中国科学院大学
University of Chinese Academy of Sciences

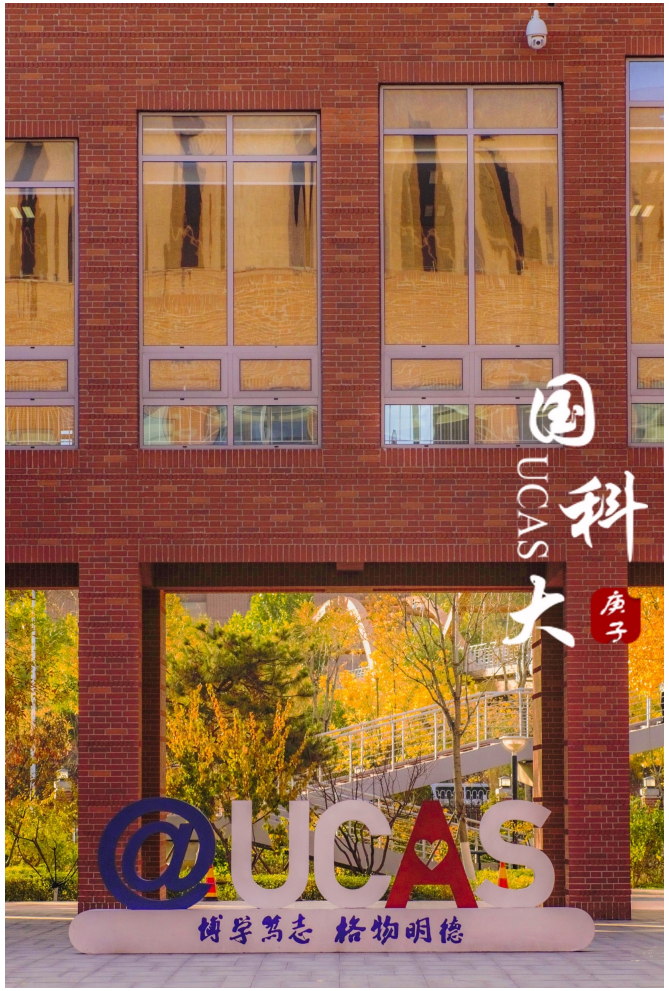
自然语言处理

第8讲 推理优化

王 石 资康莉 刘 瑜

2026年春季课程

https://ictkc.github.io/teaching/



第八讲 推理优化

天下武功，唯快不破


天下武功
无坚不破，唯快不破

中国科学院大学
University of Chinese Academy of Sciences

目 录

1 KV缓存及优化

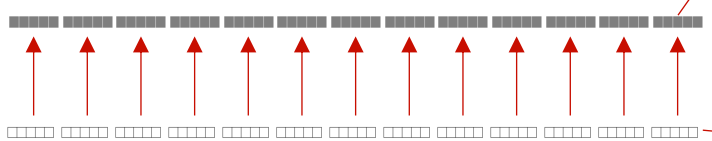
2

3

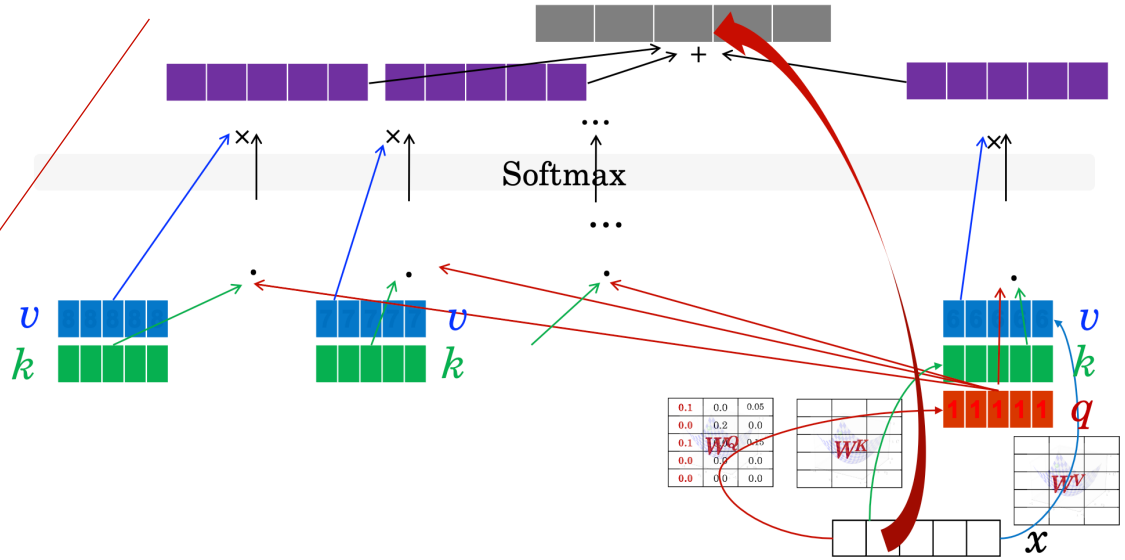
4

大模型推理过程拆解

- **Step1 – 输入读取**：**一次性读入**所有输入（提示词）Tokens的Embeddings，基于MHA计算每个token在其上文环境中的动态编码，生成第一个Token
  - 是**完全并行化的高效矩阵运算**
  - 决定用户**多久“看到第一个字”**
    TTFT（Time To First Token）



翻译英文：我爱自然语言处理 I

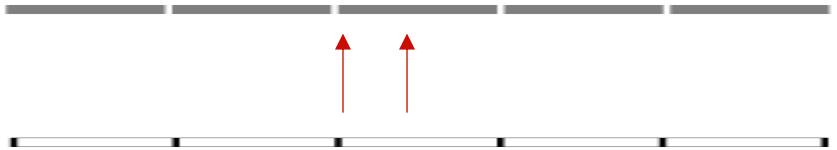


大模型推理过程拆解

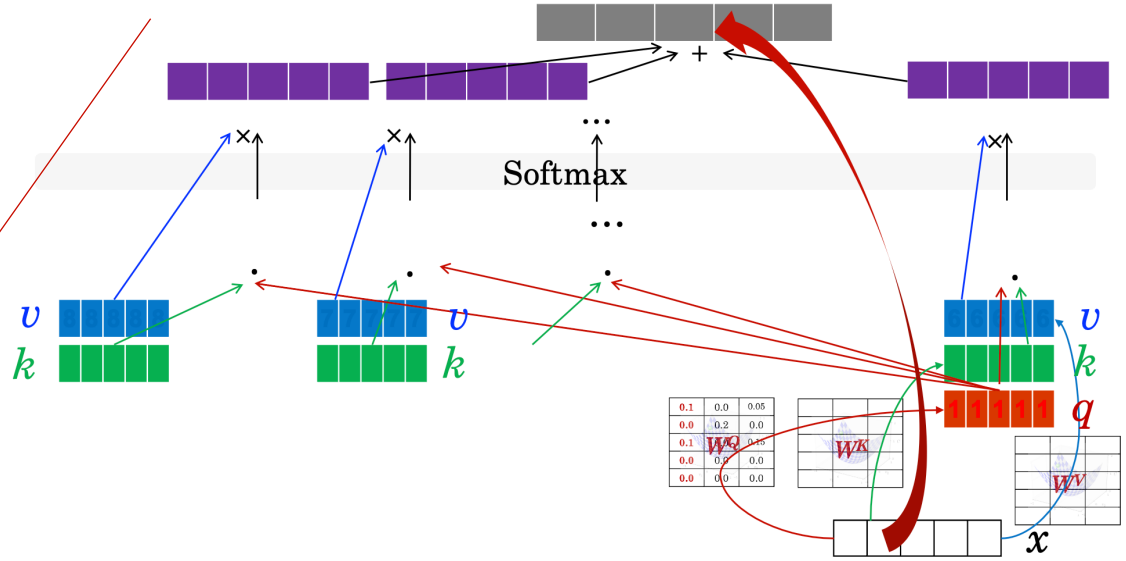
- **Step2 – 输出生成**：**逐个生成**答案Token。每个输出Token都依赖所有的上文（包括上一个刚生成的Token），因此无法并行

  - 是**串行化的逐字矩阵运算**
  - 决定**“答案生成是否流畅”**

TPOT（Time Per Output Token）

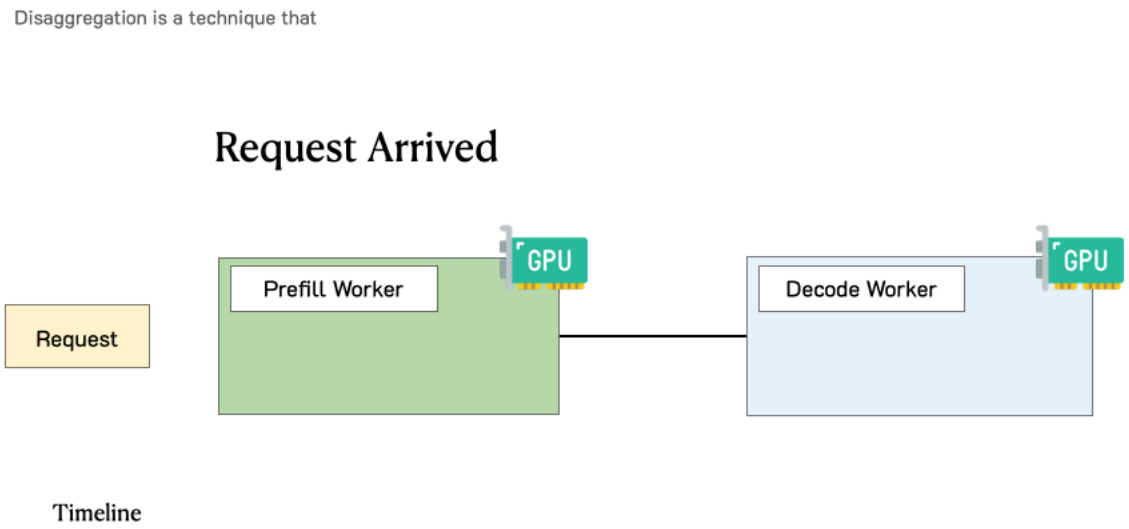


翻译英文：我爱自然语言处理 I Love NLP



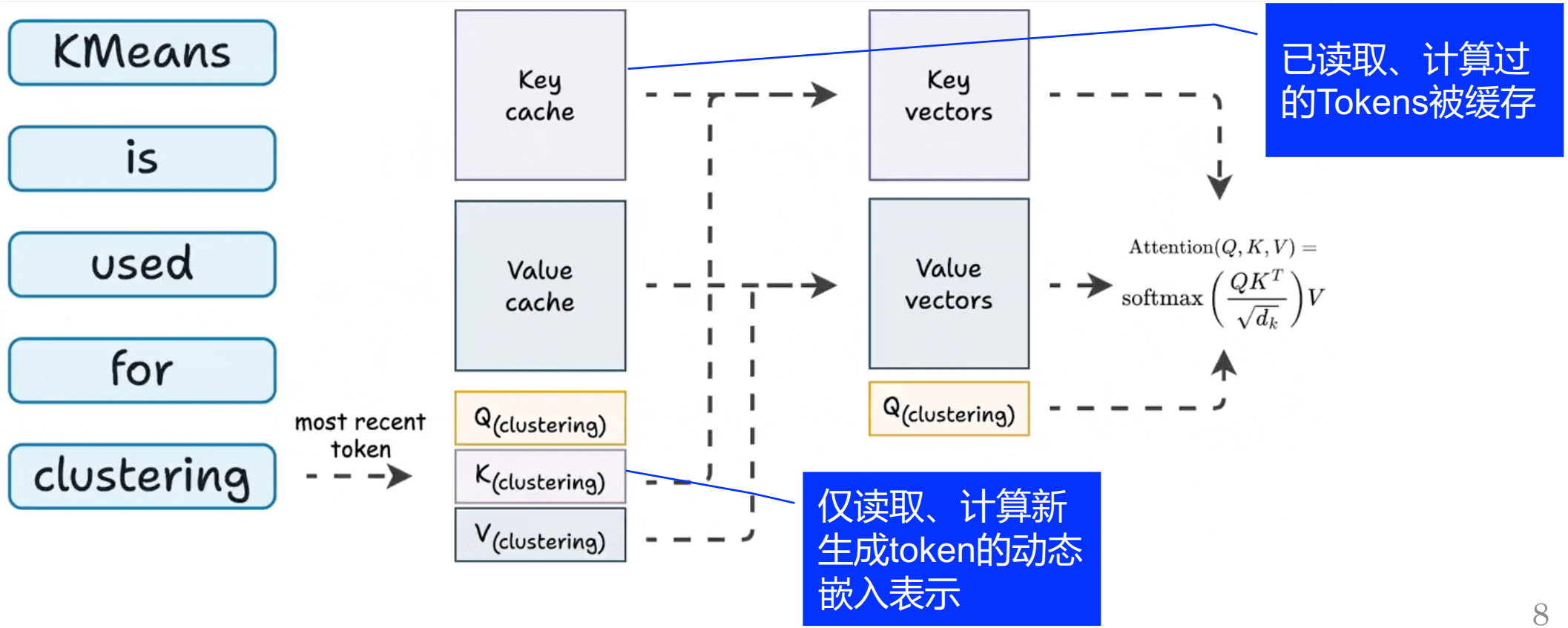
大模型推理过程拆解

- Step1 – 一次性输入读取：预填充（Prefill）
- Step2 – 逐个词答案生成：解码（Decode）



KV缓存

- 将上文Token的KV缓存到GPU内存中，避免每次重复计算



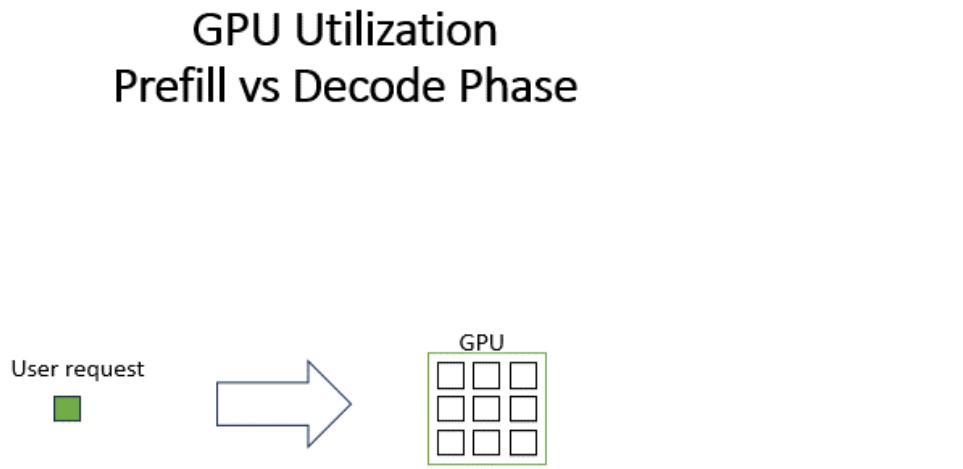
KV缓存所需内存计算

- LLM对GPU显存的需求主要是**模型权重和KV缓存**
- 以Llama2-7B（$7*10^9$）模型为例
  - 模型权重（权重以16位浮点数存储）：7B * sizeof(FP16) ~= **14G**
  - KV缓存：
    - 每Token字节数 = 2 * 网络层数 * 注意力头数 * 注意力头维度 * K/V字节数
    - 总的KV缓存大小 = **batch_size** * 上文token数 * 每个Token字节数
    - Llama2-7B为：1 * 4096 * 2 * 32 * 4096 * 2 ~= **2G**

KV Cache 成了显存和带宽的大头

batch_size：批次大小

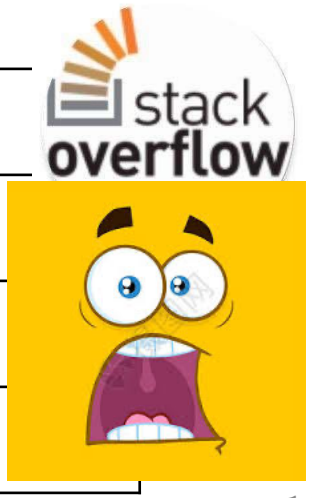
- 推理阶段batch_size是指一次向模型并发提出几个请求，并行请求可以充分利用GPU计算能力，但batch_size太大会导致显存溢出



上文Token数

- 也称“上下文长度”，是模型单次推理能处理的最大Token数，即“短期记忆容量”，直接影响长文档理解、多轮对话连贯等

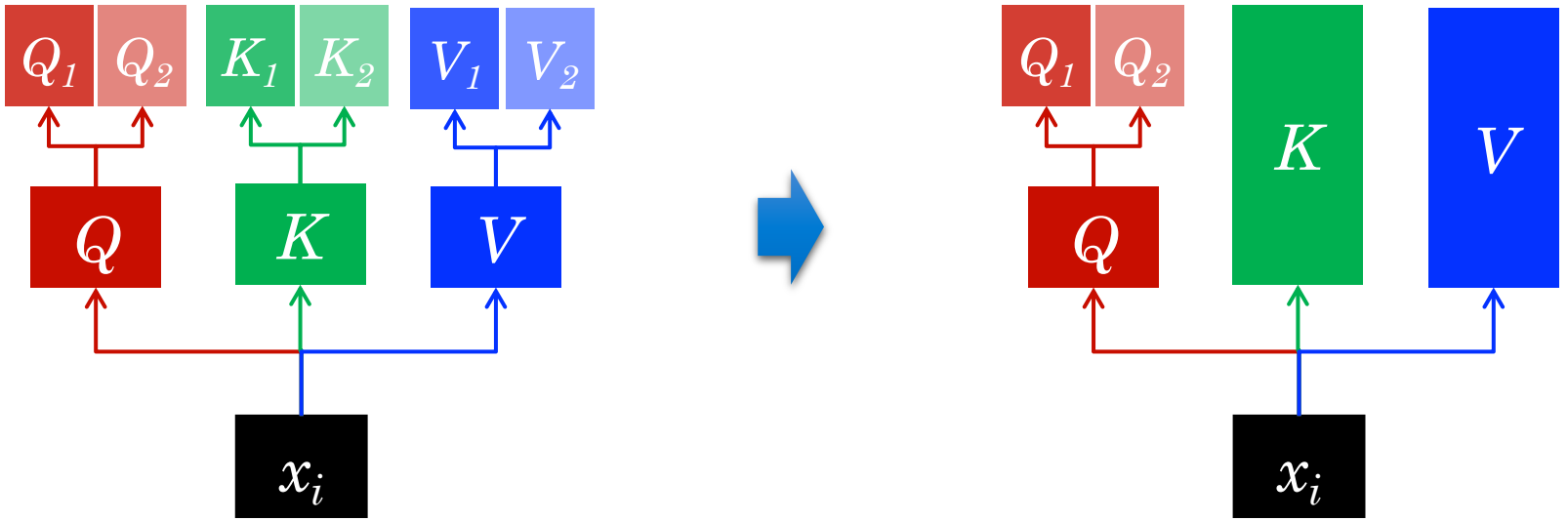
模型	上下文长度	KV缓存大小 (按Llama2 7B公式计算)
Grok-4	256K tokens	128G
GPT-5 (ChatGPT 5)	400K tokens	200G
Claude Opus 4.x / Sonnet 4	200K tokens	100G
Qwen3 Coder / Qwen3 Max	1M / 128K	496G/62G
DeepSeek-V3 / R1	64K tokens	32G



怎么降低KV大小?

1. 多查询注意力（Multi-Query Attention）

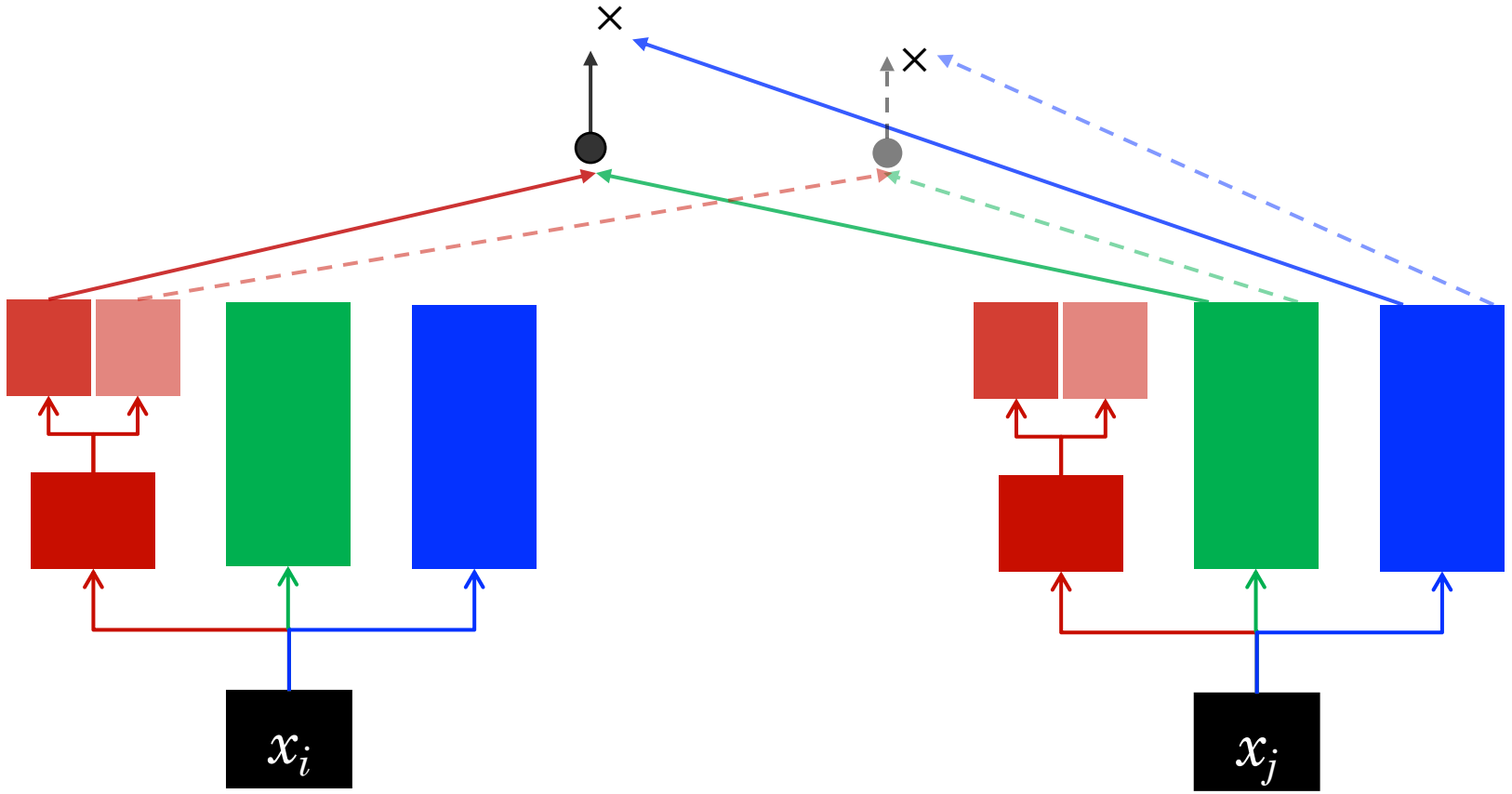
- 多个注意力头之间**共享K、V**



本质上是把K、V的特征混合编码，但由于Q不同，最终也能实现“一词多义”的动态注意力表示

1. 多查询注意力（Multi-Query Attention）

- 多个注意力头之间**共享K、V**



1. 多查询注意力（Multi-Query Attention）

Fast transformer decoding: One write-head is all you need

N Shazeer - arXiv preprint arXiv:1911.02150, 2019 - arxiv.org

Multi-head attention layers, as used in the Transformer neural sequence model, are a powerful alternative to RNNs for moving information across and between sequences. While …

☆ 保存　99 引用　被引用次数：762　相关文章　所有 2 个版本　»

[PDF] Attention is all you need

A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - digitalyouth-hub.com

… to attend to all positions in the decoder up to and including that position. We need to prevent … We implement this inside of scaled dot-product attention by masking out (setting to −∞) …

☆ 保存　99 引用　被引用次数：5676　相关文章　»

1. 多查询注意力（Multi-Query Attention）

诺姆·沙泽尔 (Noam Shazeer)

1. 多查询注意力（Multi-Query Attention）

- KV读取规模降低，推理速度提升明显

Attention Type	Training	Inference enc. + dec.	Beam-4 Search enc. + dec.
multi-head	13.2	1.7 + 46	2.0 + 203
multi-query	**13.0**	1.5 + 3.8	1.6 + 32
multi-head local	13.2	1.7 + 23	1.9 + 47
multi-query local	**13.0**	**1.5 + 3.3**	**1.6 + 16**

1. 多查询注意力（Multi-Query Attention）

- 缓存**线性降低**
  - 每个Token字节数 = 2 * 网络层数 * **注意力头数** * 注意力头维度 * K/V权重字节数
  - 总的KV缓存大小 = **batch_size** * 上文token数 * 每个Token字节数

1. 多查询注意力（Multi-Query Attention）

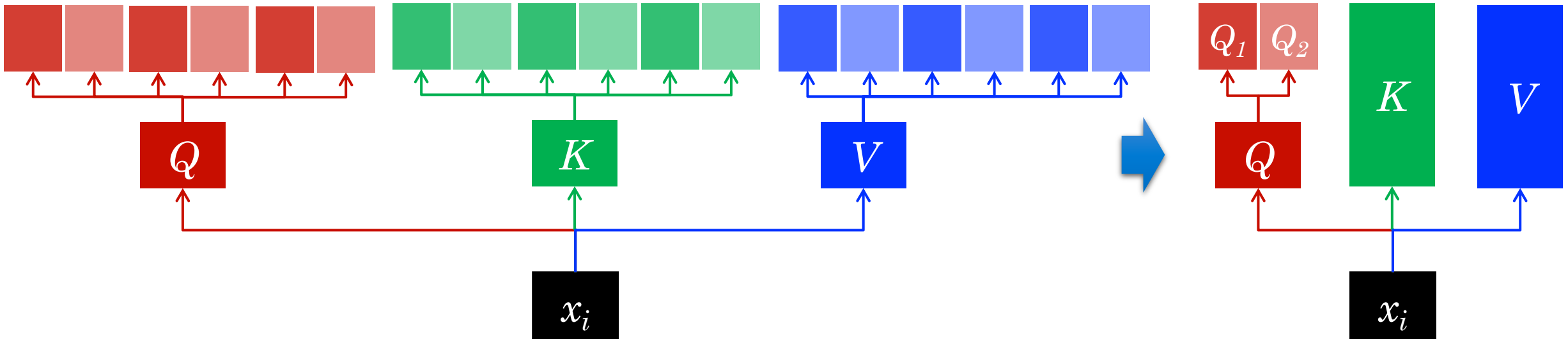
- 在某些实验中效果很好

Table 3: Billion-Word LM Benchmark Results.

Attention	h	d_k, d_v	d_{ff}	dev-PPL
multi-head	8	128	8192	**29.9**
multi-query	8	128	9088	30.2
multi-head	1	128	9984	31.2
multi-head	2	64	9984	31.1
multi-head	4	32	9984	31.0
multi-head	8	16	9984	30.9

1. 多查询注意力（Multi-Query Attention）

- 实际上在另外试验中效果下降较大



本质上是把K、V的特征**混合编码**：可以混，但不能混太多

2. 分组查询注意力（Grouped-query Attention）

- Q分组，每个组共享K、V

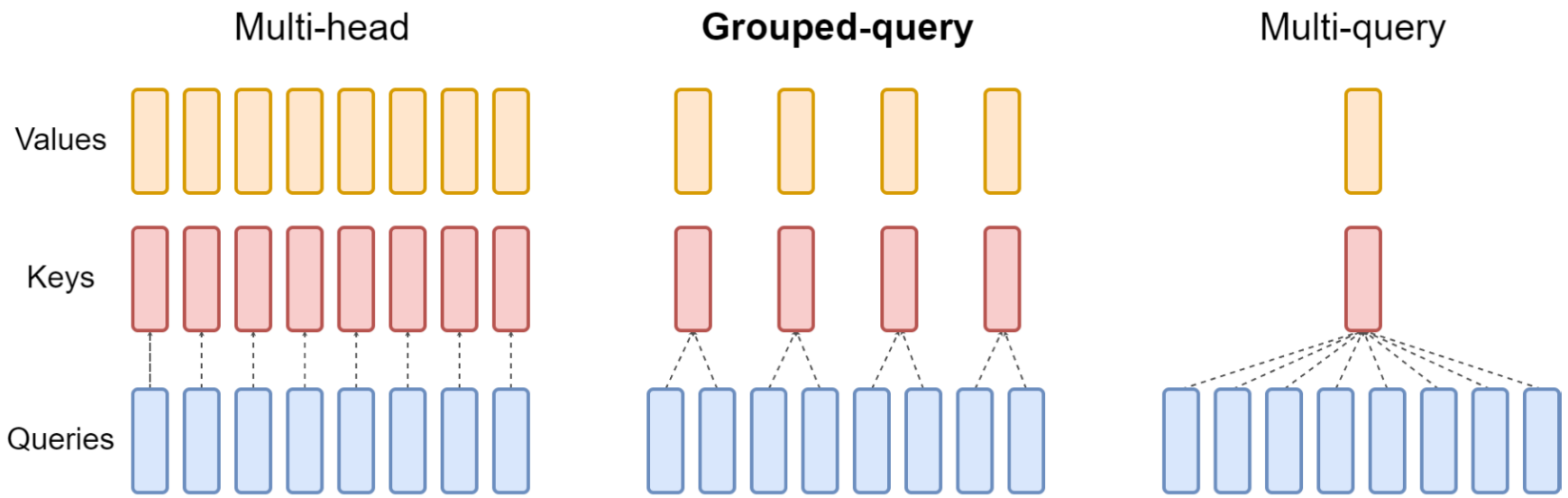


Figure 2: Overview of grouped-query method. Multi-head attention has H query, key, and value heads. Multi-query attention shares single key and value heads across all query heads. Grouped-query attention instead shares single key and value heads for each *group* of query heads, interpolating between multi-head and multi-query attention.

2. 分组查询注意力（Grouped-query Attention）

- 推理速度与MQA持平，效果与MHA持平

Model	T_{infer}	Average	CNN	arXiv	PubMed	MediaSum	MultiNews	WMT	TriviaQA
	s		R_1	R_1	R_1	R_1	R_1	BLEU	F1
MHA-Large	0.37	46.0	42.9	44.6	46.2	35.5	46.6	27.7	78.2
MHA-XXL	1.51	47.2	43.8	45.6	47.5	36.4	46.9	28.4	81.9
MQA-XXL	0.24	46.6	43.0	45.0	46.9	36.1	46.5	28.5	81.3
GQA-8-XXL	0.28	47.1	43.5	45.4	47.7	36.3	47.2	28.4	81.6

Table 1: Inference time and average dev set performance comparison of T5 Large and XXL models with multi-head attention, and 5% uptrained T5-XXL models with multi-query and grouped-query attention on summarization datasets CNN/Daily Mail, arXiv, PubMed, MediaSum, and MultiNews, translation dataset WMT, and question-answering dataset TriviaQA.

使用GQA 的模型包括 **LLAMA2-70B、LLAMA3 、DeepSeek-V1、ChatGLM2、ChatGLM3、Qwen2**等

3. 多头潜在注意力（Multi-head Latent Attention）

deepseek

DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model

DeepSeek-AI

research@deepseek.com

Contents

1 Introduction 4

2 Architecture 6

2.1 Multi-Head Latent Attention: Boosting Inference Efficiency 6

2.1.1 Preliminaries: Standard Multi-Head Attention 6

2.1.2 Low-Rank Key-Value Joint Compression 7

2.1.3 Decoupled Rotary Position Embedding 8

2.1.4 Comparison of Key-Value Cache 8

3. 多头潜在注意力（Multi-head Latent Attention）

把KV压缩存储，使用时解压

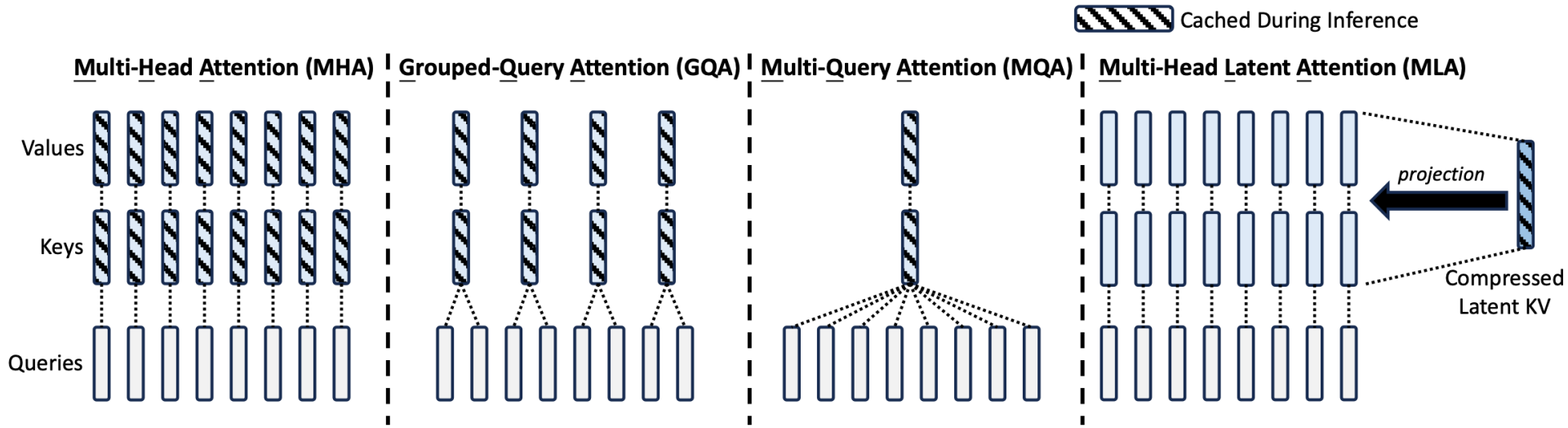
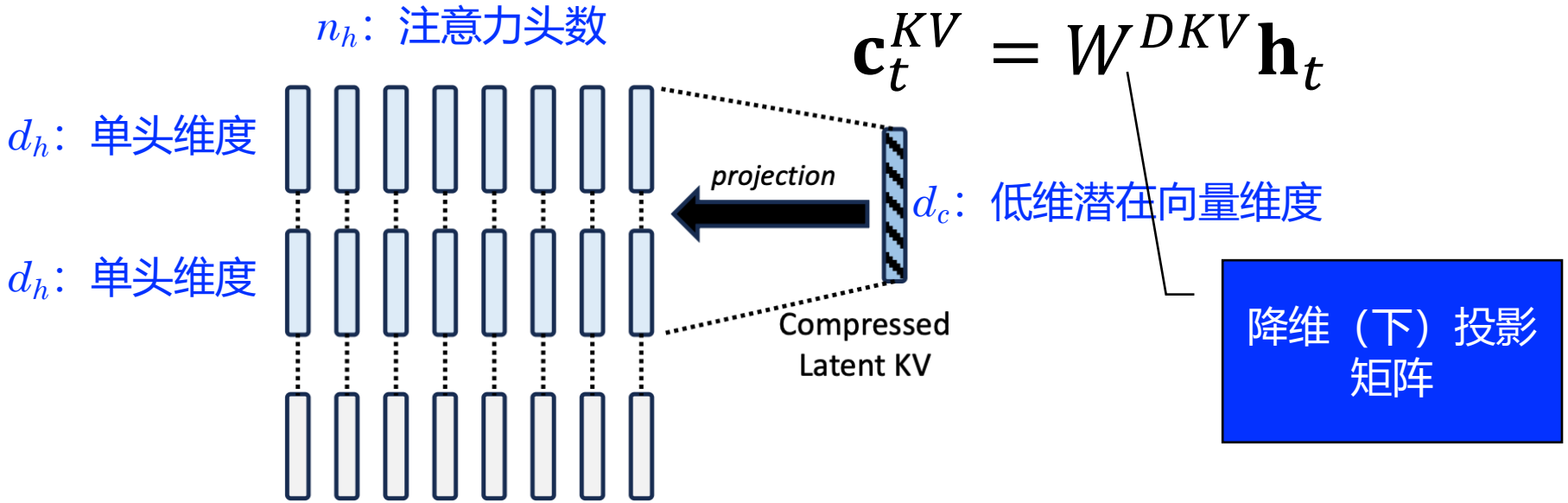


Figure 3 | Simplified illustration of Multi-Head Attention (MHA), Grouped-Query Attention (GQA), Multi-Query Attention (MQA), and Multi-head Latent Attention (MLA). Through jointly compressing the keys and values into a latent vector, MLA significantly reduces the KV cache during inference.

3. 多头潜在注意力（Multi-head Latent Attention）

把KV压缩存储，使用时解压

- Step1：将KV投影到低维潜在向量$\mathbf{c}_t^{KV}$，其维度$d_c \ll d_h n_h$



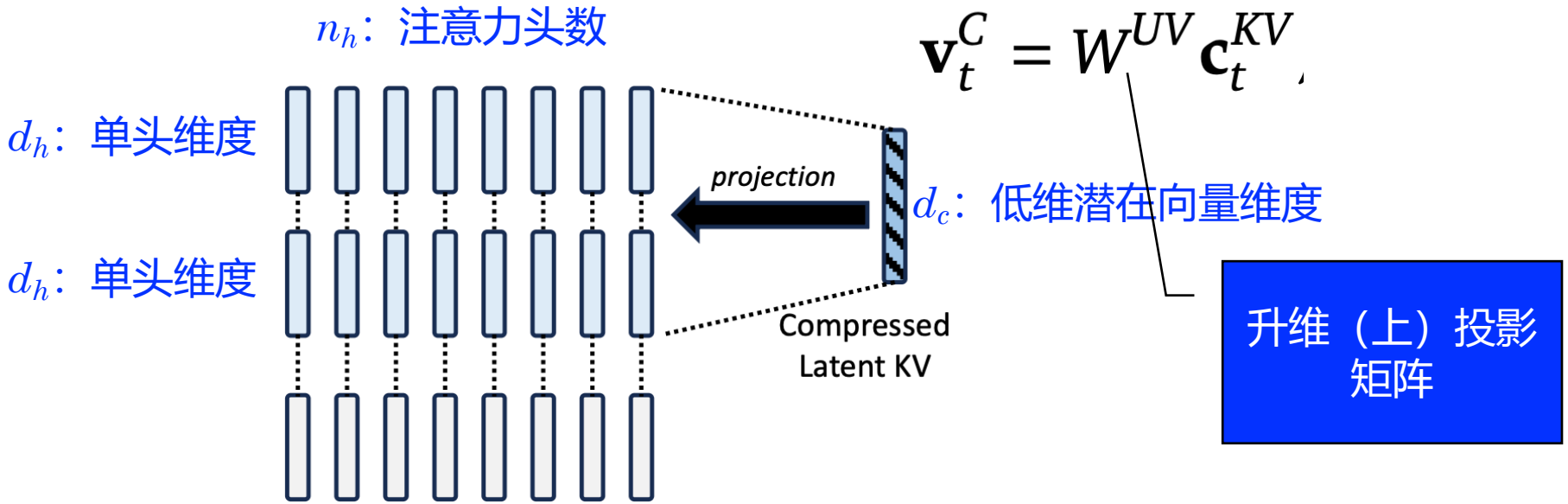
3. 多头**潜在**注意力（Multi-head Latent Attention）

把KV压缩存储，使用时解压

- Step2：将低维潜在向量$\mathbf{c}_t^{KV}$ **近似恢复**至KV

$$\mathbf{k}_t^C = W^{UK}\mathbf{c}_t^{KV}$$

$$\mathbf{v}_t^C = W^{UV}\mathbf{c}_t^{KV}$$



3. 多头潜在注意力（Multi-head Latent Attention）

把KV压缩存储，使用时解压

存储大小

Attention Mechanism	KV Cache per Token (# Element)	Capability
Multi-Head Attention (MHA)	$2n_h d_h l$	Strong
Grouped-Query Attention (GQA)	$2n_g d_h l$	Moderate
Multi-Query Attention (MQA)	$2d_h l$	Weak
MLA (Ours)	$(d_c + d_h^R)l \approx \frac{9}{2} d_h l$	Stronger

4.5*单头维度

推理速度

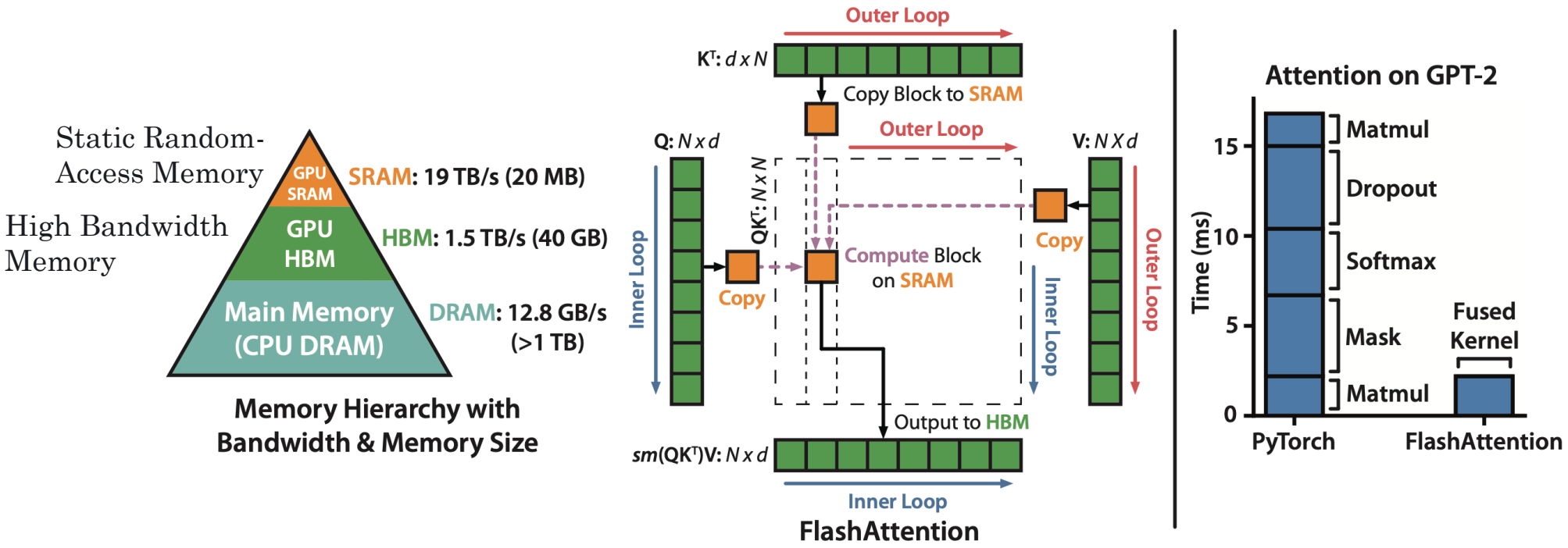
service. On a single node with 8 H800 GPUs, DeepSeek-V2 achieves a generation throughput exceeding 50K tokens per second, which is 5.76 times the maximum generation throughput of DeepSeek 67B. In addition, the prompt input throughput of DeepSeek-V2 exceeds 100K tokens per second.

小结

- KV Cache：给大模型配了“备忘录”，不用每次从头计算
- MHA → MQA/GQA → MLA：让“备忘录”既薄又好用：
  - MHA：每个人写自己的长篇备忘
  - MQA：整个团队只留一本备忘录
  - GQA：分小组，各组一本备忘录
  - MLA：备忘录压缩成“提醒摘要”，要看备忘时按需展开

还没完：4. Flash Attention

- 将QKV分割成小块在SRAM中处理，避免频繁访问HBM



https://arxiv.org/pdf/2205.14135

还没完：4. Flash Attention

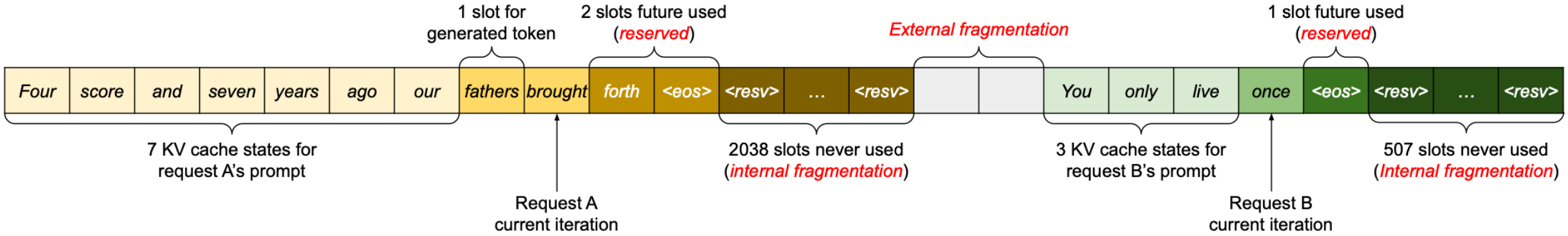
- 将QKV分割成小块在SRAM中处理，避免频繁访问HBM

Model implementations	OpenWebText (ppl)	Training time (speedup)
GPT-2 small - Huggingface [87]	18.2	9.5 days (1.0×)
GPT-2 small - Megatron-LM [77]	18.2	4.7 days (2.0×)
GPT-2 small - FlashAttention	18.2	**2.7 days (3.5×)**
GPT-2 medium - Huggingface [87]	14.2	21.0 days (1.0×)
GPT-2 medium - Megatron-LM [77]	14.3	11.5 days (1.8×)
GPT-2 medium - FlashAttention	14.3	**6.9 days (3.0×)**

https://arxiv.org/pdf/2205.14135

还没完：5. PageAttention – vLLM核心技术

- KV用最大上文长度预算分配内存，很多空间浪费，利用率低
- PageAttention按需分配小块内存

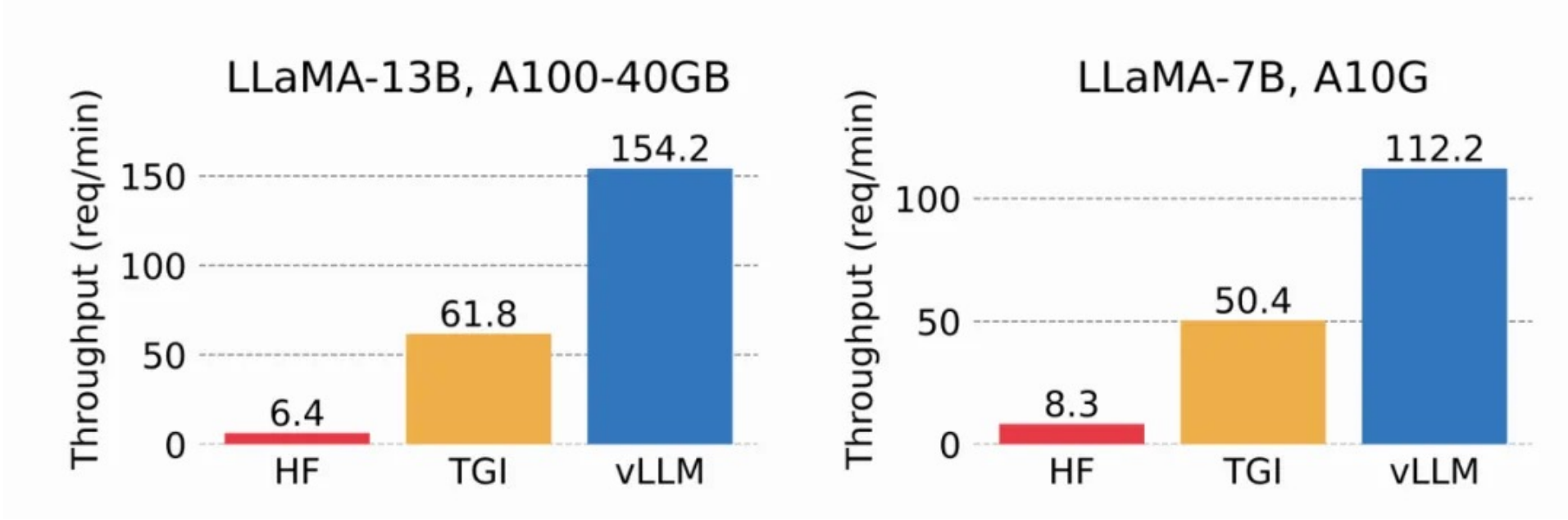


- 传统 KV Cache 像租整层办公楼，哪怕只用一个工位
- PagedAttention 则像使用共享办公空间，按需分配标准工位，灵活复用，互不干扰

https://arxiv.org/pdf/2309.06180

还没完：5. PageAttention – vLLM核心技术

- 与 HuggingFace Transformers（HF） 和 HuggingFace Text Generation Inference（TGI） 进行了对比， vLLM吞吐量最高可达 24 / 3.5 倍



https://arxiv.org/pdf/2309.06180

目录

1 KV缓存

2 模型瘦身

3

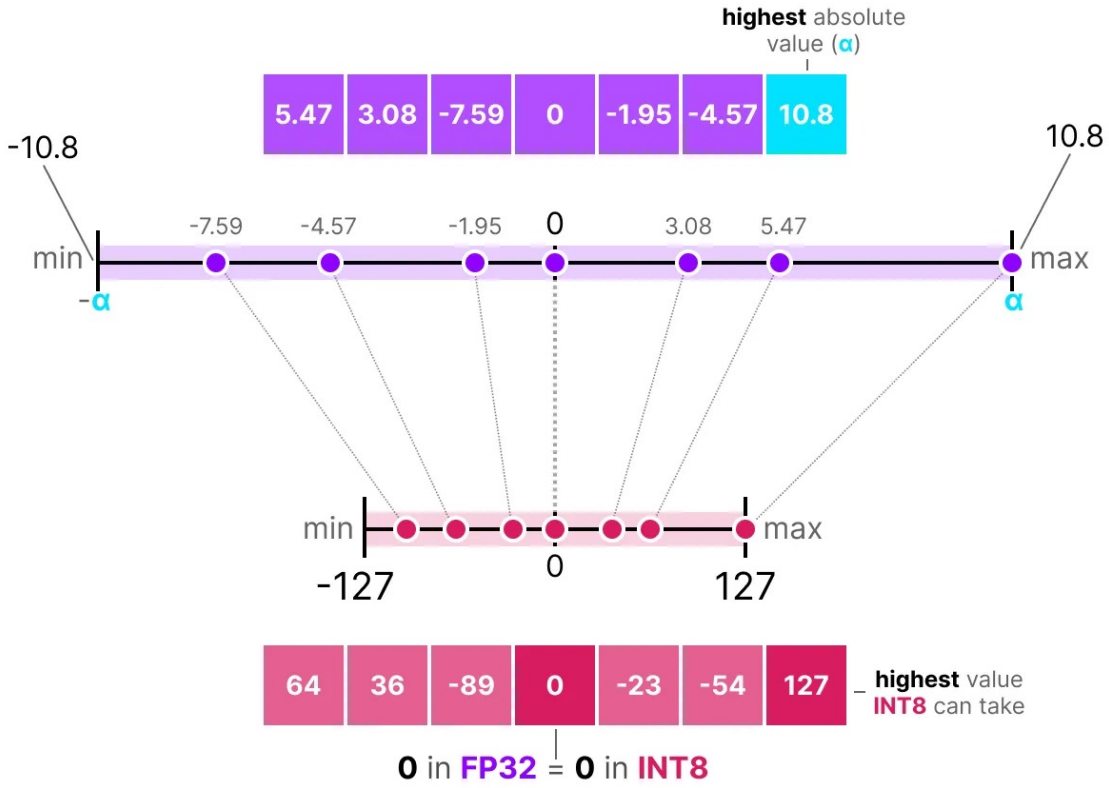
4

1. 量化（Quantization）

- 大模型参数以 32 或 16 位精度的**浮点数**存储，然而，大多数深度学习模型参数可以用每个值8位甚至更少的**整数**来有效表示

1. 量化（Quantization）

- **对称量化：**将原数据以0为中心，以一个缩放因子进行变化



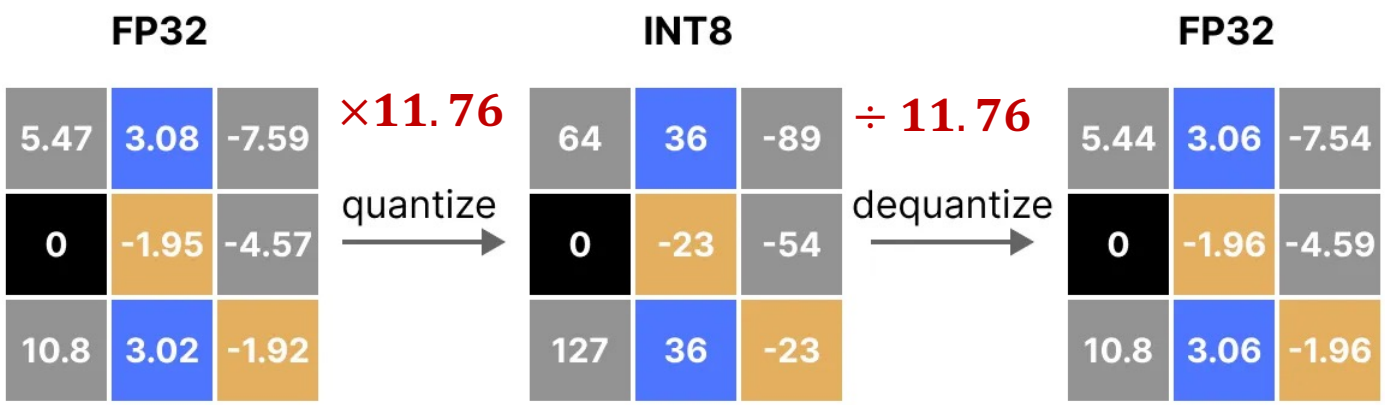
$$s = \frac{2^{b-1} - 1}{a} = \frac{127}{10.8} = 11.76$$

b：要量化为(如INT8)的字节数

a：最高的绝对值

1. 量化（Quantization）

- **对称量化：**将原数据以0为中心，以一个缩放因子进行变化



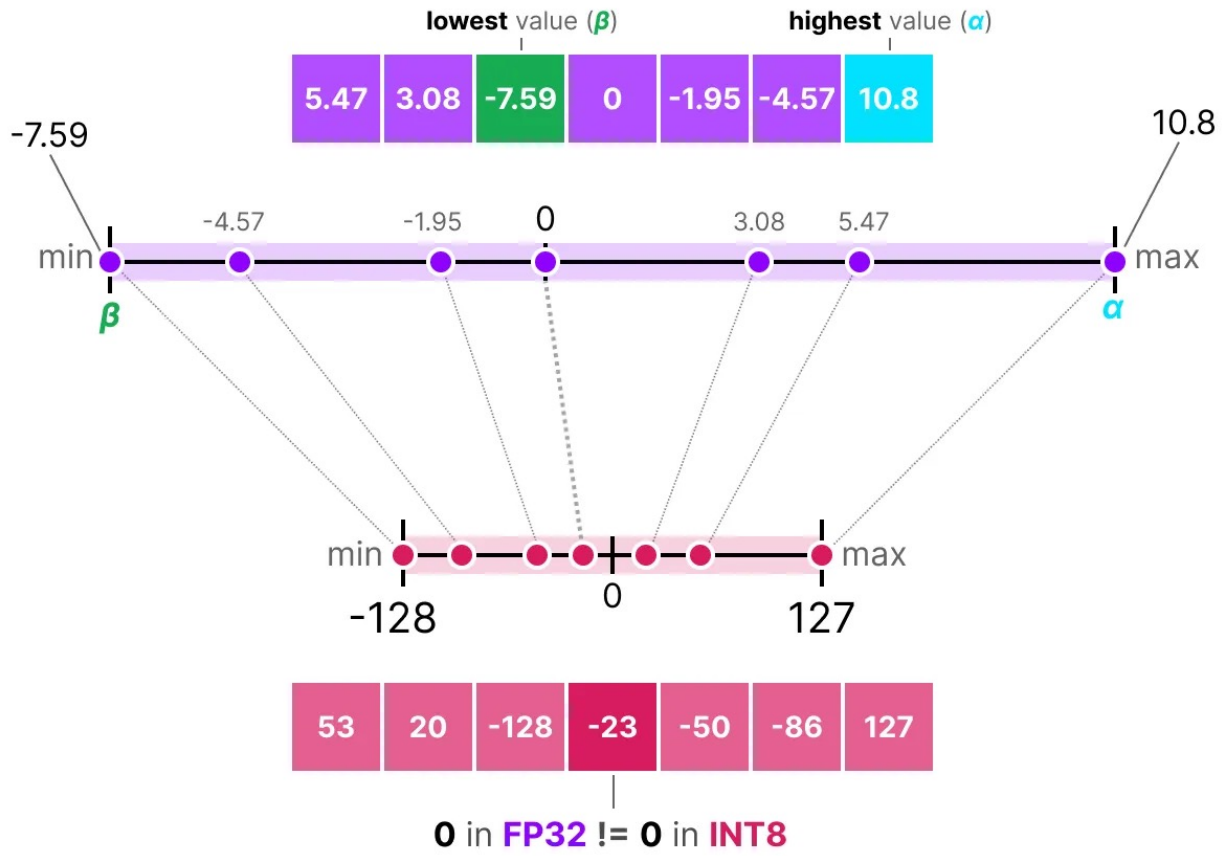
$$s = \frac{2^{b-1}-1}{a} = \frac{127}{10.8} = 11.76$$

b：要量化为(如INT8)的字节数

a：最高的绝对值

1. 量化（Quantization）

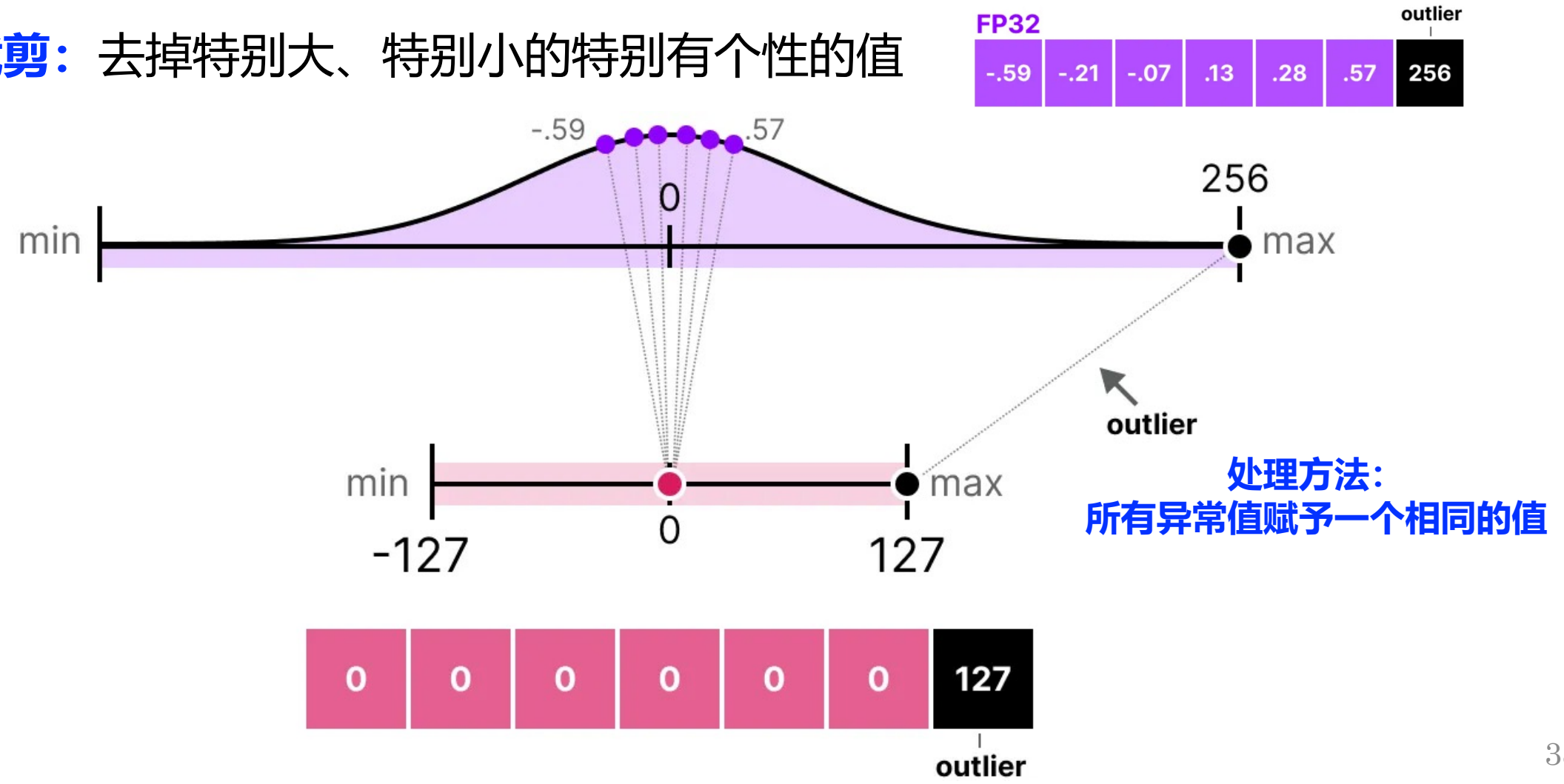
- **非对称量化：**将浮点数最小值(β)和最大值(α)缩放到量化范围的最小/大值



先计算“零点”，再计算缩放因子

1. 量化（Quantization）

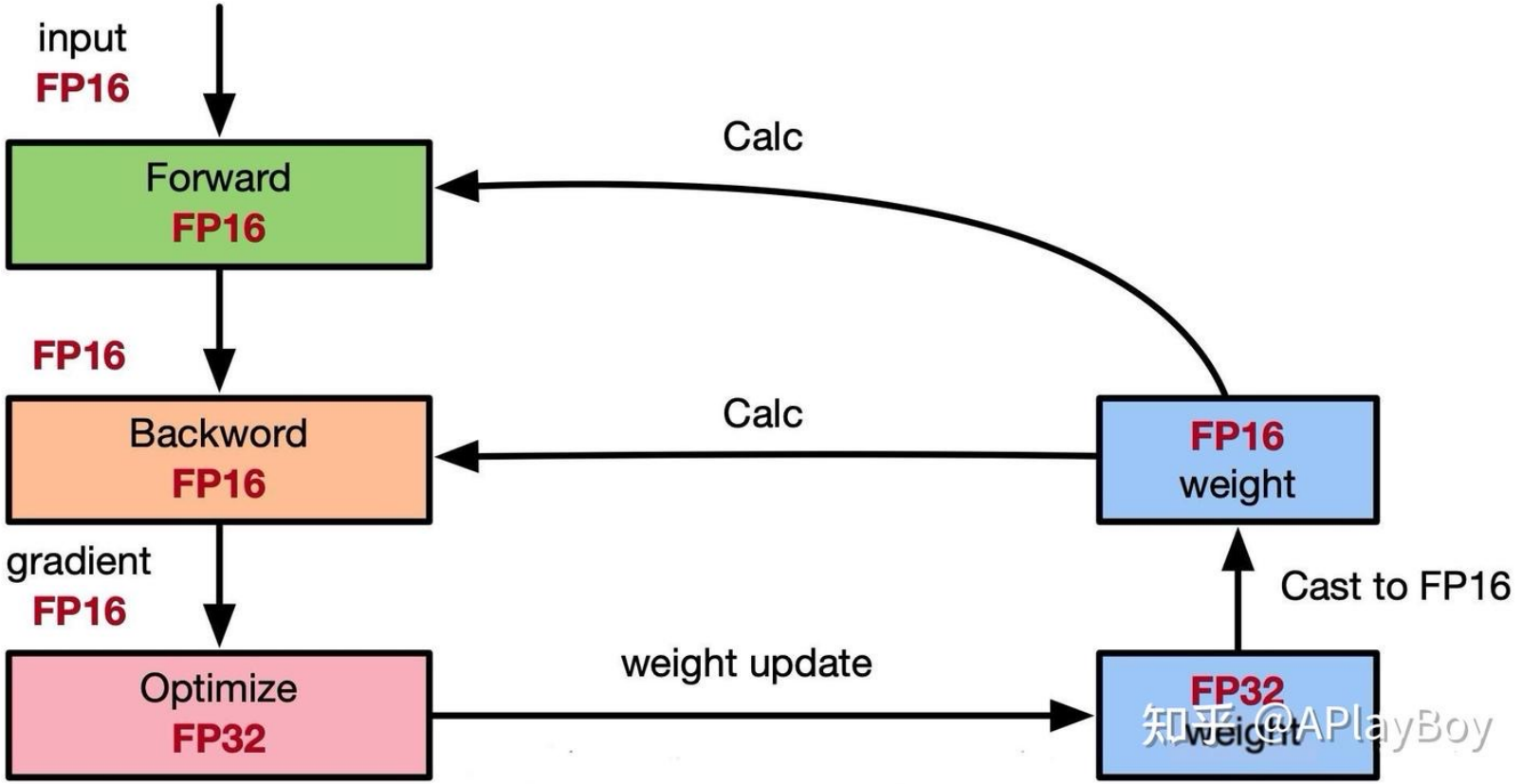
- **裁剪：**去掉特别大、特别小的特别有个性的值



处理方法：
所有异常值赋予一个相同的值

1. 量化（Quantization）

- **混合精度训练：**使用FP16在前向和后向传递，使用FP32更新梯度



2. 剪枝（Pruning）

- 识别并移除对模型性能影响极小的权重或模块：
  - 先通过权重敏感性分析，判断哪些参数是 “核心”
  - 再按比例裁剪低贡献权重（非结构化剪枝）或通道（结构化剪枝）
  - 最后微调模型，恢复因裁剪损失的少量精度

SPARSEGPT: MASSIVE LANGUAGE MODELS CAN BE ACCURATELY PRUNED IN ONE-SHOT

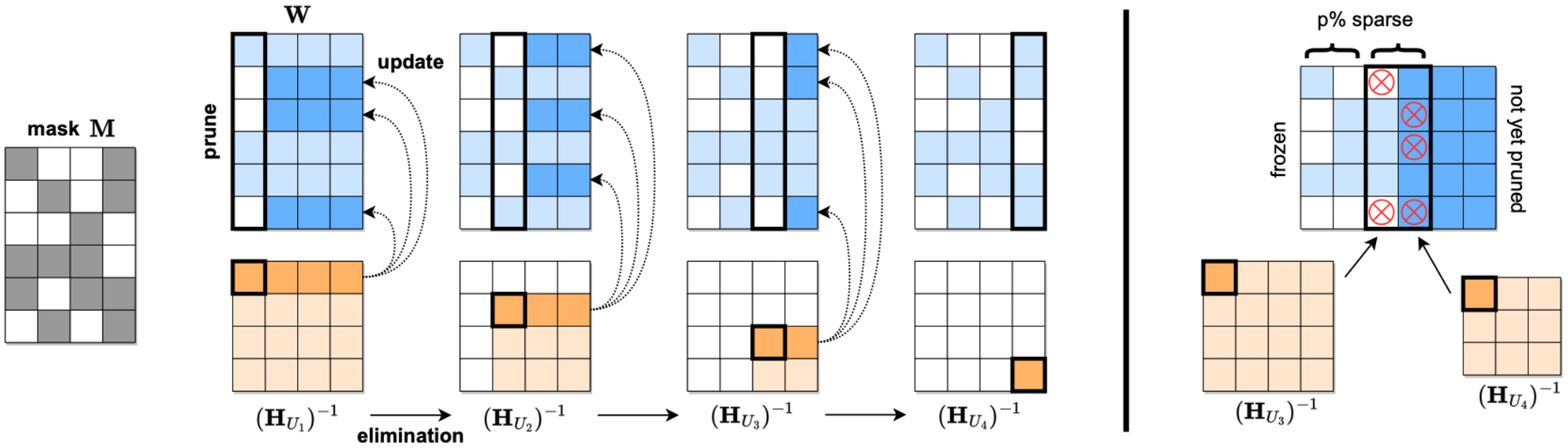
PREPRINT. VERSION JANUARY 3, 2023.

Elias Frantar
IST Austria
elias.frantar@ist.ac.at

Dan Alistarh
IST Austria & Neural Magic
dan.alistarh@ist.ac.at

2. 剪枝（Pruning）

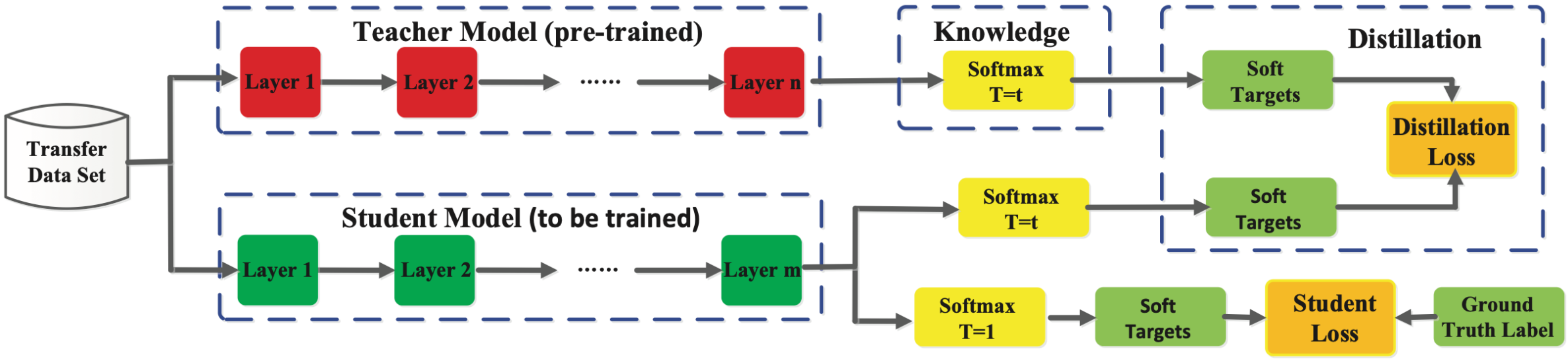
- SparseGPT：针对 GPT 规模模型的后训练方法，它不执行任何微调



给定一个固定的剪枝掩码 M，使用 Hessian 逆序列(HUj)并更新这些行中位于列“右侧”的剩余权重，逐步修剪权重矩阵 W 的每一列中的权重处理。具体来说，修剪后权重（深蓝色）“右侧”的权重将被更新以补偿修剪错误，而未修剪的权重不会生成更新（浅蓝色）

3. 蒸馏（Distillation）

- 利用教师输出的最后一层（logits）的参数值，学生使用蒸馏损失进行优化



3. 蒸馏（Distillation）

- DistilBERT：通过蒸馏将 BERT 压缩至 60% 规模，同时保持 95% 以上的语言理解性能

Table 1: **DistilBERT retains 97% of BERT performance.** Comparison on the dev sets of the GLUE benchmark. ELMo results as reported by the authors. BERT and DistilBERT results are the medians of 5 runs with different seeds.

Model	**Score**	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	STS-B	WNLI
ELMo	68.7	44.1	68.6	76.6	71.1	86.2	53.4	91.5	70.4	56.3
BERT-base	79.5	56.3	86.7	88.6	91.8	89.6	69.3	92.7	89.0	53.5
DistilBERT	77.0	51.3	82.2	87.5	89.2	88.5	59.9	91.3	86.9	56.3

中国科学院大学
University of Chinese Academy of Sciences

目录

1 KV缓存及优化

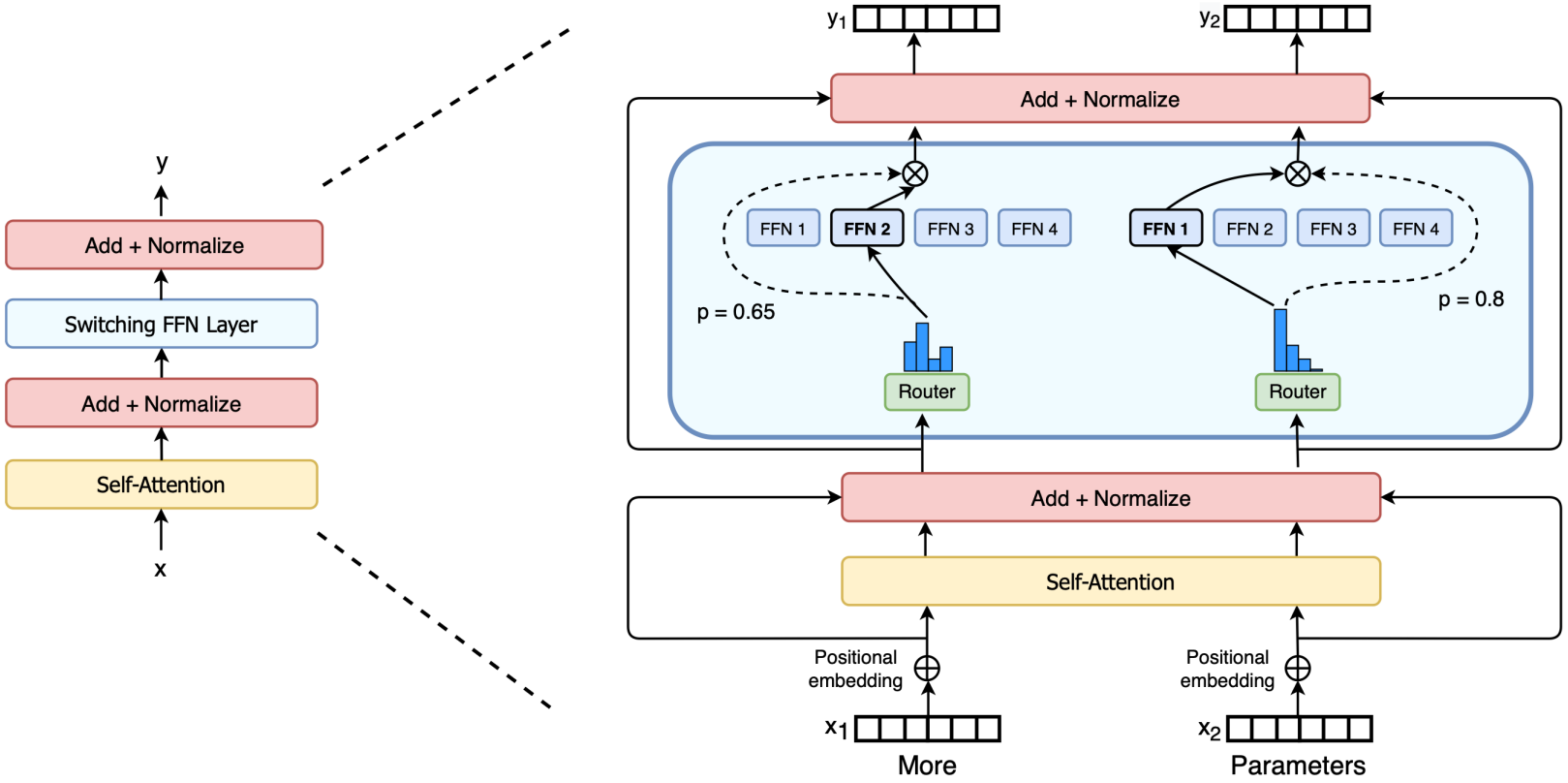
2 模型瘦身

3 MOE

4

什么是MOE（Mixed Expert Models）

- 将Transformer 中FFN层替换为MoE层，MoE 层由两个核心部分组成：路由器（或者叫门控网络）和若干数量的专家



MOE的优势

- 与Dense模型相比，在相同计算资源下训练速度更快，可以训练更大的模型
  1. 训练速度更快，效果更好
  2. 相同参数，推理成本低
  3. 扩展性好，允许模型在保持计算成本不变的情况下增加参数数量，能够扩展到非常大的模型规模（如万亿）
  4. 多任务学习能力：MoE在多任务学习中性能很好（Switch Transformer在所有101种语言上都性能提升）
- 典型模型
  - Switch Transformer大小是T5-XXL的15倍，在相同计算资源下，在达到固定困惑度 PPL 时比T5-XXL快4倍
  - Google的MoE大模型能够在相同计算资源下，以更快的速度达到相同的PPL，而且模型是T5的15倍
  - DeepSeek的16B MoE大模型，仅在40%的计算量的情况下，性能和LLaMA 2 7B效果比肩

MOE的劣势

- **训练稳定性差**：MoE在训练过程中可能会遇到稳定性问题
- **通信成本高**：在分布式训练环境中专家路由机制可能会增加通信成本
- **模型复杂性高**：MoE的设计相对复杂，需要更多的工程努力来实现和优化
- **下游任务性能**：MoE由于其稀疏性Fine-tuning过程中容易出现过拟合

1. MOE的最初

Adaptive mixtures of local experts

RA Jacobs, MI Jordan, SJ Nowlan… - Neural …, 1991 - ieeexplore.ieee.org

… the outputs of the local experts, so, to minimize the error, each local expert must make its output … effects of all the other experts. When the weights in one expert change, the residual error …

☆ 保存　99 引用　被引用次数：8240　相关文章　所有 23 个版本

Adaptive Mixtures of Local Experts

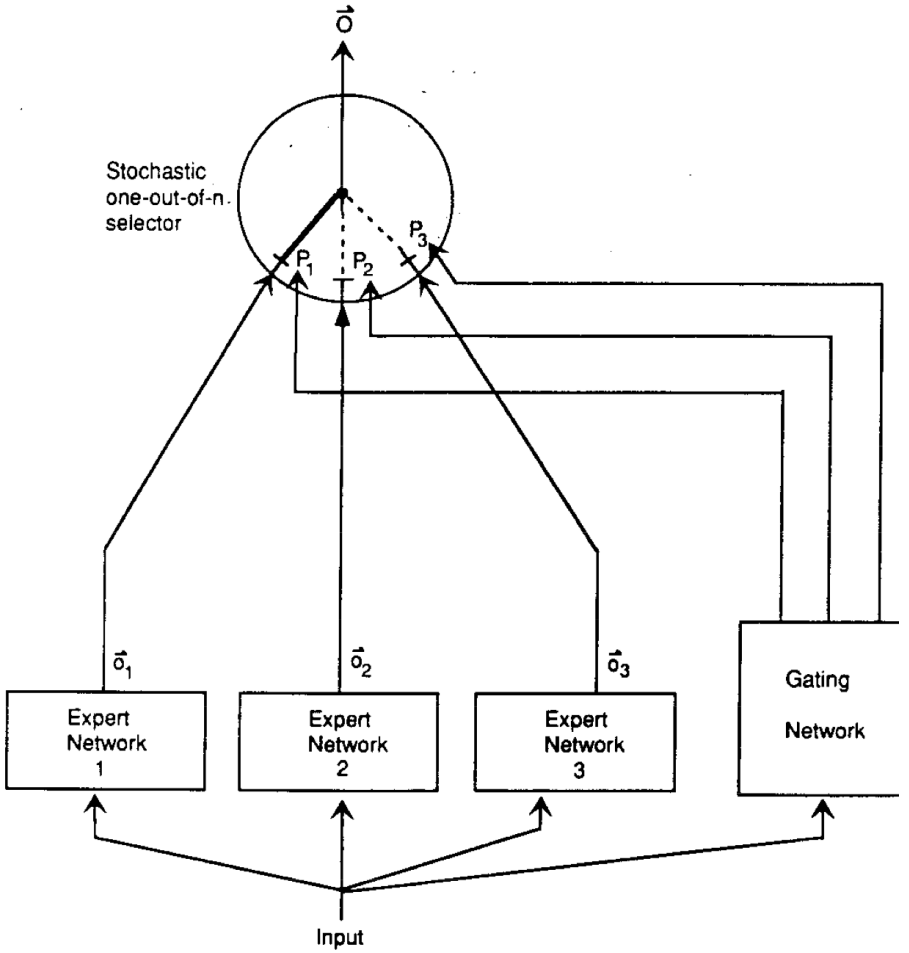
Robert A. Jacobs
Michael I. Jordan
Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA 02139 USA

Steven J. Nowlan
Geoffrey E. Hinton
Department of Computer Science, University of Toronto, Toronto, Canada M5S 1A4

We present a new supervised learning procedure for systems composed of many separate networks, each of which learns to handle a subset of the complete set of training cases. The new procedure can be viewed either as a modular version of a multilayer supervised network, or as an associative version of competitive learning. It therefore provides a new link between these two apparently different approaches. We demonstrate that the learning procedure divides up a vowel discrimination task into appropriate subtasks, each of which can be solved by a very simple expert network.

1. Adaptive mixtures of local experts

- 多层网络用来训练不同的子任务，通常会有强烈的干扰效应
  - 例如，学习一个子任务时对权重的调整可能会影响其他子任务的学习效果，因为这些权重变化会改变其他子任务的loss
- 为了解决这个问题，论文提出了使用多个模型（即专家，expert）去学习，使用一个门控网络（gating network）来决定每个数据应该被哪个模型去训练，以减轻不同类型样本之间的干扰

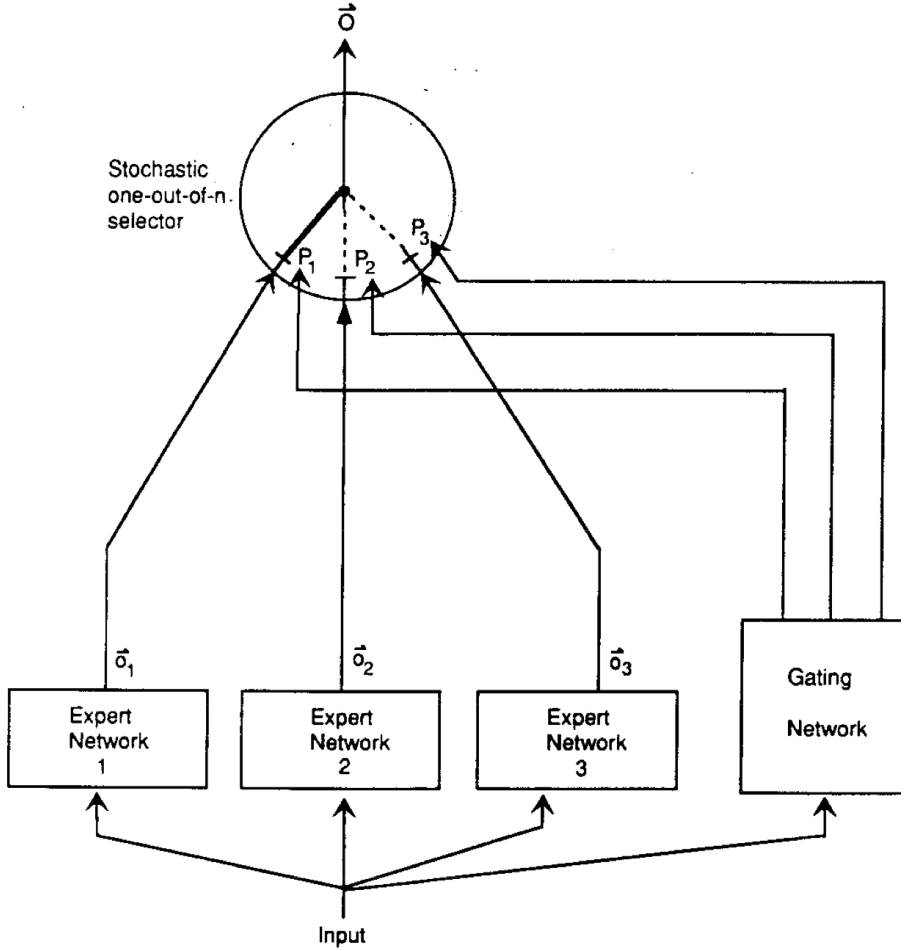


1. Adaptive mixtures of local experts

- 对于样本c，专家i输出o_i^c，期望输出d^c，专家i的权重为p_i^c，则损失函数

$$E^c = \|\mathbf{d}^c - \sum_i p_i^c \mathbf{o}_i^c\|^2$$

- 该损失函数仍可能会导致专家网络之间的强烈耦合，因为一个专家网络的权重变化会影响到其他专家网络的loss

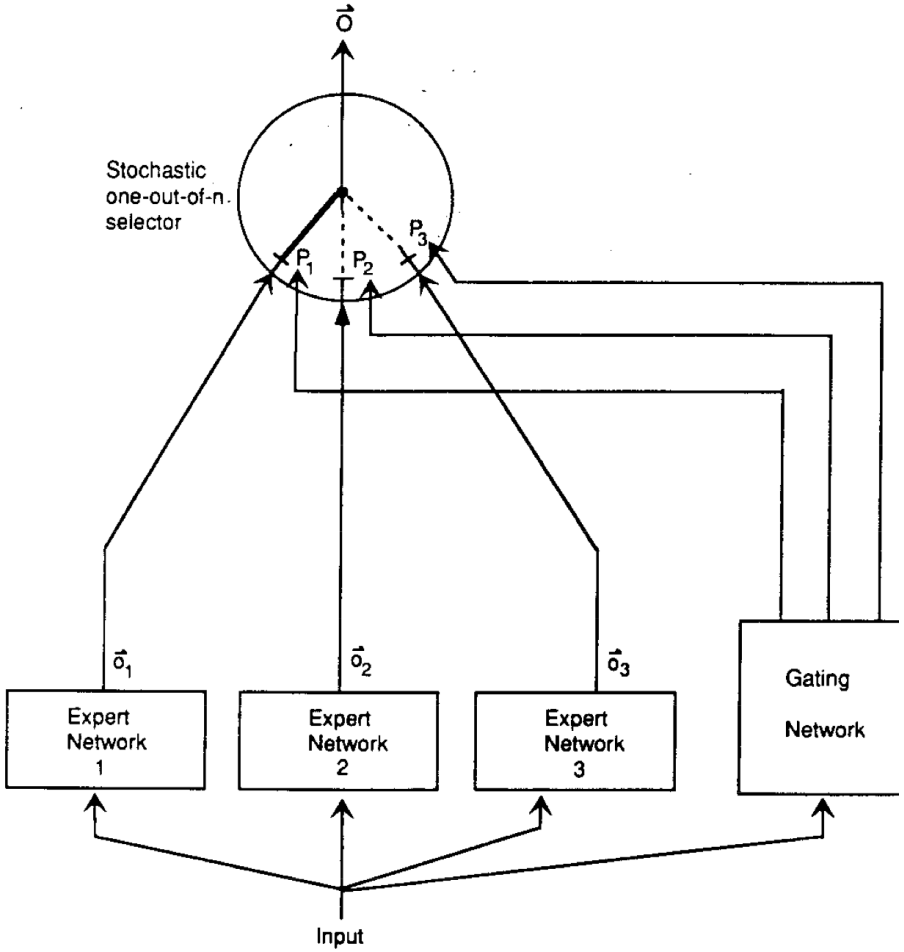


1. Adaptive mixtures of local experts

- 改进：鼓励专家网络之间的相互竞争

$$E^c = \left\langle \left\| \mathbf{d}^c - \mathbf{o}_c^i \right\|^2 \right\rangle = \sum_i p_i^c \left\| \mathbf{d}^c - \mathbf{o}_i^c \right\|^2$$

  - 当一个expert在给定样本上的的loss小于所有expert的平均loss时，它对该样本的门控score会增加；当它的表现不如平均loss时，它的门控score会减少
  - 将 Experts 和 Gating Network 一起进行训练，最终的系统就会倾向于让一个expert去处理一个样本

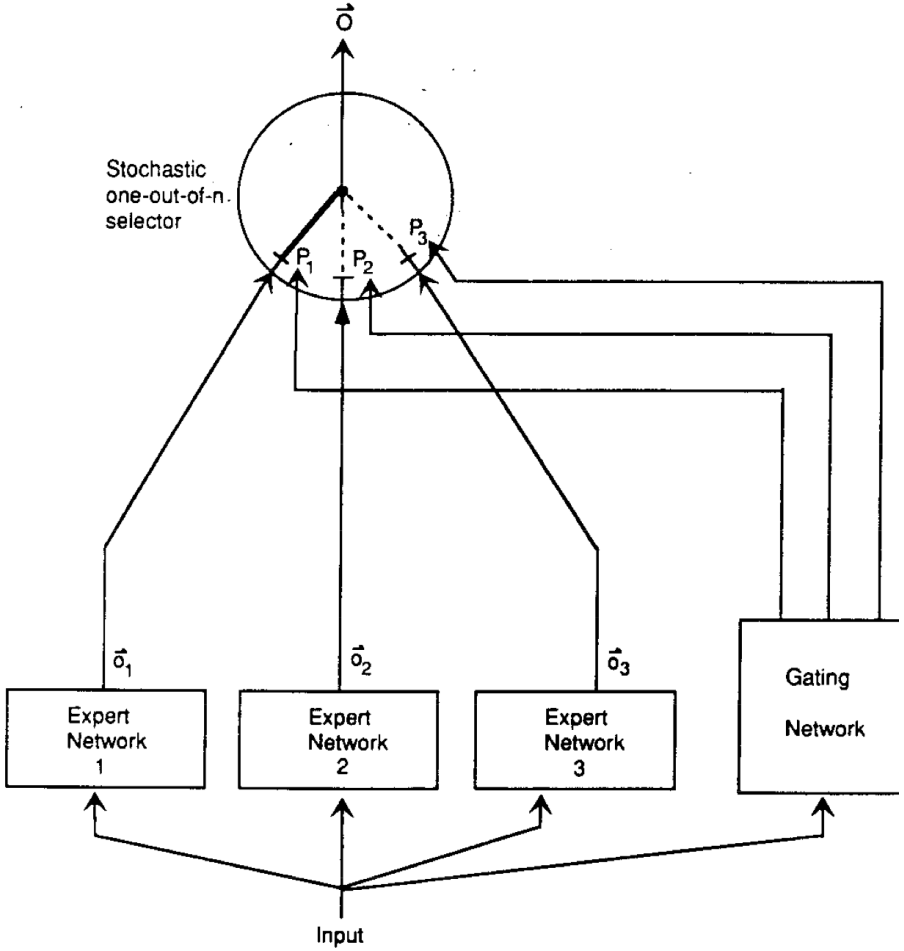


1. Adaptive mixtures of local experts

- 改进：鼓励专家网络之间的相互竞争

$$E^c = \left\langle \left\| \mathbf{d}^c - \mathbf{o}_c^i \right\|^2 \right\rangle = \sum_i p_i^c \left\| \mathbf{d}^c - \mathbf{o}_i^c \right\|^2$$

  - 当一个expert在给定样本上的的loss小于所有expert的平均loss时，它对该样本的门控score会增加；当它的表现不如平均loss时，它的门控score会减少
  - 将 Experts 和 Gating Network 一起进行训练，最终的系统就会倾向于让一个expert去处理一个样本



2. 神经网络中的MOE

Under review as a conference paper at ICLR 2017

OUTRAGEOUSLY LARGE NEURAL NETWORKS: THE SPARSELY-GATED MIXTURE-OF-EXPERTS LAYER

Noam Shazeer[1], Azalia Mirhoseini*†[1], Krzysztof Maziarz*[2], Andy Davis[1], Quoc Le[1], Geoffrey Hinton[1] and Jeff Dean[1]

[1]Google Brain, {noam,azalia,andydavis,qvl,geoffhinton,jeff}@google.com
[2]Jagiellonian University, Cracow, krzysztof.maziarz@student.uj.edu.pl

在牺牲极少的计算效率的情况下，把模型规模提升**1000多倍**

2. 神经网络中的MOE

- **Sparsely-Gated：**不是所有expert都会起作用，而是极少数的expert会被使用来进行推理。这种稀疏性，也使得可以使用海量的experts来把模型容量做的超级大
- **token-level：**不是 sample-level 的（即不同的样本使用不同的专家），是 token-level 的（一个句子中不同的 token 使用不同的专家）

2. 神经网络中的MOE

- RNN中的实现：每个 token都会有一个 MoE Layer，每个 MoE layer 中包含了多个experts，每个 expert都是一个小型的 FFN，还有一个 Gating Network 会根据当前 token，选择少数几个 expert 来进行计算

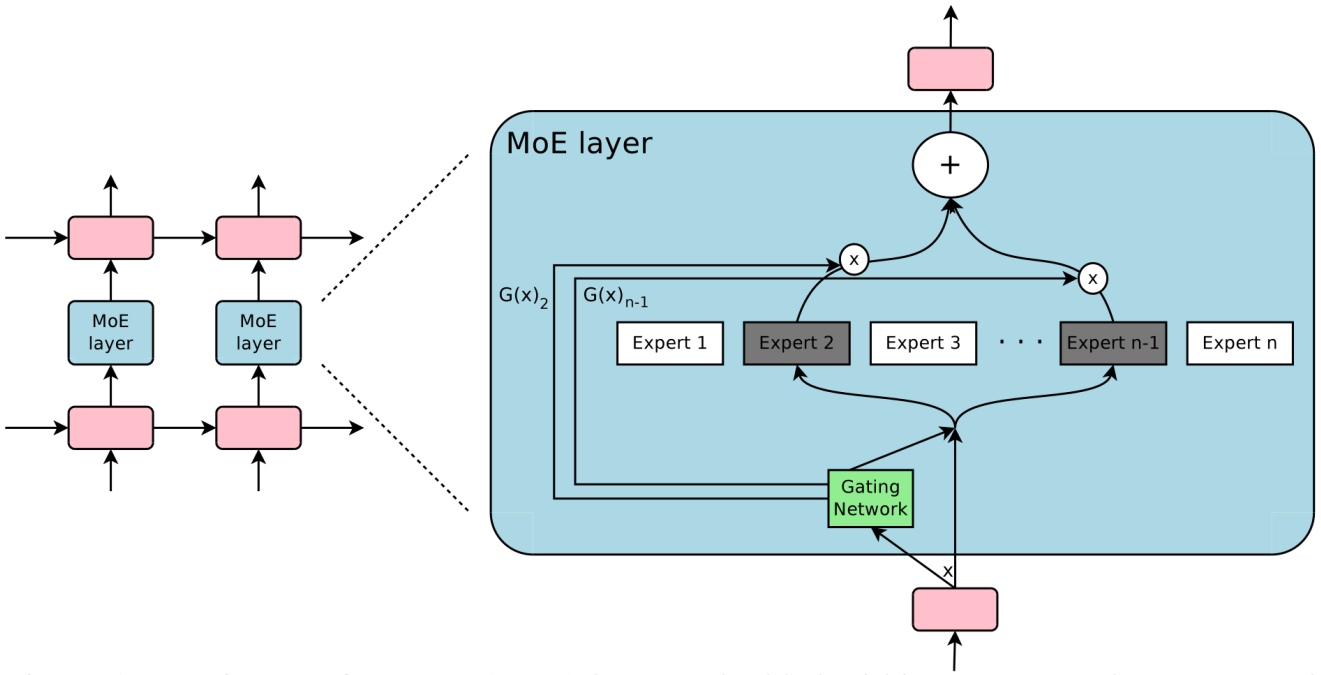


Figure 1: A Mixture of Experts (MoE) layer embedded within a recurrent language model. In this case, the sparse gating function selects two experts to perform computations. Their outputs are modulated by the outputs of the gating network.

门控网络（Gating Network）

- 门控网络负责为每个输入 token 选择一个稀疏的专家组合

门控网络输出　　专家输出

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

- 门控网络通常是一个带有 softmax 函数的简单的网络

$$G_\sigma(\boldsymbol{x}) = \text{Softmax}(\boldsymbol{x} \cdot \boldsymbol{W}_g)$$

平衡专家利用率

- 门控网络倾向于收敛到一种状态，可能导致训练效率低下，某些专家可能从未被使用过
- 软约束：

专家门控值总和

$$\text{Importance}(\boldsymbol{X}) = \sum_{\boldsymbol{x} \in \boldsymbol{X}} G(\boldsymbol{x})$$

$$L_{\text{importance}}(\boldsymbol{X}) = w_{\text{importance}} \cdot CV(\text{Importance}(\boldsymbol{X}))^2$$

缩放因子

变异系数：比较不同专家之间的波动性

2. 神经网络中的MOE

Table 5: Multilingual Machine Translation (bold values represent best results).

	GNMT-Mono	GNMT-Multi	MoE-Multi	MoE-Multi vs. GNMT-Multi
Parameters	278M / model	278M	8.7B	
ops/timestep	212M	212M	102M	
training time, hardware	various	21 days, 96 k20s	**12 days, 64 k40s**	
Perplexity (dev)		4.14	**3.35**	-19%
French → English Test BLEU	36.47	34.40	**37.46**	+3.06
German → English Test BLEU	31.77	31.17	**34.80**	+3.63
Japanese → English Test BLEU	23.41	21.62	**25.91**	+4.29
Korean → English Test BLEU	25.42	22.87	**28.71**	+5.84
Portuguese → English Test BLEU	44.40	42.53	**46.13**	+3.60
Spanish → English Test BLEU	38.00	36.04	**39.39**	+3.35
English → French Test BLEU	35.37	34.00	**36.59**	+2.59
English → German Test BLEU	**26.43**	23.15	24.53	+1.38
English → Japanese Test BLEU	**23.66**	21.10	22.78	+1.68
English → Korean Test BLEU	**19.75**	18.41	16.62	-1.79
English → Portuguese Test BLEU	**38.40**	37.35	37.90	+0.55
English → Spanish Test BLEU	34.50	34.25	**36.21**	+1.96

3. 大模型中的MOE

GShard：第一个将 MoE 思想拓展到 Transformer 上的工作

Published as a conference paper at ICLR 2021

GSHARD: SCALING GIANT MODELS WITH CONDITIONAL COMPUTATION AND AUTOMATIC SHARDING

Dmitry Lepikhin
lepikhin@google.com

HyoukJoong Lee
hyouklee@google.com

Yuanzhong Xu
yuanzx@google.com

Dehao Chen
dehao@google.com

Orhan Firat
orhanf@google.com

Yanping Huang
huangyp@google.com

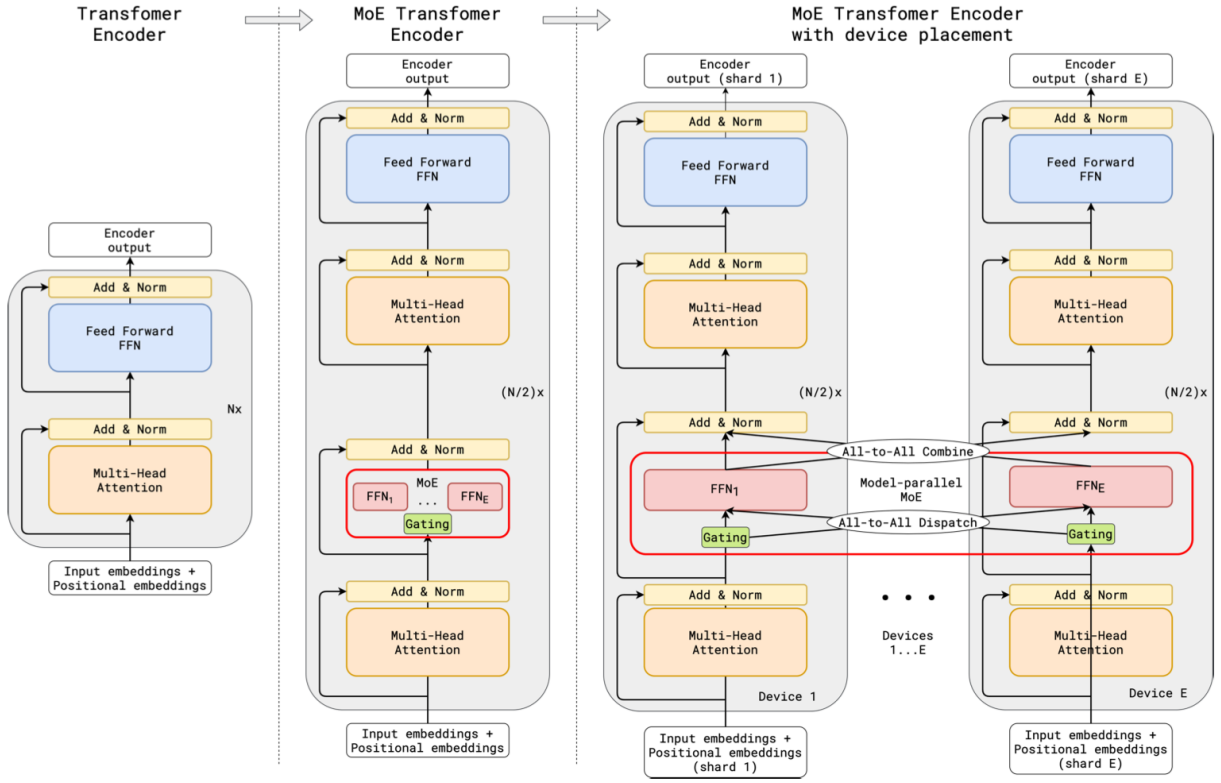
Maxim Krikun
krikun@google.com

Noam Shazeer
noam@google.com

Zhifeng Chen
zhifengc@google.com

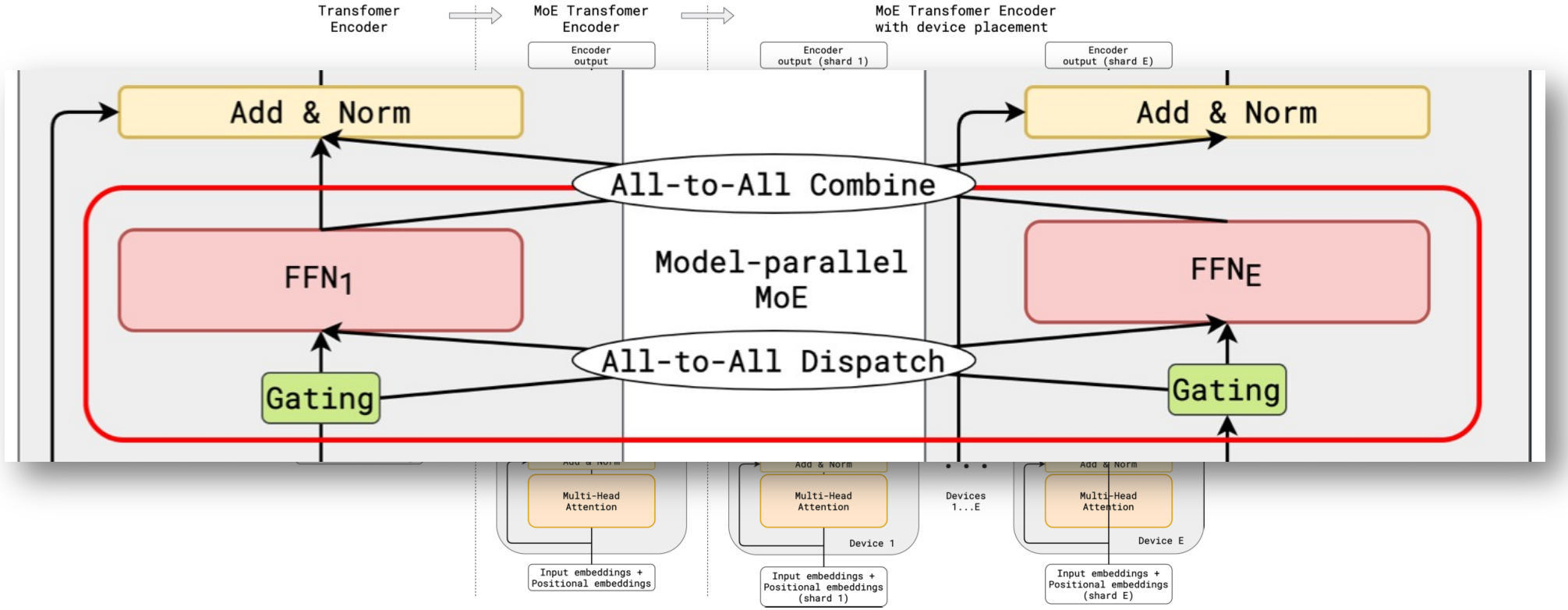
GShard

- 把Transformer的encoder和decoder中，每隔一个FFN层，替换成Top-2门控网络的MoE层



GShard

- 把Transformer的encoder和decoder中，每隔一个FFN层，替换成Top-2门控网络的MoE层



GShard

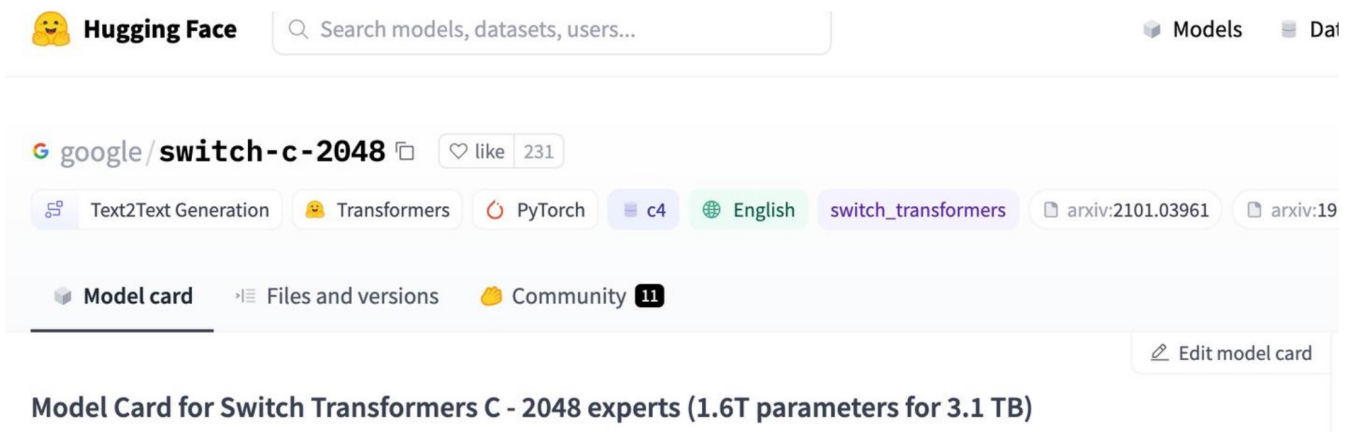
- MoE跨设备分片：GShard MoE 层中的专家网络（experts）被分布在不同的设备上。每个专家网络负责处理一部分输入数据，并且每个 token 根据门控机制的输出被分配到一个或两个专家网络中。这样，整个 MoE 层的计算被分散到了多个设备上，每个设备负责处理一部分计算任务
- GShard具有复杂性、通信成本以及训练和微调过程的不稳定性的不足之处

Switch Transformers

Journal of Machine Learning Research 23 (2022) 1-39 Submitted 8/21; Revised 3/22; Published 4/22

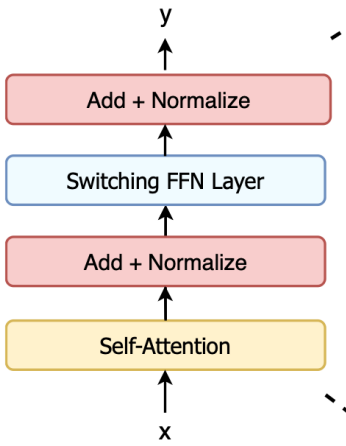
Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity

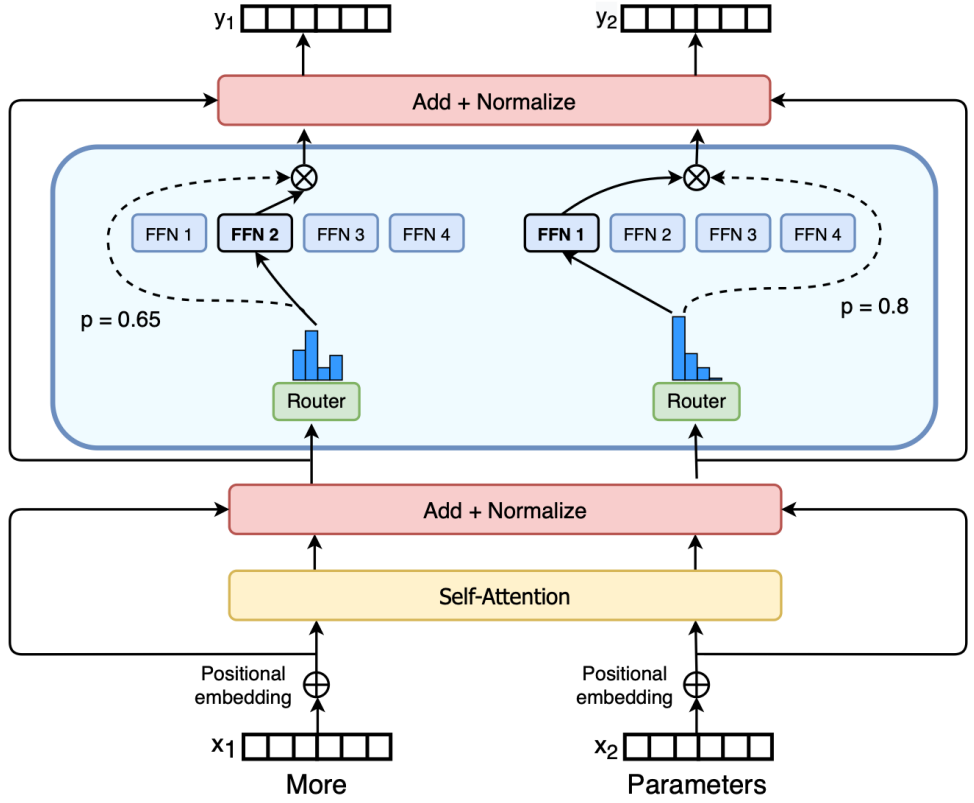
William Fedus* LIAMFEDUS@GOOGLE.COM
Barret Zoph* BARRETZOPH@GOOGLE.COM
Noam Shazeer NOAM@GOOGLE.COM
Google, Mountain View, CA 94043, USA



Switch Transformers

- 每次只选一个专家：
  - 减少了路由计算
  - 每个专家的 batch size至少可以减半
  - 简化路由的实现，降低了 MoE 通信成本





Switch Transformers

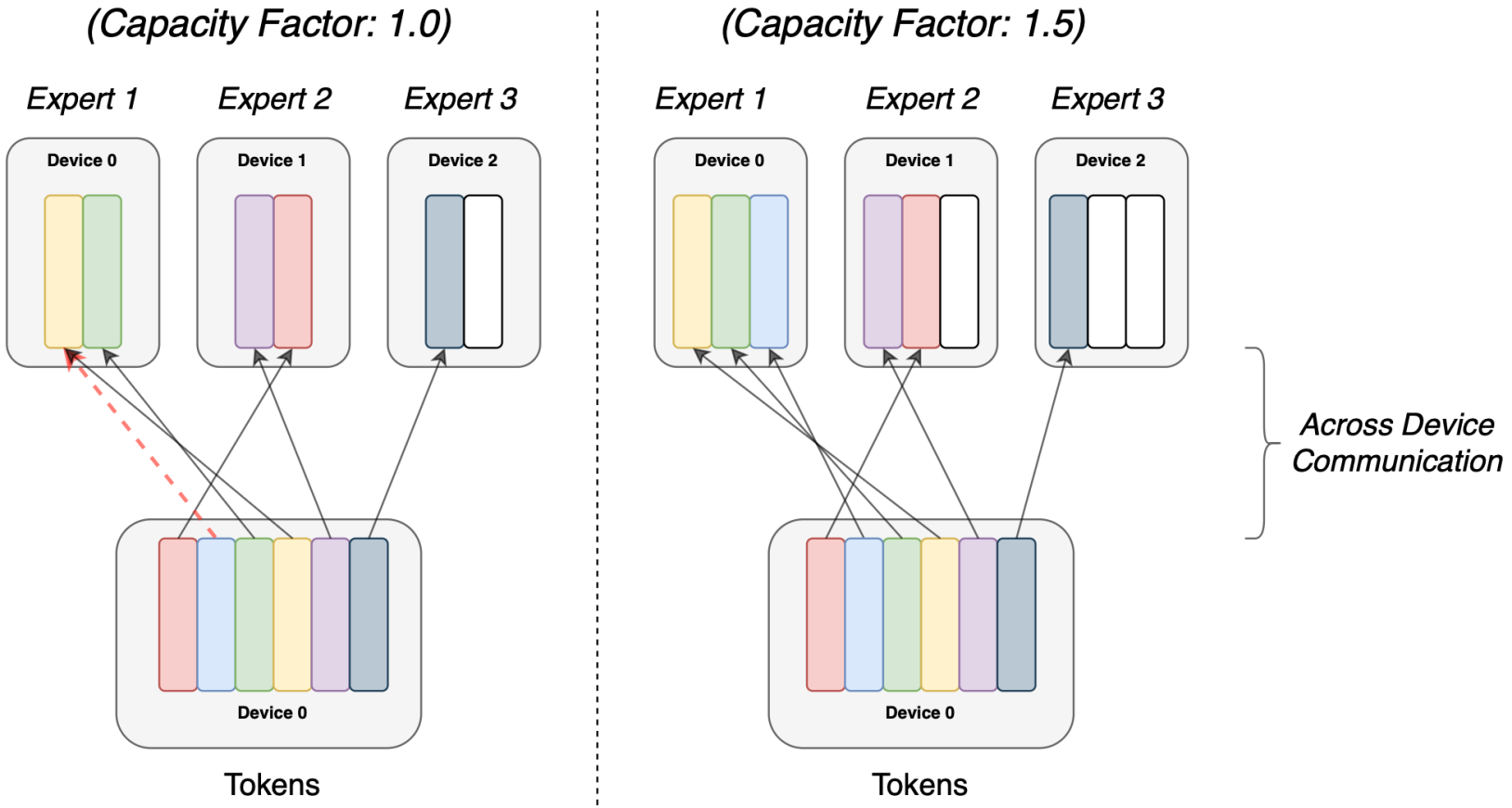
- 专家容量（Expert Capacity）：指每个专家处理的 token 数

$$\text{Expert Capacity} = \left(\frac{\text{tokens per batch}}{\text{number of experts}} \right) \times \text{capacity factor}$$

- 作用：将 batch 中的总 token 数平均分配给所有专家

Switch Transformers

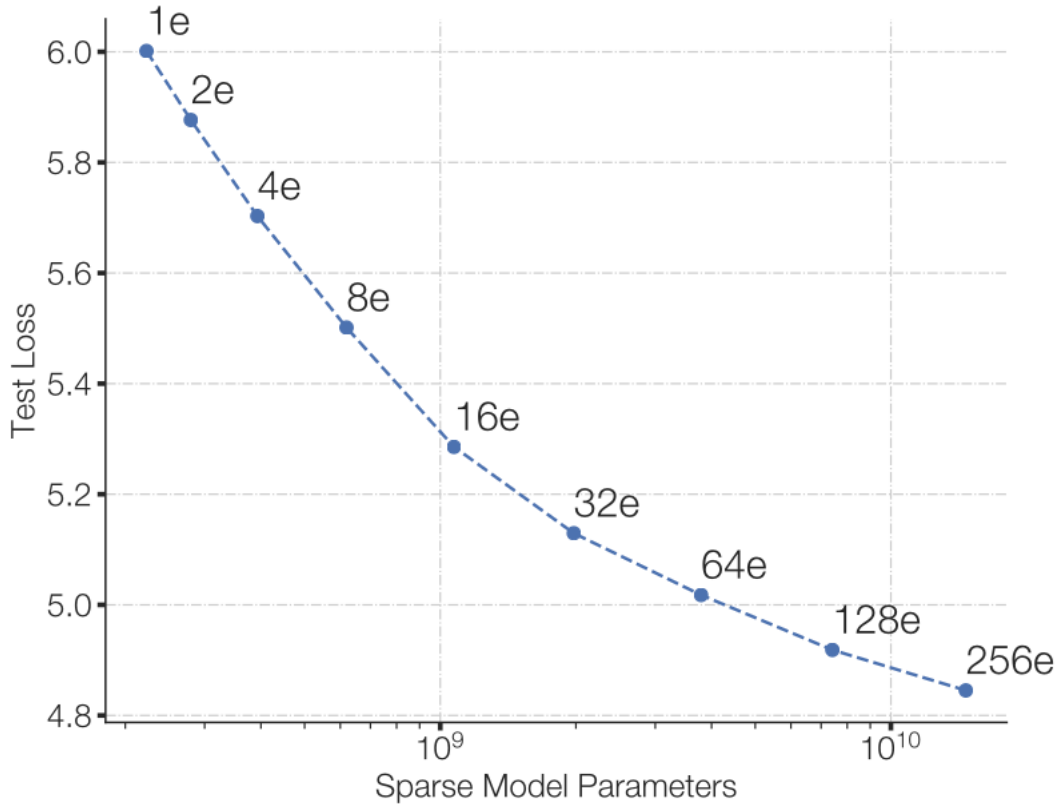
- 专家容量（Expert Capacity）：指每个专家处理的 token 数



Switch Transformers

- 和 T5 Base、T5 Large 相比，Switch Transformers 在相同计算资源情况下获得了高达 7 倍的预训练速度

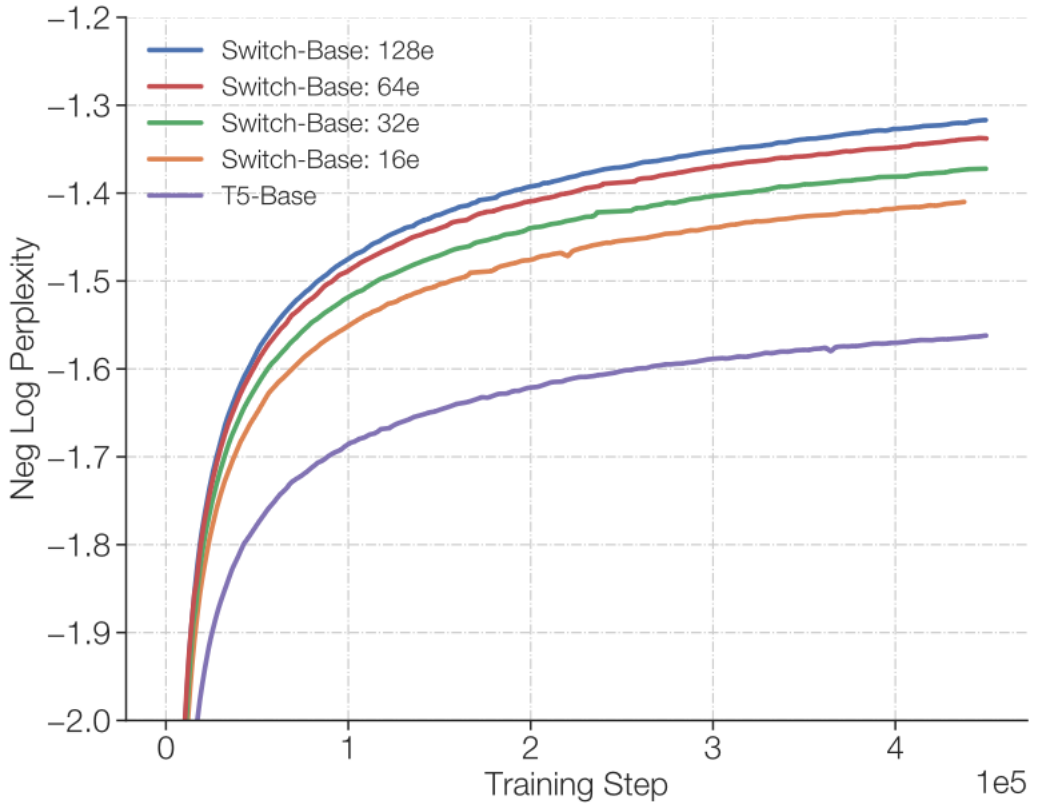
增加模型稀疏性（更多专家），loss逐渐降低



Switch Transformers

- 和 T5 Base、T5 Large 相比，Switch Transformers 在相同计算资源情况下获得了高达 7 倍的预训练速度

Switch Transformer 模型在保持相同计算资源的情况下，相对于 T5-Base 有显著的提升，而且专家数越多（模型参数越多、模型更稀疏），效果越好



中国科学院大学
University of Chinese Academy of Sciences

目 录

1 KV缓存及优化

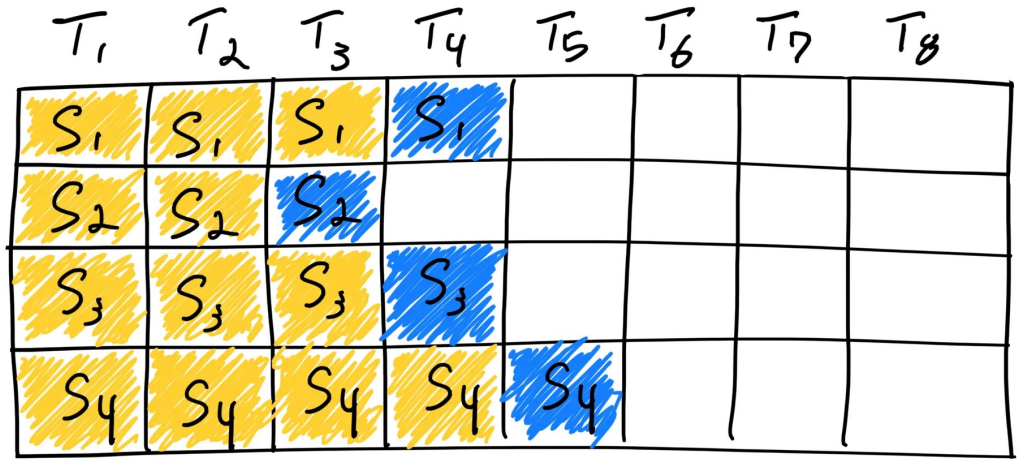
2 模型瘦身

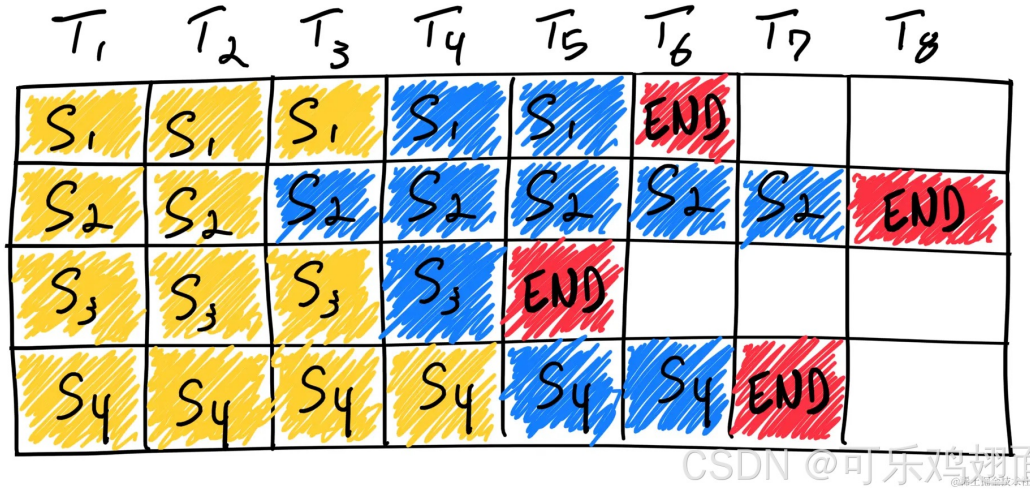
3 MOE

4 调度优化

静态批处理

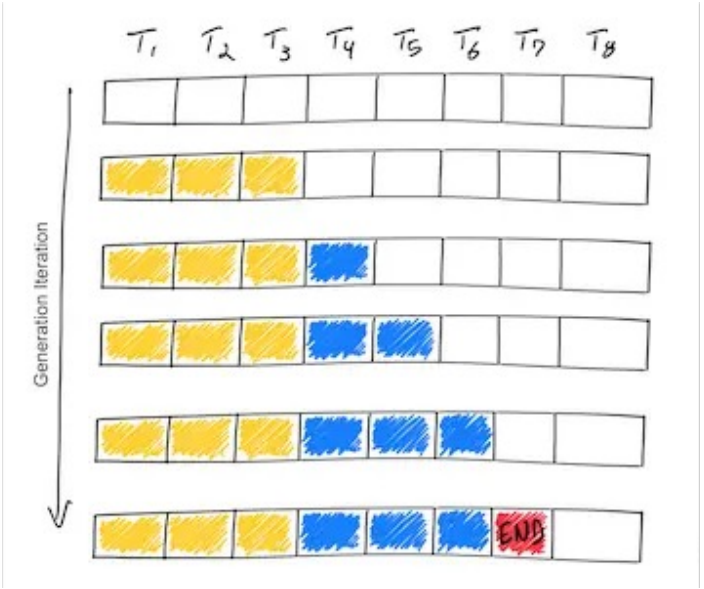
- 静态批处理（static batching）：将多个Prompt打包进行一个批处理请求，并在批处理请求中所有Prompt完成后返回响应，批处理的大小在推理完成之前保持不变。





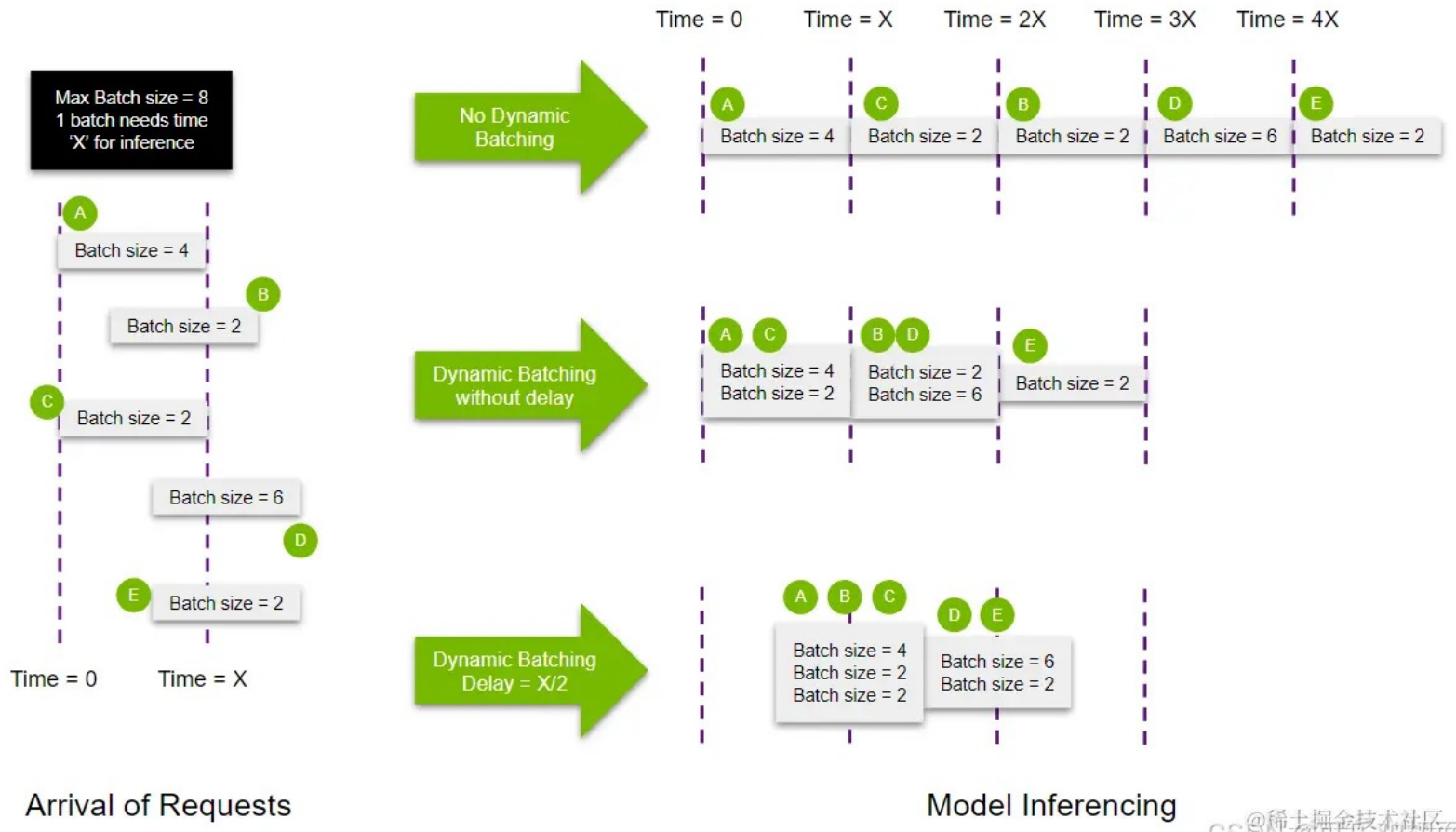
静态批处理

- 缺点：一个批次中不同序列的生成长度不同。有的Prompt在批处理中较早“完成”，但需等待这一批次中Prompt最长的生成结束，因此，GPU 未得到充分利用。



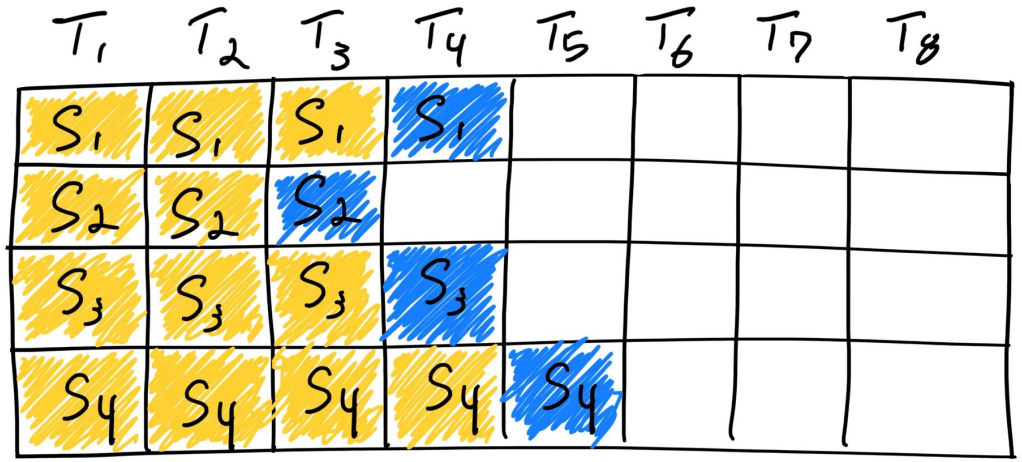
动态批处理

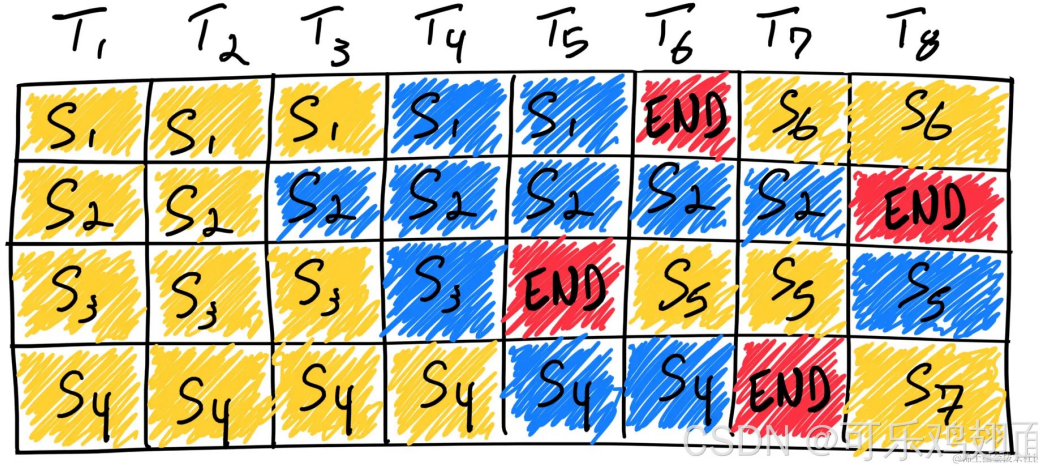
- 允许将一个或多个推理请求组合成单个批次（必须动态创建）以最大化吞吐量的功能



连续批处理

- 请求在到达时一起批量处理，但它不是等待批次中所有序列都完成，而是当一个输入提示生成结束之后，就会在其位置将新的输入Prompt插入进来

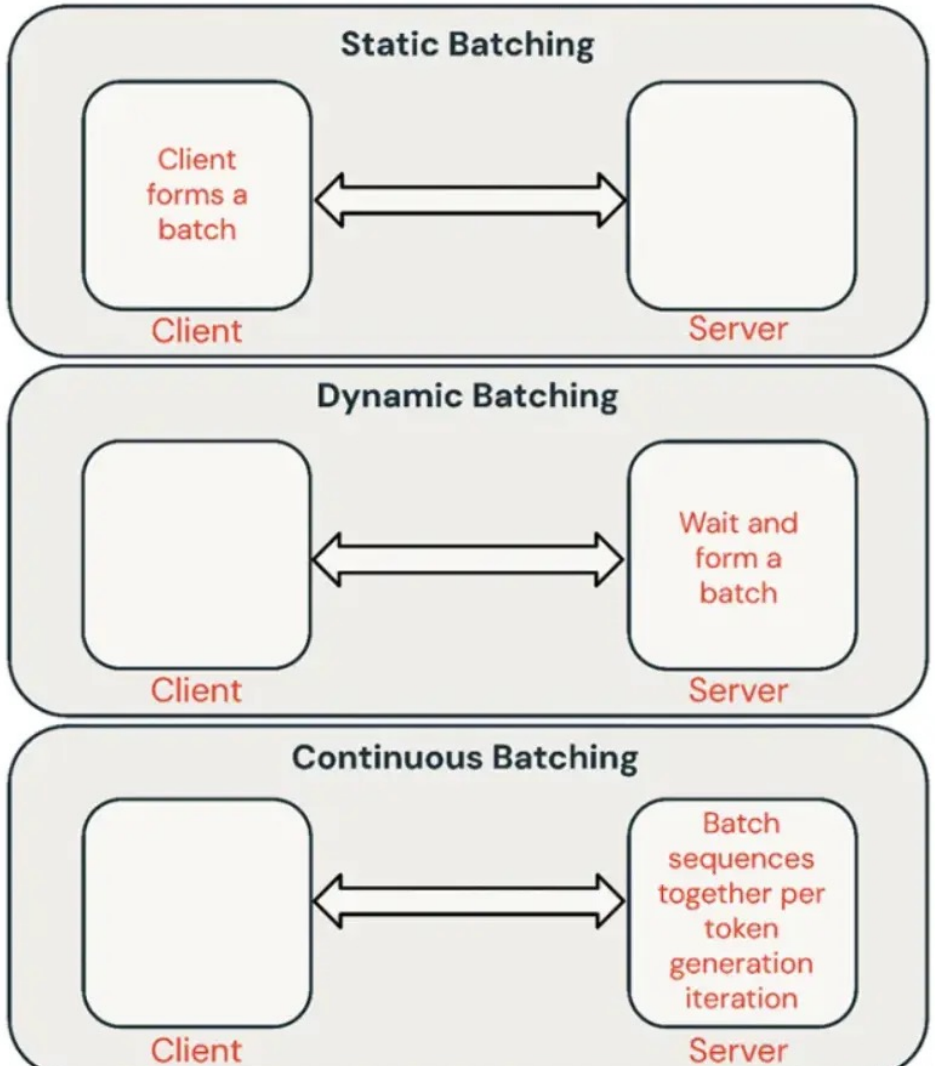




小结

- 静态批处理：客户端将多个Prompt打包进一个请求中，并在批次中所有序列完成后返回响应。通常，多数推理服务支持这种方法，但并不要求这样做。
- 动态批处理：多个请求的Prompt在服务端内部动态打包进一个批次处理。通常，这种方法的表现不如静态批处理，但如果响应短或长度一致，可以接近最优。当请求具有不同参数时，这种方法效果不佳。
- 连续批处理：将请求在到达时一起批量处理，它不是等待批次中所有序列完成，而是在迭代推理层级将序列组合在一起。它可以实现比静态批处理高10倍到20倍的吞吐量，目前是最先进的方法

小结


Static Batching
Client forms a batch
Client
Server
Dynamic Batching
Wait and form a batch
Client
Server
Continuous Batching
Batch sequences together per token generation iteration
Client
Server

本节复习

- KV Catch
- MQA，GQA，MLA
- Quantization，Pruning，Distillation
- MOE
- Static/Dynamic/Continuous Batching

参考文献

- Shazeer, Noam. "Fast transformer decoding: One write-head is all you need." arXiv preprint arXiv:1911.02150 (2019).
- Liu, Aixin, et al. "Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model." arXiv preprint arXiv:2405.04434 (2024).
- Fedus, William, Barret Zoph, and Noam Shazeer. "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity." Journal of Machine Learning Research 23.120 (2022): 1-39.


THANKS
https://ictkc.github.io/teaching/2026spring-nlp