“Small” LLMs

CMSC 473/673 - NATURAL LANGUAGE PROCESSING

Learning Objectives

Examining…

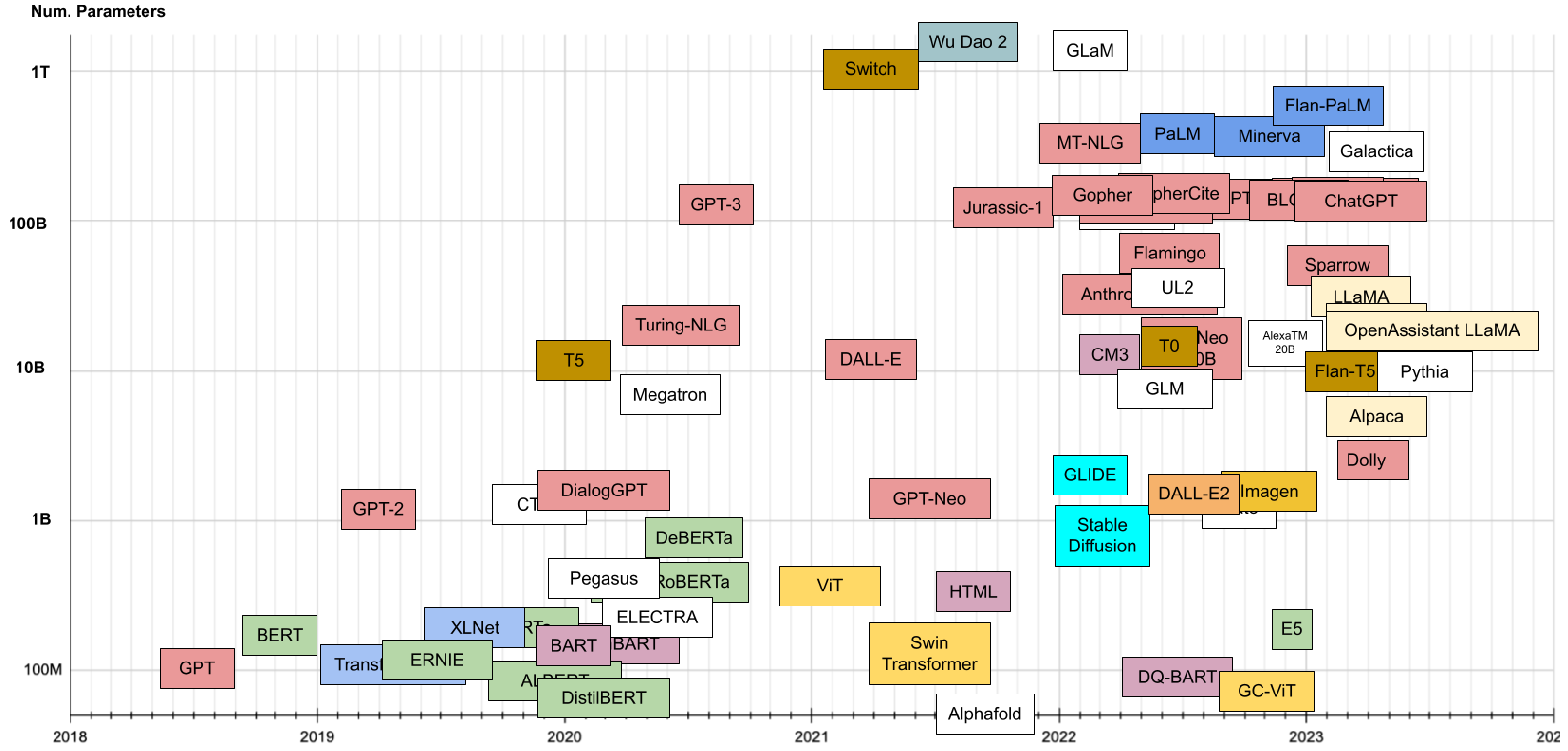
- Methods for shrinking pre-existing models
- Methods for mimicking pre-existing models with smaller models
- Methods for faster/smaller finetuning of pre-existing models
- Methods for training new models more efficiently

Finding where to implement these methods

Recognizing when to implement them

Efficient LLMs

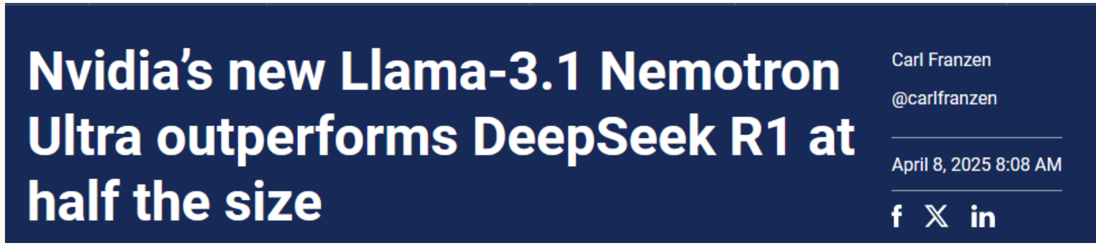
Methods for shrinking pre-existing models	Model Compression
Methods for mimicking pre-existing models with smaller models	Knowledge Distillation
Methods for faster/smaller finetuning of pre-existing models	PEFT
Methods for training new models more efficiently	Efficient Training e.g., Sparse Mixture of Experts



https://amatriain.net/blog/transformer-models-an-introduction-and-catalog-2d1e9039f376/

Category	Benchmark (Metric)	Claude-3.5-Sonnet-1022	GPT-4o 0513	DeepSeek V3	OpenAI o1-mini	OpenAI o1-1217	DeepSeek R1
	Architecture	-	-	MoE	-	-	MoE
	# Activated Params	-	-	37B	-	-	37B
	# Total Params	-	~1.8 Trillion	671B	-	-	671B
English	MMLU (Pass@1)	88.3	87.2	88.5	85.2	**91.8**	90.8
	MMLU-Redux (EM)	88.9	88.0	89.1	86.7	-	**92.9**
	MMLU-Pro (EM)	78.0	72.6	75.9	80.3	-	**84.0**
	DROP (3-shot F1)	88.3	83.7	91.6	83.9	90.2	**92.2**
	IF-Eval (Prompt Strict)	**86.5**	84.3	86.1	84.8	-	83.3
	GPQA-Diamond (Pass@1)	65.0	49.9	59.1	60.0	**75.7**	71.5
	SimpleQA (Correct)	28.4	38.2	24.9	7.0	**47.0**	30.1
	FRAMES (Acc.)	72.5	80.5	73.3	76.9	-	**82.5**

https://huggingface.co/deepseek-ai/DeepSeek-R1#4-evaluation-results


Nvidia's new Llama-3.1 Nemotron Ultra outperforms DeepSeek R1 at half the size

Carl Franzen
@carlfranzen

April 8, 2025 8:08 AM

https://venturebeat.com/ai/nvidias-new-llama-3-1-nemotron-ultra-outperforms-deepseek-r1-at-half-the-size/

Model Compression

Pruning

Remove parts of the model

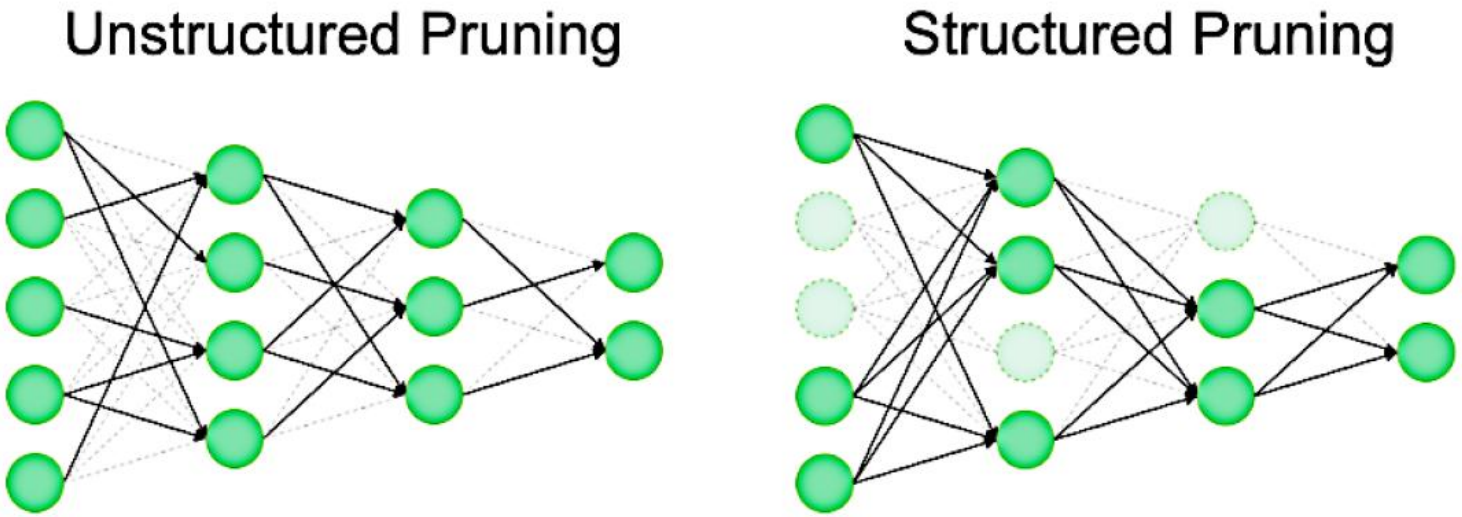
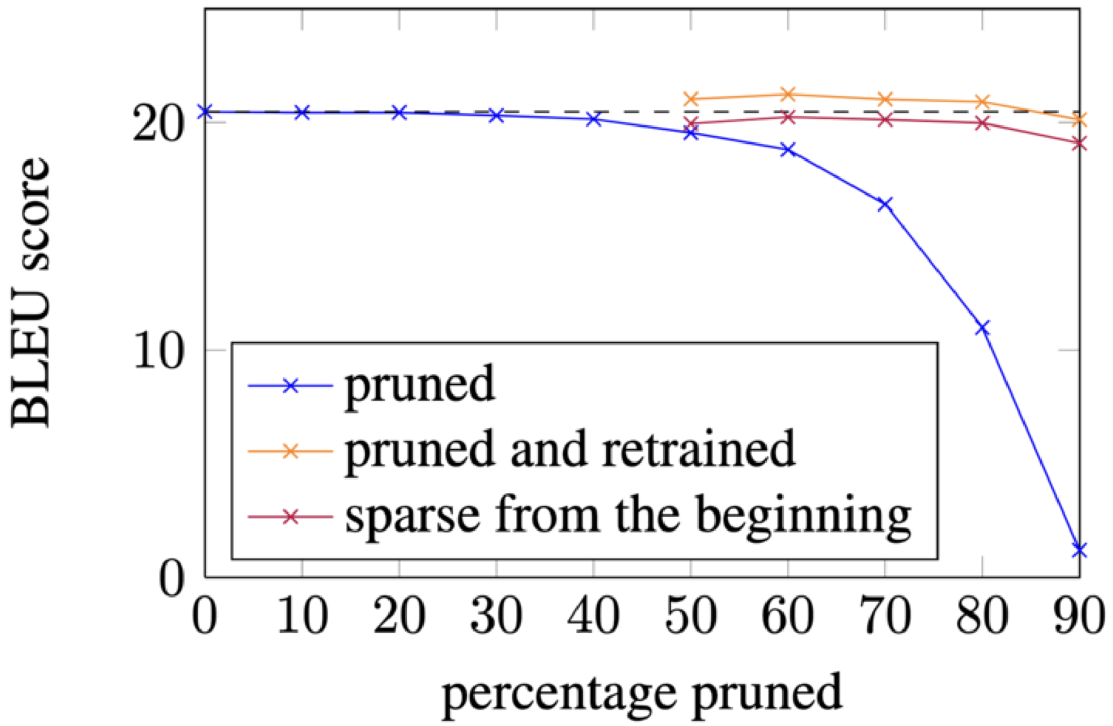


Image credits: neuralmagic.com

Implementation Tutorial: https://pytorch.org/tutorials/intermediate/pruning_tutorial.html

Magnitude Pruning



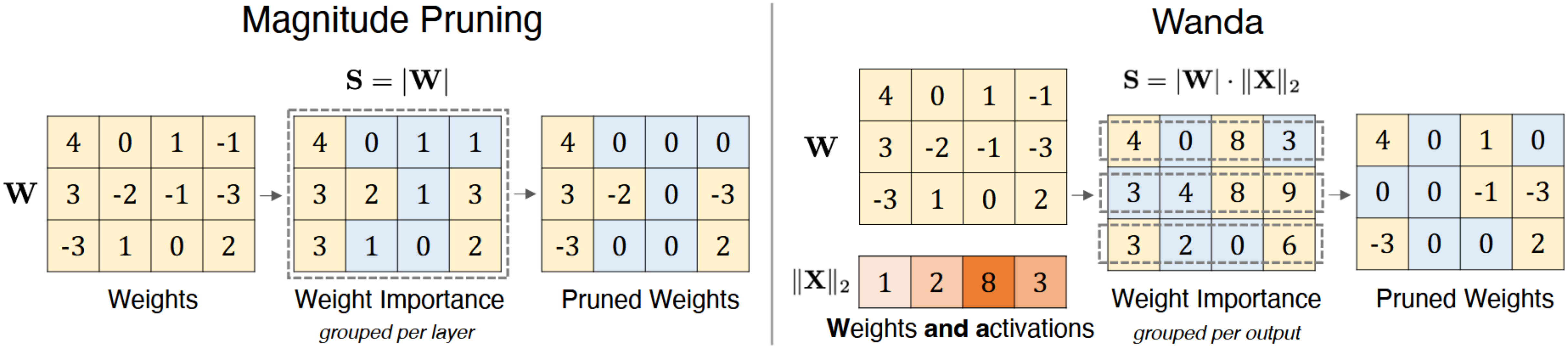
- prune weights with smallest absolute value
- prunes 40% of the weights with negligible performance loss
- by adding a retraining phase after pruning, we can prune 80% with no performance loss

Image credits: See et al. 2016

Slide by Dinesh Raghu

See, Abigail, Minh-Thang Luong, and Christopher D. Manning. 2016. “Compression of Neural Machine Translation Models via Pruning.” In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*. Berlin, Germany: Association for Computational Linguistics, 291–301. doi:10.18653/v1/K16-1029.

Wanda



Sun, Mingjie, Zhuang Liu, Anna Bair, and J. Zico Kolter. 2024. "A Simple and Effective Pruning Approach for Large Language Models." In *International Conference on Learning Representations (ICLR)*, Vienna, Austria. https://iclr.cc/virtual/2024/poster/18687

Quantizing Models

Compresses weights and activations from floating point numbers to integers (e.g., 4-bit, 8-bit)

Then use the scaling factor to get the “original” value

Implementation:

https://pytorch.org/docs/stable/quantization.html

https://pypi.org/project/bitsandbytes/

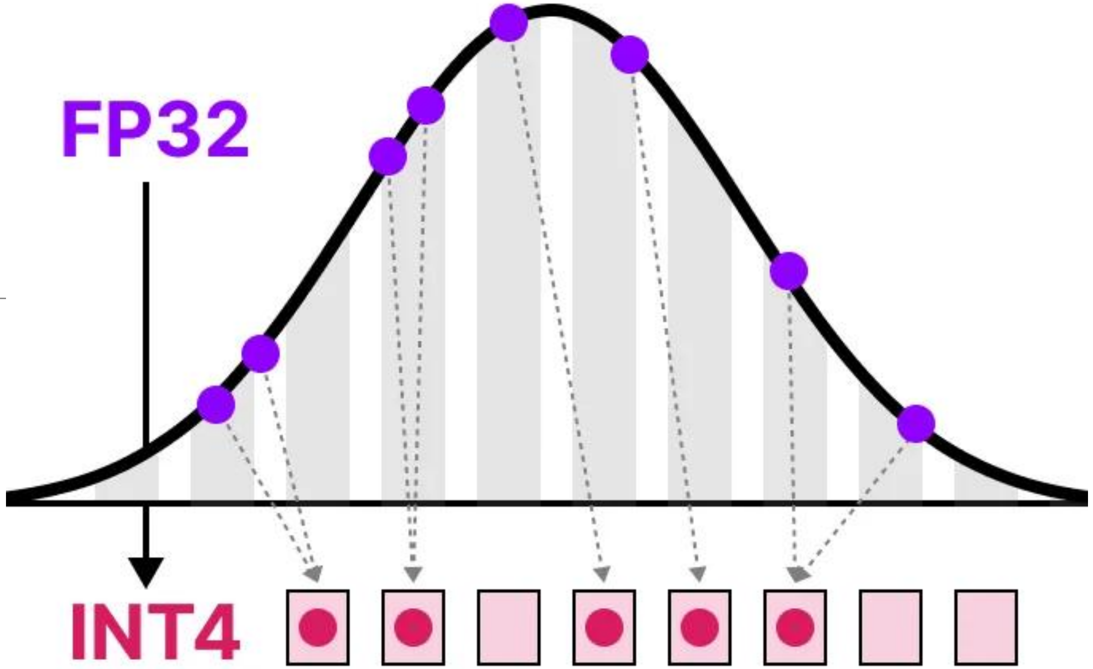
Learn more here:
https://huggingface.co/blog/hf-bitsandbytes-integration

Quantizing Models

Given a layer l with weight matrix W_l and layer input X_l, find quantized weight $\hat{W}_l$:

$$(\hat{W}_l^{\,*} = argmin_{\hat{W}_l} ||W_l X - \hat{W}_l X||_2^2)$$

https://huggingface.co/blog/merve/quantization

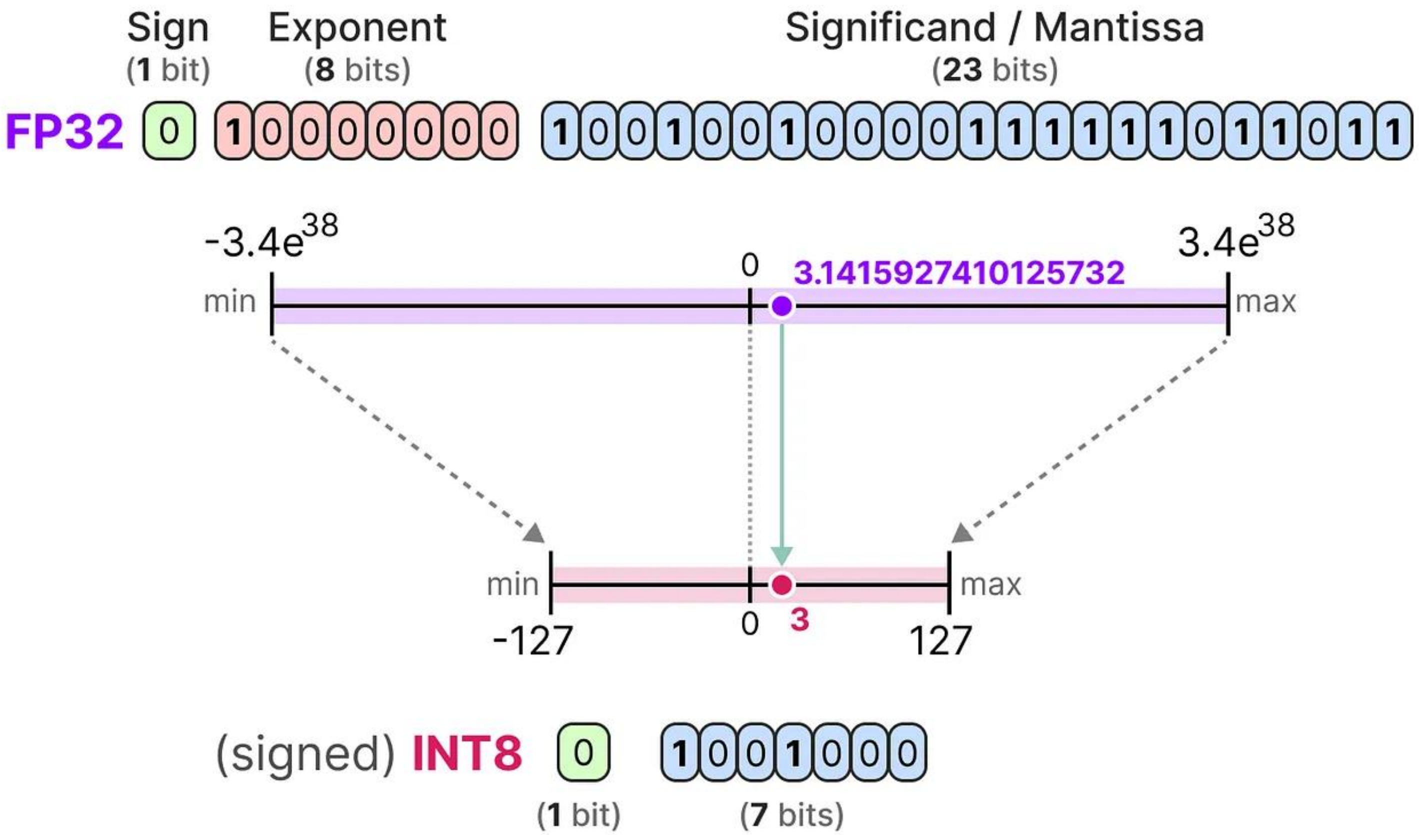


https://substack.com/home/post/p-145531349

$$\mathbf{X}^{\mathrm{Int8}} = \mathrm{round}\left(\frac{127}{\mathrm{absmax}(\mathbf{X}^{\mathrm{FP32}})}\mathbf{X}^{\mathrm{FP32}}\right) = \mathrm{round}(c^{\mathrm{FP32}} \cdot \mathbf{X}^{\mathrm{FP32}}),$$ where c is the *quantization constant*

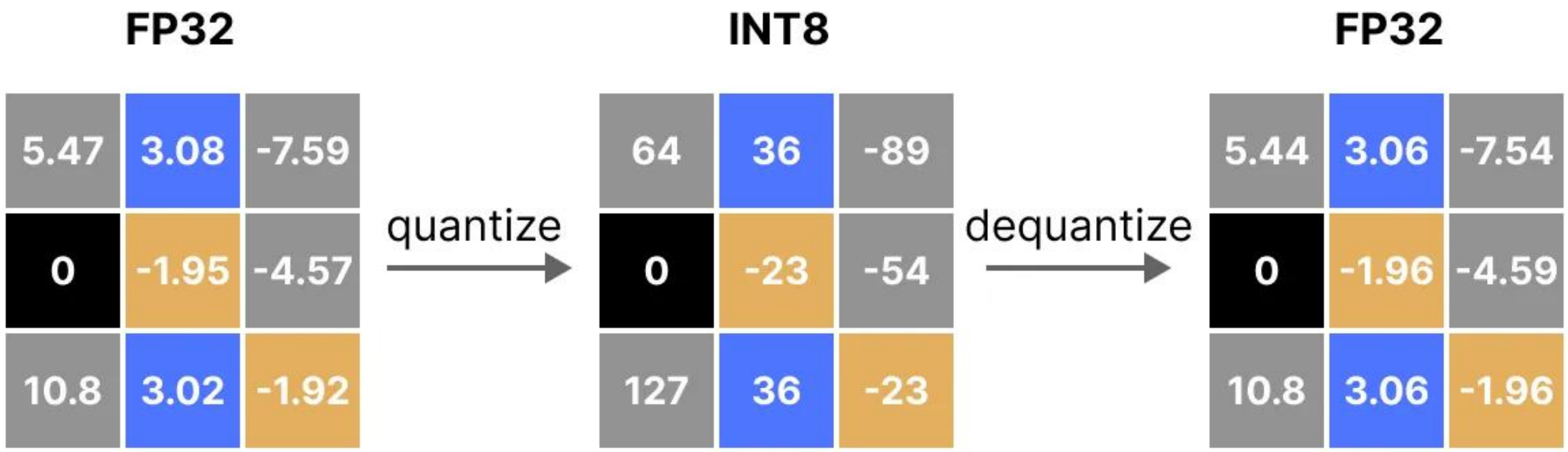
Dettmers, Tim, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. "QLoRA: Efficient Finetuning of Quantized LLMs." Advances in Neural Information Processing Systems

Mapping floating point to integer



https://substack.com/home/post/p-145531349

Quantizing → Dequantizing



https://substack.com/home/post/p-145531349

Quantizing/Pruning (Model Compression)

PROS

Save space

Saving resources (energy, time)

Start with pre-trained model

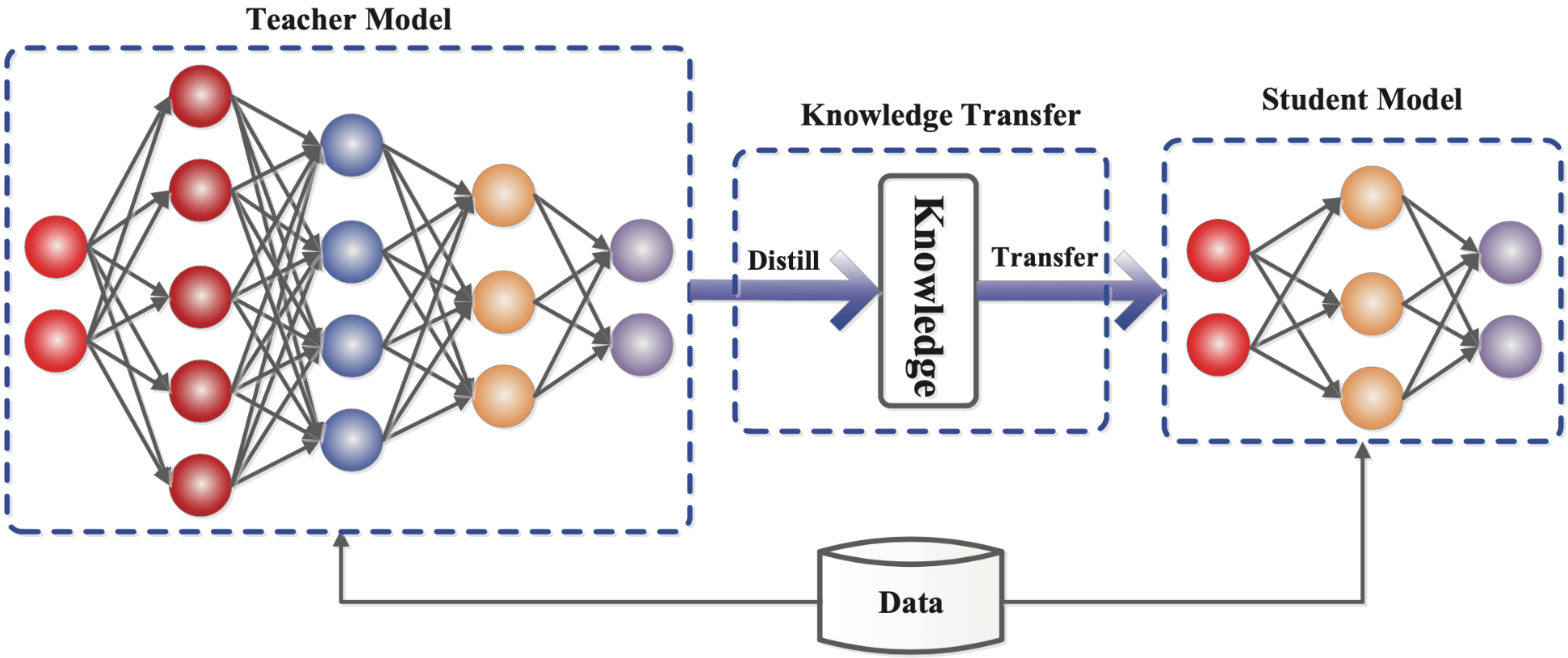
CONS

Lossy

Knowing what to prune without damaging the model

Knowledge Distillation

Knowledge Distillation

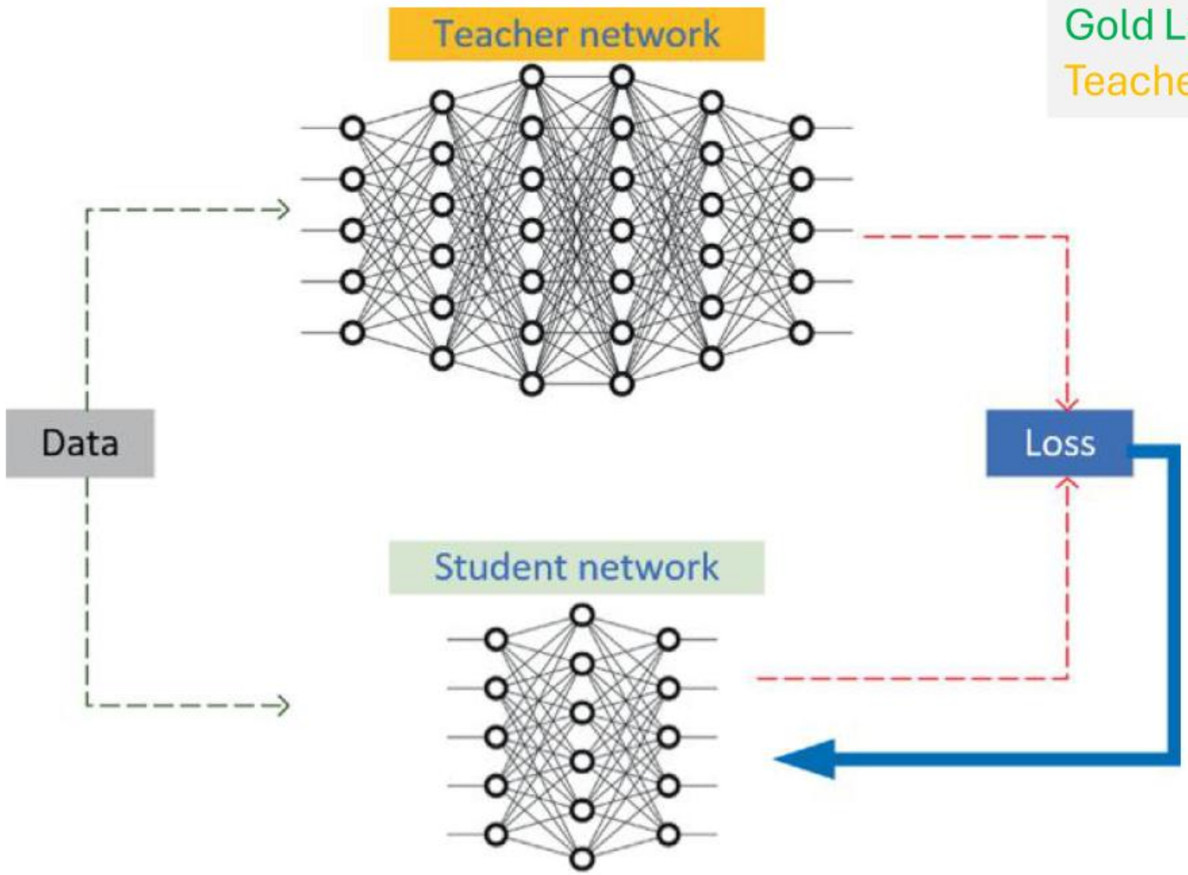


https://neptune.ai/blog/knowledge-distillation

Implementation Tutorial: https://pytorch.org/tutorials/beginner/knowledge_distillation_tutorial.html

Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean. 2014. "Distilling the Knowledge in a Neural Network." In *NeurIPS 2014 Deep Learning Workshop*, doi:10.48550/arXiv.1503.02531.

Training the Student Network



Gold Label	[	0	0	1	0	0	]
Teacher Prediction	[	0.10	0.20	0.50	0.15	0.05	]

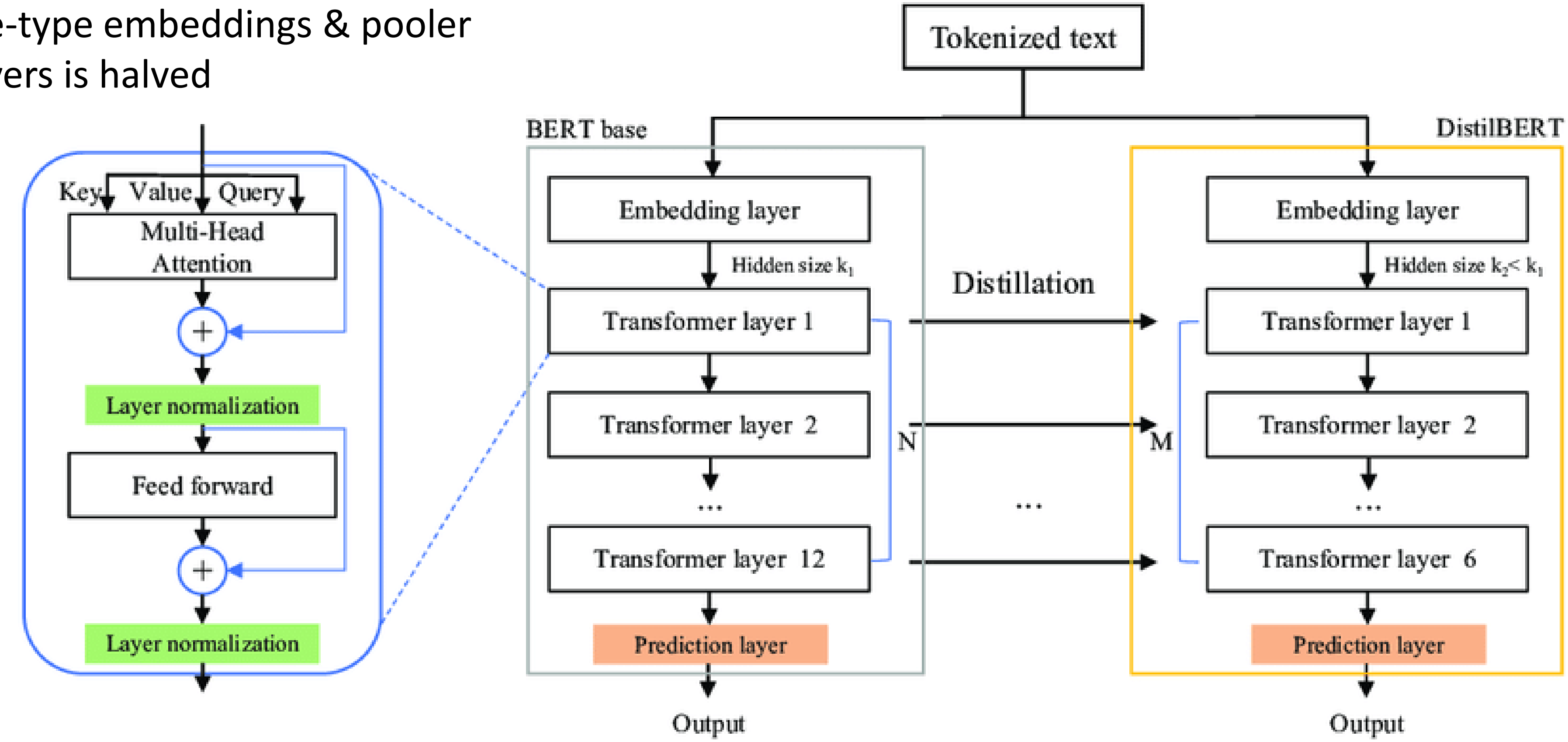
$$\mathcal{L}_{\mathrm{NLL}}(\theta) = -\sum_{k=1}^{|\mathcal{V}|} \mathbb{1}\{y = k\} \log p(y = k \mid x; \theta)$$

$$\mathcal{L}_{\mathrm{KD}}(\theta; \theta_T) = -\sum_{k=1}^{|\mathcal{V}|} q(y = k \mid x; \theta_T) \times \log p(y = k \mid x; \theta)$$

Slide by Dinesh Raghu

DistilBERT

Removed toke-type embeddings & pooler
Number of layers is halved



https://www.researchgate.net/figure/The-DistilBERT-model-architecture-and-components_fig2_358239462

Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. "DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter." In *Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019*, Vancouver, Canada. doi:10.48550/arXiv.1910.01108.

Knowledge Distillation

PROS

Student model is more manageable

New model is cheaper to run

Uses less data to train

Can turn problem into supervised learning

CONS

Student might not be as good as the teacher

You have to create a new model from scratch (architecture, training)

Relies a lot on the quality of the teacher (if teacher is bad, student will be bad)

PEFT

Parameter-efficient Fine-tuning (PEFT)

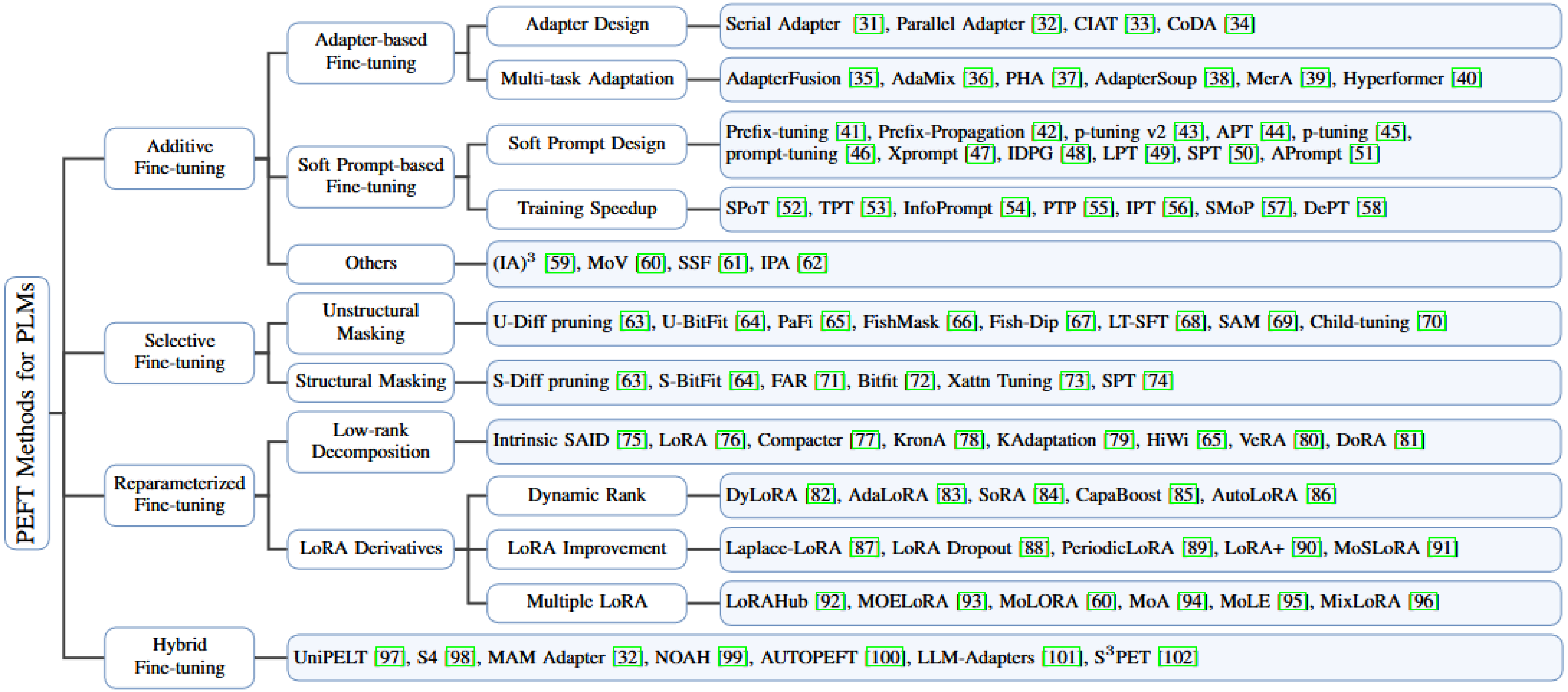


Fig. 3: Taxonomy of Parameter-Efficient Fine-Tuning Methods for Large Models.

Han, Zeyu, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. “Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey.” doi:10.48550/arXiv.2403.14608.

LoRA (Low-Rank Adaptation)

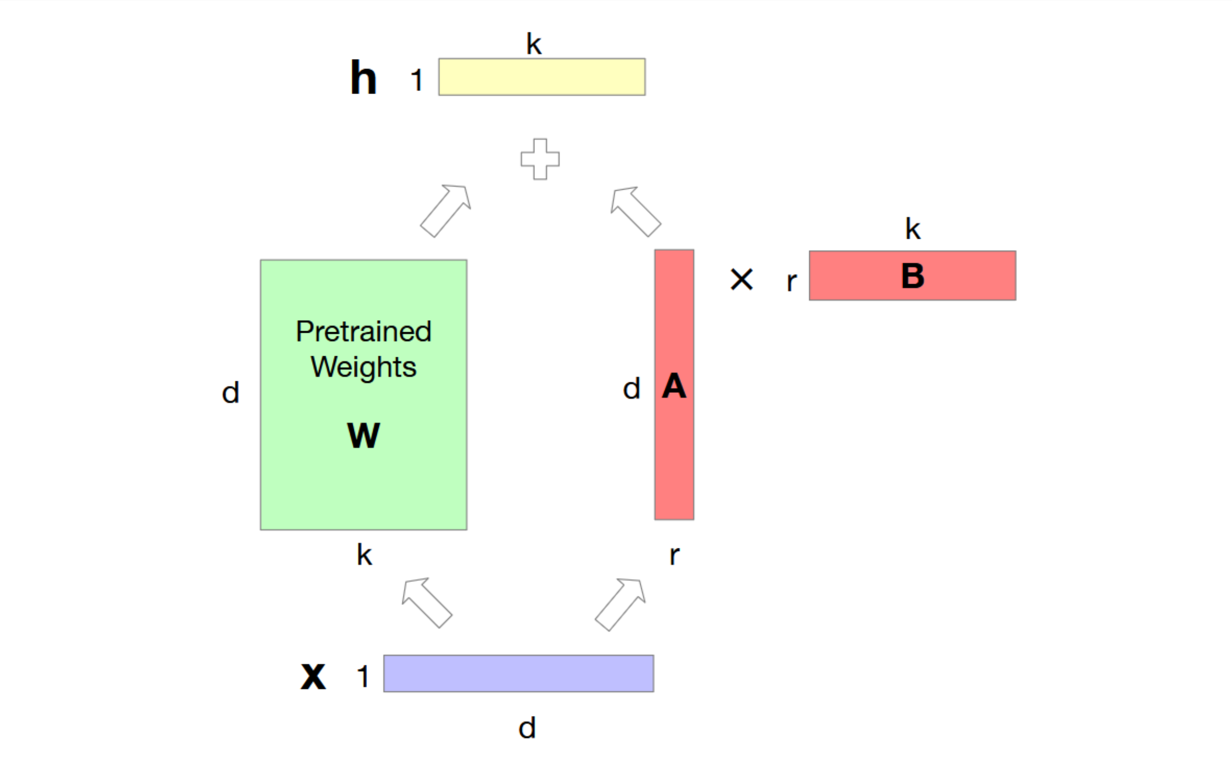


Figure 10.8 The intuition of LoRA. We freeze **W** to its pretrained values, and instead fine-tune by training a pair of matrices **A** and **B**, updating those instead of **W**, and just sum **W** and the updated **AB**.

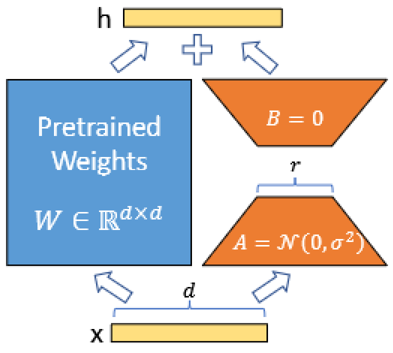
Train a model using a pretrained LLM but give the new model fewer parameters → a low-rank decomposition of the original weight matrix

$$h = xW + xAB$$

$$r << \min(d, N)$$

From SLP book Chapter 10

LoRA (Low-Rank Adaptation)



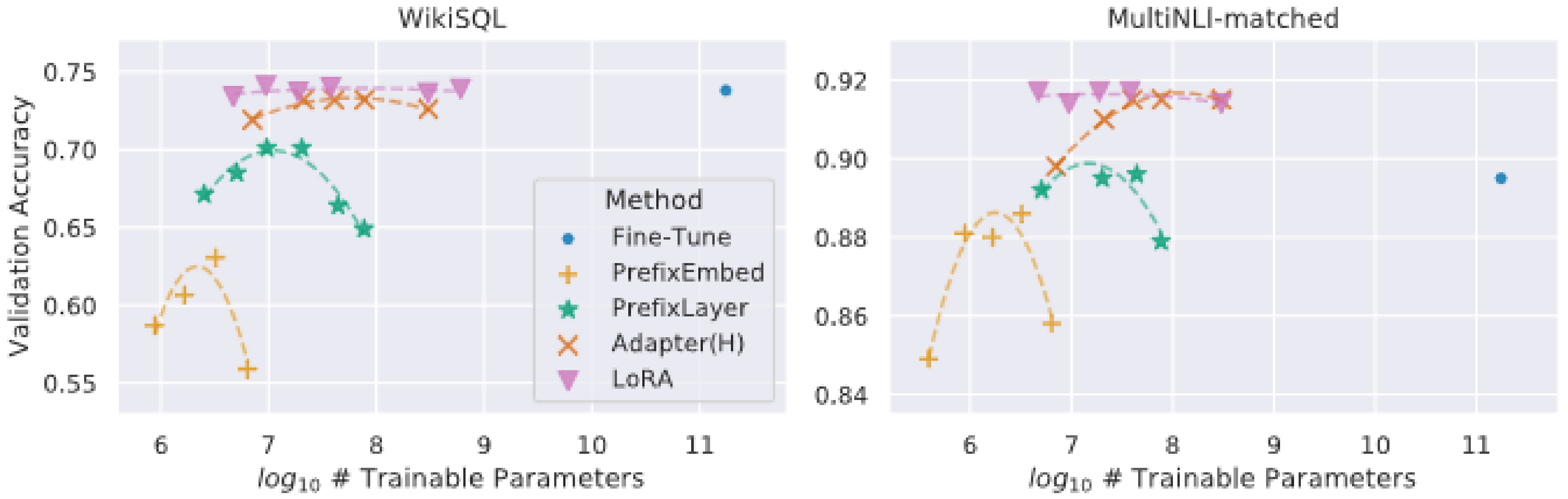
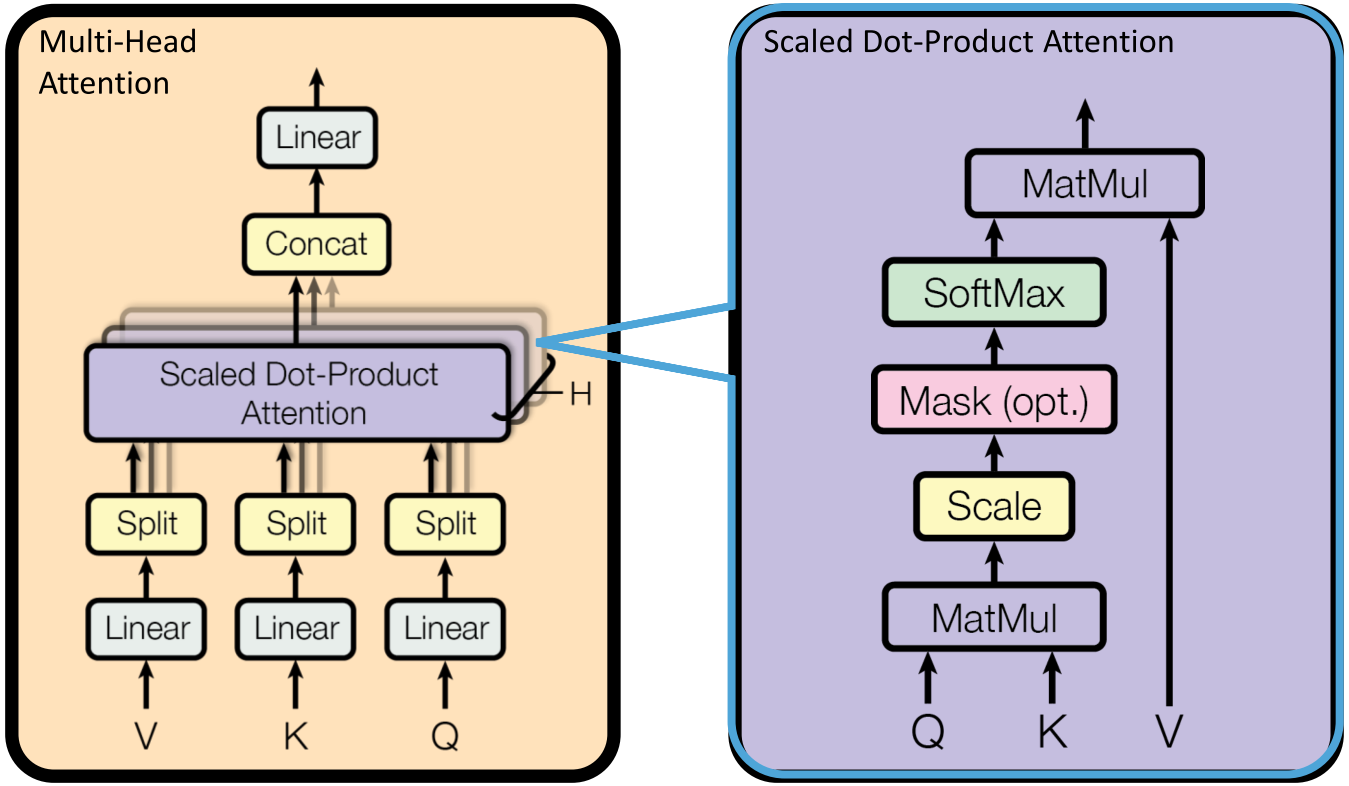


Figure 2: GPT-3 175B validation accuracy vs. number of trainable parameters of several adaptation methods on WikiSQL and MNLI-matched. LoRA exhibits better scalability and task performance. See Section I.2 for more details on the plotted data points.

Implementation:
https://github.com/microsoft/LoRA
https://huggingface.co/docs/diffusers/training/lora

Hu, Edward J., Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. "LoRA: Low-Rank Adaptation of Large Language Models." In *International Conference on Learning Representations (ICLR)*, Virtual. https://iclr.cc/virtual/2022/poster/6319

Review: Attention Mechanism



Original LoRA was just applied to the attention weights:
W_Q, W_K, W_V, and W_O

Guanaco: QLoRA

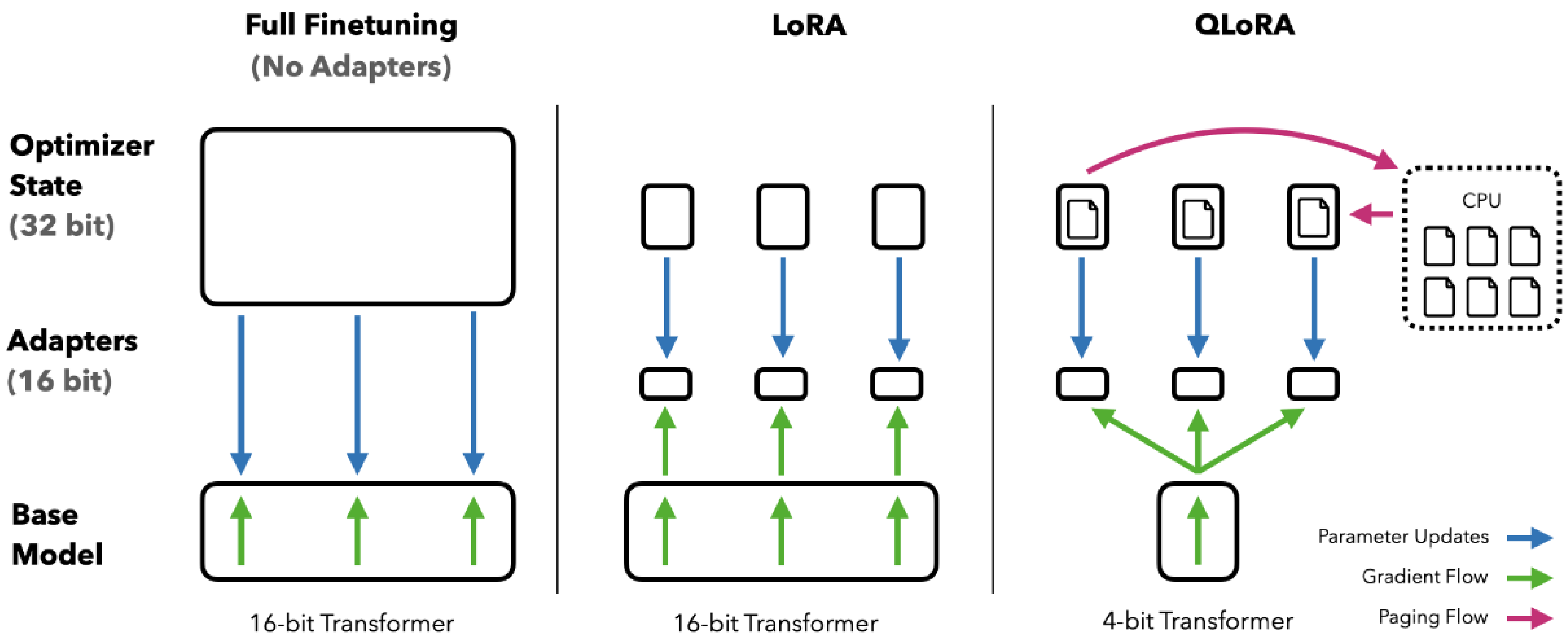


Figure 1: Different finetuning methods and their memory requirements. QLoRA improves over LoRA by quantizing the transformer model to 4-bit precision and using paged optimizers to handle memory spikes.

Dettmers, Tim, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. "QLoRA: Efficient Finetuning of Quantized LLMs." *Advances in Neural Information Processing Systems* 36: 10088–115. https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract-Conference.html

Guanaco: QLoRA

4-bit NormalFloat quantization

Uses *double quantization* (quantizing the quantization constants)

Stored as FP32

$$s = \frac{2^{b-1}-1}{\alpha}$$

Dinesh Raghu

Dettmers, Tim, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. "QLoRA: Efficient Finetuning of Quantized LLMs." *Advances in Neural Information Processing Systems* 36: 10088–115. https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract-Conference.html

Guanaco: QLoRA

Table 3: Experiments comparing 16-bit BrainFloat (BF16), 8-bit Integer (Int8), 4-bit Float (FP4), and 4-bit NormalFloat (NF4) on GLUE and Super-NaturalInstructions. QLoRA replicates 16-bit LoRA and full-finetuning.

Dataset	GLUE (Acc.)	Super-NaturalInstructions (RougeL)				
Model	RoBERTa-large	T5-80M	T5-250M	T5-780M	T5-3B	T5-11B
BF16	88.6	40.1	42.1	48.0	54.3	62.0
BF16 replication	88.6	40.0	42.2	47.3	54.9	-
LoRA BF16	88.8	40.5	42.6	47.1	55.4	60.7
QLoRA Int8	88.8	40.4	42.9	45.4	56.5	60.7
QLoRA FP4	88.6	40.3	42.4	47.5	55.6	60.9
QLoRA NF4 + DQ	-	40.4	42.7	47.7	55.3	60.9

Table 8: Evaluation of biases on the CrowS dataset. A lower score indicates lower likelihood of generating biased sequences. Guanaco follows the biased pattern of the LLaMA base model.

	LLaMA-65B	GPT-3	OPT-175B	Guanaco-65B
Gender	70.6	62.6	65.7	**47.5**
Religion	79.0	73.3	68.6	**38.7**
Race/Color	57.0	64.7	68.6	**45.3**
Sexual orientation	81.0	76.2	78.6	**59.1**
Age	70.1	64.4	67.8	**36.3**
Nationality	64.2	61.6	62.9	**32.4**
Disability	66.7	76.7	76.7	**33.9**
Physical appearance	77.8	74.6	76.2	**43.1**
Socioeconomic status	71.5	73.8	76.2	**55.3**
Average	66.6	67.2	69.5	**43.5**

Dettmers, Tim, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. "QLoRA: Efficient Finetuning of Quantized LLMs." *Advances in Neural Information Processing Systems* 36: 10088–115. https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract-Conference.html

LoRA

PROS

Train models faster and with less computation

Can have domain-specific training (b/c it's like finetuning)

CONS

Decent amount of good data

Picks up quickly on biases, etc.

Efficient Training

Review: Transformer Architecture

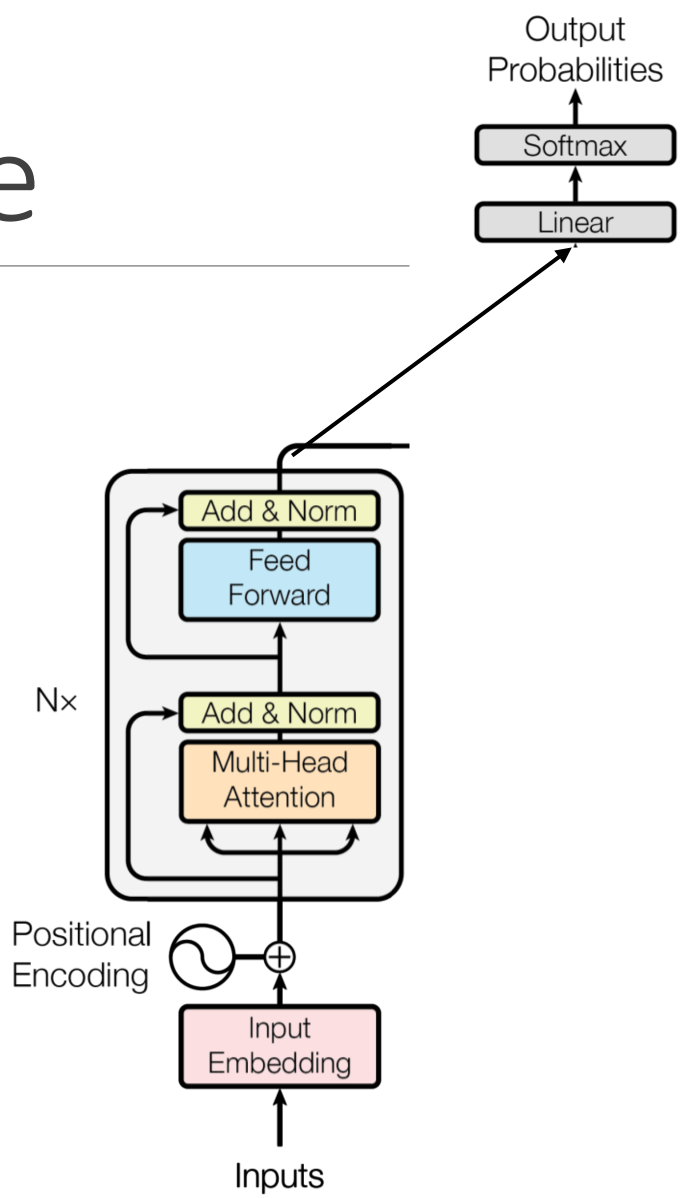
$\boxed{\text{Multi-Head Attention}} = \text{MultiHeadAtt}(\mathbf{H}_i^{enc}, \mathbf{H}_i^{enc}, \mathbf{H}_i^{enc})$

$\boxed{\text{Add \& Norm}} = \text{LayerNorm}(\boxed{\text{Multi-Head Attention}} + \mathbf{H}_i^{enc})$

$\boxed{\text{Feed Forward}} = max(0, \boxed{\text{Add \& Norm}}\mathbf{W}_1 + b_1)\mathbf{W}_2 + b_2$

$\boxed{\text{Add \& Norm (2)}} = \text{LayerNorm}(\boxed{\text{Feed Forward}} + \mathbf{H}_i^{enc})$

$\mathbf{H}_{i+1}^{enc} = \boxed{\text{Add \& Norm (2)}}$



A decoder-only architecture is very similar to the encoder of the original transformer architecture

Slide adapted from Daphne Ippolito

Sparse Mixture-of-Experts

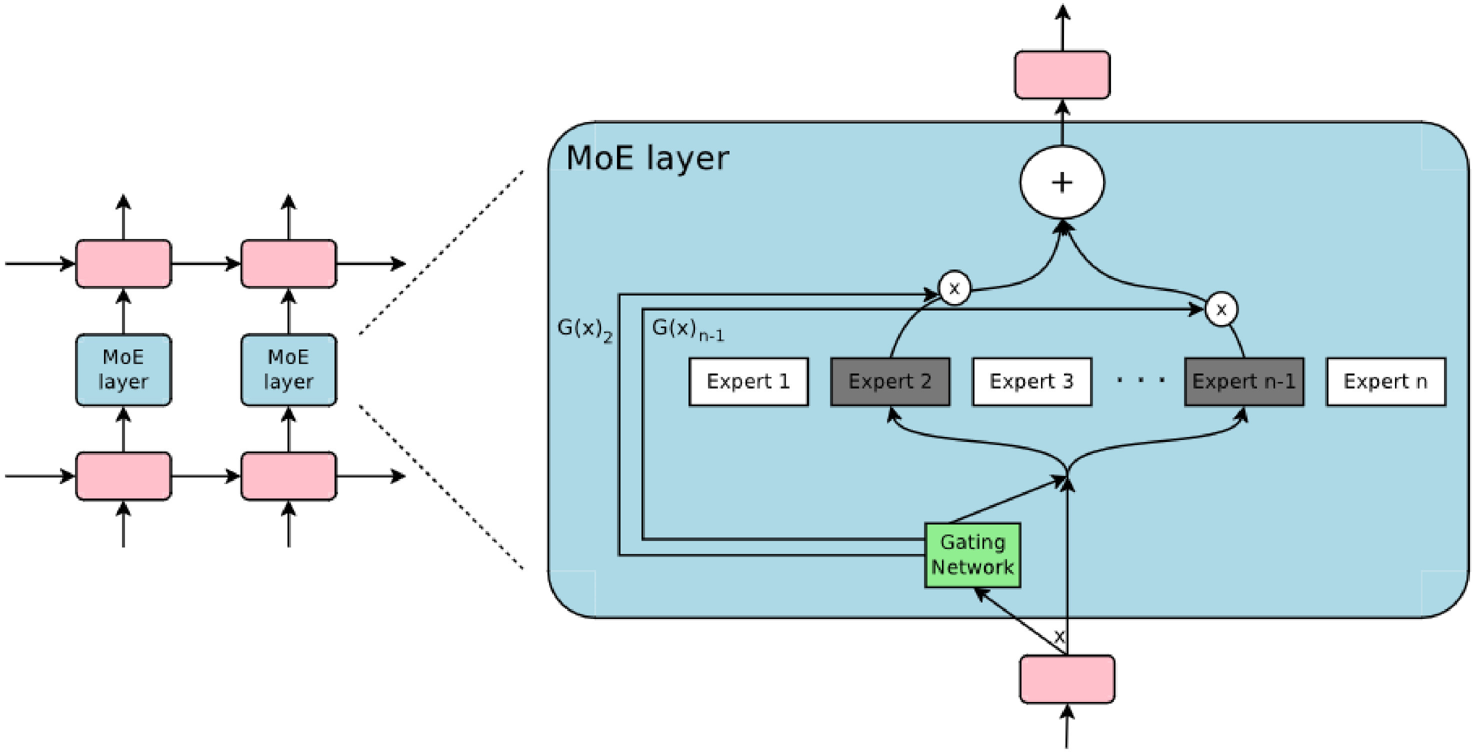


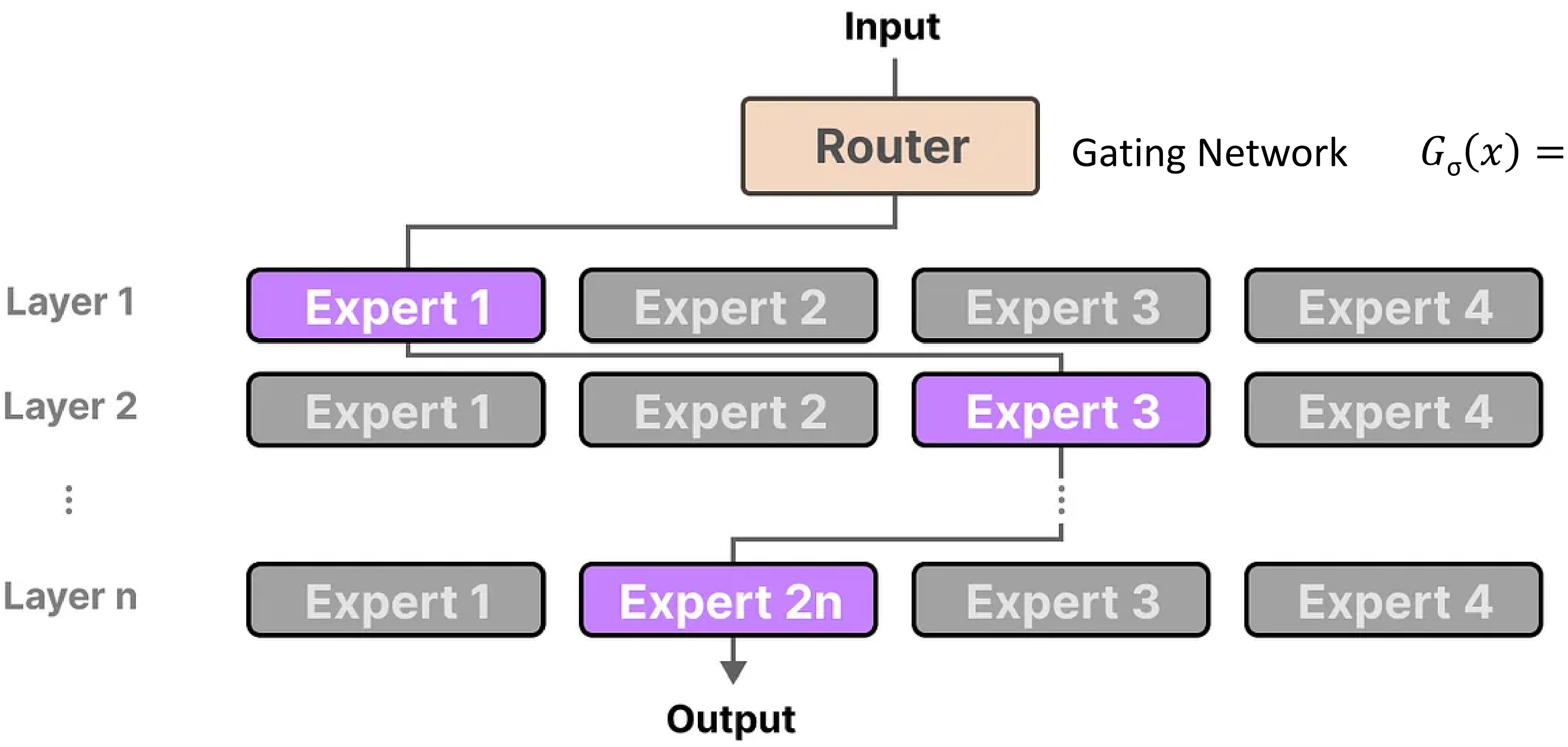
Figure 1: A Mixture of Experts (MoE) layer embedded within a recurrent language model. In this case, the sparse gating function selects two experts to perform computations. Their outputs are modulated by the outputs of the gating network.

Implementations
Megablocks: https://github.com/stanford-futuredata/megablocks
Fairseq: https://github.com/facebookresearch/fairseq/tree/main/examples/moe_lm
OpenMoE: https://github.com/XueFuzhao/OpenMoE

Shazeer, Noam, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. "Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer." In *International Conference on Learning Representations (ICLR)*, Toulon, France. https://openreview.net/forum?id=B1ckMDqlg

Mixture-of-Experts

$$y = \sum_{i=1}^{n} G(x)_i E_i(x)$$

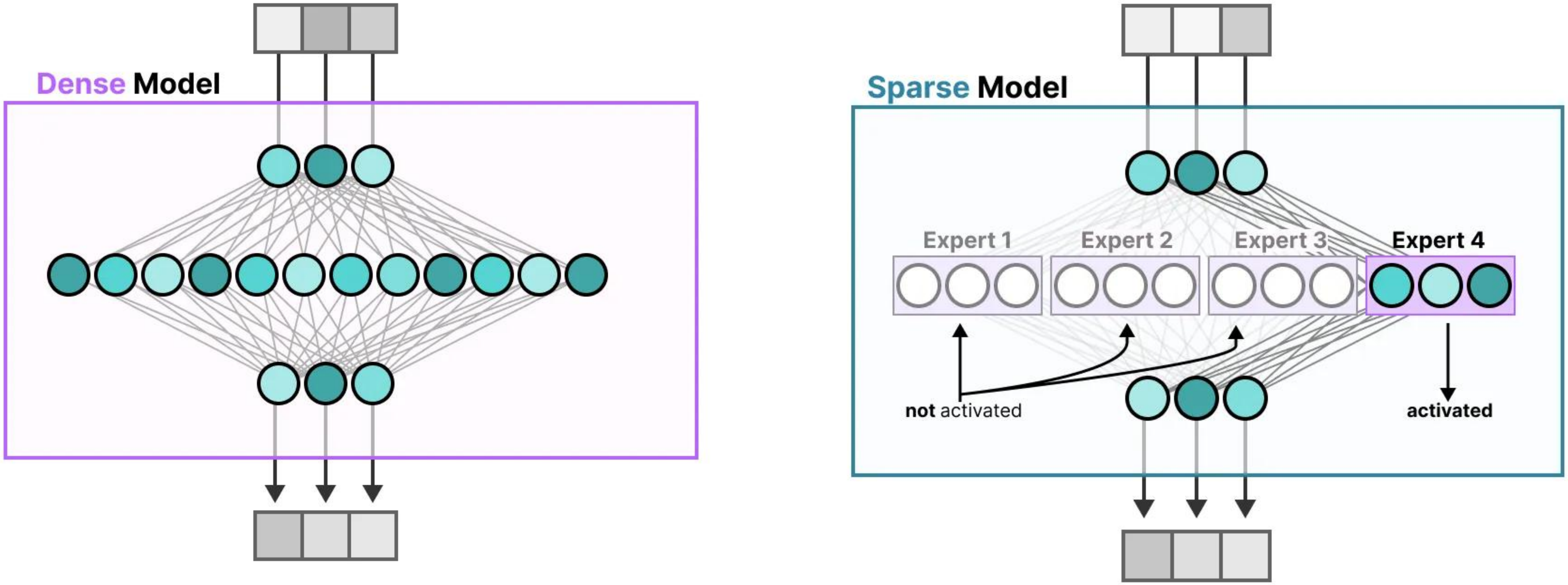


Gating Network $G_\sigma(x) = Softmax(x \cdot W_g)$

https://substack.com/home/post/p-148217245

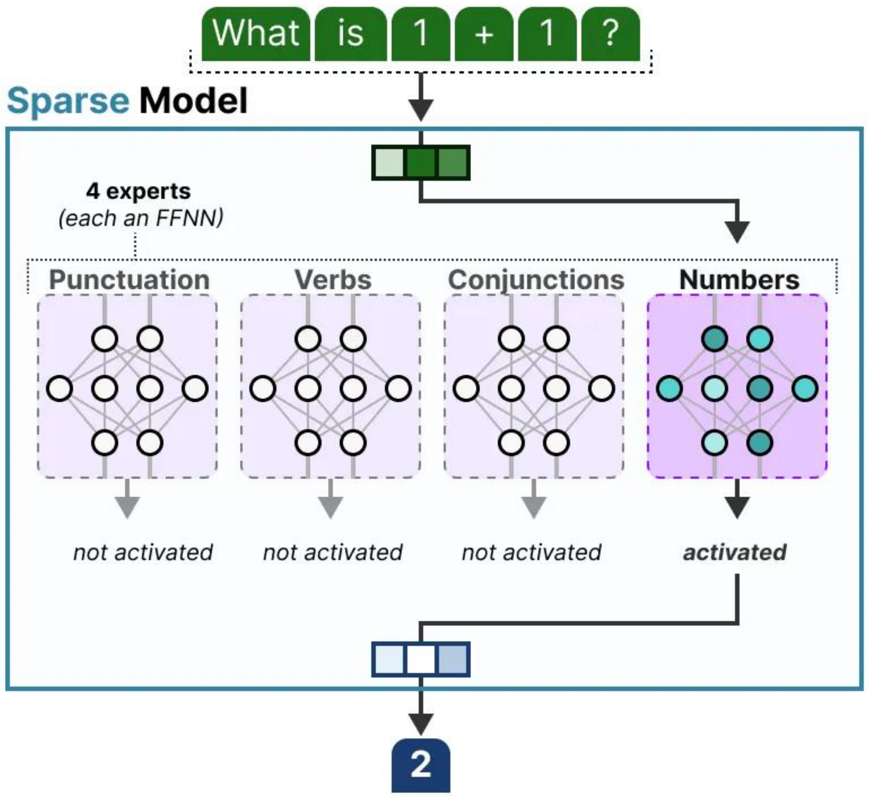
Equations from https://huggingface.co/blog/moe

Sparse MoE

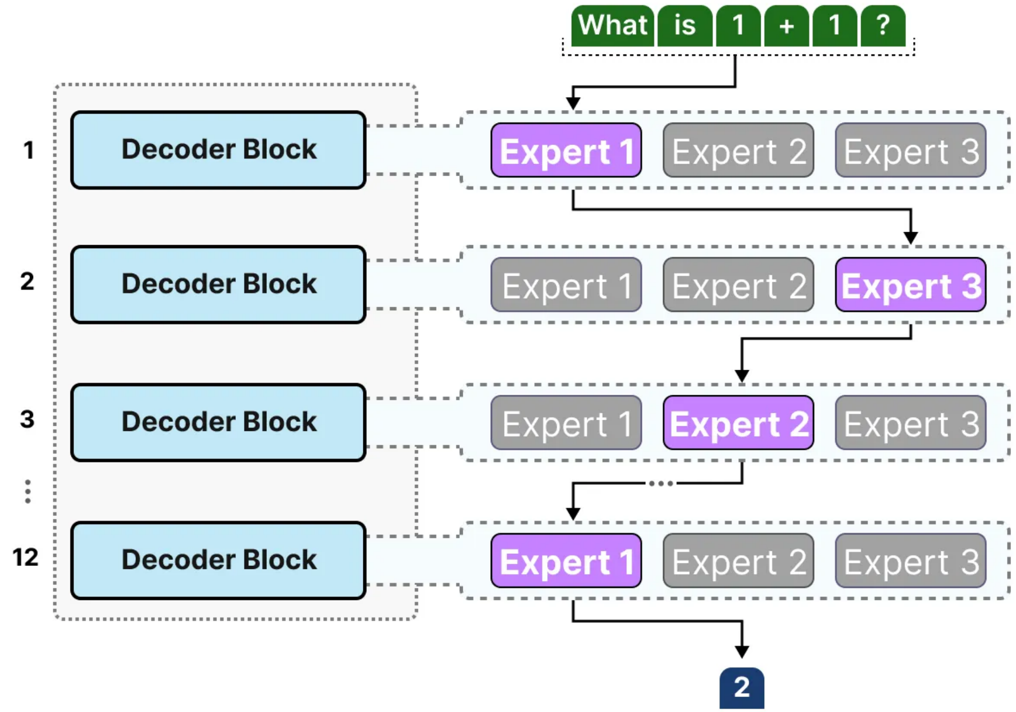


https://substack.com/home/post/p-148217245

MoE Example



Most transformers have multiple decoder blocks



https://substack.com/home/post/p-148217245

8 experts considered
Top 2 experts Selected

Mixtral 8x7B

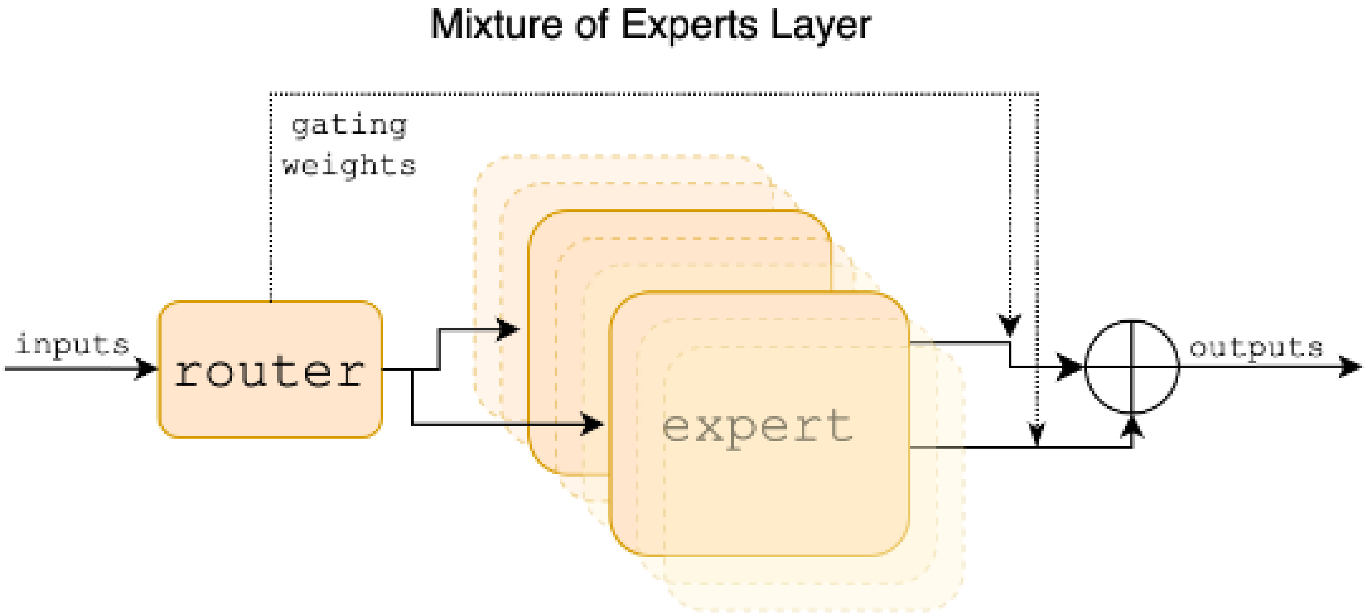


Figure 1: Mixture of Experts Layer. Each input vector is assigned to 2 of the 8 experts by a router. The layer's output is the weighted sum of the outputs of the two selected experts. In Mixtral, an expert is a standard feedforward block as in a vanilla transformer architecture.

Jiang, Albert Q., Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, et al. 2024. "Mixtral of Experts." doi:10.48550/arXiv.2401.04088.

Mixtral is 13B active parameters vs LLaMA 2's 70B

Mixtral 8x7B

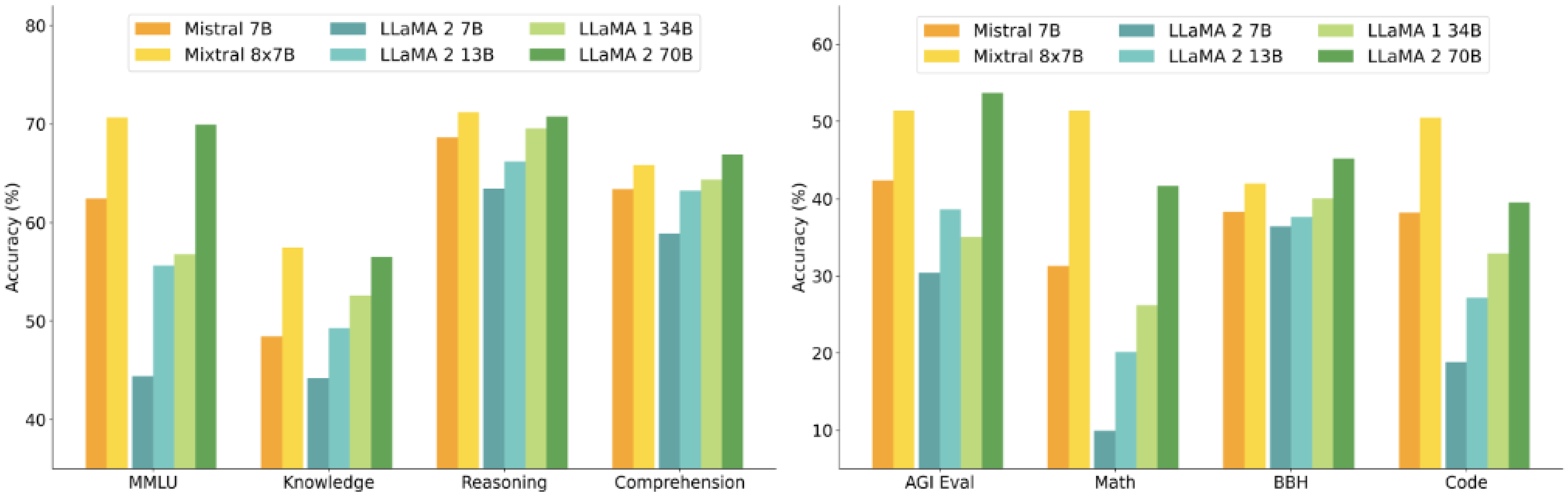
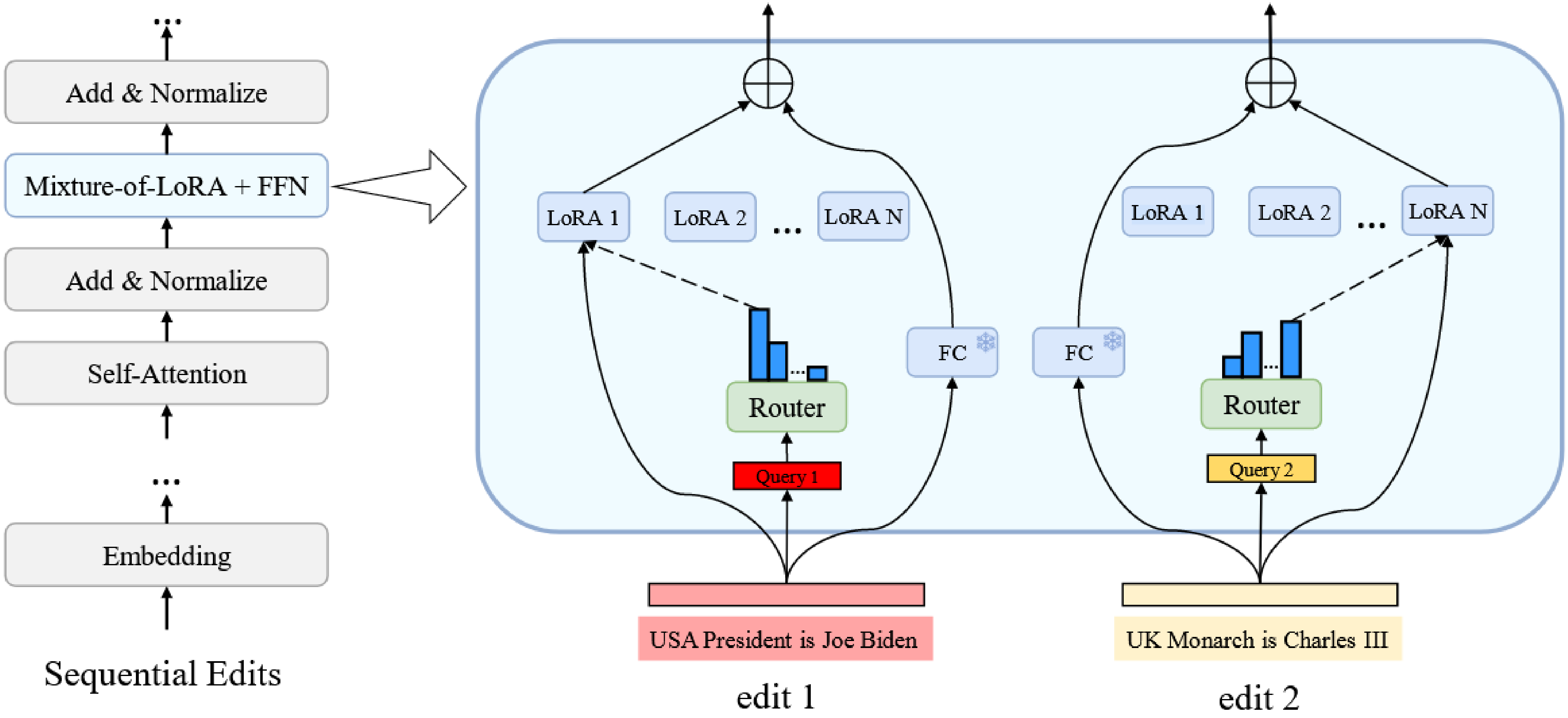


Figure 2: Performance of Mixtral and different Llama models on a wide range of benchmarks. All models were re-evaluated on all metrics with our evaluation pipeline for accurate comparison. Mixtral outperforms or matches Llama 2 70B on all benchmarks. In particular, it is vastly superior in mathematics and code generation.

Jiang, Albert Q., Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, et al. 2024. "Mixtral of Experts." doi:10.48550/arXiv.2401.04088.

ELDER: Mixture-of-LoRA



Li, Jiaang, Quan Wang, Zhongnan Wang, Yongdong Zhang, and Zhendong Mao. 2025. "ELDER: Enhancing Lifelong Model Editing with Mixture-of-LoRA." *Proceedings of the AAAI Conference on Artificial Intelligence* 39(23): 24440–48. doi:10.1609/aaai.v39i23.34622.

Mixture of Experts

PROS

Efficient training of a brand new model

Can “semantically group” ideas as it goes through the layers

CONS

Needs a variety of experts; how many experts is a good number?

Need as much data as a full LLM

Knowledge Check

When would you want to use Knowledge Distillation (training a student), PEFT/LoRA, Model Compression (quantizing/pruning), or MoE?

(Assuming they're not combined)