# ZipLLM: Efficient LLM Storage via Model-aware Synergistic Data Dedup and Compression



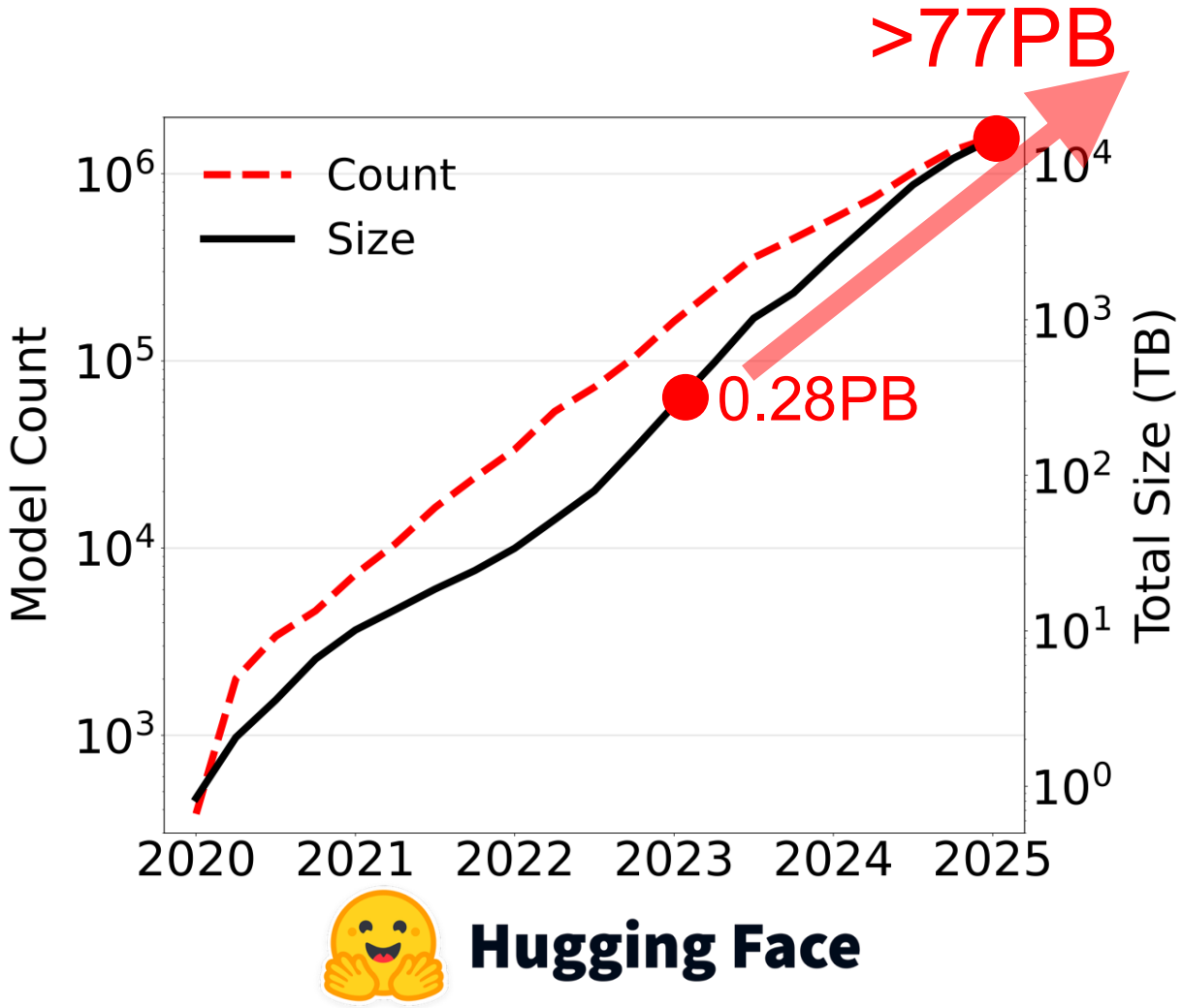


Zirui Wang[1], Tingfeng Lan[1], Zhaoyuan Su[1], Juncheng Yang[2], Yue Cheng[1]

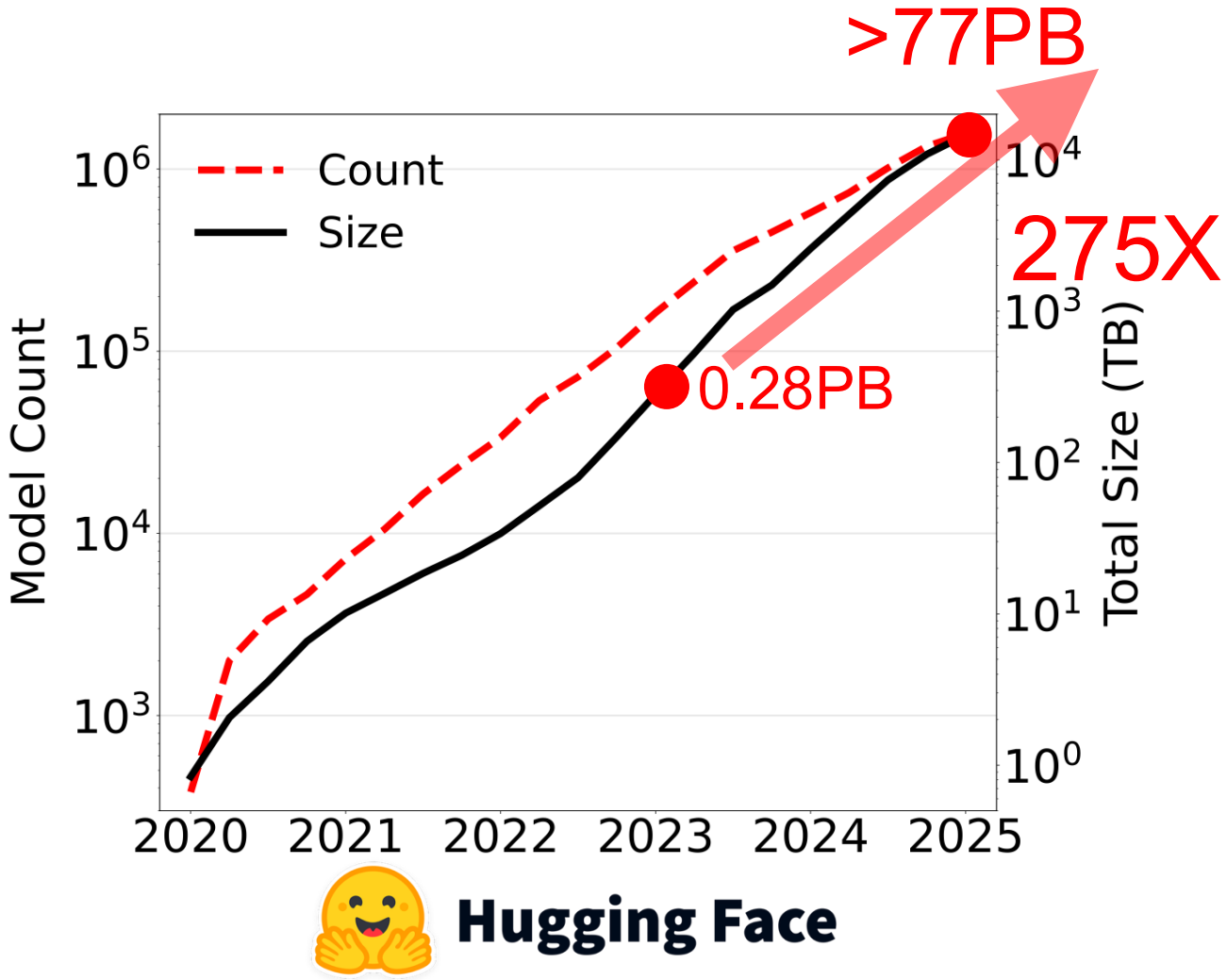
[1] University of Virginia

[2] Harvard University

# Model hub storage is growing unsustainably



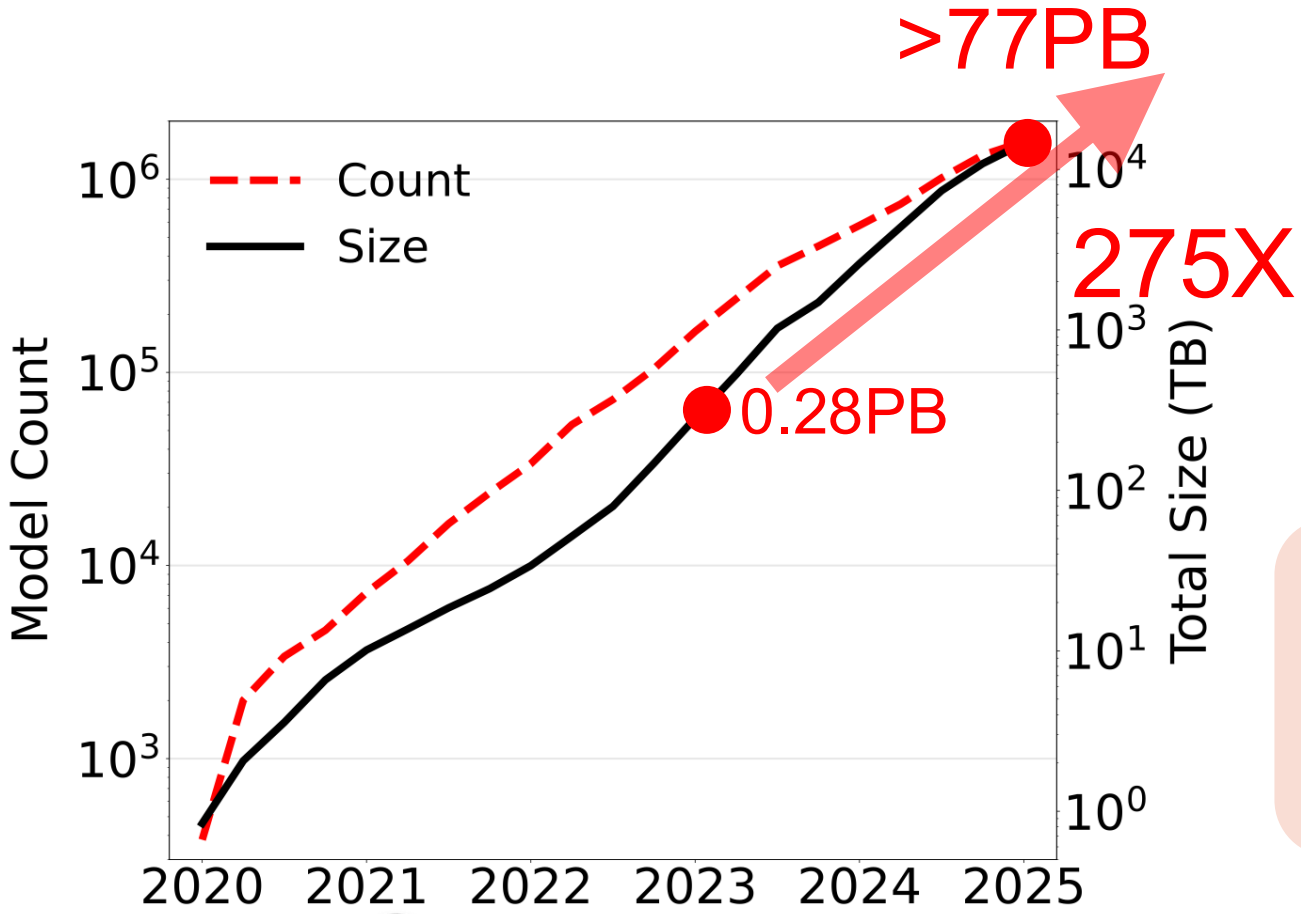
# Model hub storage is growing unsustainably




**Exponential growth**

1EB model storage by 2027

# Model hub storage is growing unsustainably




**Exponential growth**

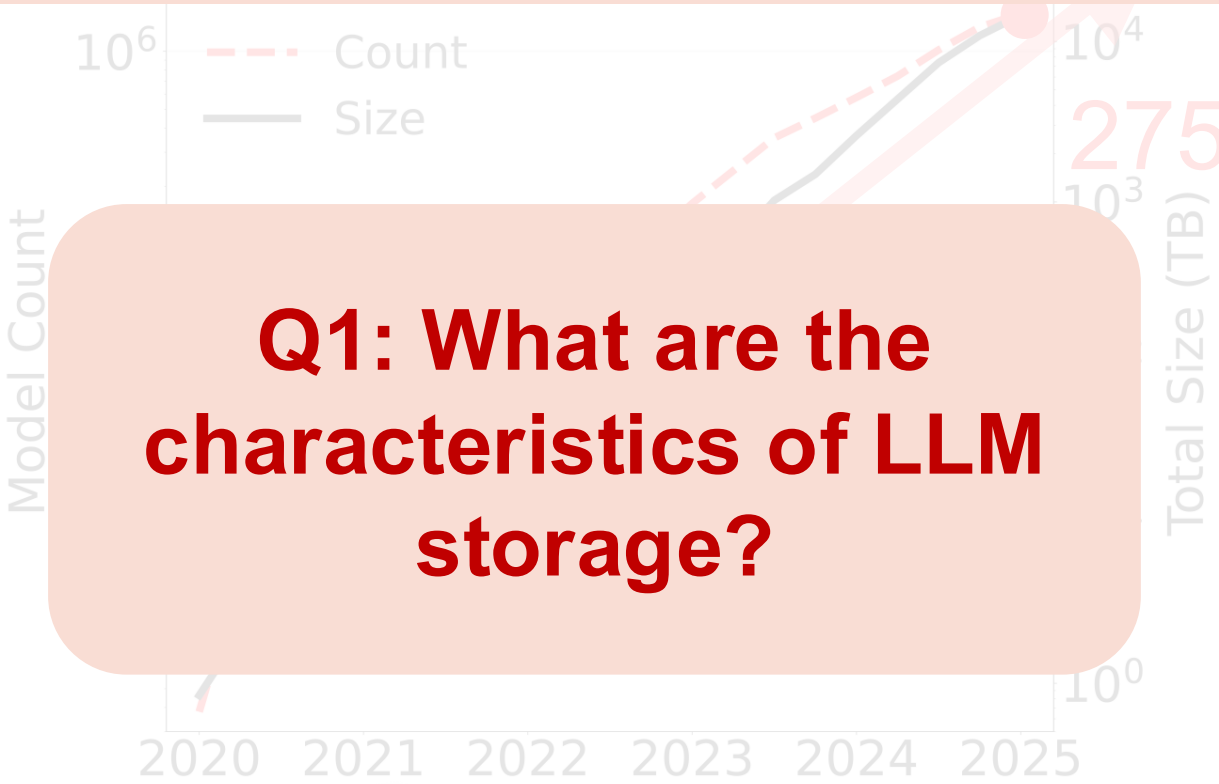
1EB model storage by 2027

**Model users constantly upload LLMs to model hubs**

**Model trainers frequently store massive checkpoints across steps/runs/versions**

# The AI model storage crisis

Current storage demands are **unsustainable** for AI models




275X

Hugging Face

Exponential growth

1EB model storage by 2027

**Q1: What are the characteristics of LLM storage?**

Model users constantly upload LLMs to model

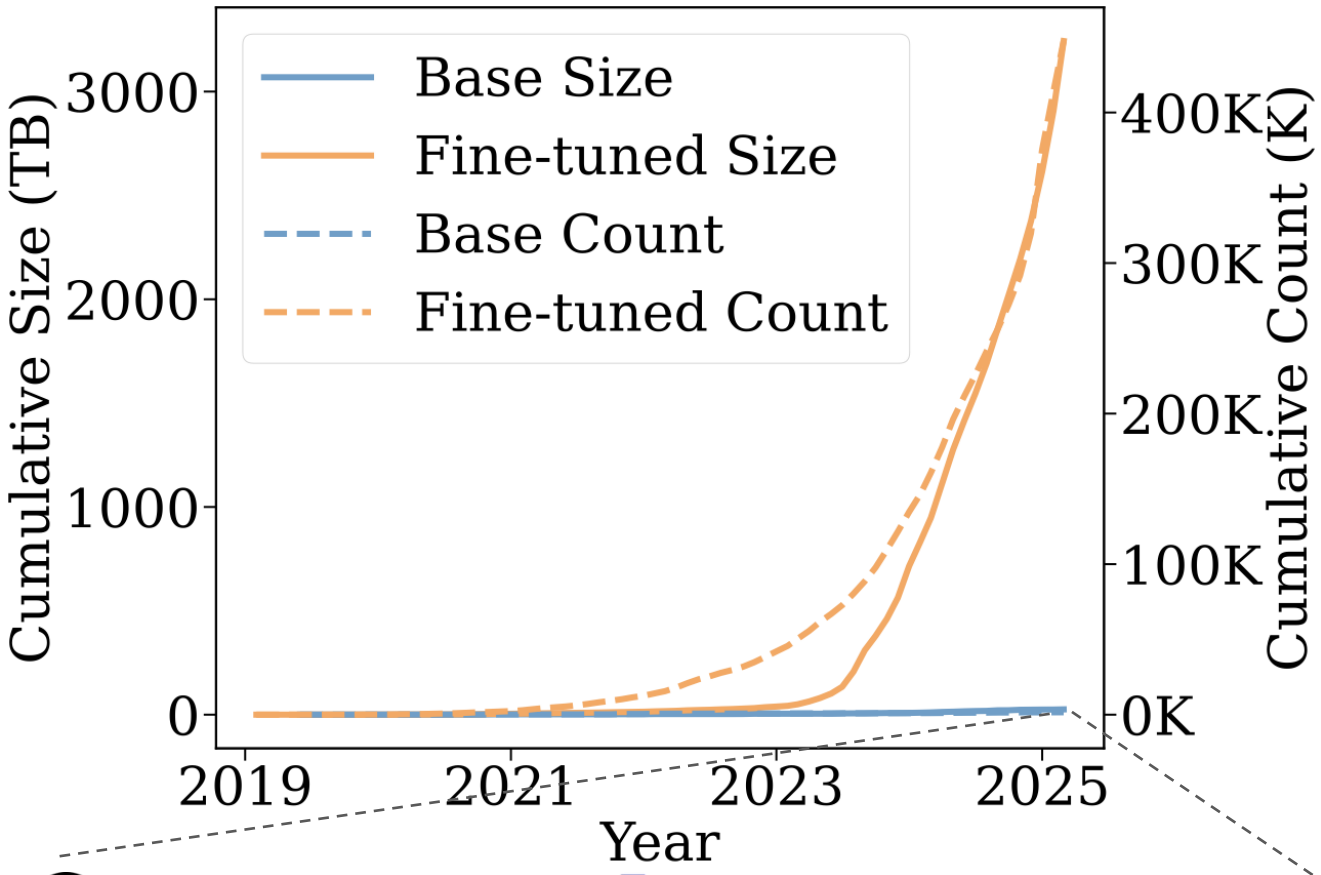
**Q2: Can existing data reduction techniques keep up with the rapid growth in model storage footprint?**

# Characterizing HF model storage

- Collected **all** LLM repos from Hugging Face (HF)
- Conducted the **first-of-its-kind**, large-scale field characterization study focusing on LLM storage

- **Observation 1**: Fine-tuned LLMs dominate
- **Observation 2**: Fine-tuned LLMs resemble their base models

# Observation 1: Fine-tuned LLMs dominate




gpt-oss deepseek ...

- Only **20+** open-source LLMs exist!
- Fine-tuned LLMs are growing much faster than base models
- Storage consumption is overwhelmingly dominated by fine-tuned LLMs

# Observation 1: Fine-tuned LLMs dominate

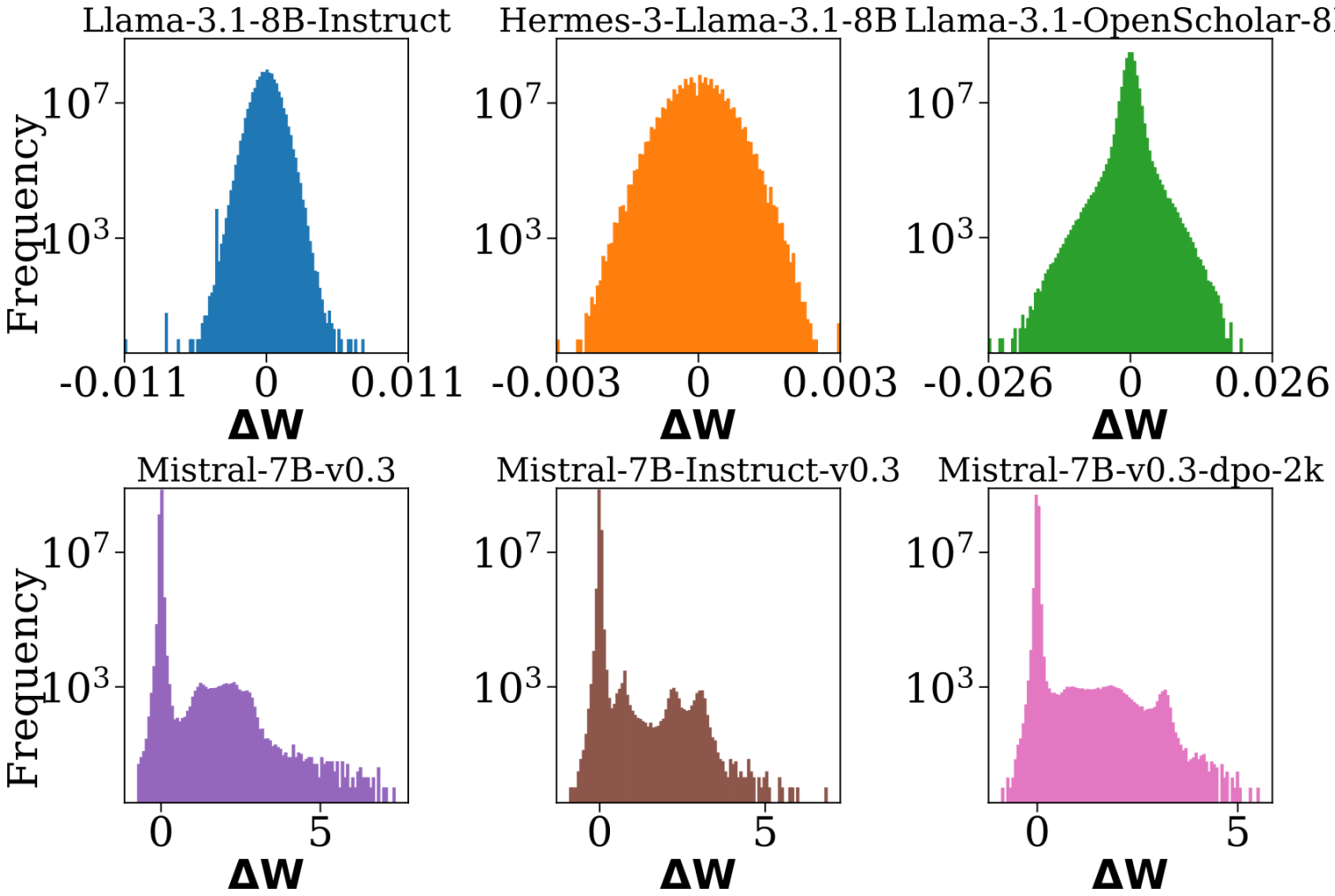

- Only **20+** open-source LLMs exist!

**Insight 1:** Fine-tuned LLMs dominate storage cost—optimization should focus here

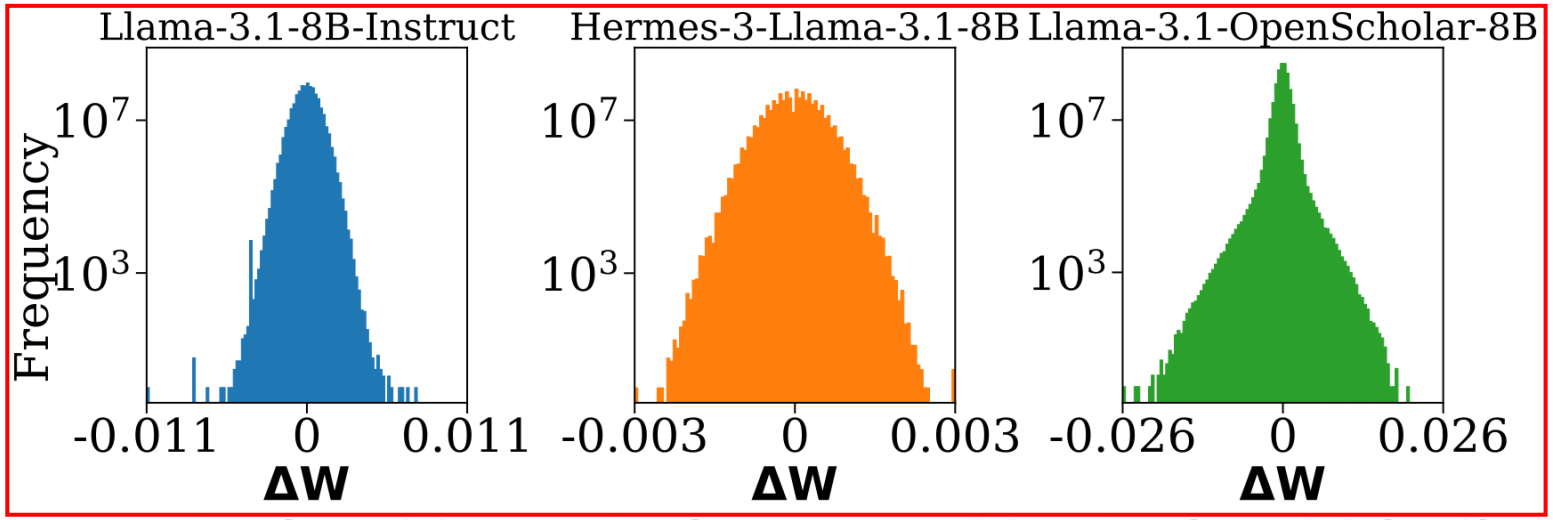
- Storage consumption is overwhelmingly dominated by fine-tuned LLMs

gpt-oss deepseek …

# Observation 2: Fine-tuned LLMs resemble their base models



# Observation 2: Fine-tuned LLMs resemble their base models







**Fine-tuned from Llama-3.1-8B**

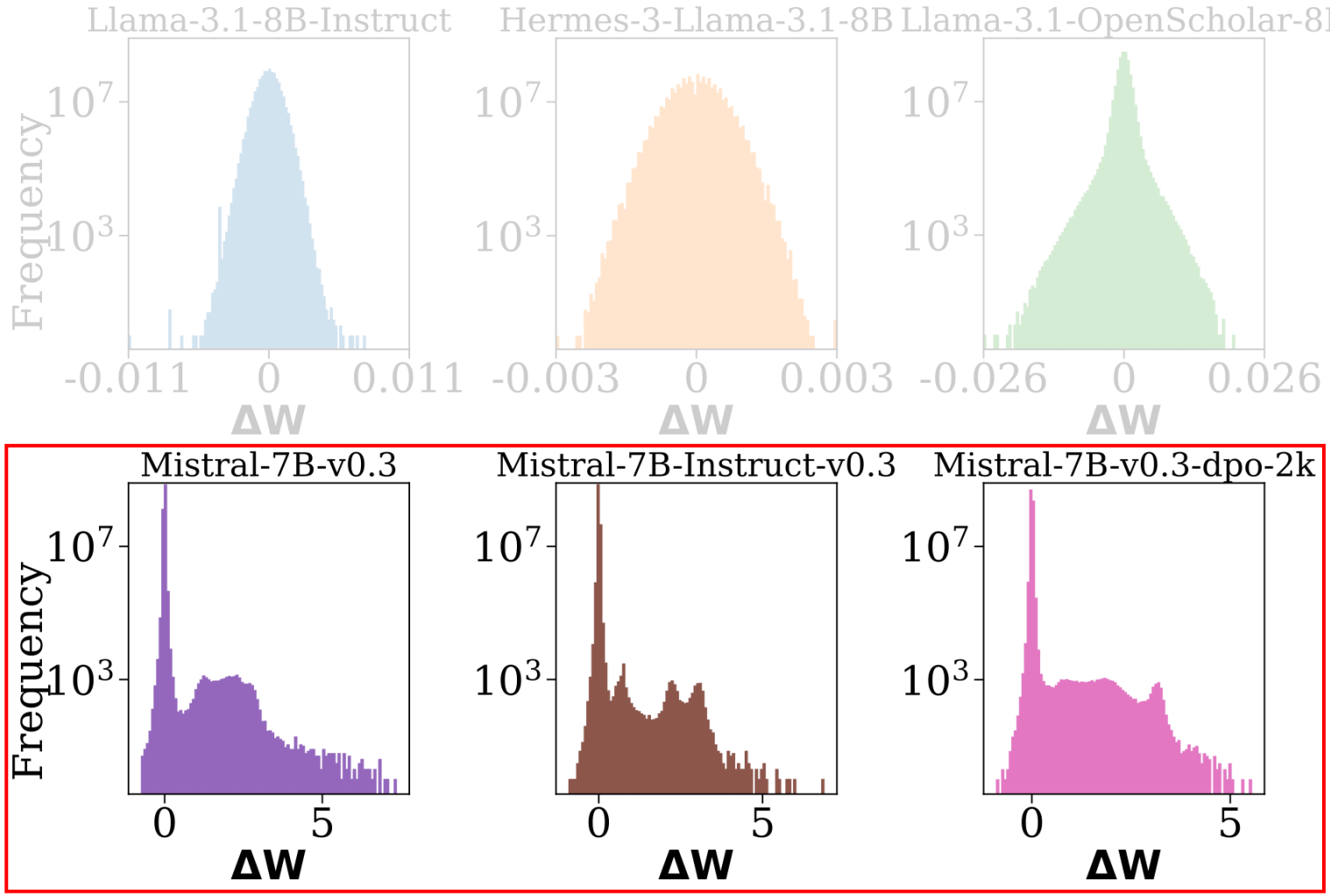
$$\triangle w_i = w_i - \widehat{w_i}$$

Fine-tuned model weight

Base model weight

- Weight deltas $\triangle w_i$ are centered around 0
- Majority of base LLM weights **unchanged** in fine-tuned variants

# Observation 2: Fine-tuned LLMs resemble their base models



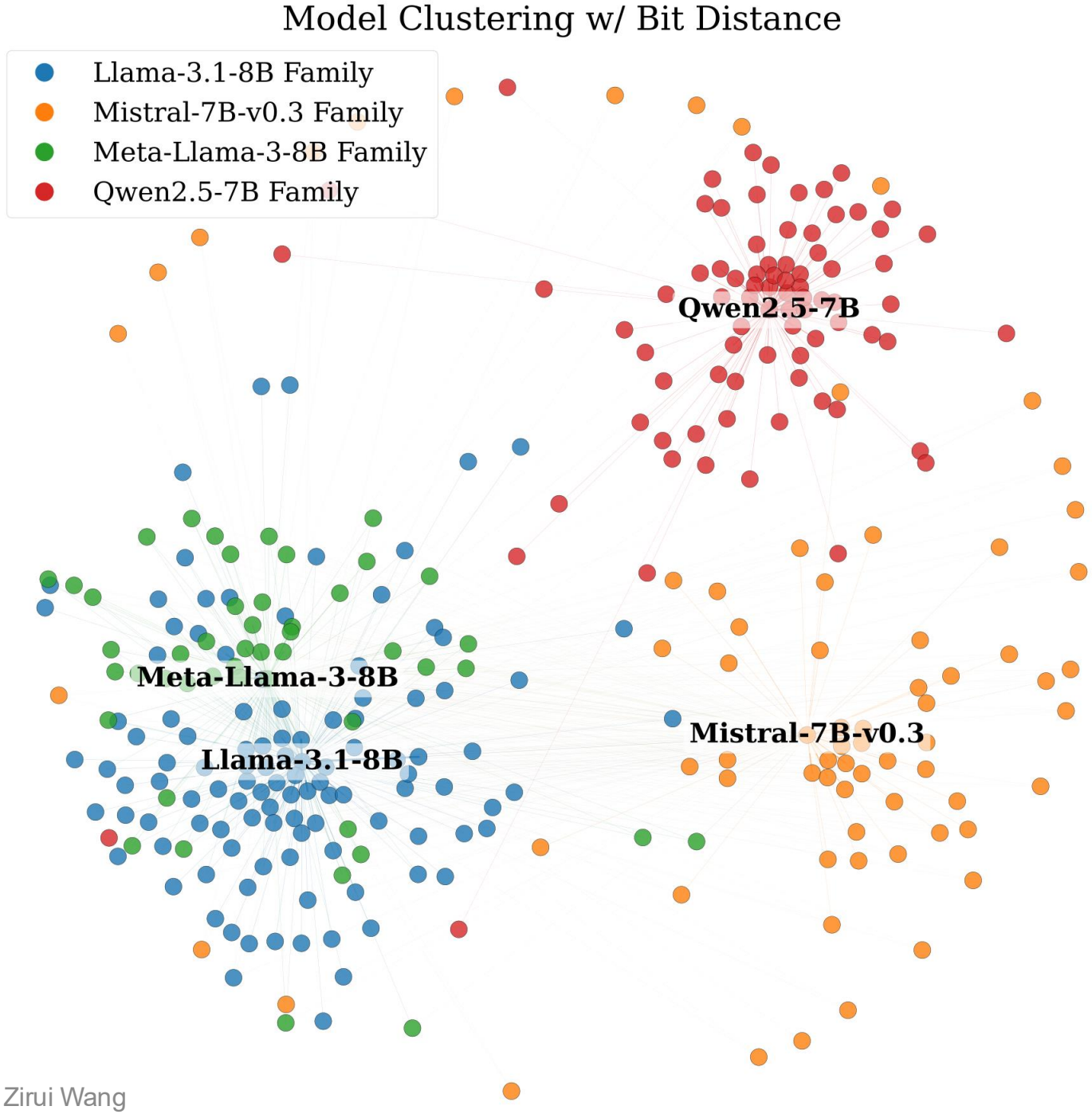

- Weight deltas from different LLM families are much wider

**Fine-tuned from Mistral-7B-v0.3**

$$\triangle w_i = w_i - \widehat{w_i}$$

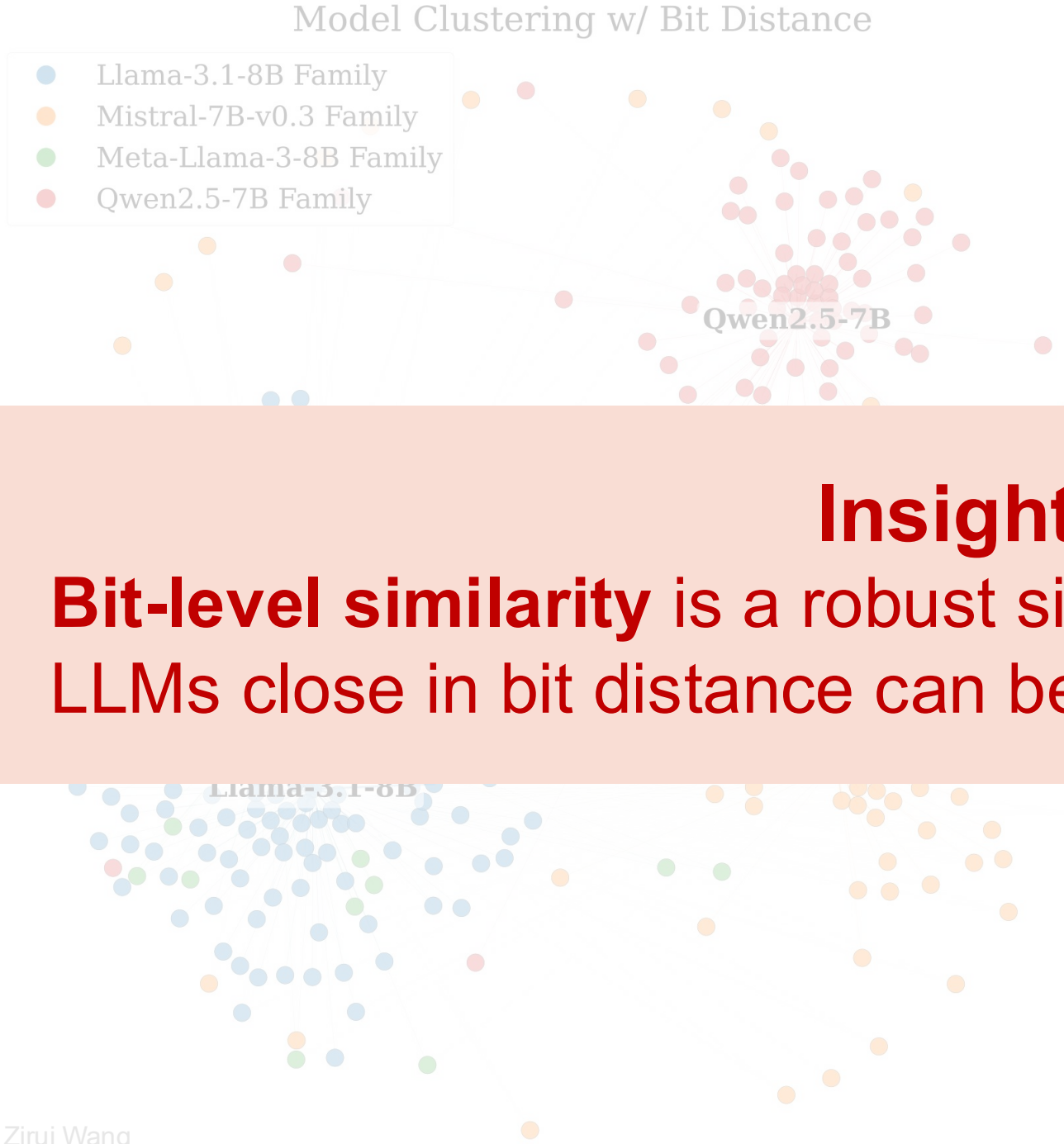
Fine-tuned model weight

**Llama-3.1-8B** base model weight




# LLM family graph

- **Bit distance:** average number of differing bits between two models
- **Within-family:** LLMs cluster tightly around base
- **Cross-family:** clear separation (closely related families cluster near each other)




# LLM family graph

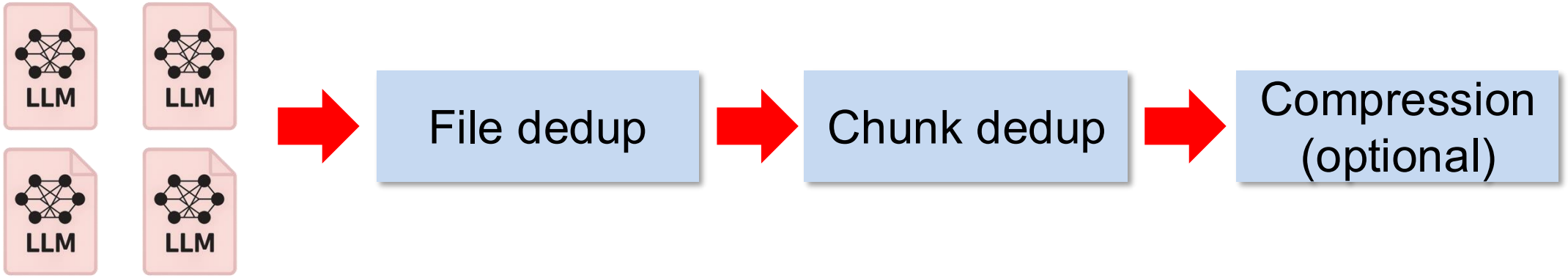
- **Bit distance:** average number of differing bits between two models

## Insight 2:

- **Bit-level similarity** is a robust signal for LLM family clustering
- LLMs close in bit distance can benefit from **delta compression**

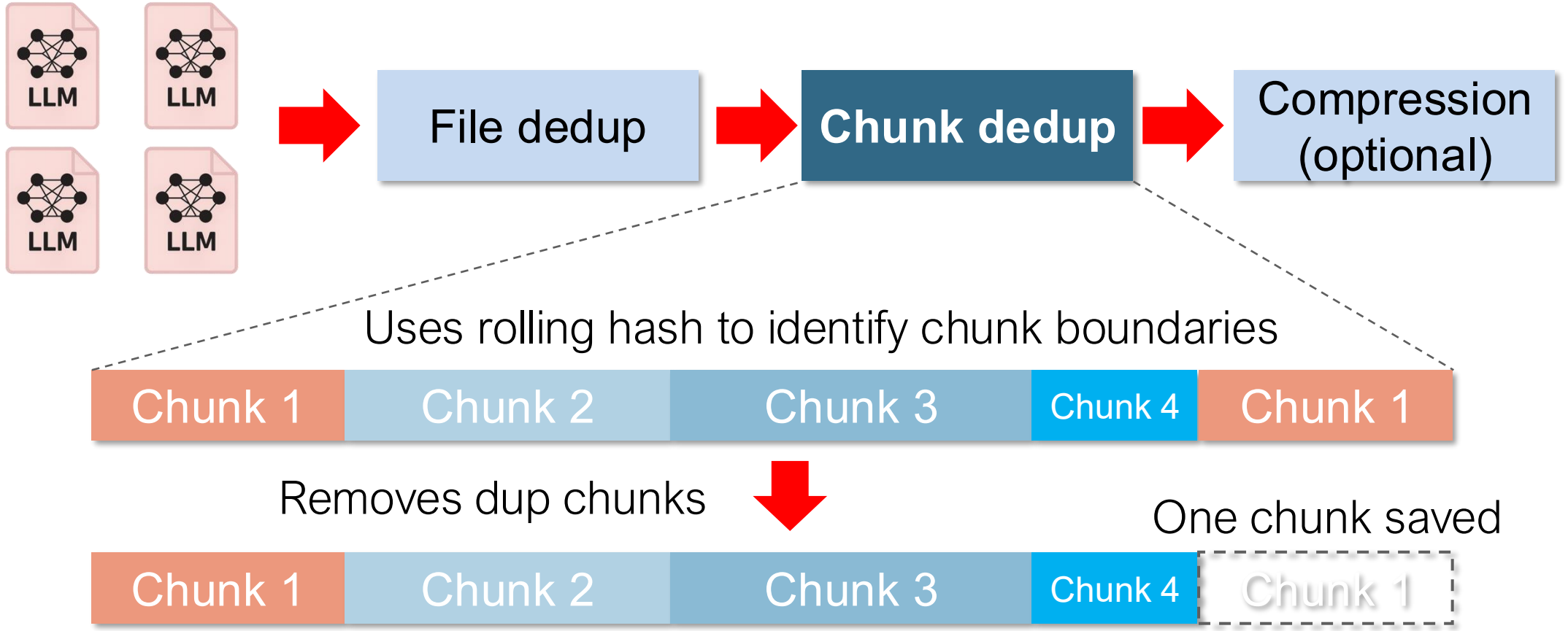
- **Cross-family:** clear separation (closely related families cluster near each other)

# Hugging Face's storage reduction pipeline



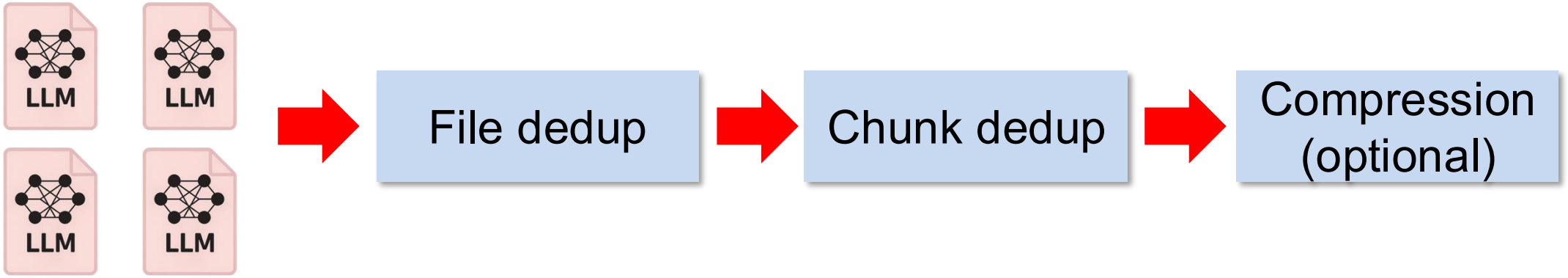

- **File dedup:** Removes identical model files across repos
- **Chunk dedup** (content-defined chunking): Splits files into variable-sized chunks by content, detects and removes duplicate chunks
- **Compression** (optional): Applies lossless compression (e.g., zipnn, zstd) on remaining data

# How content-defined chunking dedup works?



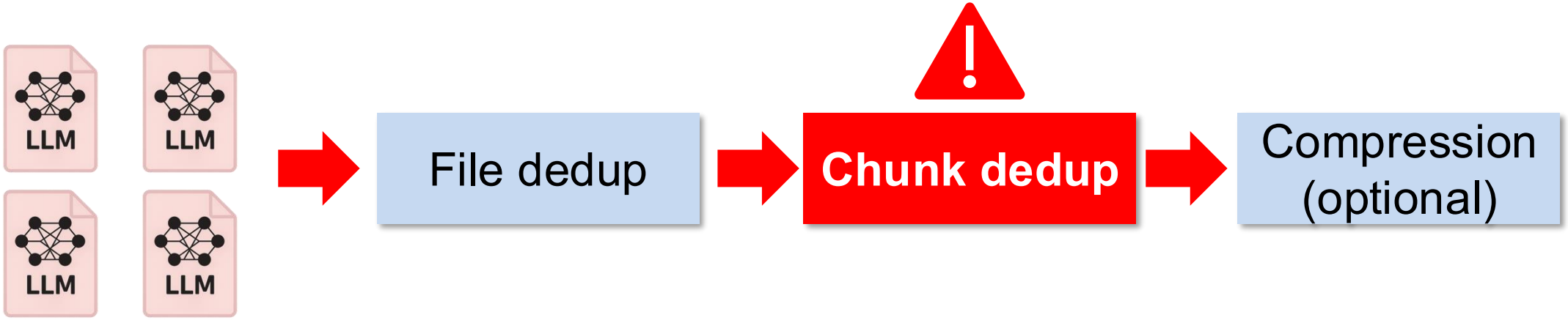

- Splits files into variable-sized chunks by content
- Identifies and removes duplicate chunks

# Hugging Face's storage reduction pipeline




- **File dedup:** Removes identical model files across repos
- **Chunk dedup** (content-defined chunking): Splits files into variable-sized chunks by content, detects and removes duplicate chunks
- **Compression** (optional): Applies lossless compression (e.g., zipnn, zstd) on remaining data

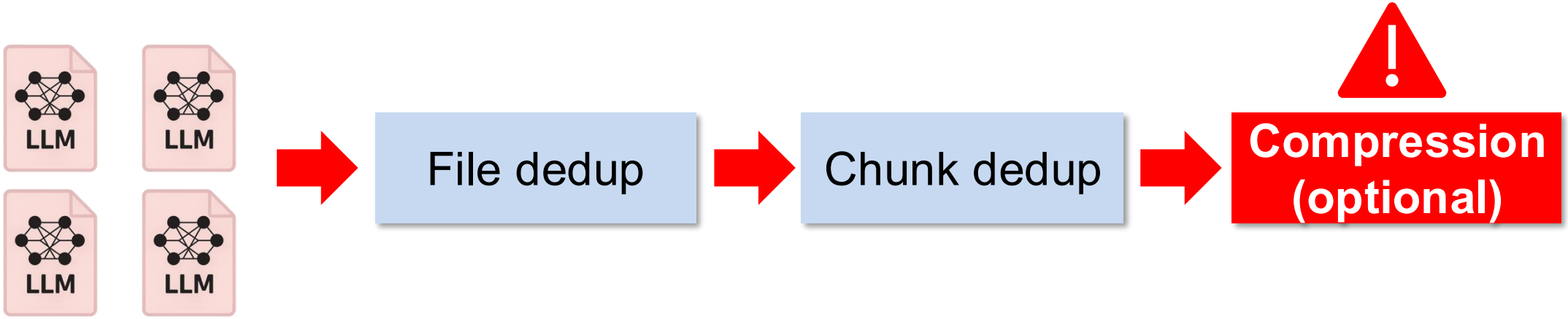
# Problem 1: CDC is inefficient




- **Metadata explosion:** Billions of small chunks
  - 1GB file ≈ 16k chunks, huge indices
- **Costly retrieval:** Lookups across massive chunk indices make reads slow
- **Sequential ingestion:** No parallelism within a file

* https://huggingface.co/blog/from-chunks-to-blocks

# Problem 2: Existing compression is not LLM-aware



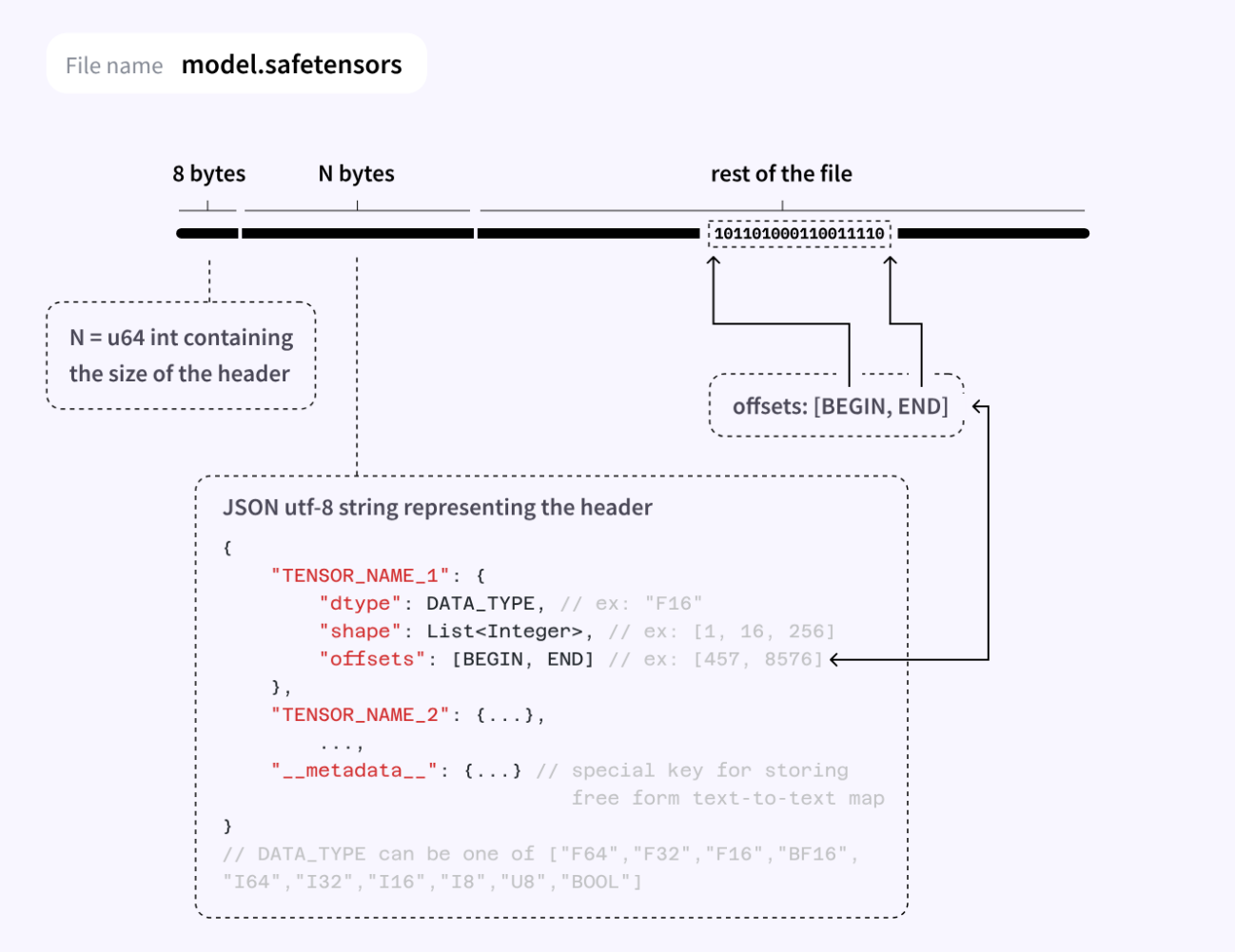

- **Targeting a single LLM/file:** Compresses each model/file separately and independently, ignoring cross-LLM redundancy
- **Byte-oriented:** Ignores float/tensor structures in model weight
- **Limited effect:** Typically, only **10-20%** data reduction ratio

# ZipLLM redesigns model hub storage pipeline

Tensor dedup → Dedup by tensors, not chunks

BitX → LLM-aware delta compression

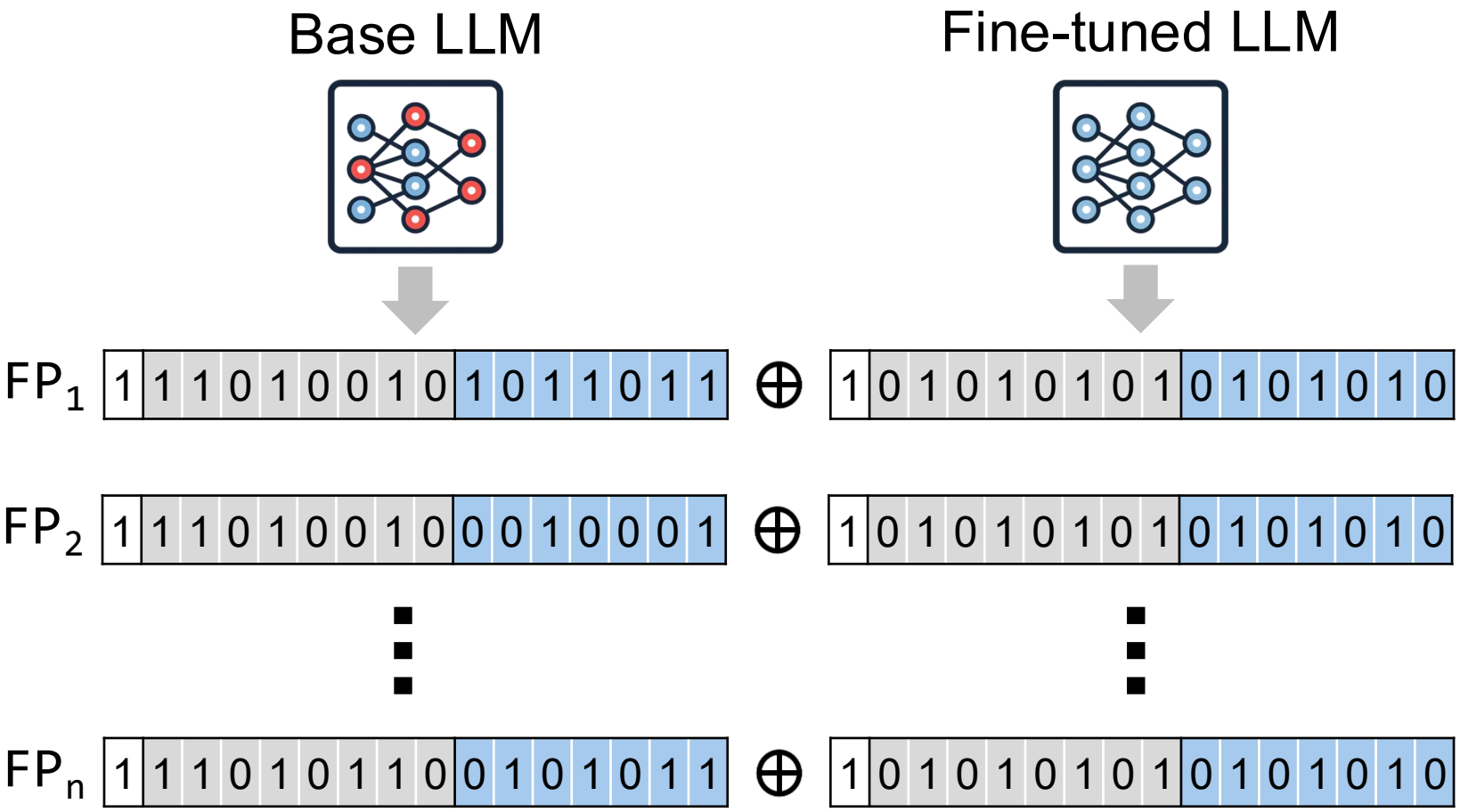
# Idea 1: Tensor dedup




- Leverages tensor boundaries from `safetensors`/`GGUF` formats
- Significantly reduced metadata size
  - Three orders of magnitude smaller indices than chunk-based
- Enables tensor-level parallel dedup
  - Higher ingestion throughput

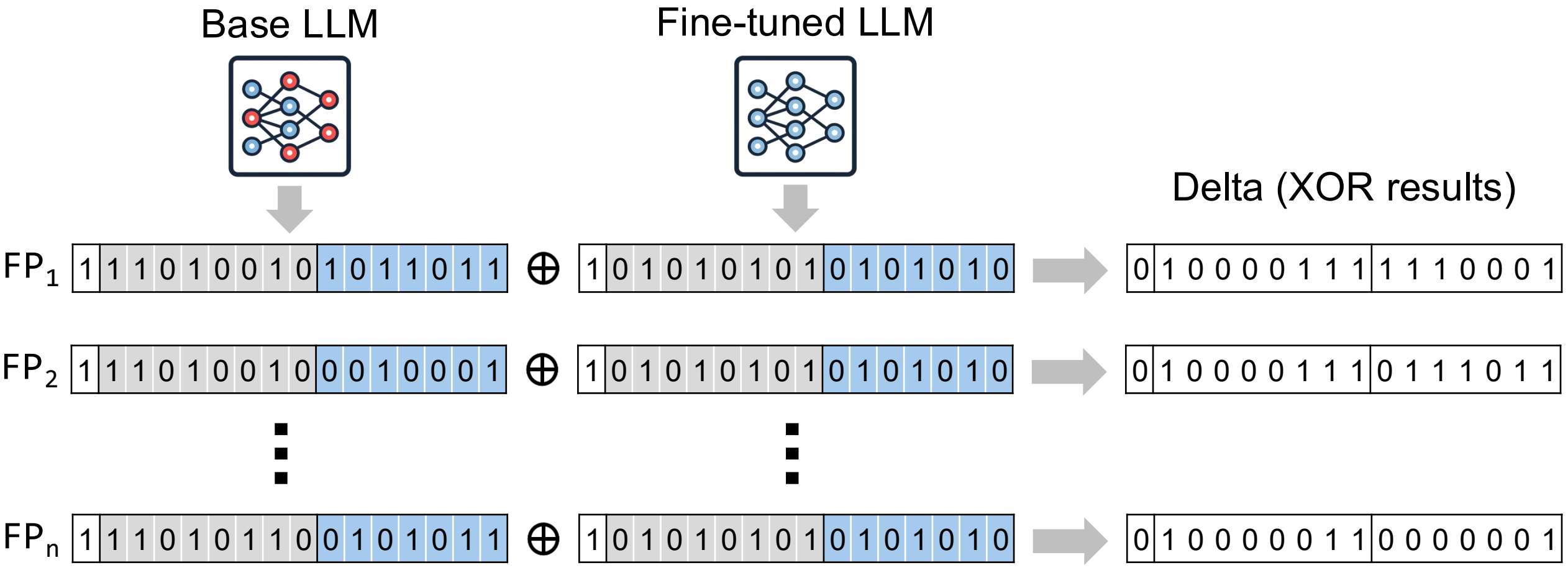
* https://huggingface.co/docs/safetensors/en/index

# Idea 2: BitX – LLM-aware delta compression



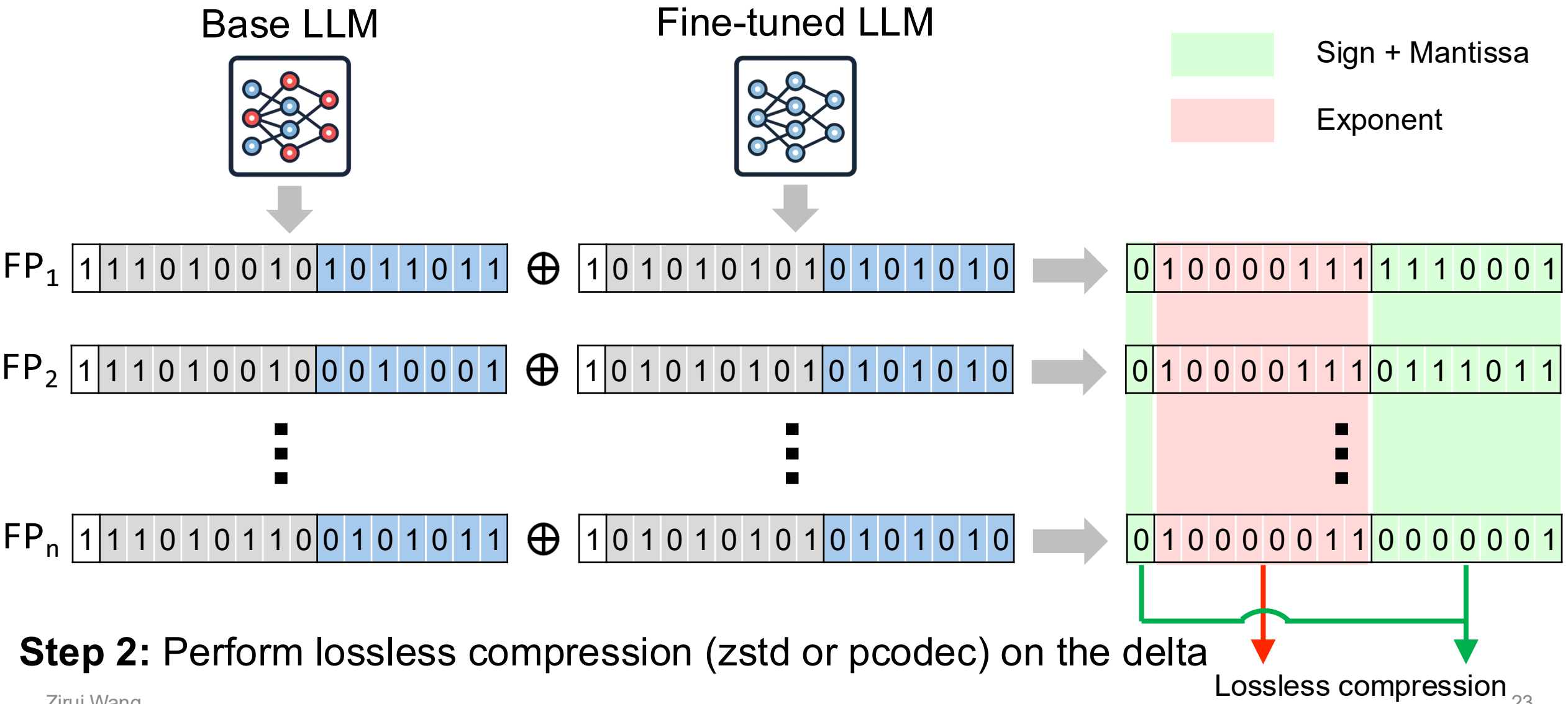

**Step 1:** Perform XOR between base and fine-tuned LLM

# Idea 2: BitX – LLM-aware delta compression



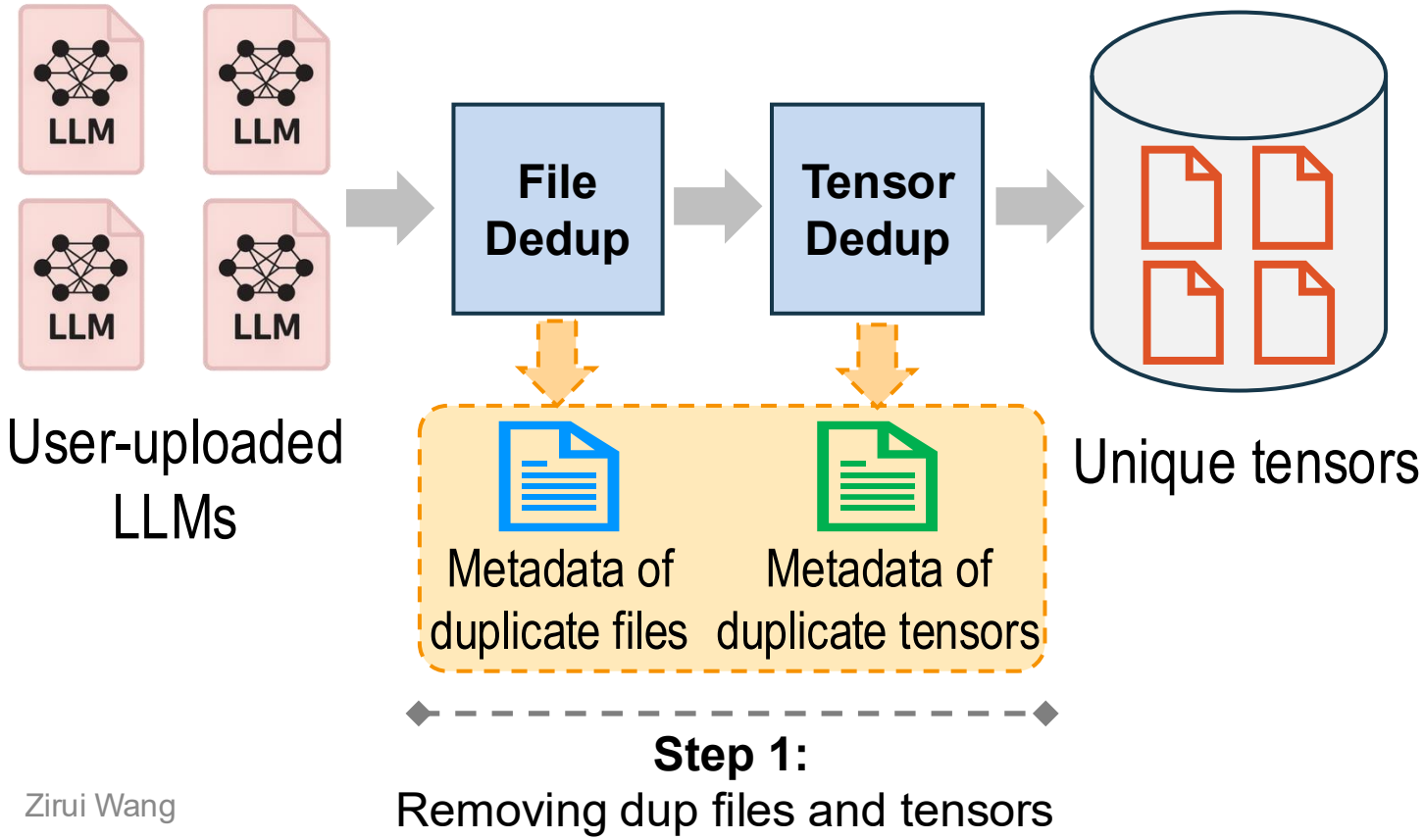

**Step 1:** Perform XOR between base and fine-tuned LLM

# Idea 2: BitX – LLM-aware delta compression

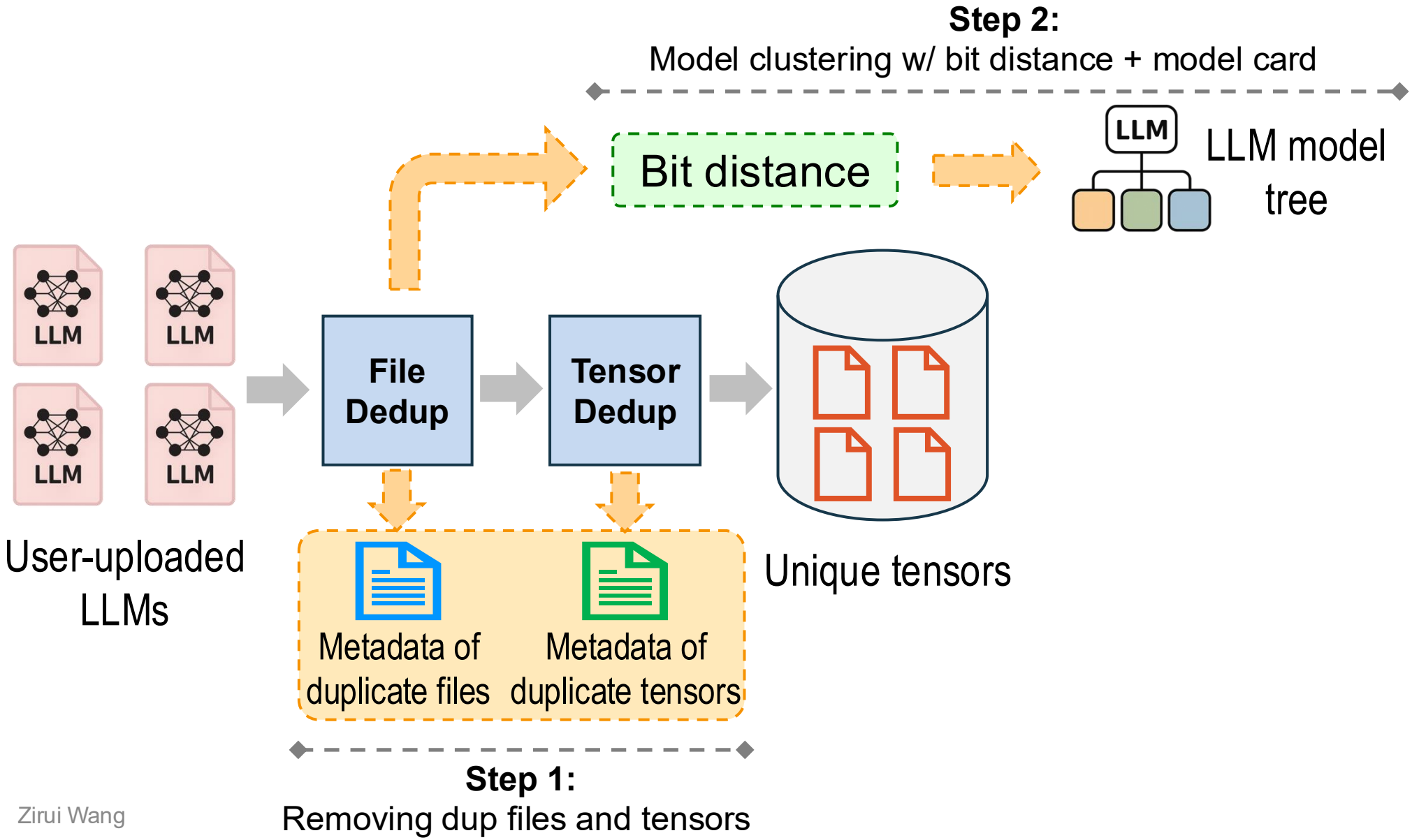



**Step 2:** Perform lossless compression (zstd or pcodec) on the delta

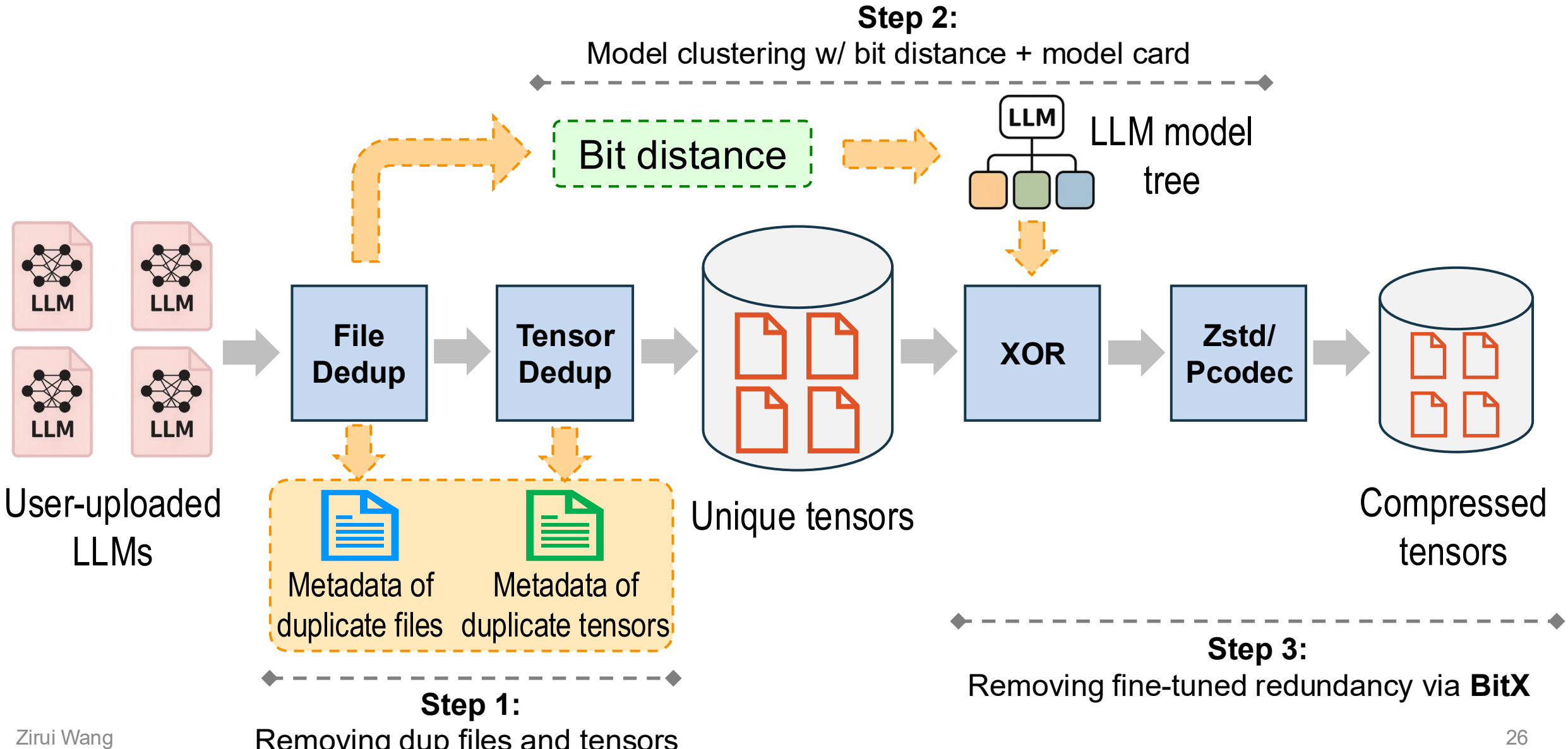
# End-to-end ZipLLM storage reduction pipeline



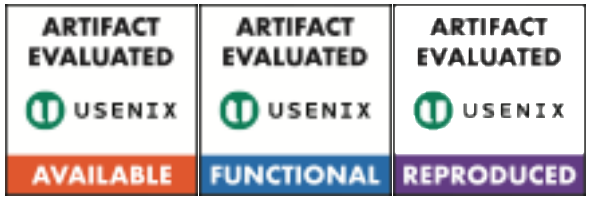
# End-to-end ZipLLM storage reduction pipeline



# End-to-end ZipLLM storage reduction pipeline



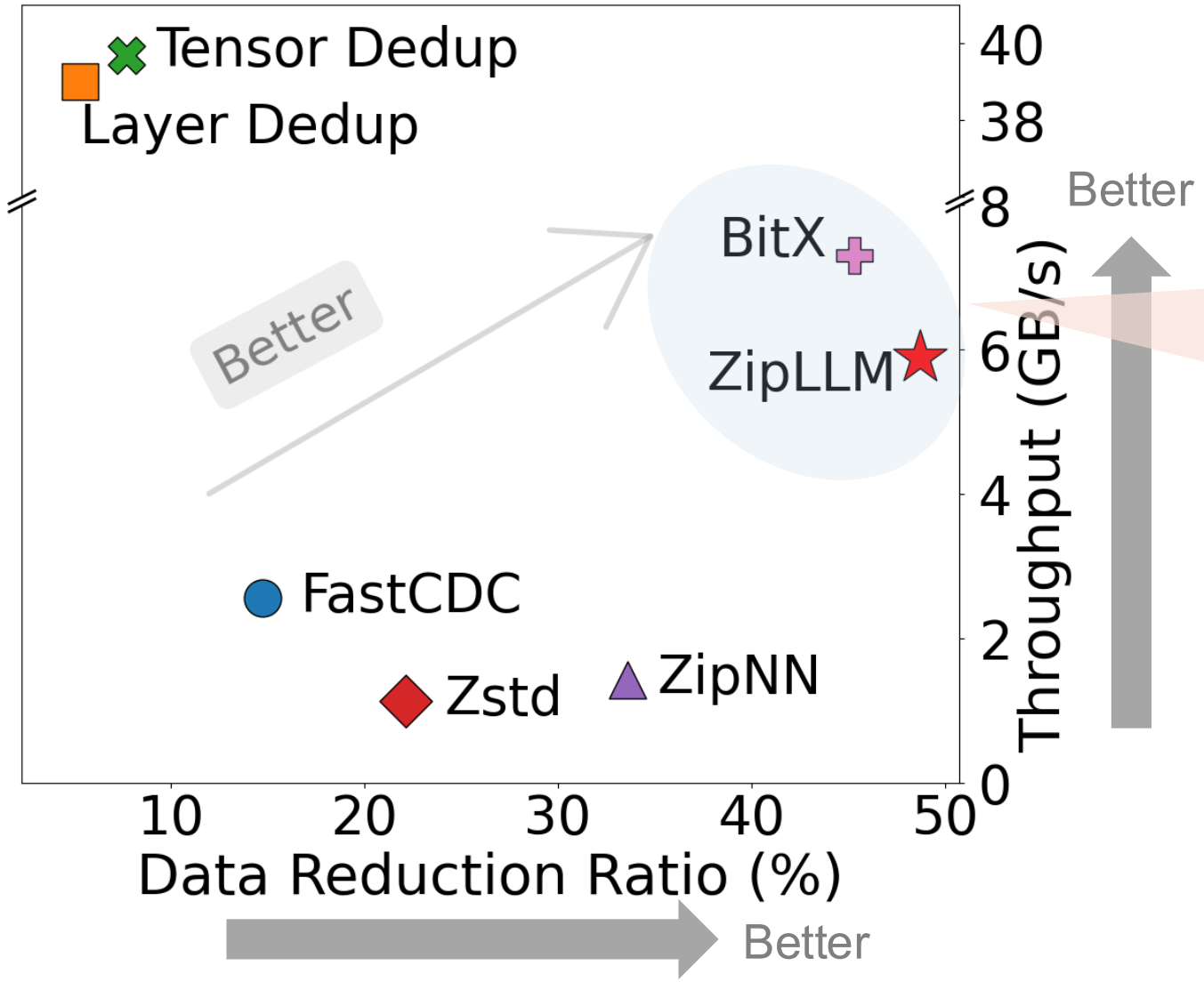
# Evaluation setup




- **Dataset:** 1,700 Hugging Face LLM repos, ~20 TB of `safetensors`
  - 563 **Qwen2.5** models, 529 **Mistral-7B** models, 480 **Llama-3.1-8B** models, 91 **Gemma-2-9B** models, and 79 **Llama-3.2-3B** models

- **Environment:** An EC2 c5a.12xlarge w/ 48 cores + 96GB DRAM

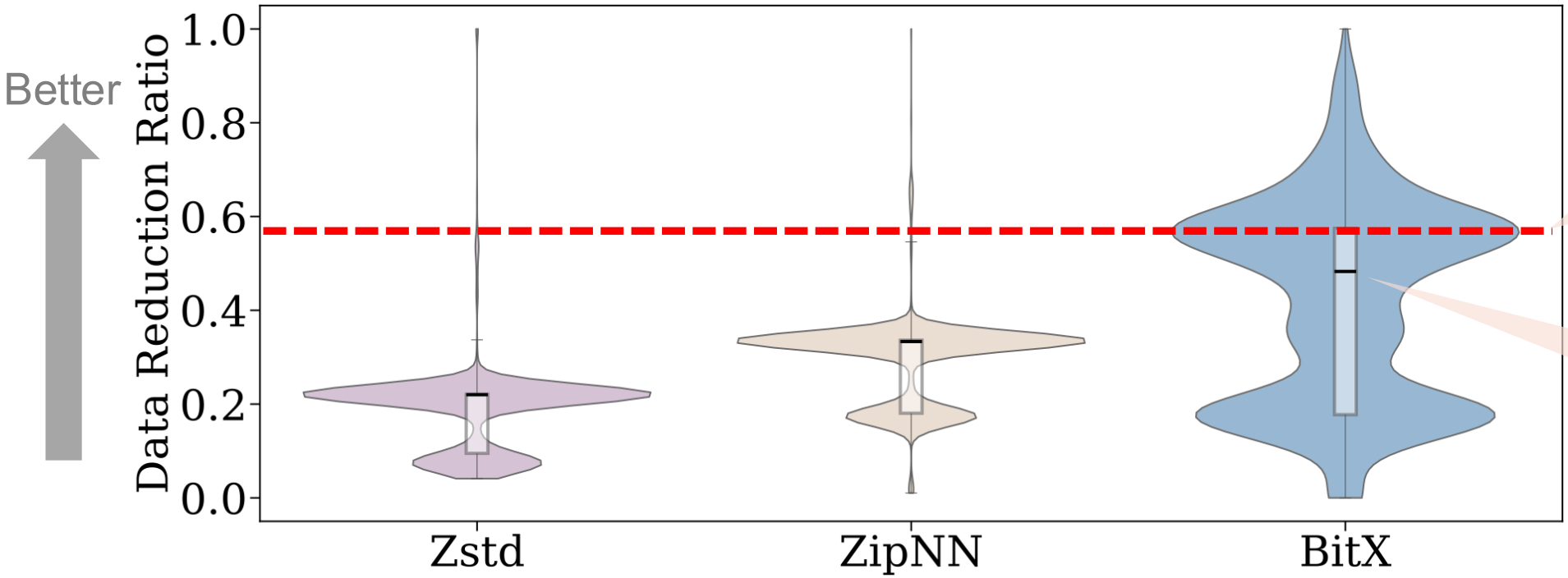
# Data reduction ratio vs. throughput




**BitX** and **ZipLLM** consistently outperform all existing and SOTA data reduction methods

**Throughput = data ingestion throughput on a 48-core machine**

# Comparing compression methods

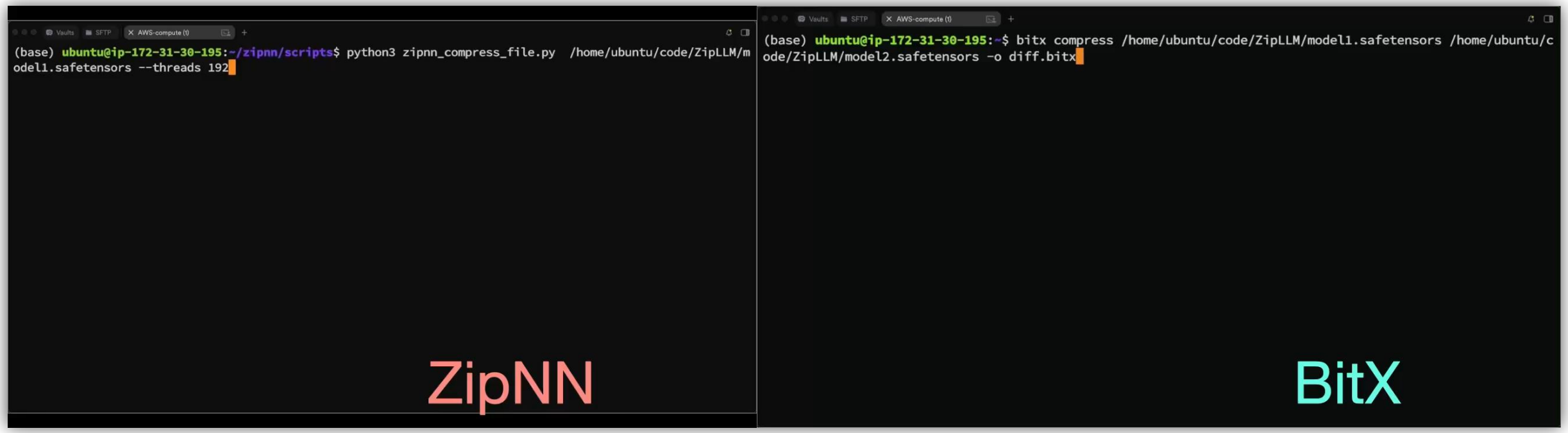



**58%** reduction for a large portion

**~50%** medium

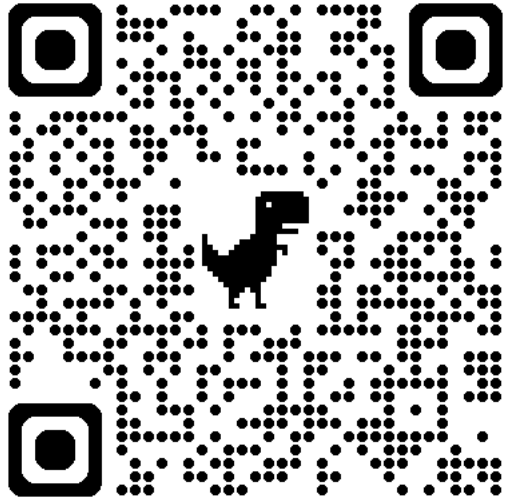
# Open-source, live gain, and impact

Rust crate: crates/bitx-codec | GitHub: https://github.com/ds2-lab/ZipLLM | Web: https://ds2-lab.github.io/ZipLLM

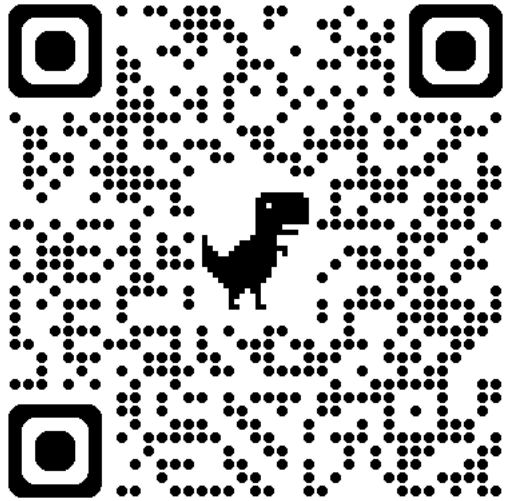


# Conclusions

- Modern AI platforms demand storage systems **tailored to AI workloads**
- Our large-scale study reveals **key redundancies** in LLM storage
- ZipLLM achieves significantly higher storage savings and throughput compared to state-of-the-art approaches



https://github.com/ds2-lab/ZipLLM



Exciting ongoing work on rethinking AI model hub with tensor-level compression

# Backup slides

# BitX on different format

- BitX natively support and is compatible for different float format and quantized format, like FP32, FP8, INT8 and so on.

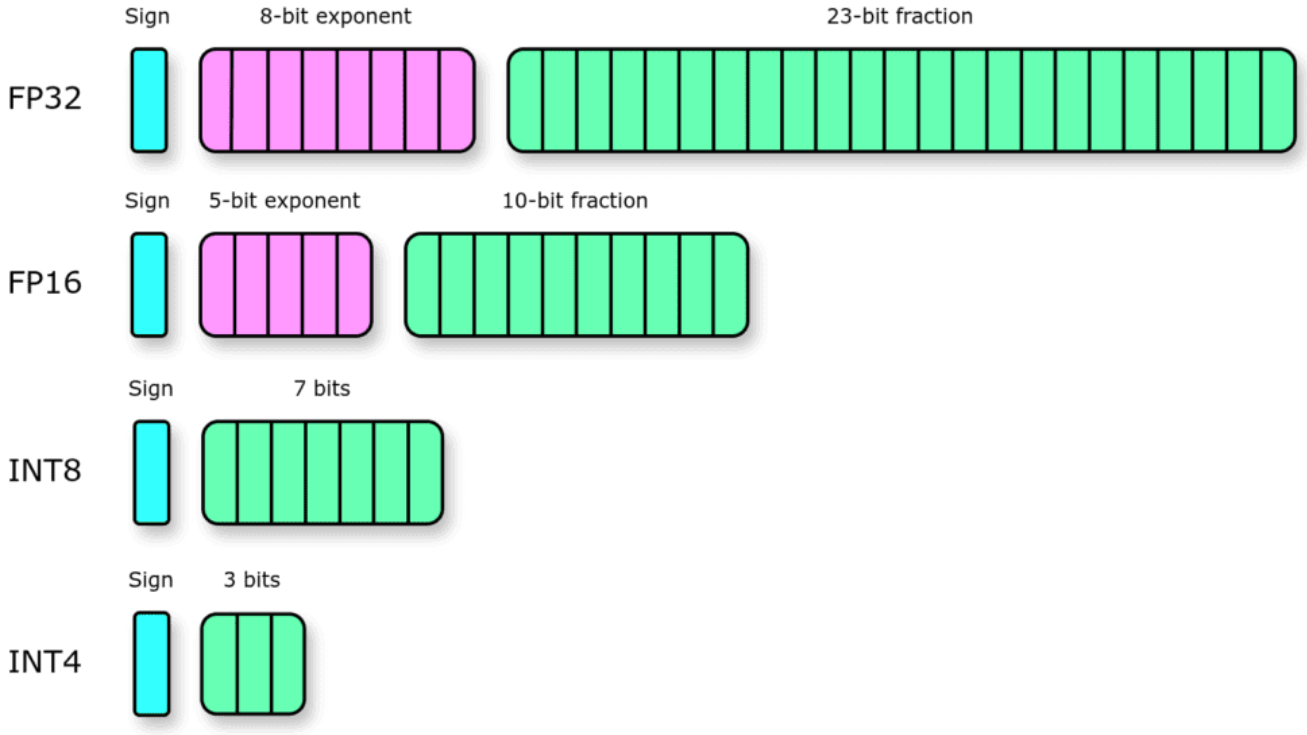



Image source: https://www.medoid.ai/blog/a-hands-on-walkthrough-on-model-quantization/

# Data reduction ratio vs. model counts