清華大學电子工程系
Department of Electronic Engineering, Tsinghua University

Generative Model Compression and Acceleration

Xuefei Ning

Affiliated with: Department of Electronic Engineering, Tsinghua University

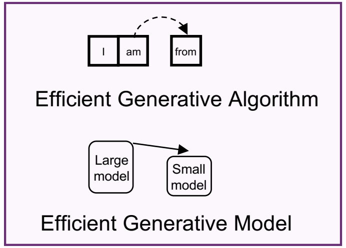
foxdoraame@gmail.com

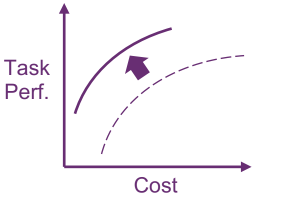
Lab Leader: Prof. Yu Wang yu-wang@tsinghua.edu.cn

NICS EFC

Team Introduction

Research Goal
Develop **efficient algorithms and models**





Improve the perf.-cost trade-off

Team Website	Bilibili	GitHub Org.
https://nics-effalg.com/	https://space.bilibili.com/642618077	https://github.com/thu-nics

NICS-EFC Lab, Tsinghua University

Professor Yu Wang is the leader of the *Nanoscale Integrated Circuits and System - Energy Efficient Computing Lab* (***NICS-EFC***) in the Dept. EE at Tsinghua.

Research Assistant Professor Xuefei Ning is the leader of the *Efficient Algorithm Team* (***EffAlg***) in the ***NICS-EFC*** lab.

目录 Contents

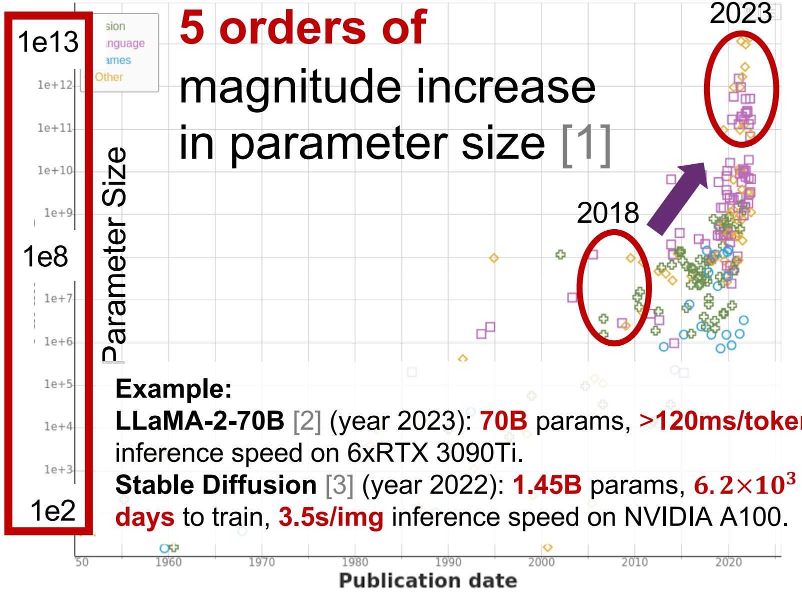
目 录 Contents

Trend of Generative Models

The model size & input/output length of generative models have being rapidly increasing

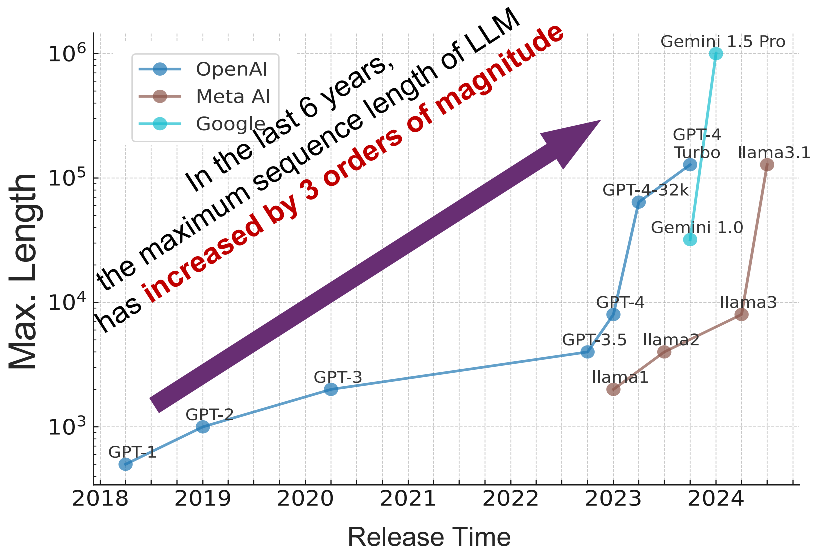
2018 - 2023

5 orders of magnitude increase in parameter size [1]



Example:
LLaMA-2-70B [2] (year 2023): **70B** params, **>120ms/token** inference speed on 6xRTX 3090Ti.
Stable Diffusion [3] (year 2022): **1.45B** params, **6.2×10^3 GPU days** to train, **3.5s/img** inference speed on NVIDIA A100.

Longer Sequence Length for Language



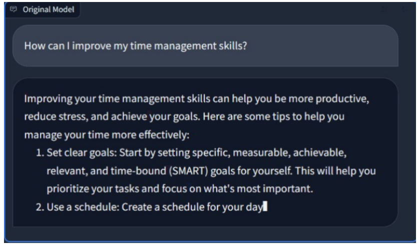
[1] Villalobos et al. “Machine Learning Model Sizes and the Parameter Gap.” arXiv 2022.
[2] Touvron, Hugo, et al. "Llama 2: Open foundation and fine-tuned chat models." *arXiv 2023*.
[3] Rombatch et al., High-Resolution Image Synthesis with Latent Diffusion Models, CVPR 2022.
[4] Yang et al., "Harnessing the Power of LLMs in Practice: A Survey on ChatGPT and Beyond“, *ACM Transactions on Knowledge Discovery from Data* 2023.

Research Overview

Research Goal：Efficient model inference for AIGC application

Application

Language Generation



Large Language Models
(e.g., LLaMA-2-7B)

Visual Generation

Diffusion Models
(e.g., Stable Diffusion 3)

Methodology: System-aware algorithm-level and model-level optimization

Technique

Algorithm-level

Diffusion Timestep Compression — Tackling Many Timesteps of Diffusion

Non-Autoregressive Generation — Tackling Full Autoregressive Generation of LLMs

Model-level

Structure Design

Model Compression

Research Summary

Efficient LLM/VLM

Overview

Survey [Under Review]

Survey on efficient LLM inference techniques

Algorithm-level

SoT [ICLR'24]

Parallel generation via prompting.
1.91~2.39x speed-up

Model-level

Sparse Attention

MoA [Under Review]

Decide the heterogeneous elastic rule of the attention span for each head.
5.5~6.7x throughput improvement

Pruning

EEP [Under Review]

Search the pruning pattern for MoE and use expert merging for finetuning.
48%~71% memory reduction,
1.11~1.40x speed-up,
better performance

Quantization

LLM-MQ [NeurIPS'23 Workshop]

Mixed-precision quantization.
2.8-bit quantization

MBQ [Under Review]

Modality-balanced quantization for VLM.
acc. improvement on MMMU: W3 up to **5.4%**, W4A8 up to **3.8%**

QLLM-Eval [ICML'24]

Evaluating the effect of quantization.
Providing knowledge and suggestions

Efficient Vision Generation

Algorithm-level

Time Step Compression

LCSC [Under Review]

Linear combination of checkpoints.
15~23x training acceleration,
1.25~2x timestep compression

USF [ICLR'24] **OMS-DPM** [ICML'23]

Search for optimal diffusion schedulers.
1.5~2x speed-up

DD [Under Review]

Distill AR into Flow Matching, can achieve **>100x** speedup for Image AR model

Model-level

Quantization

MixDQ [ECCV'24]

Mixed-precision quantization.
3x memory decrease,
1.5x speed-up

ViDiT-Q [Under Review]

Quantization for DiT.
2.5x memory improvement,
1.5x speed-up

Local Attn. & Act. Sharing

DiTFastAttn [NeurIPS'24]

Window & reused attention for DiT.
1.6x speed-up

Fast Compression

FlashEval [CVPR'24]

10x evaluation acceleration

目录 Contents

Efficient Techniques for LLMs

Directions to improve Large Language Models' efficiency

Prefill Stage

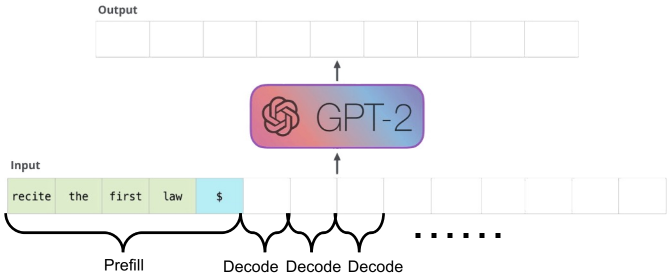
Reduce $t_{prefill}$,

M_{weight}, M_{other_act}

Decode Stage

Reduce t_{decode},

M_{weight}, $M_{kv\ cache}$



Overall Cost

(for each request)

Total Latency: $t_{prefill} + t_{decode} * N_{token}$

Total Memory: $M_{weight} + M_{kv\ cache} + M_{other\_act}$

Overview of Efficient Techniques

Lower Latency | Lower Storage | Higher Throughput | Lower Power Consumption

What algorithm property?	Cause what?	Solutions
Model Scale	• Large computation • Large memory access • Large memory footprint	**Data-level Optimization** • Prompt pruning • Soft prompt tuning • … • **Output Organization** Input Compression Output Organization
Attention Operation	• Input-quadratic computation • Input-quadratic memory access • Input-quadratic memory footprint	**Model-level Optimization** • Dynamic MoE • Low-complexity attention • **Quantization** • **Sparse Attention** • **Weight Pruning** • … Structure Design Model Compression
Decoding Approach	• Low arithmetic intensity (i.e., computation / memory access) cause under-utilization • Varying length -> Dynamically increasing KV cache cause fragmented memory, increasing both footprint and access	**System-level Optimization** • Graph and Operator Optimization • Speculative decoding • Memory Management • Batching • … Inference Engine Serving Framework

[1] Zhou, Zixuan*, **Ning, Xuefei***, et al. "A Survey on Efficient Inference for Large Language Models." arXiv 2024.

Skeleton-of-Thought (SoT)

To accelerate the sequential generation of LLM inference, SoT relies on the LLMs' planning ability and proposes a two-stage parallel generation scheme to achieve up to 2.39× end-to-end speed-up.

Challenge: Existing LLMs use the sequential generation that generates tokens one by one, resulting in a significant end-to-end latency and a low hardware utilizatio
cannot be parallelized

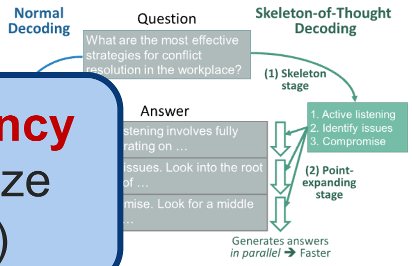
Motivation: Can we re
LLMs themselves to int
generation process, and thus do parallel generation to improve the hardware utilization and latency.

- Humans do not always think about questions in a sequential fashion. For many question types,
  - We first derive the skeleton according to some protocols and strategies
  - Then add details to explain each point.

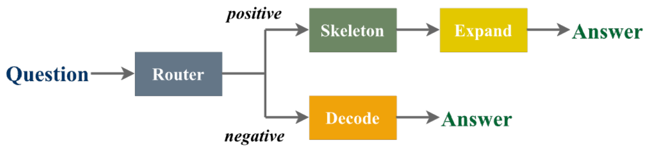
The 1st agentic generation method for efficiency
(relying on the LLM's emerging ability to organize the output, so as to introduce parallelizability)

Methodology: We propose SoT with two stages:

- **Skeleton Stage**: Guide the LLM to output a concise



We further extend **SoT with router (SoT-R)** to make the overall solution more practical. The router first decides whether to apply the SoT decoding mode based on the user's prompt.



[1] **Ning, Xuefei***, Zinan Lin*, et. al., "Skeleton-of-Thought: Prompting LLMs for Efficient Parallel Generation." ICLR 2024.

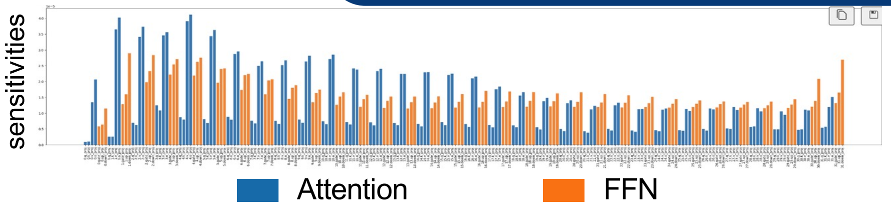
Mixed-precision Quantization (LLM-MQ)

To explore ultra-low bit-width weight quantization for accelerating LLM decoding, LLM-MQ proposes a mixed-precision quantization method and achieve 1.2~1.4× e2e speed-up

Challenge: Using a **consistent low bit-width format across all layers** is hard to push the bit-width to 3-bit without significant accuracy loss.

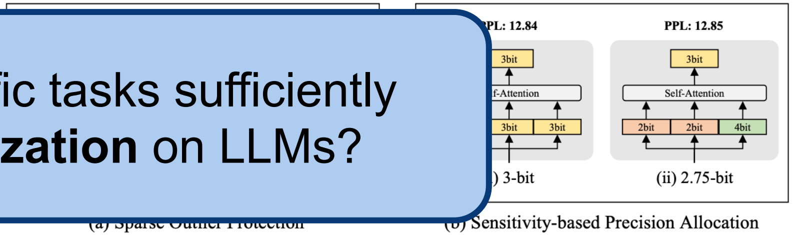
Motivation: Different layers have significant sensitivities, requirin… we aim to develop a… **method** for ultra lov…

Does the accuracy on specific tasks sufficiently reflect the **effect of quantization** on LLMs?



Sensitivity Analysis of LLaMA2-13B

Methodology:
- Keep the outliers with FP16 format.
- Assign high bit-width to sensitive layers in order to minimize the change in model output.



- For zero-shot understanding tasks:
  - When the average accuracy loss is around 0.1%, the model can be quantized to an average of 3.6 bits.
  - When the average accuracy loss is around 1%, the model can be quantized to an average of 2.8 bits.

[1] Li, Shiyao, **Ning, Xuefei** et. al., "LLM-MQ: Mixed-precision Quantization for Efficient LLM Deployment." NeurIPS Workshop 2023

Evaluating Quantized LLMs (QLLM Eval)

To optimize LLMs for efficiency across diverse scenarios, we propose a comprehensive evaluation for 11 LLM families under W, WA, KV quant on various tasks

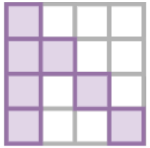
Knowledge Level	Key Knowledge
Tensor-level	**1. Tensor type (Sec. 3.2)**: The larger the model, the higher the tolerance for Weight-only and KV Cache Quantization, while the tolerance for Activation Quantization is lower. **2. Tensor position (Sec. 3.2)**: The sensitivity to quantization varies significantly across different tensor positions due to their distinct data distributions.
Model-level	**1. (Sec. 3.3)** The relative rankings of quantized LLMs are generally consistent with those of the FP16 LLMs when the bit-width is higher than W4, W4A8, and KV4. **2. (Sec. 3.3)** Leveraging MoE to increase the model size can improve the model's performance but may not improve the tolerance to quantization.
Task-level	**1. Emergent abilities (Sec. 4)**: The tolerance of Multi-Step Reasoning and Self-Calibration to quantization is lower than that of Instruction-Following and In-Context Learning abilities. **2. Dialogue tasks (Sec. 6)**: As the bit-width decreases, sentence-level repetition occurs first, followed by token-level repetition, and token-level randomness. **3. Long-Context tasks (Sec. 7)**: The longer the text, the larger the performance loss caused by Weight and KV Cache quantization. Most LLMs are more sensitive to KV Cache Quantization than Weight-only and Weight-Activation Quantization.
Bit-width Recommendation	**1. Basic NLP tasks (Sec. 3)**: W4, W4A8, KV4, W8KV4. **2. Emergent (Sec. 4)**: W8, W8A8, KV8 ($<$ 13B); W4, W4A8, KV4 ($\geq$ 13B). **3. Trustworthiness (Sec. 5)**: W8, W8A8, KV8 ($<$ 7B); W4, W4A8, KV4 ($\geq$ 7B). **4. Dialogue (Sec. 6)**: W8, W8A8, KV4. **5. Long-Context (Sec. 7)**: W4, W4A8, KV4 (token $<$ 4K); W4, W4A8, KV8 (token $\geq$ 4K). *(Note: Within 2% accuracy loss on the evaluated tasks. The recommended quantization bit-width may not generalize to other LLMs or tasks)*

[1] Li, Shiyao, **Ning, Xuefei,** et al. "Evaluating Quantized Large Language Models." ICML 2024.

Mixture of Attention (MoA)

For long-context LLM decoding, MoA assigns the heterogeneous elastic sparse pattern for each attention head and improves the inference throughput by 1.7-1.9x compared to vLLM.

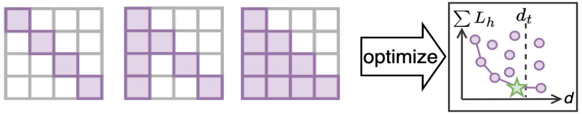
Challenge & Motivation: Existing methods employ a **uniform sparse attention mask** across each head and **fail to capture the diverse attention patterns** in LLMs.



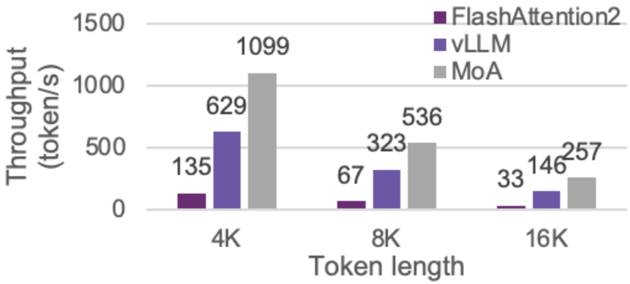
Unified Mask

Methodology: MoA automatically **tailors distinct sparse attention configurations to different heads and layers**.

- MoA constructs and navigates a search space of various attention patterns and their scaling rules relative to input sequence lengths



MoA: Heterogeneous Mask



On the Vicuna-7B, MoA achieves a **6.6-8.2x** throughput improvement compared to FlashAttn2 and a **1.7-1.9x** compared to vLLM.

	Token length			
Methodology	**32k**	**64k**	**128k**	**256k**
Dense Model	0.98	0.93	0.76	0.37
InfLLM	0.43	0.32	0.25	Out-Of-Time
StreamingLLM	0.52	0.48	0.41	0.25
MoA	**1.00**	**0.92**	**0.83**	**0.46**

A **50%** density mask is used to search for a **12k** sequence length. The pattern can be directly used on longer input up to 256k and achieves improvements.

[1] Fu, Tianyu*, Huang, Haofeng*, **Ning, Xuefei***, et al. "MoA: Mixture of Sparse Attention for Automatic Large Language Model Compression." arXiv 2024.

Efficient Expert Pruning (EEP)

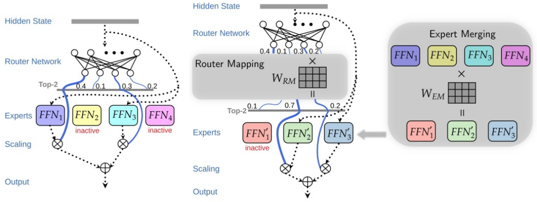
For MoE LLM inference, we propose to merge expert to inherit knowledge, achieving 75%/50% total/active expert sparsity. EEP can generalize on OOD data.

Challenge & Motivation:

- The mixture-of-expert (MoE) architecture has a large parameter size when there are many **experts**.
- We aim to **reduce the latency & peak memory through expert pruning** and **recover the accuracy through expert merging**.
- How to efficiently and best **inherit** the knowledge from original model when pruning experts?

Methodology

1. Use **weight merging** to preserve the "knowledge" of pruned experts
2. Use **evolutionary search** to optimize merging matrix
3. Use **absolute performance** as the objective.



Use Cases

1. Reduce total expert
2. Reduce active expert

$$M = x_{weight}\downarrow + M_{other}$$

$$t_{decode} \approx \frac{\#Mem\ Access}{BW \times BW\_Util}$$

$$t_{prefill} \approx \frac{\#OP}{Peak\_perf \times Comp\_Util}$$

[1] Liu, Enshu*, Zhu, Junyi*, Lin, Zinan+, **Xuefei Ning+**, et al. "Efficient Expert Pruning for Sparse Mixture-of-Experts Language Models: Enhancing Performance and Reducing Inference Costs" arXiv 2024.

Efficient Techniques for LLMs

Overall Cost
(for each request)

Total Latency: $t_{prefill} + t_{decode} * N_{token}$
Total Memory: $M_{weight} + M_{kv\ cache} + M_{other\_act}$

SoT
(Skeleton-of-Thought)

Total Latency: $t_{prefill} + t_{decode} * N_{token} / B$
Total Memory: $M_{weight} + M_{kv\ cache} + M_{other\_act}$

LLM-MQ, QLLM-Eval
(Quantization)

Total Latency: $t_{prefill} + t_{decode}\downarrow * N_{token}$
Total Memory: $M_{weight}\downarrow + M_{kv\ cache} + M_{other\_act}$

MoA
(Mixture of Attention)

Total Latency: $t_{prefill} + t_{decode}\downarrow * N_{token}$
Total Memory: $M_{weight} + M_{kv\ cache}\downarrow + M_{other\_act}$

EEP
(Efficient Expert Pruning)

Total Latency: $t_{prefill} + t_{decode}\downarrow * N_{token}$
Total Memory: $M_{weight}\downarrow + M_{kv\ cache} + M_{other\_act}$

目录 Contents

How DMs do inference

- Forward Process: Gradually add gaussian noise of different levels
- Backward Process: Gradually denoise the gaussian noise
- Intuition: the NN learns to predict the "noise" at each timestep.

[1] Ho, Jonathan et al. "Denoising Diffusion Probabilistic Models." ArXiv 2020.

Efficient Techniques for DMs

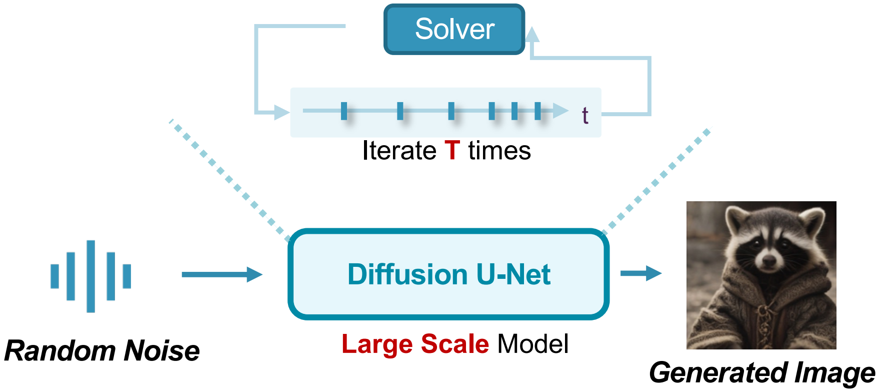
Directions to improve Diffusion Models' efficiency

Algorithm-level

Reduce $N_{timestep}$

Model-level

Reduce t_{model}, M_{weight}, M_{act}



Overall Cost

(for each iter)

Total Latency: $t_{model} * N_{timestep}$

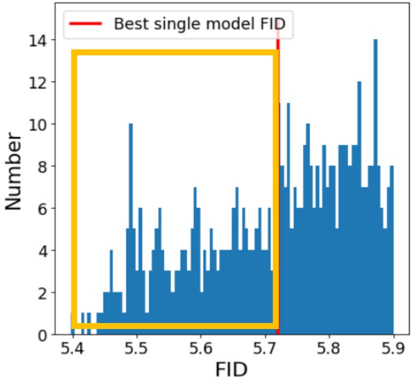
Total Memory: $M_{weight} + M_{activation}$

Optimizing the Model Schedule (OMS-DPM)

For Text-to-Image generation tasks, we propose a predictor-based search algorithm to optimize the model schedule, achieve 2× speed-up.

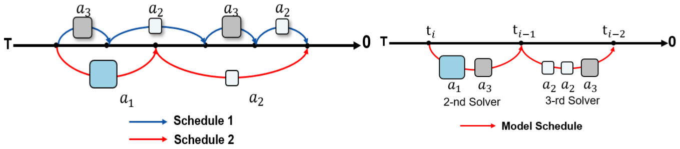
Challenge: Diffusion models require multiple forward passes of **large models** to generate images, leading to **high latency**.

Motivation: We find that smaller models outperform large models at some timesteps, suggesting that **mixing small and large models** can not only reduce latency but also potentially improve performance..

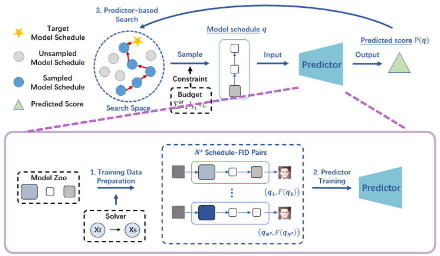


Methodology:

- Design a model schedule to **mix small and large diffusion models** during generation.
- **Train a predictor** to search better model schedule.



Model Schedule



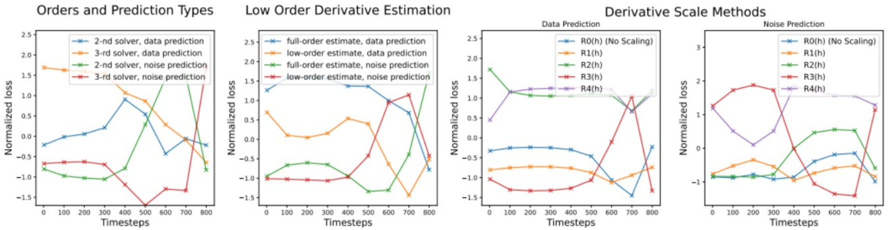
Predictor-based Search Method

[1] Liu, Enshu*, **Ning, Xuefei***, et al. "OMS-DPM: Optimizing the Model Schedule for Diffusion Probabilistic Models." ICML 2023.

Unified Sampling Framework (USF)

For text-to-image generation, we propose to optimize solver schedule, achieving 2× speed-up and enables sampling with very low NFE.

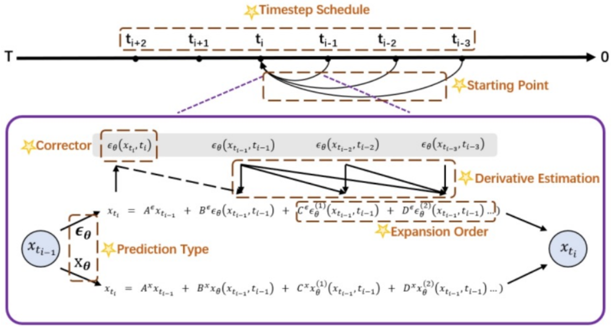
Challenge: Most of the current solvers use sub-optimal **empirical strategies**, cause poor quality with few number of function evaluations (<10).



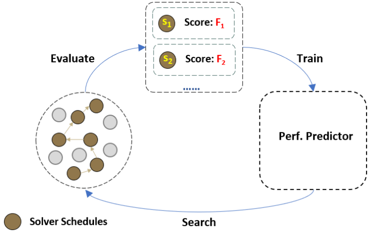
The ranking of all strategies changes over timestep

Motivation: Instead of using the empirical solver strategies, we aim to propose **a unified sampling framework to automatically search** for better solver strategies.

Methodology: Design a unified sampling framework with **6 different components**.



Construct a **performance predictor** to enable the fast evaluation of different solver schedules.

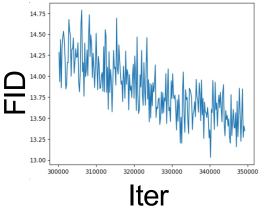


[1] Liu, Enshu, **Ning, Xuefei+,** et al. "A Unified Sampling Framework for Solver Searching of Diffusion Probabilistic Models". ICLR 2024.

Linear Combination of Saved Checkpoints (LCSC)

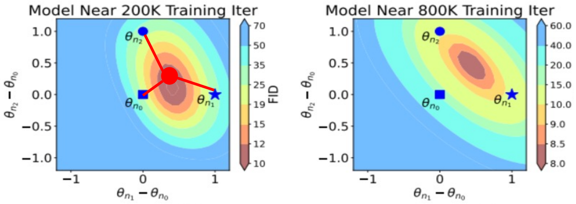
For the training of Consistency and Diffusion Models, we propose to combine checkpoints and optimize the coefficients, achieving 15~23× training speed-up on CM and 1.25~1.7× inference speed-up on DM.

Challenge: The **training variance** of DMs and CMs **is relatively high**, leading to suboptimal performance and large time cost (e.g., **>600 GPU hours** on CIFAR-10)

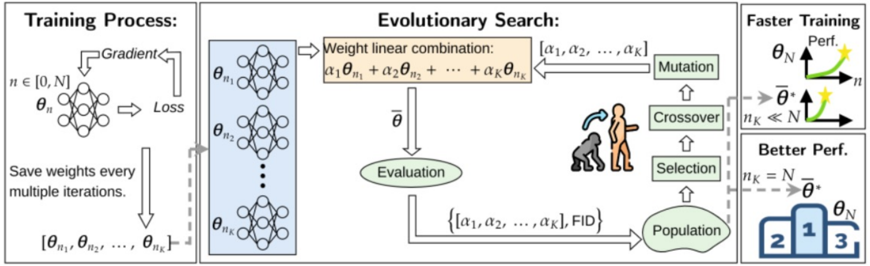


Perf. fluctuates greatly

Motivation: Combining checkpoints during the training process instead of relying solely on gradient-based training might help.



Methodology: Search the combination coefficients of saved checkpoints with evolutionary search.



Use Case: LCSC can (1) accelerate training process and (2) enhancing the performance of converged DMs and CMs.

[1] Liu, Enshu*, Zhu, Junyi*, Lin, Zinan+, **Ning, Xuefei+**, et al. "Linear Combination of Saved Checkpoints Makes Consistency and Diffusion Models Better." arXiv 2024.

Distilled Decoding of Image AR model (DD)

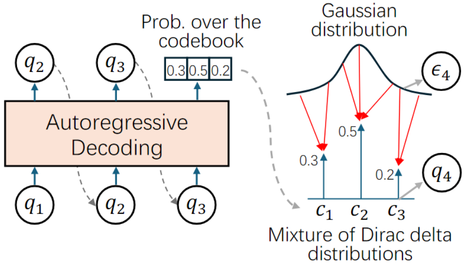
For AR image generation, we introduce noise token and propose distilled decoding, which generates image in 0.02s and can achieve >100x speedup with acceptable performance loss.

Challenge & Motivation: Auto-regressive (AR) image generation model often takes a significant inference latency **since the token generation cannot be parallelized**. The typical solutions try to model the distribution of multiple steps simultaneously, and **don't work for very few steps generation**.

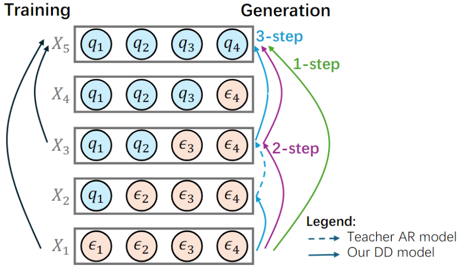
Methodology:

- Introduce **noise token** and **flow-matching** to construct an auto-regressive trajectory
- Train the model to skip further along the trajectory

1. Construct the trajectory



2. Training & Sampling



[1] Liu, Enshu, **Ning Xuefei**, Yu Wang, Zinan Lin. "Distilling Autoregressive Models Into Few Steps 1: Image Generation." arXiv 2024.

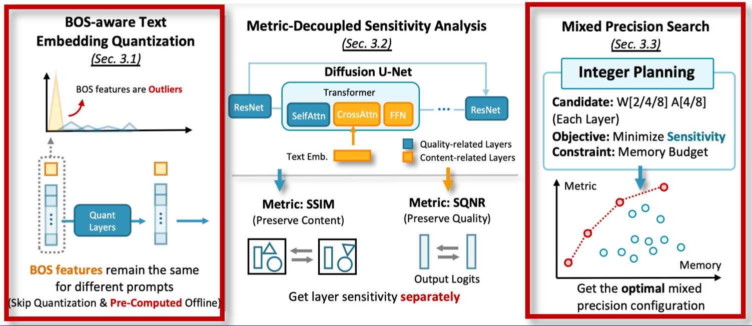
Mixed-precision Quantization (MixDQ)

For few-step DMs, MixDQ adopts a metric-decoupled sensitivity analysis method for mixed precision bitwidth allocation to achieve W4A8 quantization of 1-step DMs, achieving 2-3x peak memory reduction and 1.4x speed-up on RTX3090 (SDXL-Turbo, 1024x1024).

Challenge & Motivation: Few-step text-to-image DMs is hard to be quantized because of the **large outliers in text embeddings** and the existence of **highly sensitive layers**.

Methodology:

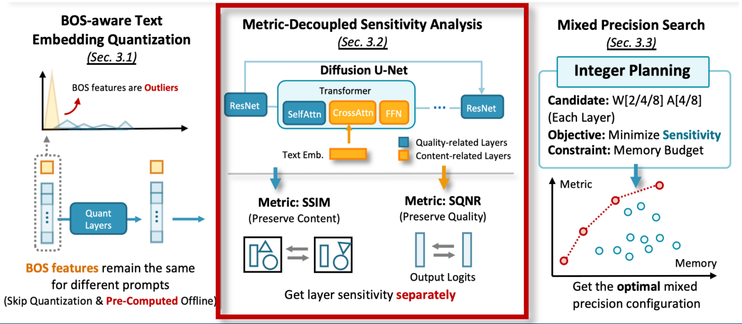
- Design **BOS-aware quantization** technique to protect the large outliers in text embeddings without quantization.
- Design **mixed-precision bit-width allocation** to assign high-precision for sensitive layers



Challenge & Motivation: Quantization affects both the **image quality and content**, so we need to consider both effects when assessing the sensitivity of each layer.

Methodology:

- Design **"Metric-decoupled" sensitivity analysis** to guide the bit-width allocation process.



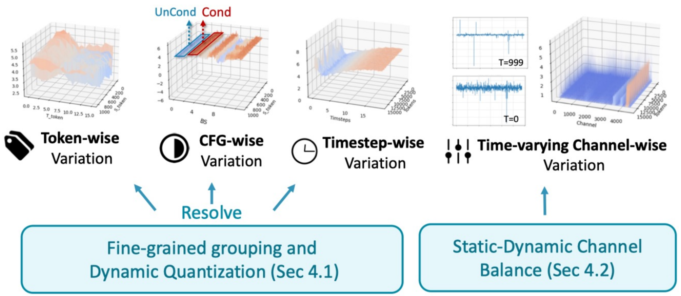
[1] Zhao, Tianchen*, **Ning, Xuefei*+**, et al. "MixDQ: Memory-Efficient Few-Step Text-to-Image Diffusion Models with Metric-Decoupled Mixed Precision Quantization." ECCV 2024.

Video and Image DiT Quantization (ViDiT-Q)

For DiT models and video generation, ViDiT-Q uses a static-dynamic channel balancing technique and metric-decoupled mixed-precision quantization to achieve lossless W8A8 with a 1.4-1.7x E2E speed-up on A100 (OpenSoRA and Pixart-α/σ).

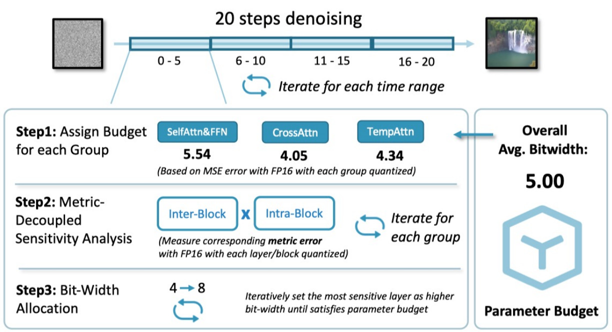
Challenge & Motivation: DiT (Diffusion Transformers) have unique properties for quantization**:**
- Highly variant activation distribution along token, CFG and timestep levels.
- Time-varying Channel-wise Imbalance.

Methodology:
- **Fine-grained group-wise dynamic quantization** for activations.
- **Static-Dynamic Channel Balance:** Combine the advantage of current **scale-based** and **rotation-based** balancing methods.



Challenge & Motivation: Quantization **affects multiple evaluation aspects** on video generation task.

Methodology: Quantization scheme tailored for visual generation task
- **Decouple the quantization's effect** on **multiple aspects** to preserve performance for each aspect.



[1] Zhao, Tianchen, …,**Ning, Xuefei+**, et al. "ViDiT-Q: Efficient and Accurate Quantization of Diffusion Transformers for Image and Video Generation." arXiv 2024.

Attention Compression (DiTFastAttn)

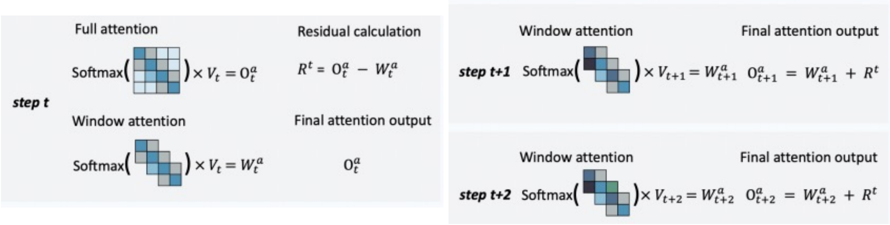
To reduce the attention overhead in DiTs, we leverage local attention and activation sharing, achieving 76% FLOPs and 1.8x speed-up on A100 (Pixart-σ, 2048x2048).

Challenge: DiTs excel at image & video generation but face computational challenges due to the O(N^2) complexity of self-attn.

Motivation: The changes in the attention output across different timesteps are **primarily driven by a local attention window**.

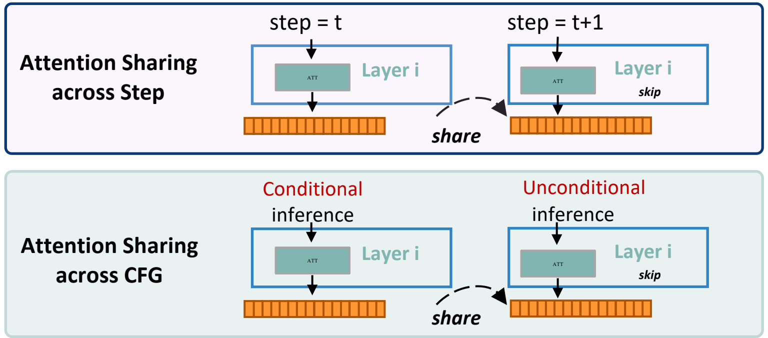
Methodology: Design window attention with residual

- In each timestep, we only compute the **local attention** and then add the **residual of previous global attention**, without the need to recompute the full global attention.



Motivation: We observe that in different **timesteps and CFG**, the attention outputs of a certain attention head exhibit **significant similarity**.

Methodology: Design to share the attention results across timesteps and CFG



[1] Yuan, Zhihang*, Lu, Pu*, Zhang, Hanling*, **Ning, Xuefei+**, et al. "DiTFastAttn: Attention Compression for Diffusion Transformer Models". NeurIPS 2024.

Efficient Techniques for DMs

Overall Cost (for each iter)	**Total Latency:** $t_{model} * N_{timestep}$ **Total Memory:** $M_{weight} + M_{activation}$
LCSC & OMS-DPM & USF & DD (Schedule Optimization)	**Total Latency:** $t_{model} * N_{timestep} \downarrow$ **Total Memory:** $M_{weight} + M_{activation}$
MixDQ & ViDiT-Q (Mixed-precision quantization)	**Total Latency:** $t_{model} \downarrow * N_{timestep}$ **Total Memory:** $M_{weight} \downarrow + M_{activation} \downarrow$
DiTFastAttn (Attention Compression)	**Total Latency:** $t_{model} \downarrow * N_{timestep}$ **Total Memory:** $M_{weight} + M_{activation} \downarrow$

目录 Contents

Research Summary

Overview

Survey
[Under Review]

Survey on efficient LLM inference techniques

Algorithm-level

SoT
[ICLR'24]

Parallel generation via prompting.
1.91~2.39x speed-up

Model-level

Sparse Attention

MoA
[Under Review]

Decide the heterogeneous elastic rule of the attention span for each head.
5.5~6.7x throughput improvement

Pruning

EEP
[Under Review]

Search the pruning pattern for MoE and use expert merging for finetuning.
48%~71% memory reduction,
1.11~1.40x speed-up,
better performance

Quantization

LLM-MQ
[NeurIPS'23 Workshop]

Mixed-precision quantization.
2.8-bit quantization

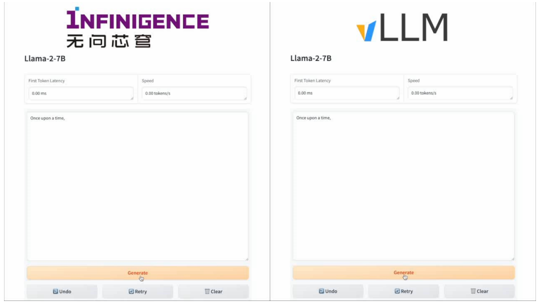
MBQ
[Under Review]

Modality-balanced quantization for VLM.
acc. improvement on MMMU: W3 up to **5.4%**, W4A8 up to **3.8%**

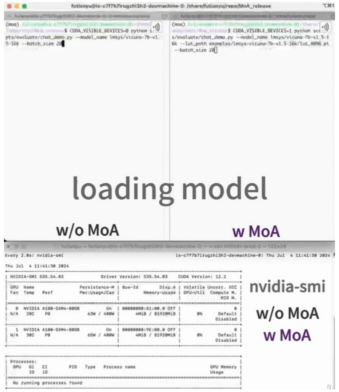
QLLM-Eval
[ICML'24]

Evaluating the effect of quantization.
Providing knowledge and suggestions

Efficient LLM/VLM



LLaMA-2-7B on AMD MI210
2× throughput improvement



Vicuna-7B on Nvidia-A100
batch size 20
end-to-end latency 2.3x

Research Summary

Algorithm-level
Time Step Compression

LCSC [Under Review]
Linear combination of checkpoints. **15~23x training acceleration, 1.25~2x timestep compression**

USF [ICLR'24]

OMS-DPM [ICML'23]

Search for optimal diffusion schedulers. **1.5~2x speed-up**

DD [Under Review]
Distill AR into Flow Matching, can achieve **>100x** speedup for Image AR model

Fast Compression

FlashEval [CVPR'24]
10x evaluation acceleration

Model-level
Quantization

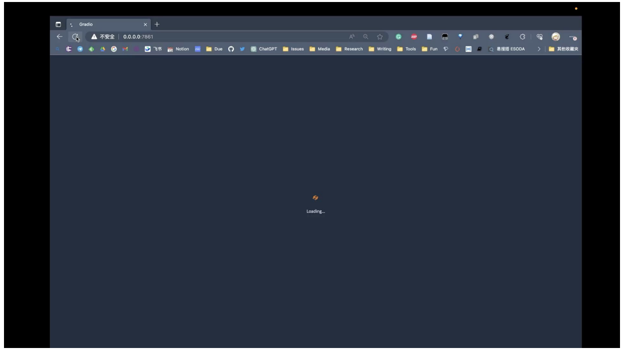
MixDQ [ECCV'24]
Mixed-precision quantization. **3x memory decrease, 1.5x speed-up**

ViDiT-Q [Under Review]
Quantization for DiT. **2.5x memory improvement, 1.5x speed-up**

Local Attn. & Act. Sharing

DiTFastAttn [NeurIPS'24]
Window & reused attention for DiT. **1.6x speed-up**

Efficient Vision Generation



Stable Diffusion on a single NVIDIA A100 GPU, Achieving **6.9×** speed-up and reducing **1.5×** memory

Pixart-Sigma, 2K generation on NVIDIA A100 GPU
1.8x latency speedup

OpenSORA, 512x512x16 Frames, on NVIDIA A100 GPU,
2x Memory Savings, **1.7x** latency speedup

References

- **Efficient LLM/VLM**

1. **SoT**: "Skeleton-of-Thought: Large Language Models Can Do Parallel Decoding." ICLR 2024. https://arxiv.org/abs/2307.15337
2. **LLM-MQ**: "LLM-MQ: Mixed-precision Quantization for Efficient LLM Deployment." NeurIPS Workshop' 23.
3. **QLLM-Eval**: "Evaluating Quantized Large Language Models." ICML 2024. https://arxiv.org/pdf/2402.18158
4. **Survey**: "A Survey on Efficient Inference for Large Language Models." arXiv 2024. https://arxiv.org/abs/2404.14294
5. **MoA**: "MoA: Mixture of Sparse Attention for Automatic Large Language Model Compression." Under review. https://arxiv.org/abs/2406.14909
6. **EEP**: "Efficient Expert Pruning for Sparse Mixture-of-Experts Language Models." Under review. https://arxiv.org/abs/2407.00945
7. **MBQ**: "MBQ: Modality-Balanced Quantization for Large Vision-Language Models." Under review.

- **Efficient Vision Generation**

1. **OMS-DPM**: "OMS-DPM: Optimizing the Model Schedule for Diffusion Probabilistic Models." ICML 2023. https://arxiv.org/abs/2306.08860
2. **USF**: "A Unified Sampling Framework for Solver Searching of Diffusion Probabilistic Models." ICLR 2024. https://arxiv.org/abs/2312.07243
3. **FlashEval**: "FlashEval: Towards Fast and Accurate Evaluation of Text-to-image Diffusion Generative Models." CVPR 2024. https://arxiv.org/abs/2403.16379
4. **LCSC**: "Linear Combination of Saved Checkpoints Makes Consistency and Diffusion Models Better." Under review. https://arxiv.org/abs/2404.02241
5. **MixDQ**: "MixDQ: Memory-Efficient Few-Step Text-to-Image Diffusion Models with Metric-Decoupled Mixed Precision Quantization. " ECCV 2024. https://arxiv.org/abs/2405.17873
6. **ViDiT-Q**: "ViDiT-Q: Efficient and Accurate Quantization of DiTs for Image and Video Generation. " Under review. https://arxiv.org/abs/2406.02540
7. **DiTFastAttn**: "DiTFastAttn: Attention Compression for DiT Models." NeurIPS 2024. https://arxiv.org/abs/2406.08552
8. **DD**: "Distilling Autoregressive Models into Few Steps 1: Image Generation." Under review.

We're Now Working On …

- **[Application-driven]** Applying and analyzing efficiency techniques on *multi-modality understanding models & video generative models*, to use them well
- **[Application-driven]** Developing methods for efficient *long-context inference*
- **[Application-driven]** *Pushing to the edge*: We want high compression ratio or a small model from scratch
    - Training-free techniques -> Training-based techniques
    - *Integrating efficiency techniques together*, to understand their interplay and use them well
    - How can we still *inherit the knowledge* well, or there is not difference from training from scratch?
- **[Algorithm-driven]** *Developing efficient generative algorithm*: Combining the benefits of data-space autoregressive models and flow matching

Thank You !

New Chinese Book
《高效深度学习：模型压缩与设计（全彩）》

Team Website

Xuefei Ning foxdoraame@gmail.com

Affiliated with: Department of Electronic Engineering, Tsinghua University

Lab Leader: Prof. Yu Wang yu-wang@tsinghua.edu.cn