NYU

Recitation 8

Devaditya Mohanty

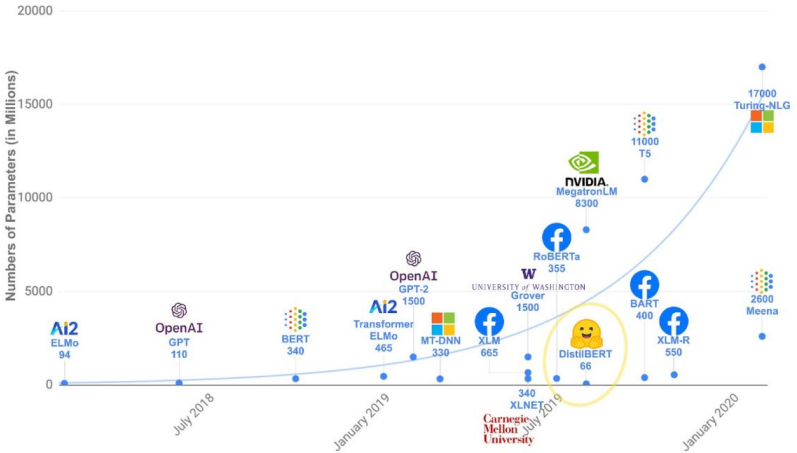
DS-GA 1011, Fall'25
October 29, 2025

Agenda

- Efficiency Challenge
- Quantization
  - What is quantization and how to quantize?
- Pruning
  - Pruning before, during and after training?
- Knowledge Distillation
  - Distillation on outputs, weights and features

Efficiency Challenge

- What is inference? __
- Size of the model makes __

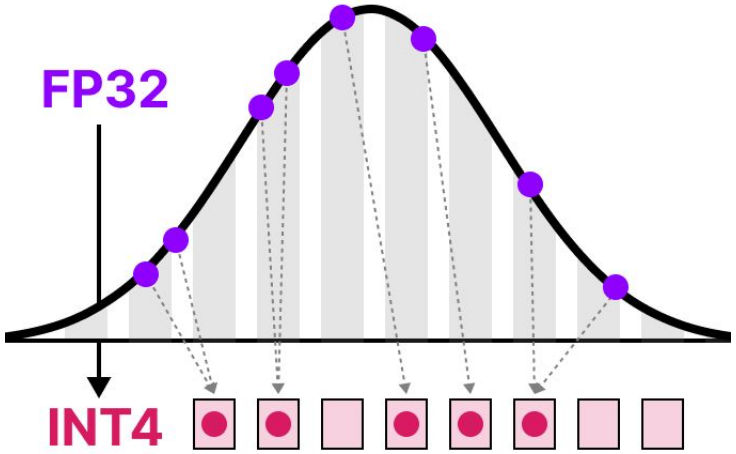


NYU

What is quantization?

- In general, it is the process of mapping a continuous or large set of values to a smaller, discrete set of values

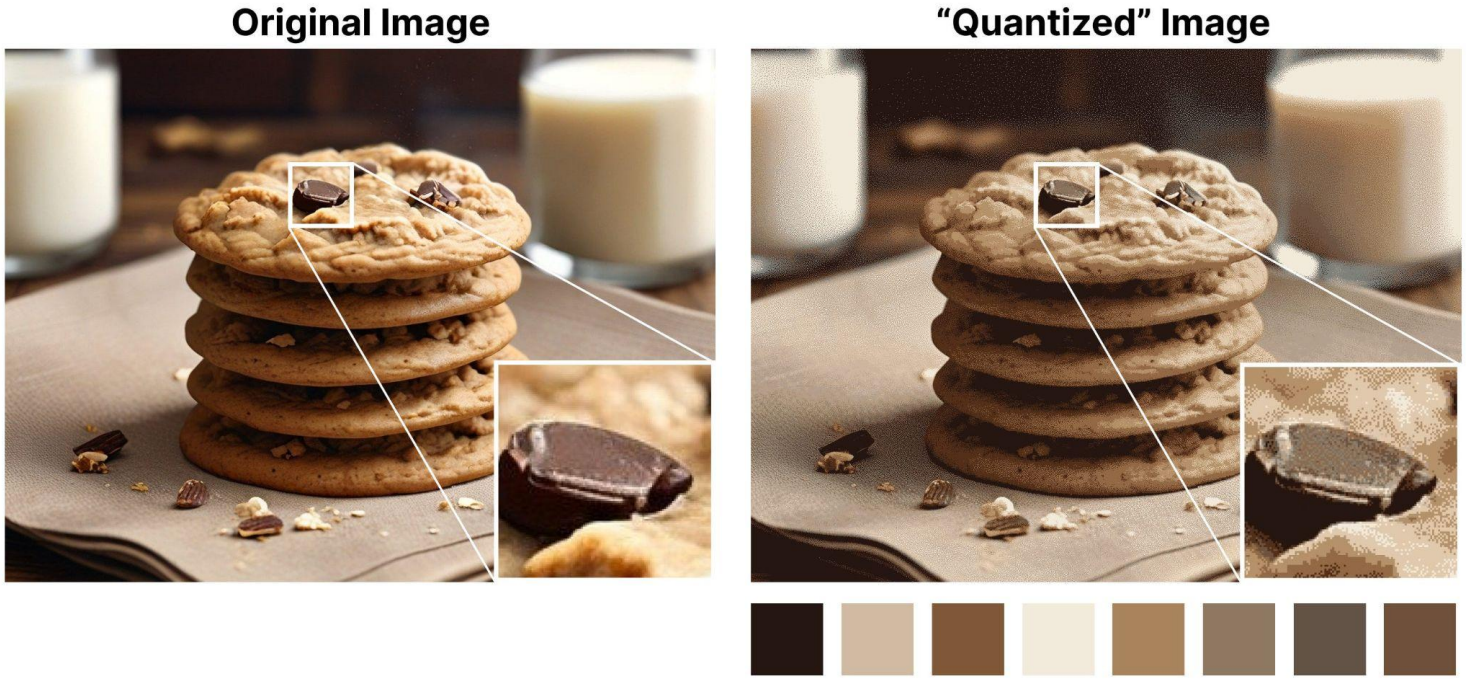
- It is simplifying data by **rounding it to low-precision data types** while **preserving important details**



Adapted from [2]

NYU

How does quantization look like in real-world?

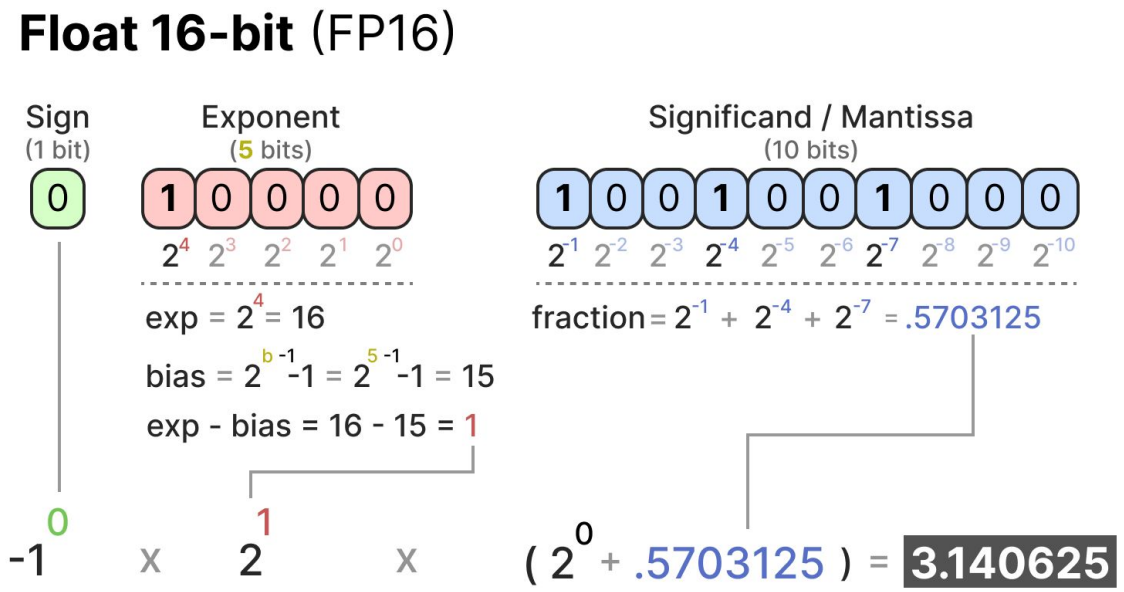


Using only 8 colors to represent the image. Adapted from [2].
Reducing the image to use just 8 colors leads to a **loss of detail and precision**

NYU

How to represent numerical values?

- Represented by “**bits**”, or binary digits
- Bits use **sign, exponent and fraction (mantissa)** to represent a value

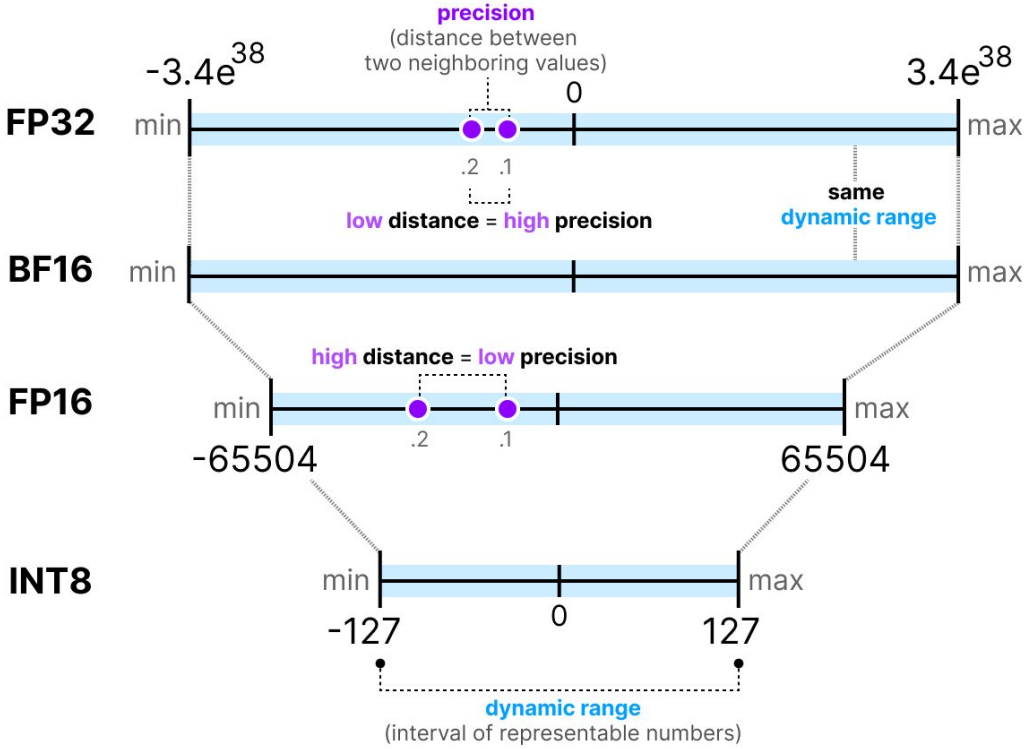


Adapted from [2]

What is precision? Common data types.. ?

- **Precision** is a measure of how precisely a number can be represented

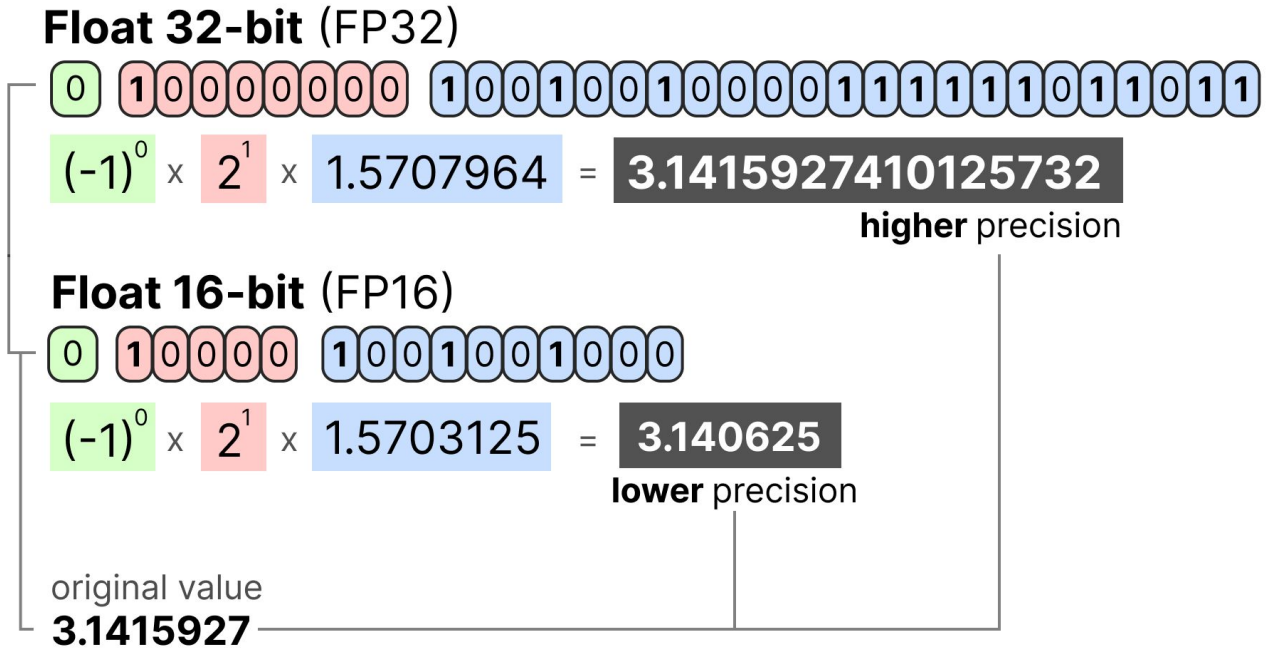
- What is the difference between BF16 and FP16?



Adapted from [2]

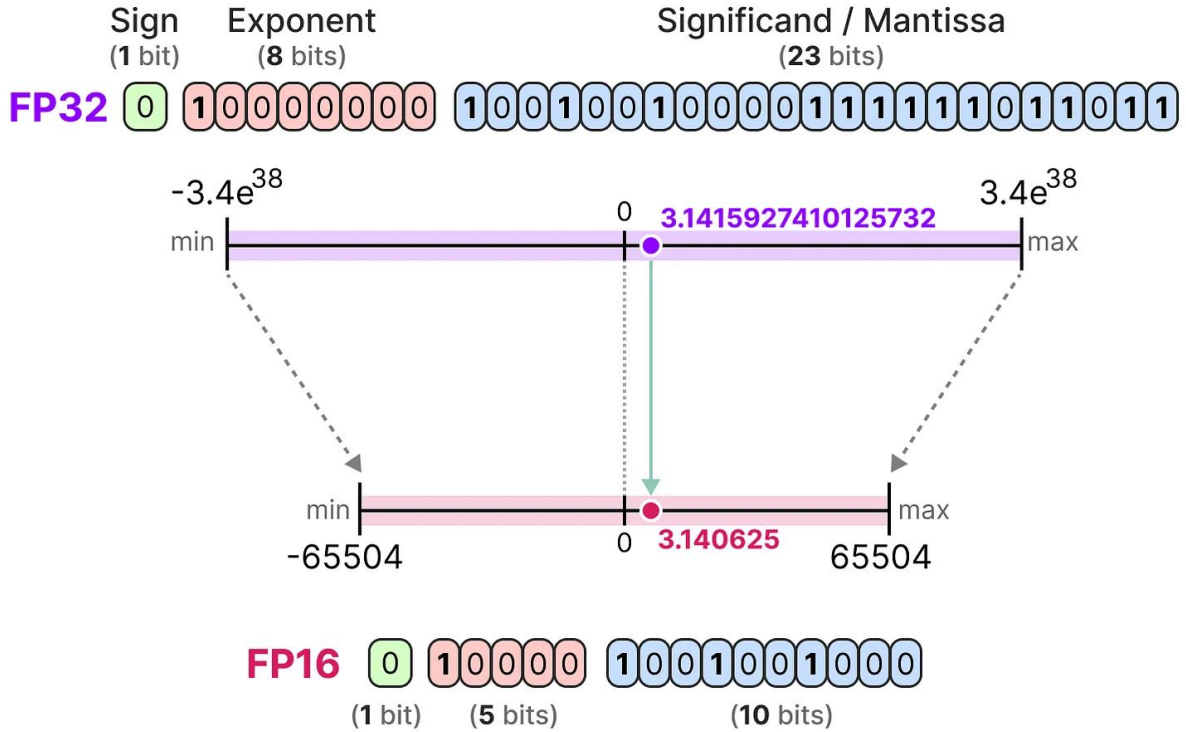
Common data types

$(-1)^{sign} \times base^{exponent} \times fraction$



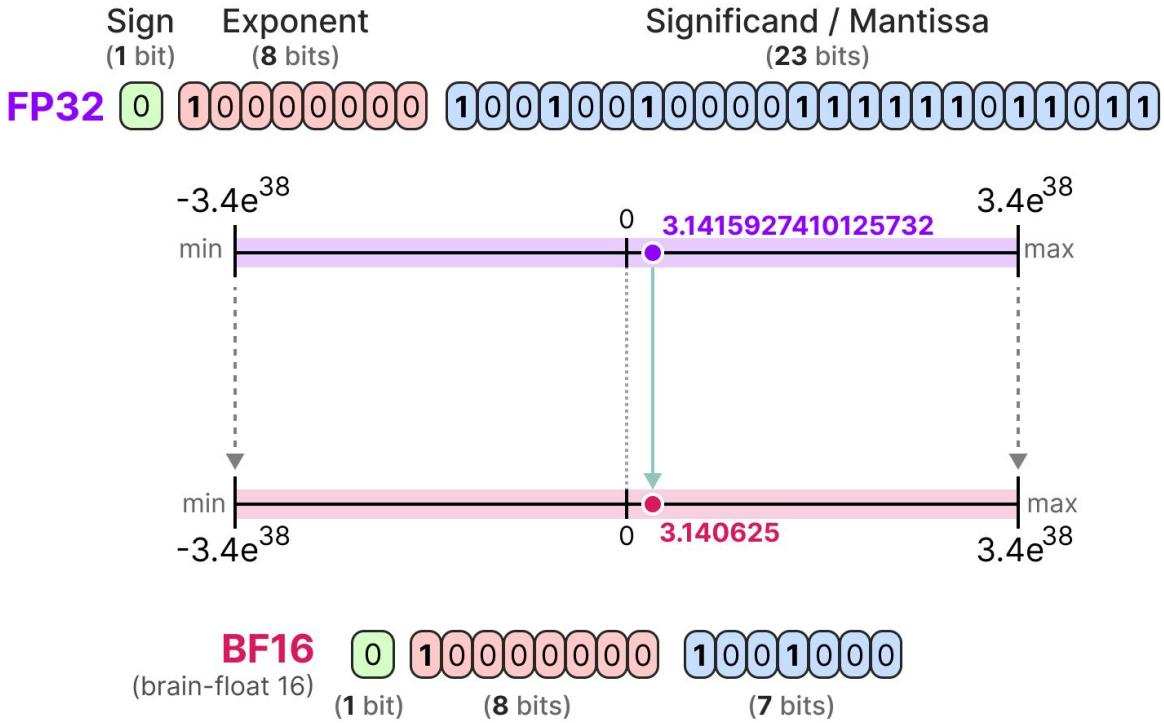
Using more bits **increases the precision** of a value. From [2]

Common data types



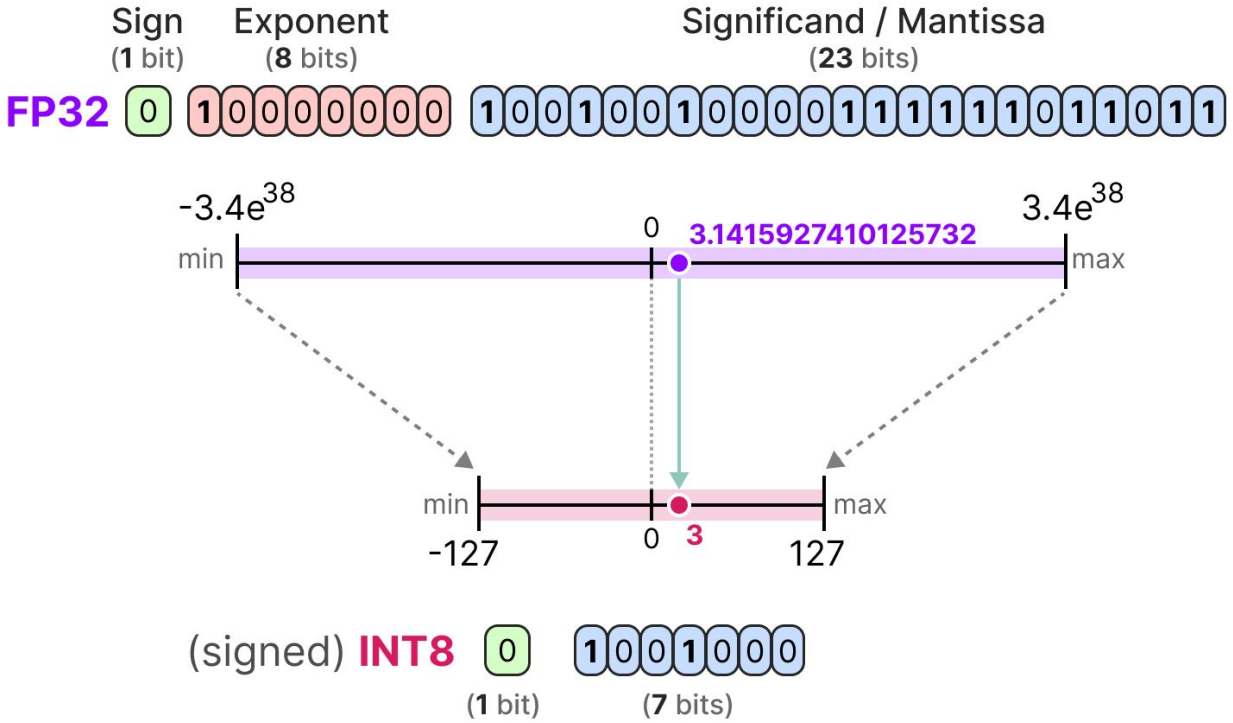
The **range of values reduces** with quantization. From [2]

Common data types



BF16 introduced to **have same range** as **FP32**. From [2]

Common data types



Can convert FP32 to 8-bit integer representations to save a lot of memory. From [2]

How much memory do we use.. ?

- **Memory = [# bits/ 8] * # params**

- Let's say you have a *Llama-3-8B*, how much memory would you need to just load the model weights?
  - **FP32** -> 32-bits = 32/8 * 8B ~ 32 GB
  - **FP16** -> 16-bits = 16/8 * 8B ~ 16 GB
  - **BF16** -> 16-bits = 16/8 * 8B ~ 16 GB
  - **INT-8** -> 8-bits = 8/8 * 8B ~ 8 GB
  - **4-bit** -> 4-bits = 4/8 * 8B ~ 4 GB

NYU

*Actual GPU memory usage when running a model depends on several other factors such as: CUDA/runtime overhead, activation buffers, attention caches (for generation), context size etc, and if training — gradients and optimizer/momentum states (can multiply memory by 2–6x)

Absolute maximum (absmax) quantization

- How would you scale a number into the range [-127, 127] ?
  - Divide by its absolute maximum value and multiply by 127

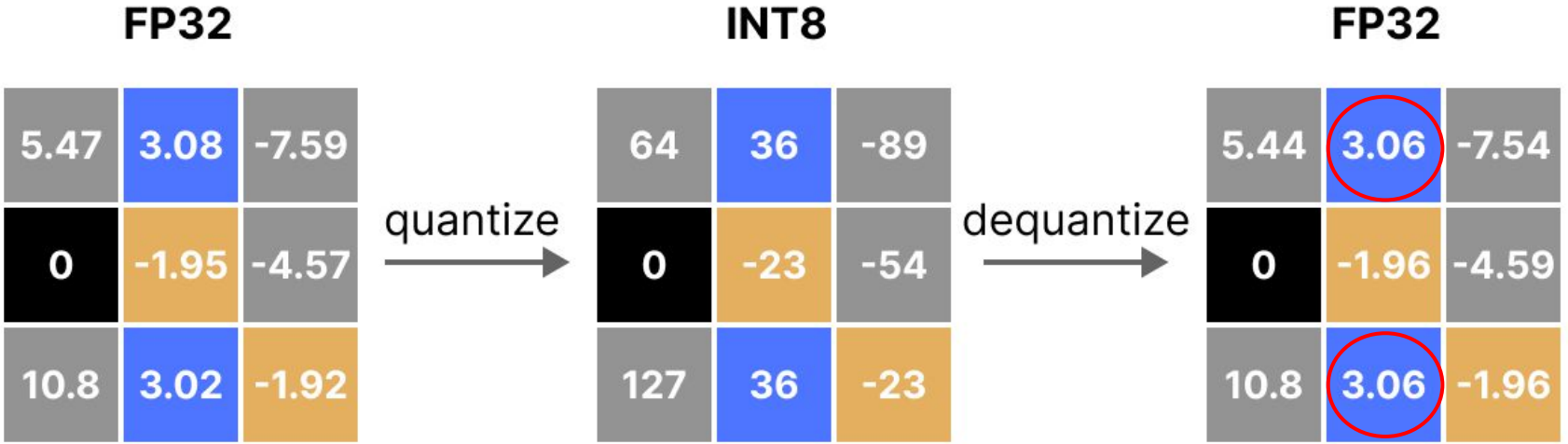
$$\mathbf{X}_{\text{dequant}} = \frac{\max|\mathbf{X}|}{127} \cdot \mathbf{X}_{\text{quant}}$$

$$\mathbf{X}_{\text{quant}} = \text{round}\left(\frac{\mathbf{X}}{\max|\mathbf{X}|} \cdot 127\right)$$

NYU

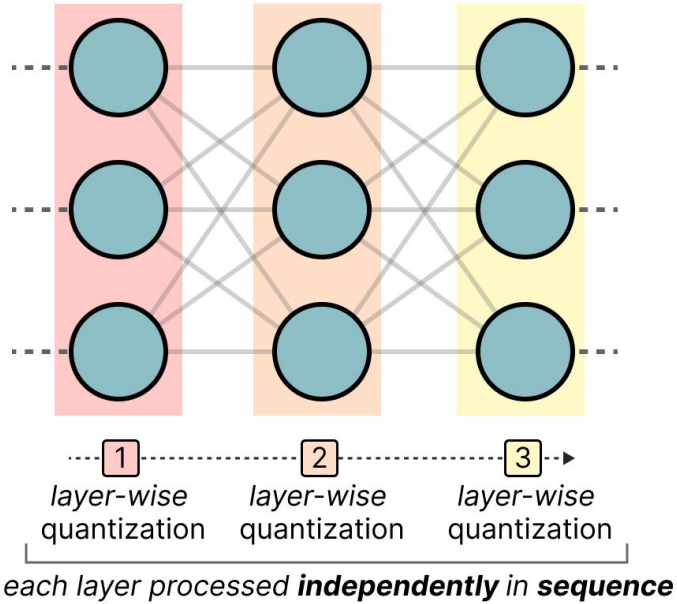
Quantization error

- Applying the quantization and then dequantization process to retrieve the original – leads to quantization error (= delta b/w original and dequantized)



GPTQ

- Each layer is quantized independently. Given a *layer l* with a weight matrix, W_l find the quantized weights $\widehat{W}_l$ **[3]**



$$\widehat{W}_l^* = \arg\min_{W_l} \|W_l X - \widehat{W}_l X\|^2$$

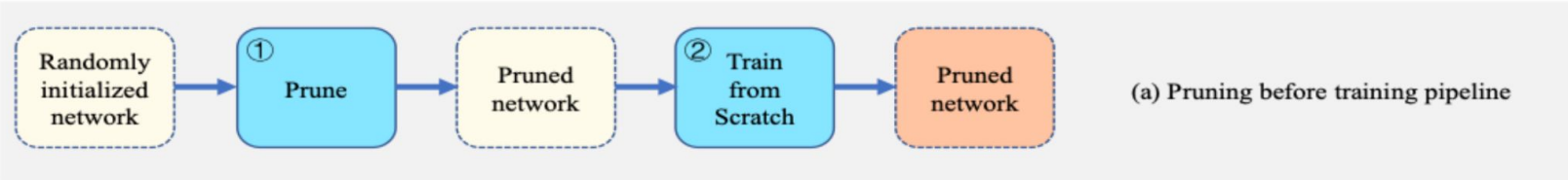
Model Pruning

- **What is model pruning?**
  - Removing parts of a neural network that contribute little to its output
  - Reduces model size and speeds up inference – without affecting output quality much

- **How is it different from quantization?**
  - Pruning reduces model size by reducing # parameters, while quantization does so by reducing precision of each parameters

Pruning before training

- **Procedure**: Prune the randomly initialized network followed by training from scratch. Reduces pre-training cost significantly
- Implementation: Weights are zeroed out using a binary mask

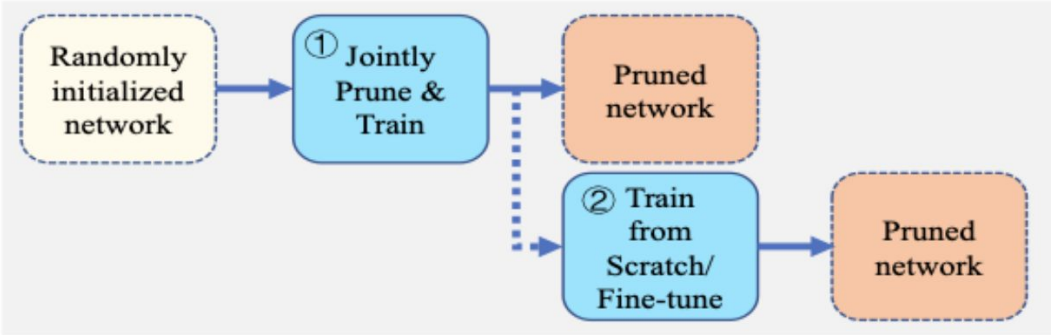


Adapted from [8]

- Examples:
  - **SNIP** [4] – one-shot pruning before training based on sensitivity
  - **GraSP** [5] – gradient-based pruning preserving training signal flow
  - **RANDOM** [6] – random pruning performs strikingly well

Pruning during training

- **Procedure**: Jointly prune and train the weights and the mask during training.

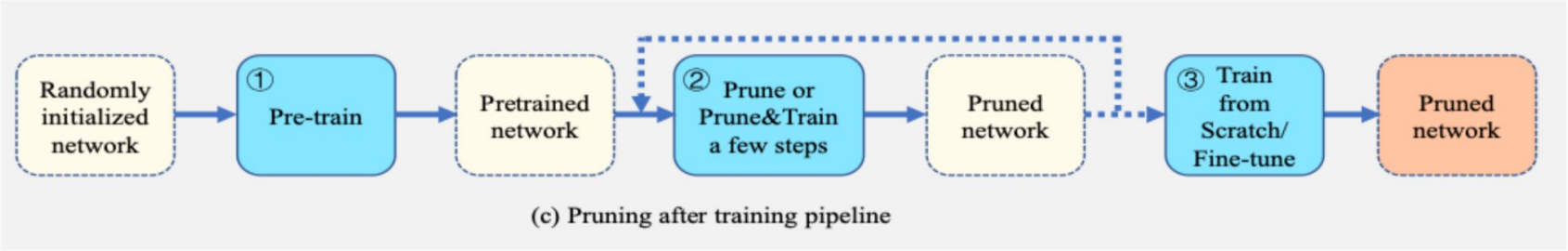


* Dashed lines indicate that subsequent operations are not required

- Examples:
  - **Sparse Regularization** - To impose a sparse constraint on the loss function, for example adding a L_1 regularization term [7]
  - **Score based methods** - Evaluate the scores of parameters during model training and directly prune parameters with low scores [8]
    - **Sheared Llama** [12] - performs structured pruning interleaved with training

Pruning after training

- **Procedure**: Jointly prune and train the weights and the mask during training
  - Follows a pretrain-prune-retrain process, as shown in the figure below.



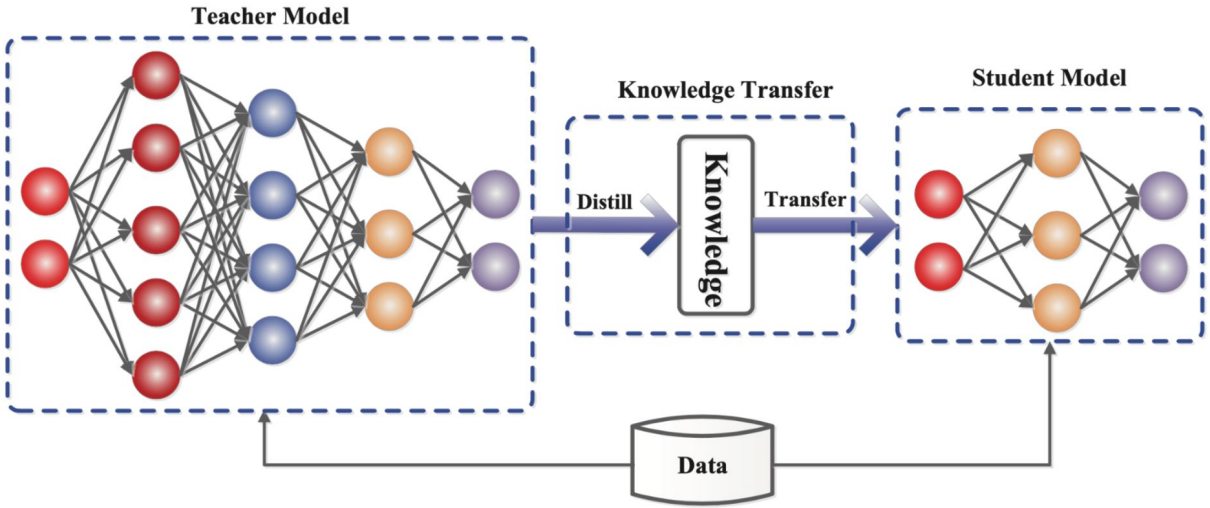
(c) Pruning after training pipeline

* Dashed lines indicate that subsequent operations are not required

- Examples:
  - **Lottery Ticket Hypothesis [9]**: Every overparameterized network has a smaller winning subnet that can perform just as well when trained independently
  - **Sparse GPT [10]:** LLMs can be pruned to 50% sparsity in one-shot, without any retraining
  - **Early BERT [11]:** Identifies structured winning tickets in the early stage of BERT

Knowledge Distillation

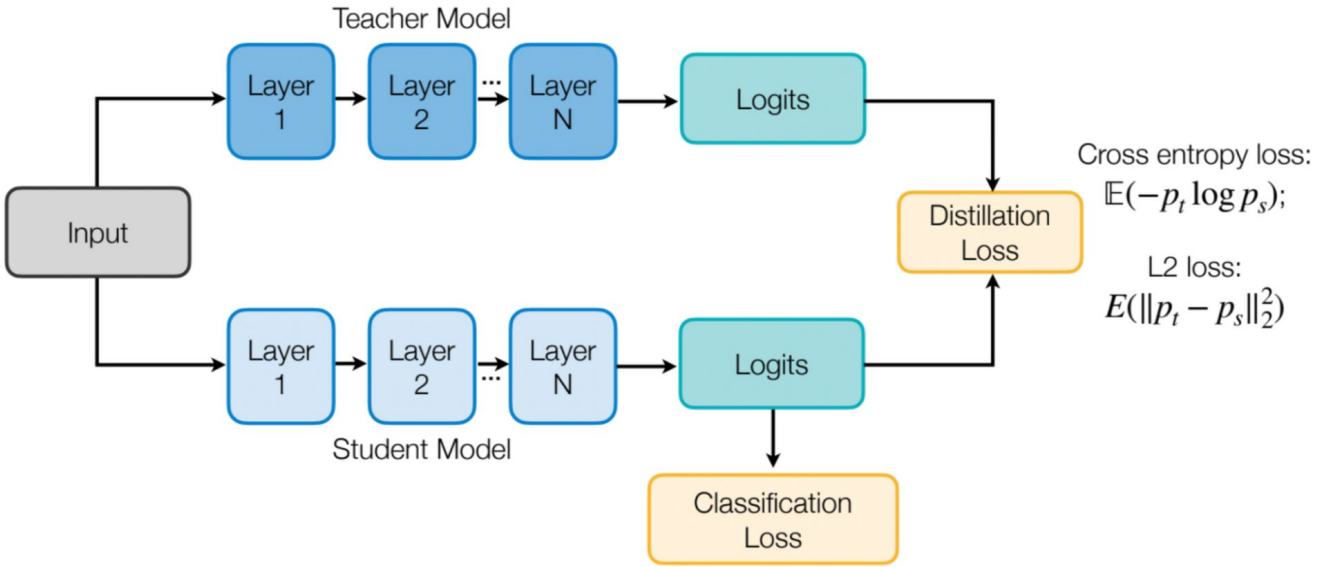
- **Goal**: To **transfer knowledge** from a **larger model** to a **student model**



- Why not train a student model **from scratch** instead?

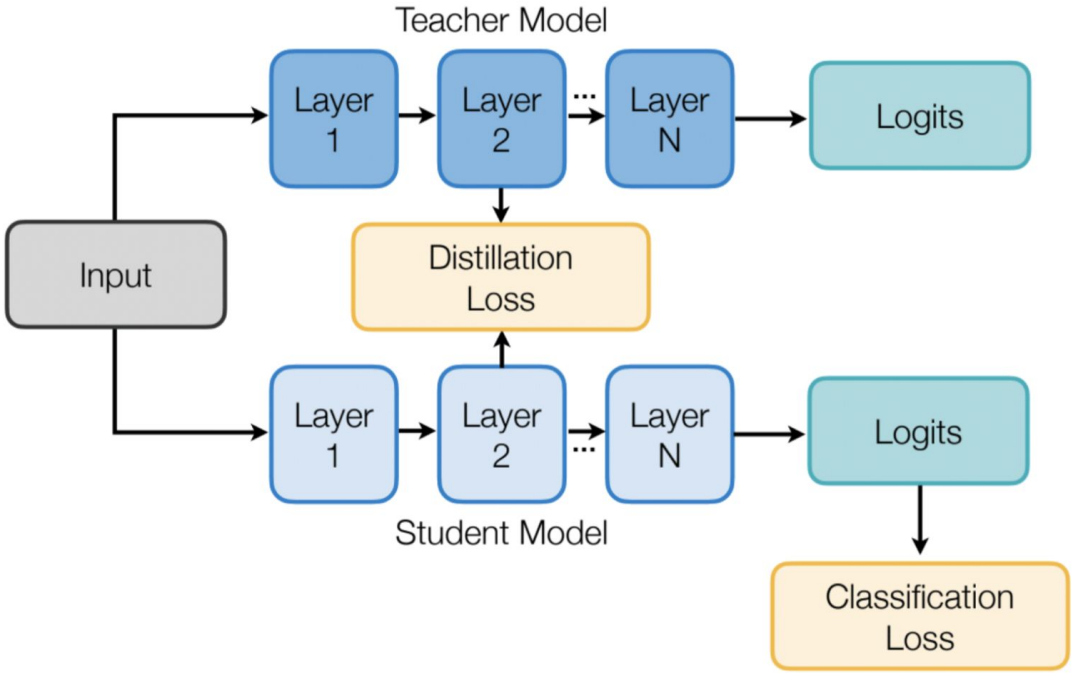
Knowledge Distillation

- Simplest way is to **match the outputs** using a **distance metric**



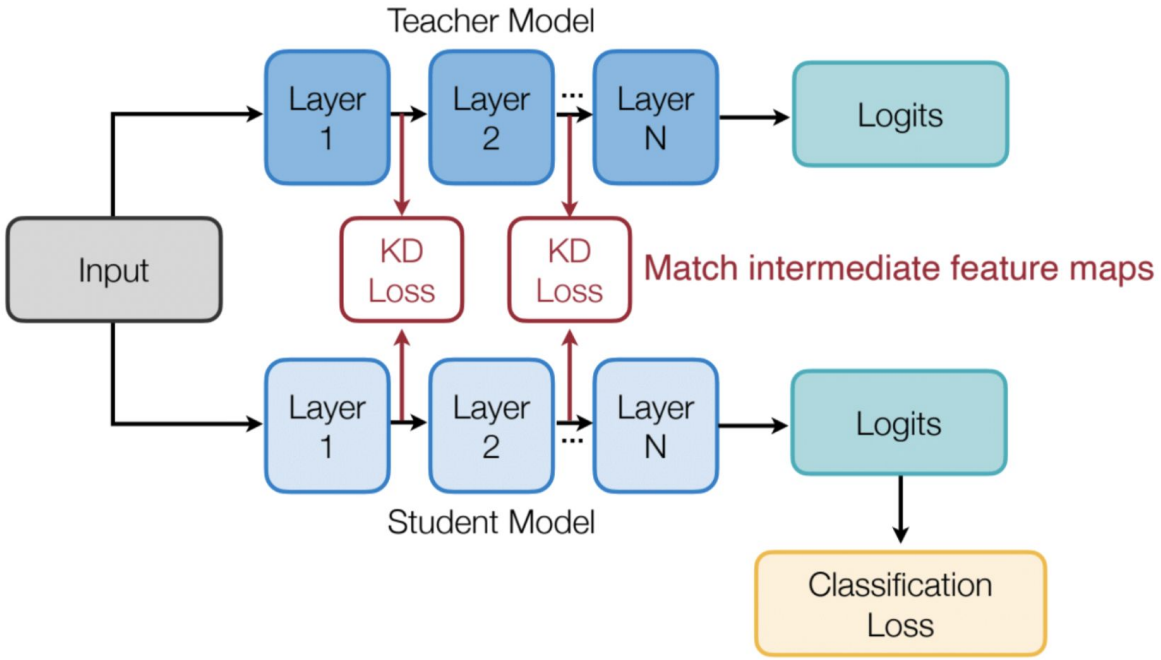
Knowledge Distillation

- Match the **weights in intermediate layers** using a **distance metric**



Knowledge Distillation

- Match the **intermediate feature maps** using a distance metric. Ex. DistilBERT

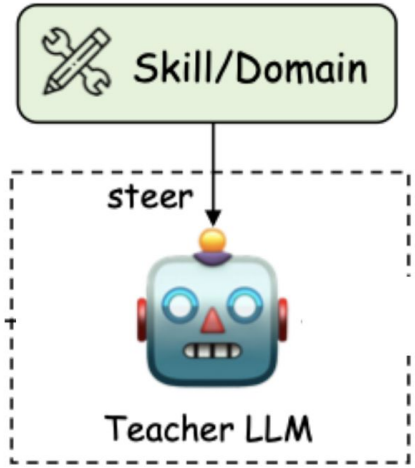


Knowledge Distillation in LLM Era

- **Inaccessible parameters**, but we still want to transfer knowledge from LLMs

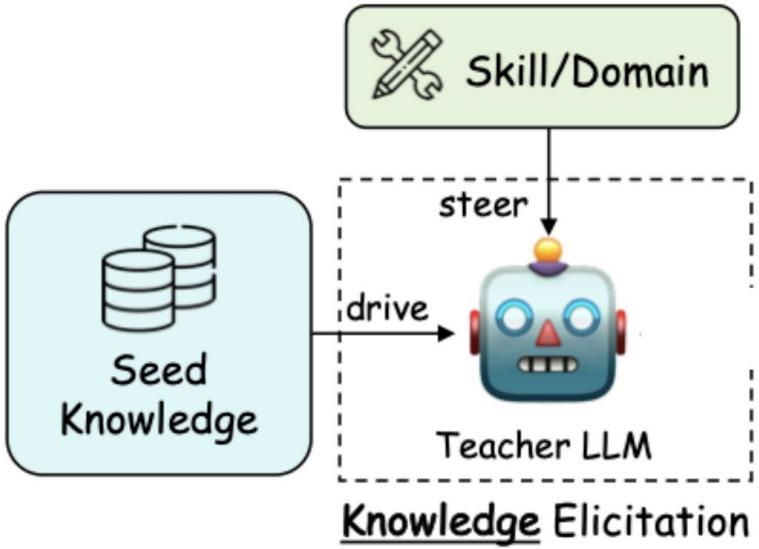
Knowledge Distillation in LLM Era

- **Steer the LLM** for a target skill or domain



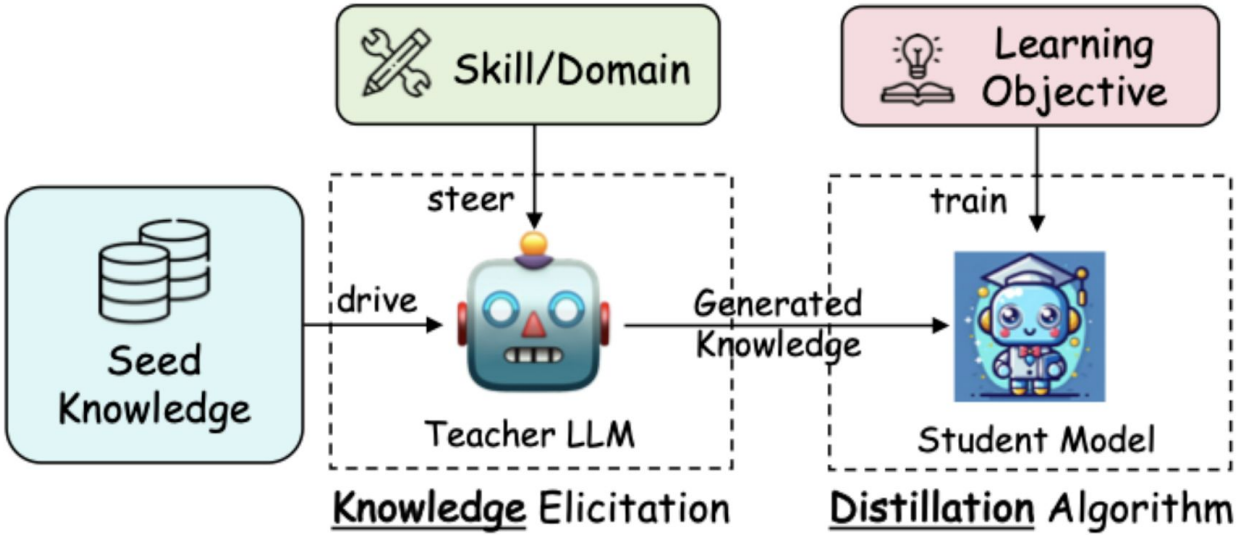
Knowledge Distillation in LLM Era

- **Feed the LLM** with a small dataset as seed knowledge



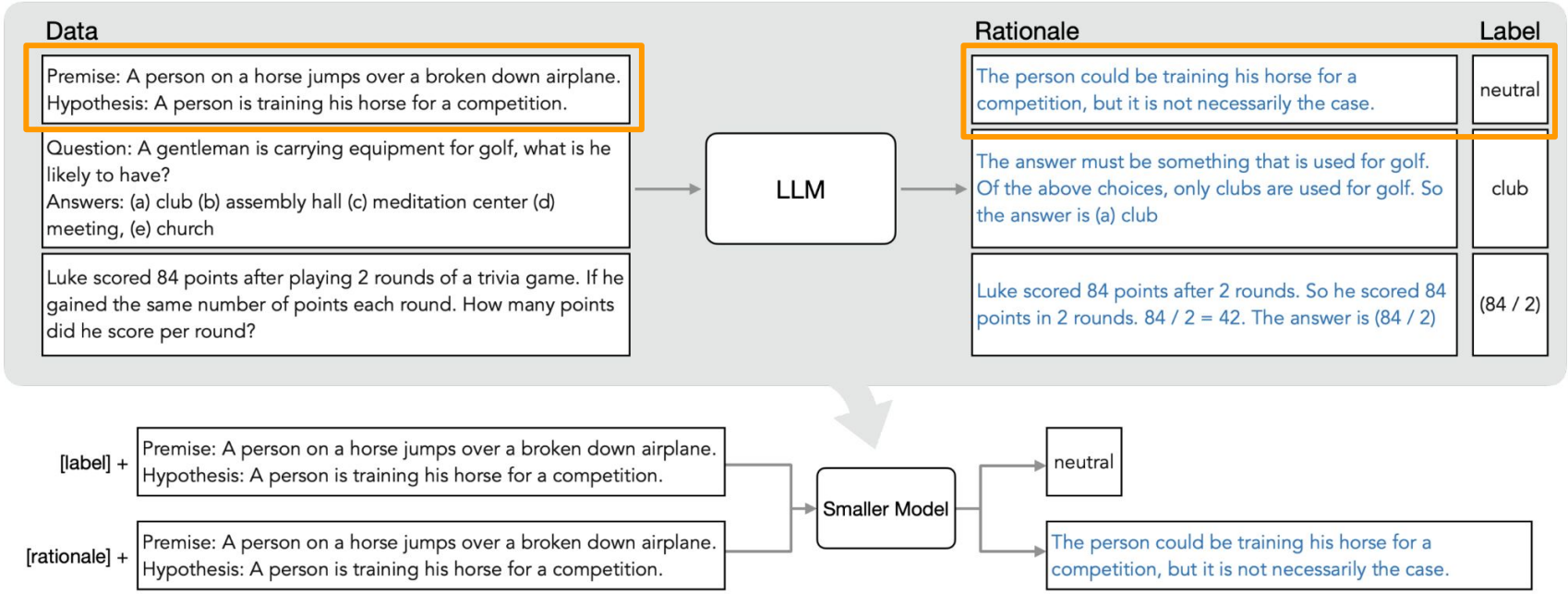
Knowledge Distillation in LLM Era

- **Generate knowledge** from the LLM with a small dataset as seed knowledge



Knowledge Distillation in LLM Era

- Can also distill on reasoning steps (rationales) extracted from LLMs



Adapted from C.-Y. Hsieh, et. al., "Distilling step-by-step! Outperforming larger language models with less training data and smaller model sizes," ACL, 2023.

Key Takeaways

- **Efficient inference** is important to **deploy models for resource-constrained devices**
- **Quantization** aims to simplify data by **rounding it to low-precision data types** while preserving information
- **Pruning** aims to **remove redundancy** in networks before, during or after training
- **Knowledge distillation** distills knowledge from a **teacher to student network**

References

1. Adapted from NYU DSGA-1011 (Fall'24) section material by Divyam Madaan
2. https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-quantization
3. E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate post-training quantization for generative pre-trained transformers," ICLR, 2023
4. **SNIP** - N. Lee, T. Ajanthan, and P. H. S. Torr, "**SNIP**: Single-shot network pruning based on connection sensitivity," ICLR, 2019. Available: https://arxiv.org/abs/1810.02340
5. **GraSP** - C. Wang, G. Zhang, and R. Grosse, "Picking winning tickets before training by preserving gradient flow," ICLR, 2020. Available: https://arxiv.org/abs/2002.07376
6. S. Liu, et. al., "The unreasonable effectiveness of **random** pruning: Return of the most naive baseline for sparse training," ICLR 2022. Available: https://arxiv.org/abs/2202.02643
7. W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in NIPS, 2016.
8. H. Cheng, M. Zhang, and J. Q. Shi, "A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations," arXiv preprint arXiv:2308.06767, 2024. [Online]. Available: https://arxiv.org/abs/2308.06767
9. J. Frankle and M. Carbin, "The **lottery ticket hypothesis**: finding sparse, trainable neural networks," in ICLR, 2019
10. E. Frantar and D. Alistarh, "**SparseGPT**: Massive language models can be accurately pruned in one-shot," arXiv preprint arXiv:2301.00774, 2023
11. X. Chen, Y. Cheng, S. Wang, Z. Gan, Z. Wang, and J. Liu, "**EarlyBERT**: Efficient BERT training via early-bird lottery tickets," in ACL-IJCNLP, 2021, pp. 2195–2207.
12. M. Xia, T. Gao, Z. Zeng, and D. Chen, "**Sheared LLaMA**: Accelerating language model pre-training via structured pruning," ICLR, 2024. Available: https://arxiv.org/abs/2310.06694