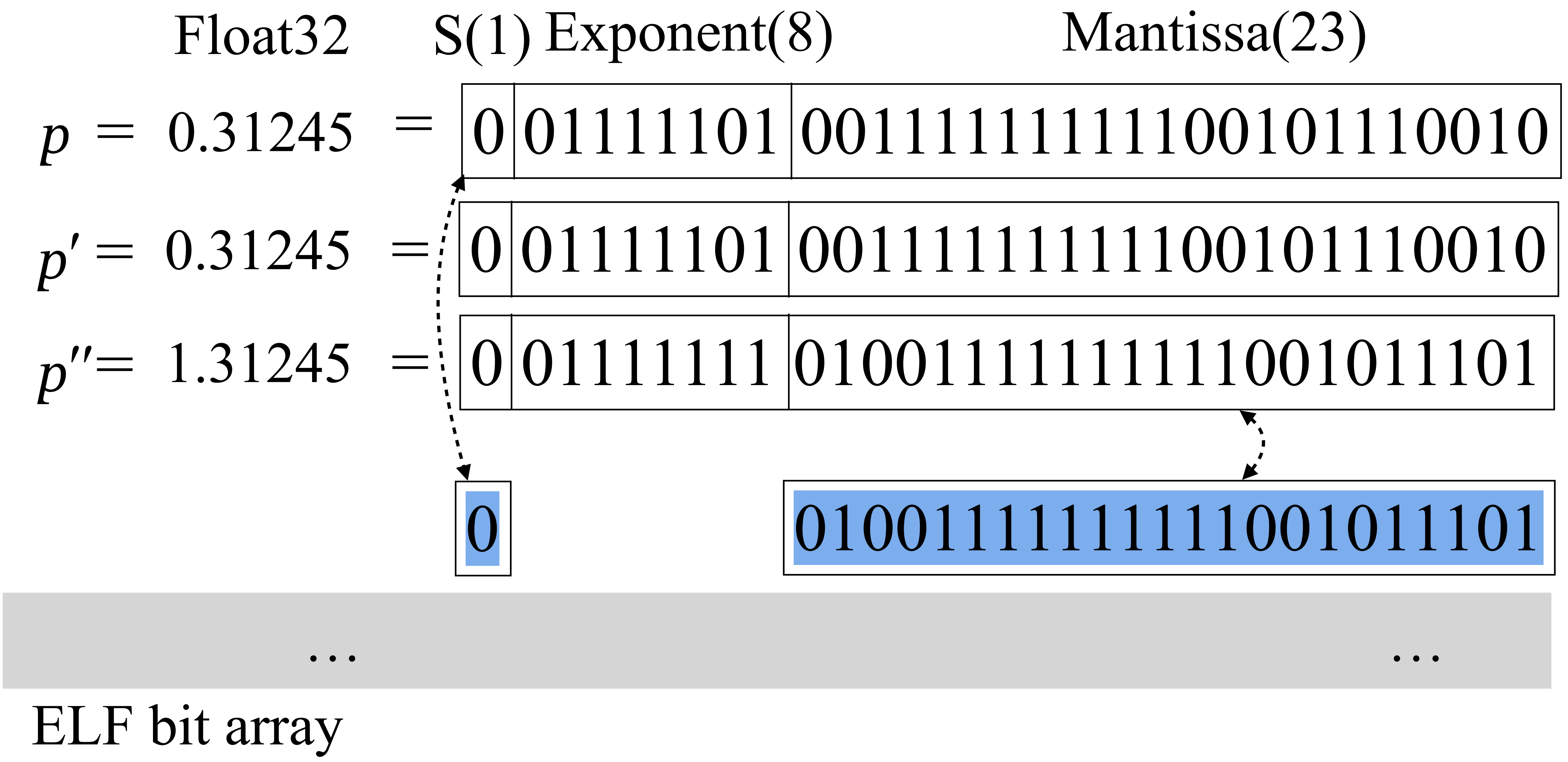
CR = 10 * 32 / (10 * 2 + 2 + 32) = 5.93

Exponent-less Floating-point (ELF) Compression

Float32		Binary Representation S(1)	Exponent(8)	Mantissa(23)
0.31245	=	0	01111101	00111111111100101110010
0.214235	=	0	01111100	10110110110000001101011
-0.0958372	=	1	01111011	10001000100011001001011
0.0001473	=	0	01110010	00110100111010010001011

		S(1)	Exponent(8)	Mantissa(23)
1.31245	=	0	01111111	01001111111111001011101
1.214235	=	0	01111111	00110110110110000001101
-1.0958372	=	1	01111111	00011000100010001100101
1.0001473	=	0	01111111	00000000000010011010100

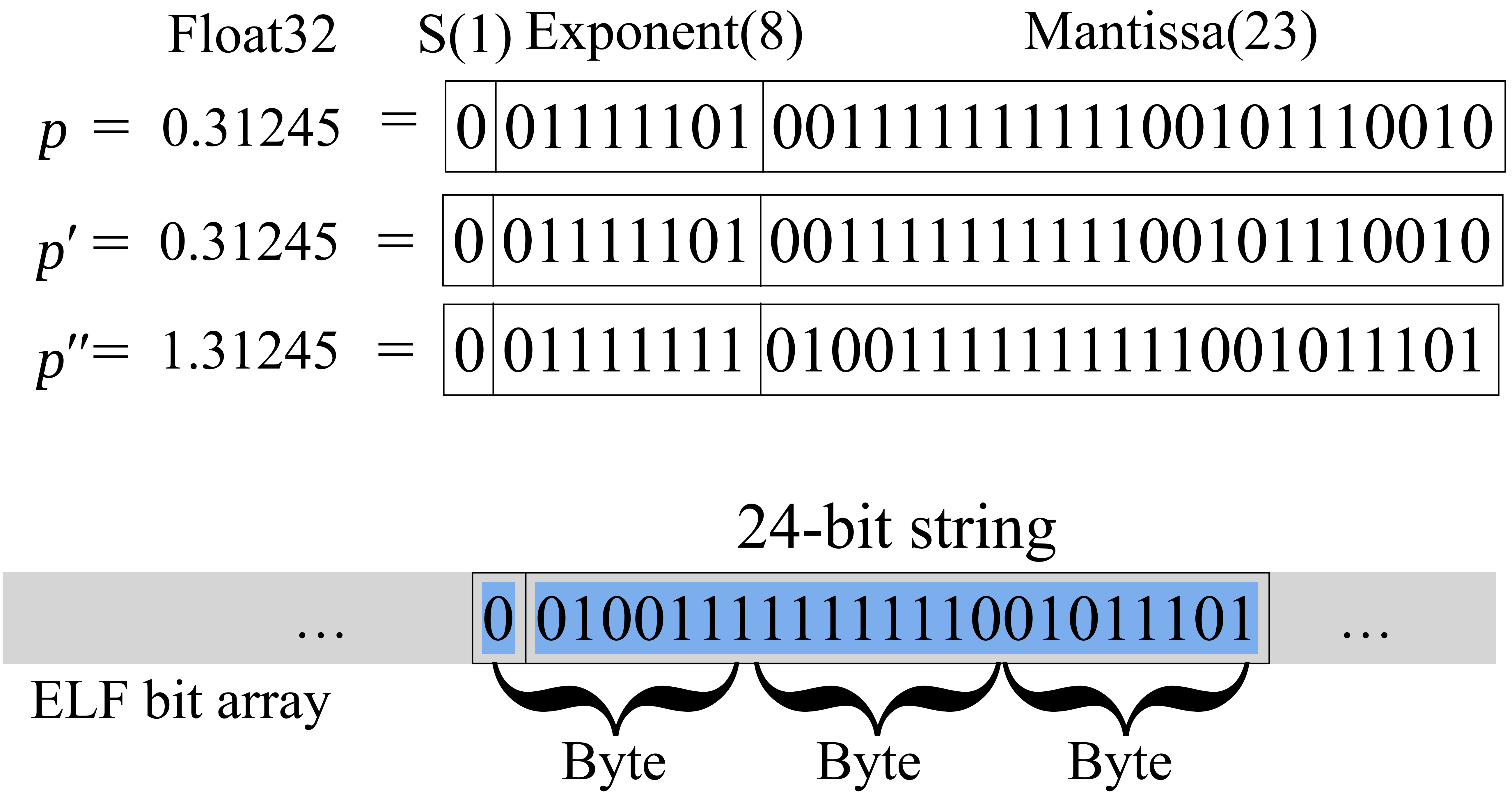
Exponent-less Floating-point (ELF) Compression



ELF Compression

1. Recording the sign bit of $p \in (-1,1)$.
2. Obtaining $p' \in [0,1)$, the absolute value of p.
3. Converting p' to $p'' \in [1,2)$ by adding 1.
4. Recording the mantissa of p''.
5. Appending 24-bit string into the bit array and storing them as 3 bytes.

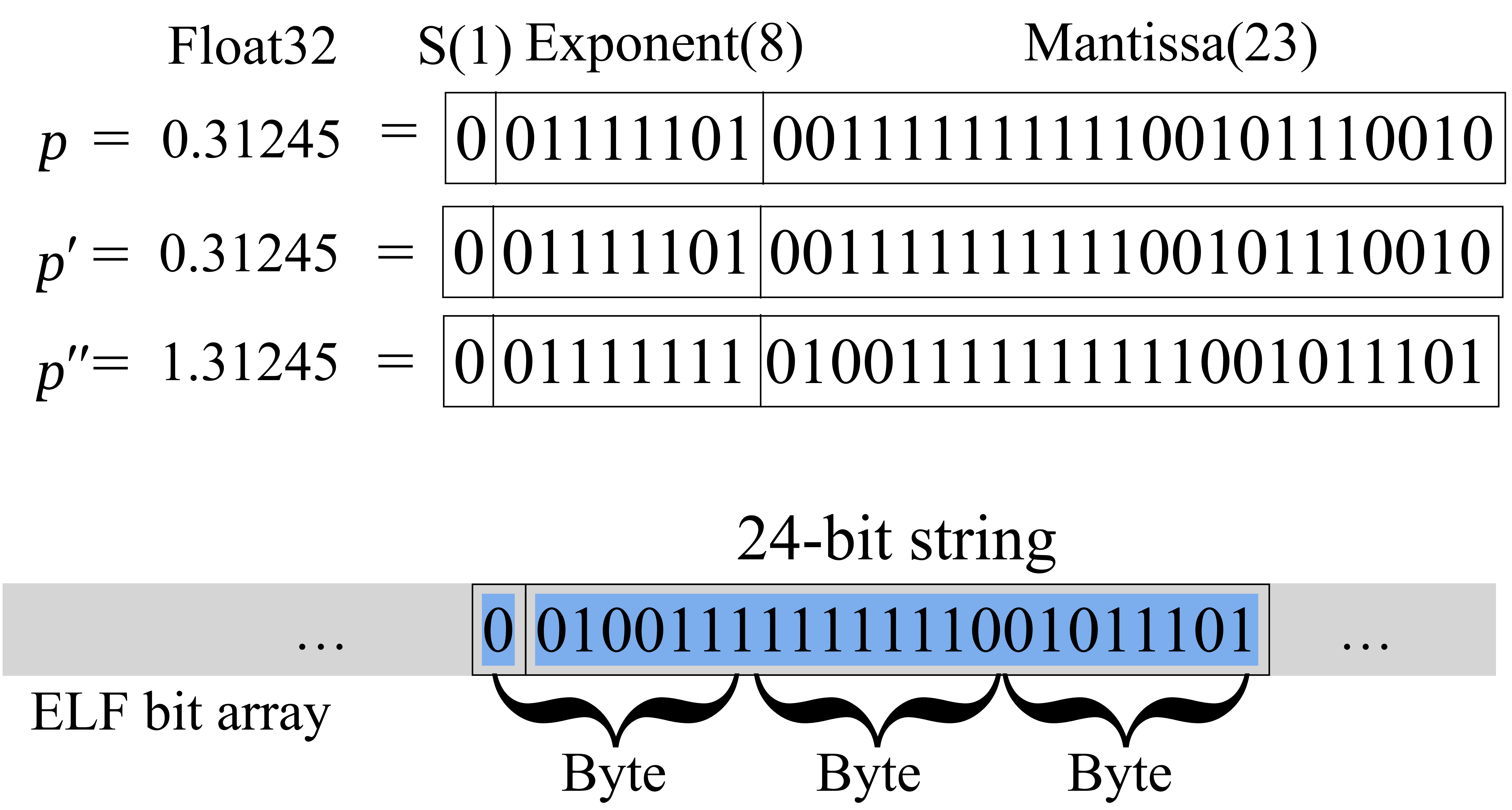
Exponent-less Floating-point (ELF) Compression



ELF Compression

1. Recording the sign bit of $p \in (-1,1)$.
2. Obtaining $p' \in [0,1)$, the absolute value of p.
3. Converting p' to $p'' \in [1,2)$ by adding 1.
4. Recording the mantissa of p''.
5. Appending 24-bit string into the bit array and storing them as 3 bytes.

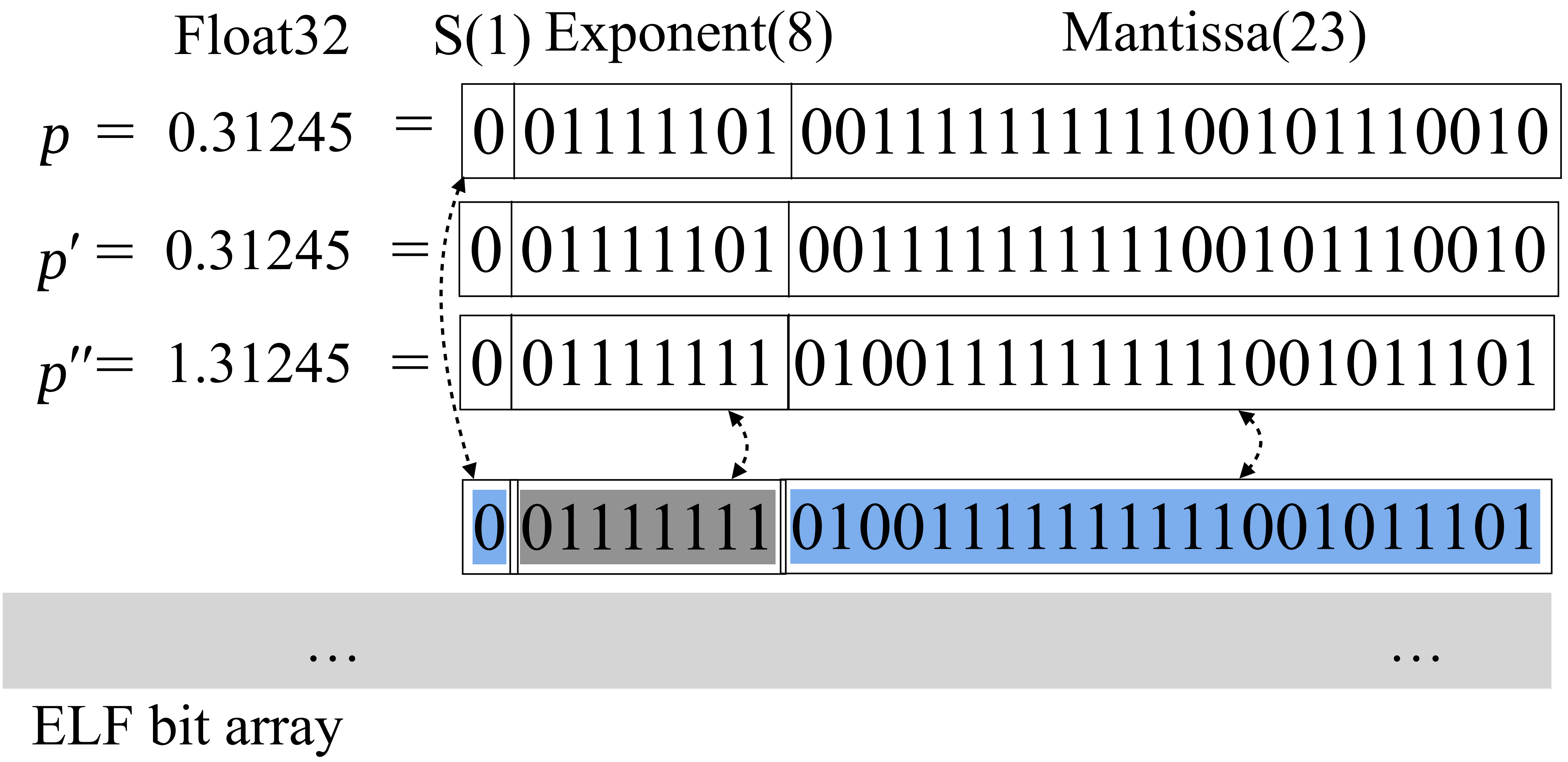
Exponent-less Floating-point (ELF) Compression



ELF Decompression

1. Obtaining the 24-bit string from bit array.
2. Bring the fixed exponent back.
3. Constructing $p'' \in [1,2)$ from the given bits.
4. Calculating $p' \in [0,1)$ from p''.
5. Adding back the sign to restore the $p \in (-1,1)$.

Exponent-less Floating-point (ELF) Compression

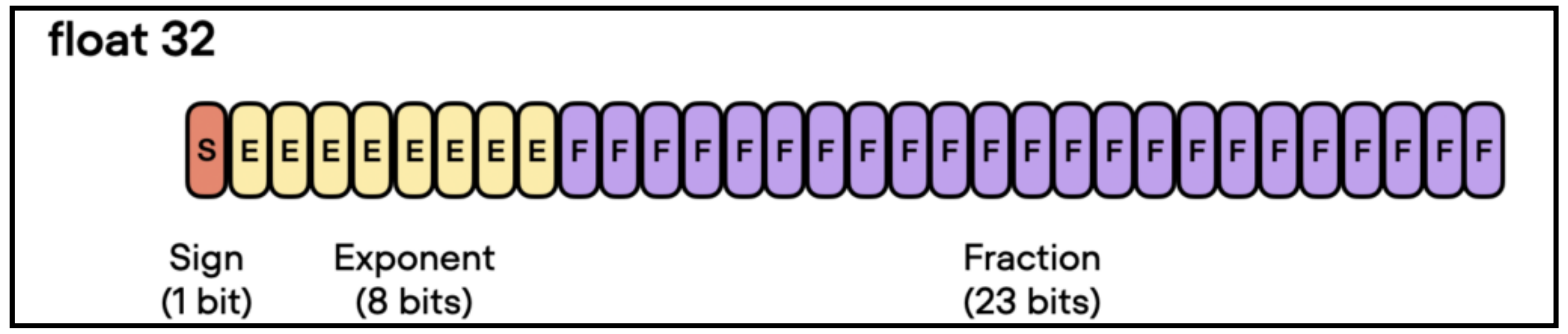


ELF Decompression

1. Obtaining the 24-bit string from bit array.
2. Bring the fixed exponent back.
3. Constructing $p'' \in [1,2)$ from the given bits.
4. Calculating $p' \in [0,1)$ from p''.
5. Adding back the sign to restore the $p \in (-1,1)$.

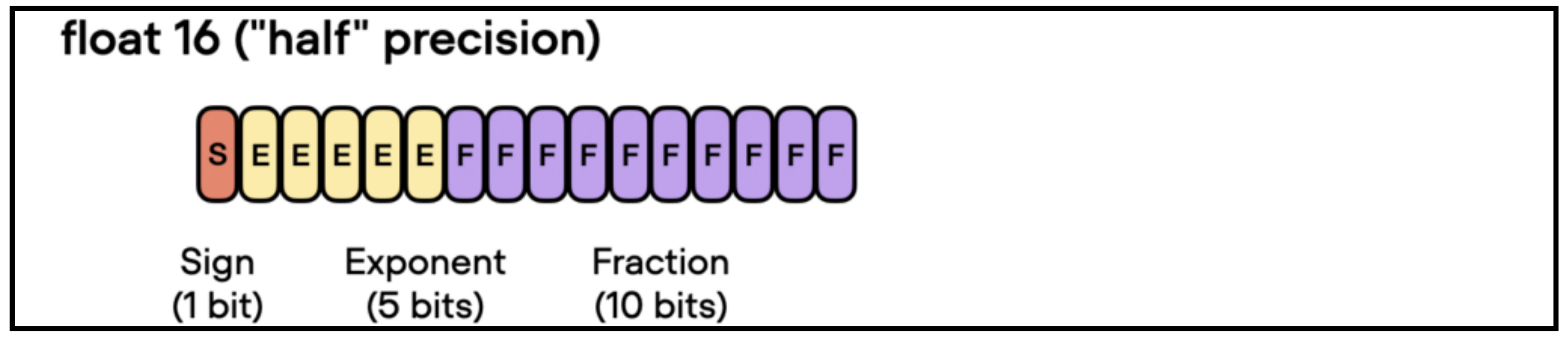
Exponent-less Floating-point (ELF) Compression

Question: What is the ELF CR for floating-32 points?



CR = 32 / (32-8) = 1.3333

Question: What is the ELF CR for floating-16 points?



CR = 16 / (16-5) = 1.4545

Summary

Today, we reviewed and discussed

- The observations and insights from PTM analysis:
  - Most parameters are **float32** numbers within the range **(-1, 1)**.

- The **binary representations** for floating-point numbers.

- Model pruning:
  - **Global magnitude pruning**

- Model quantization:
  - **Uniform quantization**
  - Non-uniform quantization

- **Exponent-less floating-point(ELF) compression**

References

1. Su, Zhaoyuan, et al. "Everything You Always Wanted to Know About Storage Compressibility of Pre-Trained ML Models but Were Afraid to Ask." *arXiv preprint arXiv:2402.13429* (2024).
2. Han, Song, Huizi Mao, and William J. Dally. "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding." *arXiv preprint arXiv:1510.00149* (2015).
3. Brendan McMahan, Daniel Ramage. "Federated Learning: Collaborative Machine Learning without Centralized Training Data." Google Research blog (2017).
4. Fajardo, Carlos, Oscar M Reyes, and Ana Ramirez. "Seismic data compression using 2D lifting-wavelet algorithms." *Ingeniería y Ciencia* 11.21 (2015): 221-238.
5. Abjieet Pujara, "Concept of AlexNet:Convolutional Neural Network", https://medium.com/analytics-vidhya/concept-of-alexnet-convolutional-neural-network-6e73b4f9ee30 (2020).
6. Sebastian Raschka, "Accelerating Large Language Models with Mixed-Precision Techniques", https://lightning.ai/pages/community/tutorial/accelerating-large-language-models-with-mixed-precision-techniques/ (2023).