

Beyond Decoding: Meta-Generation Algorithms for Large Language Models

Presenters: Matthew Finlayson, Hailey Schoelkopf, Sean Welleck

December 11, 2024

Efficient meta-generation

Scope:

- Basics of efficient generation
- How can we make meta-generation faster?
- Which specific meta-generators are most efficient?

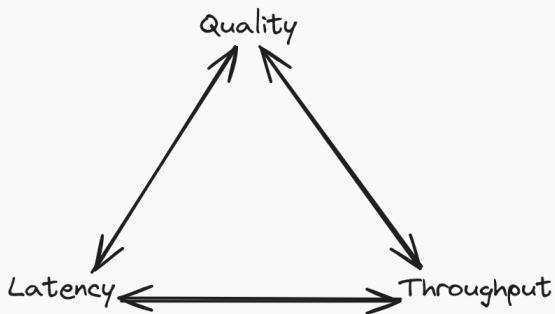
How do we measure “efficiency”?

- **Latency**

- *How long does a user wait for a response?*

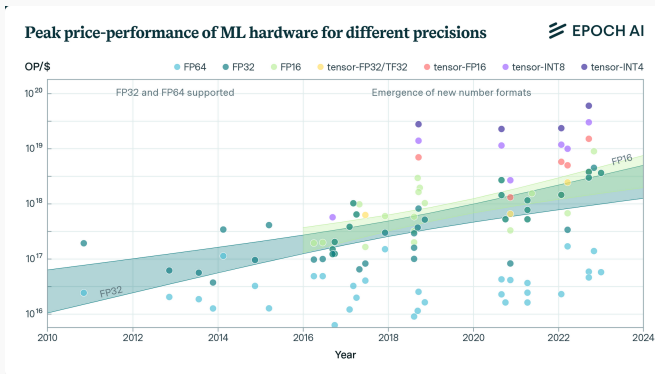
- **Throughput**

- *How many requests are completed per second?*



Latency, Throughput, and Quality often trade off at a given budget.

Hardware improvements have driven model improvements ²¹



The largest efficiency wins come from mapping operations onto hardware (more) effectively!

²¹Figure: [Hobbhahn et al., 2023]

How do ML accelerator designs impact generation efficiency?

- How much data can we keep on-device?
 - **VRAM (GB)**
- How many operations/second can the device perform?
 - **FLOP/s**
- How long does it take to send operands from GPU memory (HBM) to the processor?
 - **Memory Bandwidth (GB/s)**

- Loading inputs (activations) from memory
 - **Memory Bandwidth**
- Loading *weights* from memory
 - **Memory Bandwidth**
- Performing computation
 - **FLOP/s**
- Communicating across devices
 - **Communication Speeds (GB/s)**
- ...

Time per operation can be modeled as²²:

$$\text{Time} = \max \left(\frac{\text{Operation FLOP}}{\text{Device FLOP/s}}, \frac{\text{Data Transferred (GB)}}{\text{Memory Bandwidth (GB/s)}} \right)$$

Operations are either “compute-bound” or “memory-bound”²³

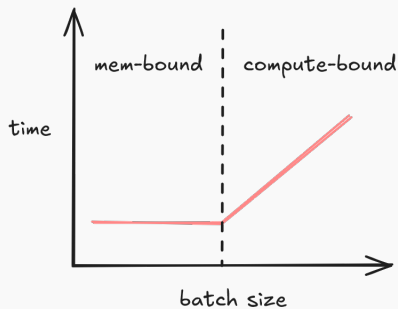
²²[He, 2022]

²³H100 SXM: BF16 dense tensor core max FLOP/s $\approx 1 \times 10^{15}$ FLOP/s, Memory bandwidth $\approx 3.35 \times 10^{12}$ byte/s. $\gg 100$ FLOP/byte is “free”!

Efficiency | batching



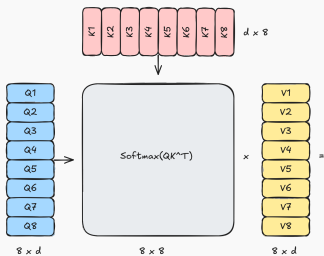
Inputs to a model can be “batched” together and computed simultaneously.



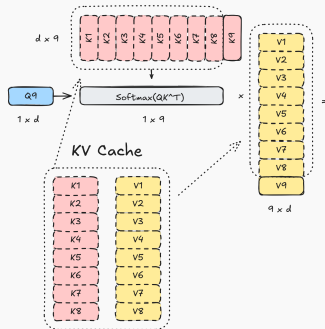
Batching can be cost-free for memory-bound operations!^a

^a<https://www.artfintel.com/p/how-does-batching-work-on-modern>

Efficiency | KV cache



Prefill Stage: process prompt all at once. Keys and values retained and initialize the “KV Cache”.



Decode Stage: use cached KV values to compute attention for current timestep. Append new K, V to KV cache

$$\text{Size} = (\text{batch} \cdot \text{n_ctx}) \cdot (2 \cdot \text{n_layer} \cdot \text{n_heads} \cdot \text{head_dim}) \cdot (\text{n_bytes})$$

Efficient meta-generation

How to speed up sampling a single token?

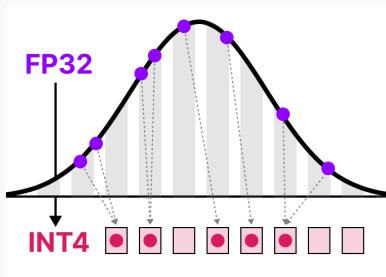
For a single decoding step, how do we work around hardware constraints?

- Memory Bandwidth ↓
- FLOP/s ↑
- FLOP ↓

Memory Bandwidth ↓: reduce data transferred

- Quantize weights or activations²⁴

$$\text{(bytes per parameter)} \cdot \text{(total parameters)}$$



- Compress or distill model

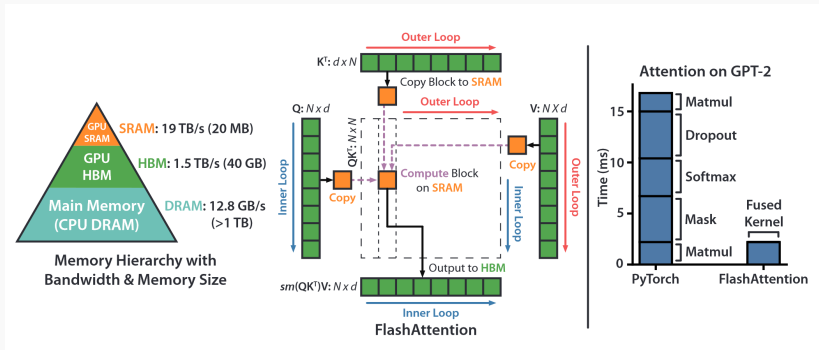
$$\text{(bytes per parameter)} \cdot \text{(total parameters)}$$

²⁴Visual from

Efficiency | single-token

FLOP/s \uparrow : improve hardware utilization

(FLOP per second) \cdot (total operation FLOP)

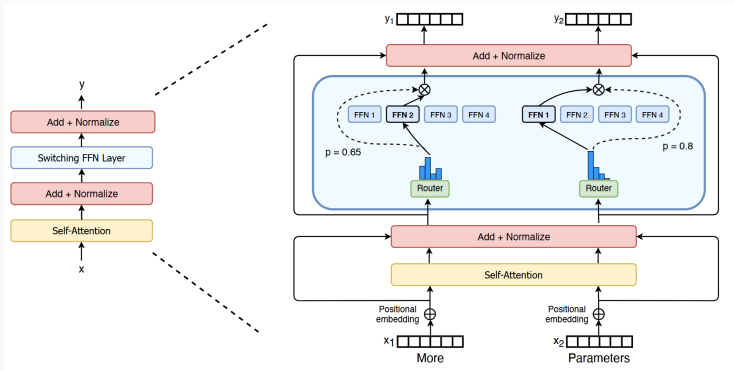


Flash Attention [Dao et al., 2022] performs the same operations, but optimizes the implementation to achieve far greater speed

Efficiency | single-token

FLOP ↓: reduce operations required

(FLOP per second) · (total operation FLOP)



Mixture-of-Experts models use fewer FLOP per token than equi-parameter dense models [Fedus et al., 2022]

Efficient meta-generation

How to speed up a single generation?

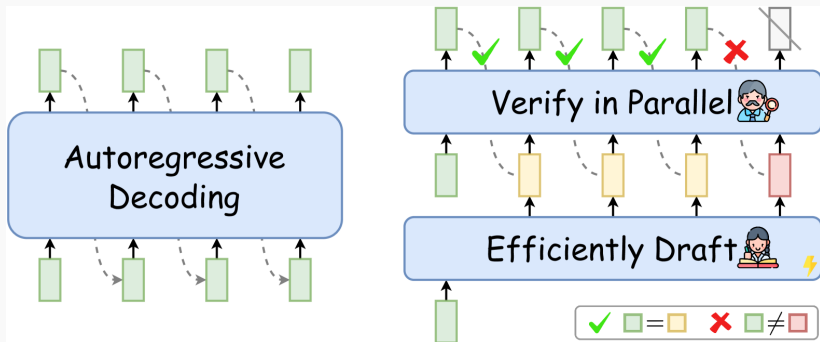
Generation of long outputs is bottlenecked by sequential next-token prediction. But not all tokens are created equal!

... The **cow** jumped over the moon . <EOS>

How can we spend less time on “easier” tokens?

Efficiency | single-generation

Decoding is typically memory-bound.

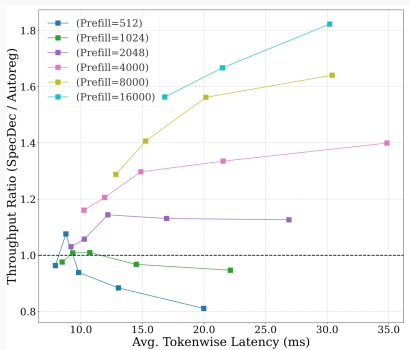


Speculative decoding uses a smaller **draft** model to produce “guesses” for the next N tokens cheaply, which are then “accepted” or “rejected” in parallel by the main model [Xia et al., 2024]

In speculative decoding:

- A lighter-weight *draft* model generates N “proposal” tokens
- These N “proposal” tokens can be passed **in parallel** into the main generator
- All tokens which match the main generator’s predictions are retained, and ones that do not are discarded

Efficiency | single-generation



Speculative decoding can harm throughput at low context but improves both throughput and latency at long context lengths [Chen et al., 2024a]

Efficient meta-generation

How to speed up meta-generation?

- How do meta-generators interact with **real-world efficiency** and **hardware utilization**?
- **Which** meta-generators are the fastest? Can we design more efficient meta-generators?

Shared Prefix

You are ChatGPT, a large language model trained by OpenAI, based on the GPT-4 architecture.

Knowledge cutoff: 2023-04

Current date: 2023-11-16

Image input capabilities: Enabled

When you send a message containing Python code to python, it will be executed in a stateful Jupyter notebook environment. Python will respond...

Unique Suffixes

Hi, can you write a...

Tell me a funny...

Who is Alan Turing?

Debug this Python...

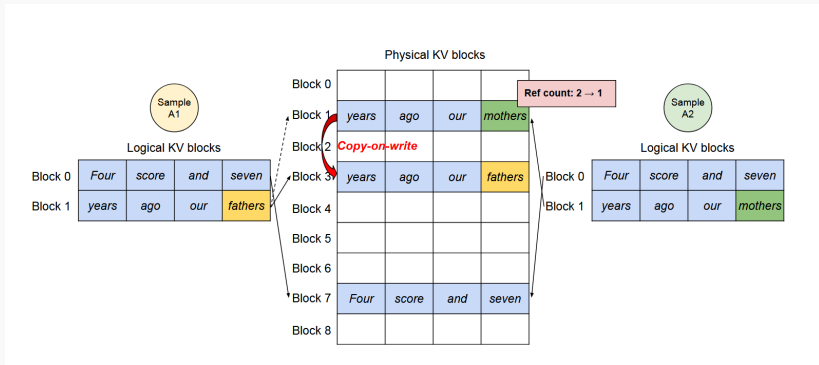
Ignore all previous...

Shared Prefix Setting

Common deployment and parallel generation scenarios have redundant **shared prefix** content in prompts²⁵

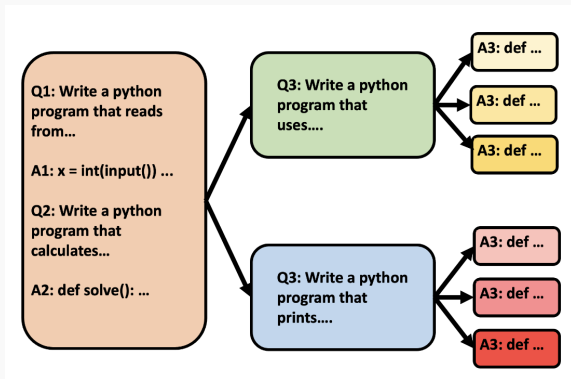
²⁵Figure from [Juravsky et al., 2024]

Efficiency | meta-generators | KV Cache reuse



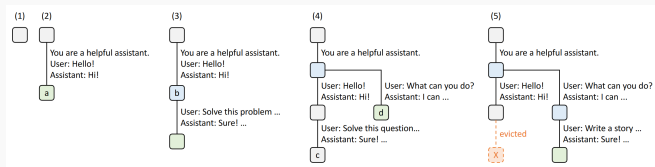
PagedAttention [Kwon et al., 2023] prevents redundant storage costs by mapping KV cache blocks to physical “pages” of VRAM

KV Cache reuse is not limited to single-level shared prefixes!



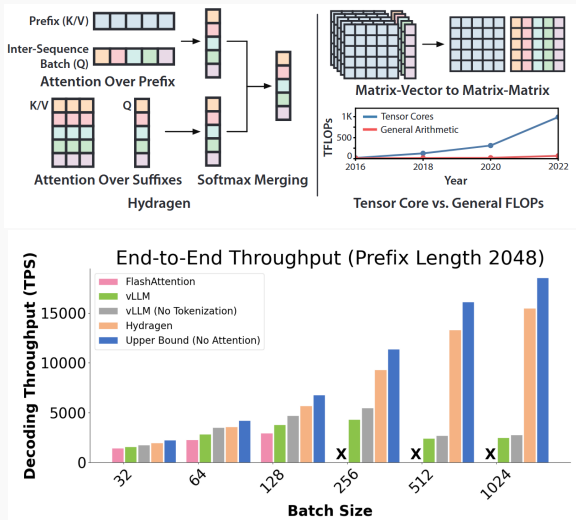
Multiple levels of prefix sharing can arise frequently: for example, combining a long few-shot prompt with Best-of-N generation²⁶

Efficiency | meta-generators | KV Cache reuse



RadixAttention enables complex prefix sharing patterns [Zheng et al., 2024], evicting least-recently-used KV cache blocks from memory when needed

Efficiency | meta-generators | KV Cache reuse



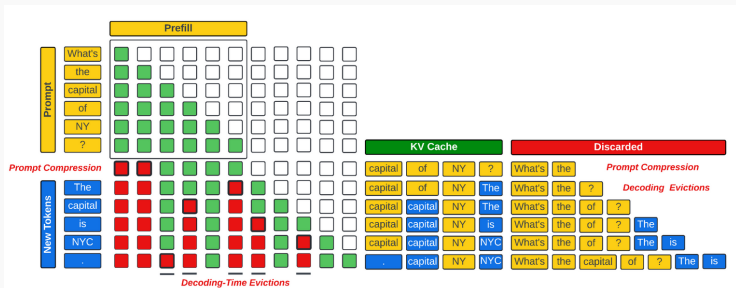
Hydragen [Juravsky et al., 2024] makes shared-prefix attention components faster via leveraging Tensor Cores

KV Cache size is a key bottleneck to larger batches and to longer context inference

- **Token Dropping:** Selectively remove tokens from the KV Cache
- **Quantization:** Modify KV Cache datatype
- **Architectural Modification:** Reduce inherent size of a prospective model's KV Cache

Token Dropping:

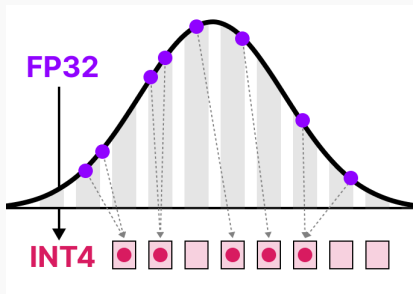
$$(\text{batch} \cdot n_{\text{ctx}}) \cdot (2 \cdot n_{\text{layer}} \cdot n_{\text{heads}} \cdot \text{head}_{\text{dim}}) \cdot (n_{\text{bytes}})$$



An overview of approaches to control KV Cache size via *token dropping* [Adams et al., 2024]

Quantization:

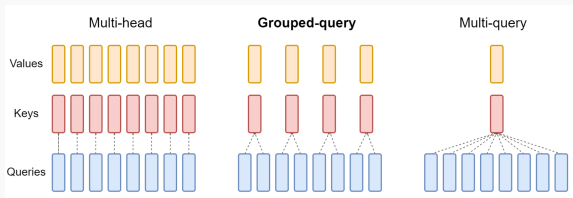
$$(\text{batch} \cdot \text{n_ctx}) \cdot (2 \cdot \text{n_layer} \cdot \text{n_heads} \cdot \text{head_dim}) \cdot (\text{n_bytes})$$



As with model weights, elements of the KV cache can be *quantized* to reduce memory overheads

Architectural Modification:

$$(\text{batch} \cdot \text{n_ctx}) \cdot (2 \cdot \text{n_layer} \cdot \text{n_heads} \cdot \text{head_dim}) \cdot (\text{n_bytes})$$






Architectural tweaks such as Multi-Query Attention [Shazeer, 2019] or Grouped-Query Attention [Ainslie et al., 2023] reduce the number of Key + Value attention heads to shrink the required KV Cache size




Which meta-generators are most efficient?

- **Parallelizable**: trajectories can be run in parallel; not sequentially bottlenecked
- **Prefix-shareable**: long inputs are presented as identical shared prefix content, whose KV Caches can be reused across many model calls

Token budget is not the only indicator of meta-generator efficiency!

-  Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985).
A learning algorithm for boltzmann machines.
Cognitive Science, 9(1):147–169.
-  Adams, G., Ladhak, F., Schoelkopf, H., and Biswas, R. (2024).
Cold compress: A toolkit for benchmarking kv cache compression approaches.
-  Aggarwal, P., Parno, B., and Welleck, S. (2024).
Alphaverus: Bootstrapping formally verified code generation through self-improving translation and treefinement.
<https://arxiv.org/abs/2412.06176>.

References ii

-  Ainslie, J., Lee-Thorp, J., de Jong, M., Zemlyanskiy, Y., Lebrón, F., and Sanghai, S. (2023).
Gqa: Training generalized multi-query transformer models from multi-head checkpoints.
-  Ankner, Z., Paul, M., Cui, B., Chang, J. D., and Ammanabrolu, P. (2024).
Critique-out-loud reward models.
-  Asai, A., He*, J., Shao*, R., Shi, W., Singh, A., Chang, J. C., Lo, K., Soldaini, L., Feldman, Tian, S., Mike, D., Wadden, D., Latzke, M., Minyang, Ji, P., Liu, S., Tong, H., Wu, B., Xiong, Y., Zettlemoyer, L., Weld, D., Neubig, G., Downey, D., Yih, W.-t., Koh, P. W., and Hajishirzi, H. (2024).
OpenScholar: Synthesizing scientific literature with retrieval-augmented language models.

Arxiv.



Basu, S., Ramachandran, G. S., Keskar, N. S., and Varshney, L. R. (2021).

Mirostat: a neural text decoding algorithm that directly controls perplexity.

In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.



Bertsch, A., Xie, A., Neubig, G., and Gormley, M. (2023).

It's MBR all the way down: Modern generation techniques through the lens of minimum Bayes risk.

In Elazar, Y., Ettinger, A., Kassner, N., Ruder, S., and A. Smith, N., editors, *Proceedings of the Big Picture Workshop*, pages 108–122, Singapore. Association for Computational Linguistics.



Brown, B., Juravsky, J., Ehrlich, R., Clark, R., Le, Q. V., Ré, C., and Mirhoseini, A. (2024).

Large language monkeys: Scaling inference compute with repeated sampling.

<https://arxiv.org/abs/2407.21787>.



Chen, J., Tiwari, V., Sadhukhan, R., Chen, Z., Shi, J., Yen, I. E.-H., and Chen, B. (2024a).


Magicdec: Breaking the latency-throughput tradeoff for long context generation with speculative decoding.



Chen, X., Lin, M., Schärli, N., and Zhou, D. (2024b).


Teaching large language models to self-debug.

In The Twelfth International Conference on Learning Representations.

 Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H., Petrov, S., Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., and Wei, J. (2022).




Scaling instruction-finetuned language models.




<https://arxiv.org/abs/2210.11416>.

 Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. (2021).

Training verifiers to solve math word problems.

<https://arxiv.org/abs/2110.14168>.

-  Dao, T., Fu, D. Y., Ermon, S., Rudra, A., and R'e, C. (2022).
Flashattention: Fast and memory-efficient exact attention with io-awareness.
ArXiv preprint, abs/2205.14135.
-  Dohan, D., Xu, W., Lewkowycz, A., Austin, J., Bieber, D., Lopes, R. G., Wu, Y., Michalewski, H., Saurous, R. A., Sohl-dickstein, J., Murphy, K., and Sutton, C. (2022).
Language model cascades.
<https://arxiv.org/abs/2207.10342>.
-  Fan, A., Lewis, M., and Dauphin, Y. (2018).
Hierarchical neural story generation.
In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 889–898. Association for Computational Linguistics.

-  Fedus, W., Zoph, B., and Shazeer, N. (2022).
Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity.
-  Feng, G., Zhang, B., Gu, Y., Ye, H., He, D., and Wang, L. (2023).
Towards revealing the mystery behind chain of thought: A theoretical perspective.
In Thirty-seventh Conference on Neural Information Processing Systems.
-  Finlayson, M., Hewitt, J., Koller, A., Swayamdipta, S., and Sabharwal, A. (2024).
Closing the curious case of neural text degeneration.
In The Twelfth International Conference on Learning Representations.



Freitag, M. and Al-Onaizan, Y. (2017).

Beam search strategies for neural machine translation.

In *Proceedings of the First Workshop on Neural Machine Translation*, pages 56–60. Association for Computational Linguistics.



He, H. (2022).

Making deep learning go brrrr from first principles.



Hewitt, J., Manning, C., and Liang, P. (2022).

Truncation sampling as language model desmoothing.

In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3414–3427. Association for Computational Linguistics.



Hobbhahn, M., Heim, L., and Aydos, G. (2023).

Trends in machine learning hardware.

Accessed: 2024-11-26.



Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. (2020).

The curious case of neural text degeneration.

In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020.




OpenReview.net.






Huang, J., Chen, X., Mishra, S., Zheng, H. S., Yu, A. W., Song, X., and Zhou, D. (2024).

Large language models cannot self-correct reasoning yet.

In The Twelfth International Conference on Learning Representations.

-  Jiang, A. Q., Welleck, S., Zhou, J. P., Lacroix, T., Liu, J., Li, W., Jamnik, M., Lampl, G., and Wu, Y. (2023).
Draft, sketch, and prove: Guiding formal theorem provers with informal proofs.
In The Eleventh International Conference on Learning Representations.
-  Juravsky, J., Brown, B., Ehrlich, R., Fu, D. Y., Ré, C., and Mirhoseini, A. (2024).
Hydragen: High-throughput llm inference with shared prefixes.
-  Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. (2020).
Scaling laws for neural language models.
<https://arxiv.org/abs/2001.08361>.

-  Khattab, O., Santhanam, K., Li, X. L., Hall, D. L. W., Liang, P., Potts, C., and Zaharia, M. A. (2022).
Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive nlp.
ArXiv, abs/2212.14024.
-  Kim, S., Suk, J., Longpre, S., Lin, B. Y., Shin, J., Welleck, S., Neubig, G., Lee, M., Lee, K., and Seo, M. (2024).
Prometheus 2: An open source language model specialized in evaluating other language models.
<https://arxiv.org/abs/2405.01535>.
-  Koh, J. Y., McAleer, S., Fried, D., and Salakhutdinov, R. (2024).
Tree search for language model agents.
arXiv preprint arXiv:2407.01476.



Kudo, T. (2018).

Subword regularization: Improving neural network translation models with multiple subword candidates.



In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia.


Association for Computational Linguistics.



Kumar, A., Zhuang, V., Agarwal, R., Su, Y., Co-Reyes, J. D., Singh, A., Baumli, K., Iqbal, S., Bishop, C., Roelofs, R., Zhang, L. M., McKinney, K., Shrivastava, D., Paduraru, C., Tucker, G., Precup, D., Behbahani, F., and Faust, A. (2024).


Training language models to self-correct via reinforcement learning.

-  Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. (2023).
Efficient memory management for large language model serving with pagedattention.
-  Li, X. L., Holtzman, A., Fried, D., Liang, P., Eisner, J., Hashimoto, T., Zettlemoyer, L., and Lewis, M. (2023a).
Contrastive decoding: Open-ended text generation as optimization.
In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors,
Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 12286–12312. Association for Computational Linguistics.

 Li, Y., Choi, D., Chung, J., Kushman, N., Schrittwieser, J., Leblond, R., Eccles, T., Keeling, J., Gimeno, F., Lago, A. D., Hubert, T., Choy, P., de Masson d'Autume, C., Babuschkin, I., Chen, X., Huang, P.-S., Welbl, J., Goyal, S., Cherepanov, A., Molloy, J., Mankowitz, D. J., Robson, E. S., Kohli, P., de Freitas, N., Kavukcuoglu, K., and Vinyals, O. (2022).

Competition-level code generation with alphacode.

Science, 378(6624):1092–1097.

 Li, Y., Lin, Z., Zhang, S., Fu, Q., Chen, B., Lou, J.-G., and Chen, W. (2023b).

Making language models better reasoners with step-aware verifier.

In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), pages 5315–5333, Toronto, Canada. Association for Computational Linguistics.



Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. (2024).

Let's verify step by step.

In The Twelfth International Conference on Learning Representations.



Liu, A., Han, X., Wang, Y., Tsvetkov, Y., Choi, Y., and Smith, N. A. (2024).

Tuning language models by proxy.



Liu, A., Sap, M., Lu, X., Swayamdipta, S., Bhagavatula, C., Smith, N. A., and Choi, Y. (2021).

DExperts: Decoding-time controlled text generation with experts and anti-experts.


In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 6691–6706. Association for Computational Linguistics.



Lu, X., Brahman, F., West, P., Jung, J., Chandu, K., Ravichander, A., Ammanabrolu, P., Jiang, L., Ramnath, S., Dziri, N., Fisher, J., Lin, B., Hallinan, S., Qin, L., Ren, X., Welleck, S., and Choi, Y. (2023).

Inference-time policy adapters (IPA): Tailoring extreme-scale LMs without fine-tuning.

In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6863–6883. Association for Computational Linguistics.

-  Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegrefe, S., Alon, U., Dziri, N., Prabhume, S., Yang, Y., Gupta, S., Majumder, B. P., Hermann, K., Welleck, S., Yazdanbakhsh, A., and Clark, P. (2023).

Self-refine: Iterative refinement with self-feedback.

In Thirty-seventh Conference on Neural Information Processing Systems.

-  Meister, C., Cotterell, R., and Vieira, T. (2020).

If beam search is the answer, what was the question?

In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 2173–2185.

Association for Computational Linguistics.



Meister, C., Pimentel, T., Wiher, G., and Cotterell, R. (2022).

Locally typical sampling.

Transactions of the Association for Computational Linguistics,
11:102–121.



Meister, C., Pimentel, T., Wiher, G., and Cotterell, R. (2023).

Locally typical sampling.

Transactions of the Association for Computational Linguistics,
11:102–121.



Merrill, W. and Sabharwal, A. (2024).


The expressive power of transformers with chain of thought.

*In The Twelfth International Conference on Learning
Representations.*

 Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., Jiang, X., Cobbe, K., Eloundou, T., Krueger, G., Button, K., Knight, M., Chess, B., and Schulman, J. (2022).

Webgpt: Browser-assisted question-answering with human feedback.

<https://arxiv.org/abs/2112.09332>.

 Nebius (2024).

Leveraging training and search for better software engineering agents.

<https://nebius.com/blog/posts/training-and-search-for-software-engineering-agents>.



Nowak, F., Svete, A., Butoi, A., and Cotterell, R. (2024).

On the representational capacity of neural language models with chain-of-thought reasoning.

In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12510–12548, Bangkok, Thailand. Association for Computational Linguistics.



OpenAI (2024).

Learning to reason with llms.

<https://openai.com/index/learning-to-reason-with-llms/>.



Polu, S. and Sutskever, I. (2020).

Generative language modeling for automated theorem proving.



Press, O., Zhang, M., Min, S., Schmidt, L., Smith, N., and Lewis, M. (2023).

Measuring and narrowing the compositionality gap in language models.

In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711. Association for Computational Linguistics.



Schlag, I., Sukhbaatar, S., Celikyilmaz, A., tau Yih, W., Weston, J., Schmidhuber, J., and Li, X. (2023).


Large language model programs.

<https://arxiv.org/abs/2305.05364>.




Shazeer, N. (2019).

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

 Shi, C., Yang, H., Cai, D., Zhang, Z., Wang, Y., Yang, Y., and Lam, W. (2024).

A thorough examination of decoding methods in the era of LLMs.

In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N., editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8601–8629, Miami, Florida, USA. Association for Computational Linguistics.

 Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016).

Mastering the game of go with deep neural networks and tree search.

Nature, 529:484–503.



Stahlberg, F. and Byrne, B. (2019).

On nmt search errors and model errors: Cat got your tongue?

ArXiv, abs/1908.10090.



Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F. (2020).

Learning to summarize with human feedback.

In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 3008–3021. Curran Associates, Inc.



Sun, Z., Yu, L., Shen, Y., Liu, W., Yang, Y., Welleck, S., and Gan, C. (2024).

Easy-to-hard generalization: Scalable alignment beyond human supervision.

In The Thirty-eighth Annual Conference on Neural Information Processing Systems.



Uesato, J., Kushman, N., Kumar, R., Song, F., Siegel, N., Wang, L., Creswell, A., Irving, G., and Higgins, I. (2022).

Solving math word problems with process- and outcome-based feedback.



Wang, P., Li, L., Shao, Z., Xu, R., Dai, D., Li, Y., Chen, D., Wu, Y., and Sui, Z. (2024a).

Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations.


In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, Bangkok, Thailand. Association for Computational Linguistics.



Wang, X., Wei, J., Schuurmans, D., Le, Q. V., Chi, E. H., Narang, S., Chowdhery, A., and Zhou, D. (2023).

Self-consistency improves chain of thought reasoning in language models.

In *The Eleventh International Conference on Learning Representations*.

-  Wang, Y., Wu, Y., Wei, Z., Jegelka, S., and Wang, Y. (2024b).
A theoretical understanding of self-correction through in-context alignment.
<https://arxiv.org/abs/2405.18634>.
-  Wei, J., Wang, X., Schuurmans, D., Bosma, M., brian ichter, Xia, F., Chi, E. H., Le, Q. V., and Zhou, D. (2022).
Chain of thought prompting elicits reasoning in large language models.
In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K., editors,
Advances in Neural Information Processing Systems.



Welleck, S., Bertsch, A., Finlayson, M., Schoelkopf, H., Xie, A., Neubig, G., Kulikov, I., and Harchaoui, Z. (2024).

From decoding to meta-generation: Inference-time algorithms for large language models.

<https://arxiv.org/abs/2406.16838>.




Welleck, S., Kulikov, I., Roller, S., Dinan, E., Cho, K., and Weston, J. (2020).

Neural text generation with unlikelihood training.

In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020.

OpenReview.net.


 Welleck, S., Lu, X., West, P., Brahman, F., Shen, T., Khashabi, D., and Choi, Y. (2023).

Generating sequences by learning to self-correct.

In The Eleventh International Conference on Learning Representations.




 Weston, J. and Sukhbaatar, S. (2023).



System 2 attention (is something you might need too).

 Wu, I., Fernandes, P., Bertsch, A., Kim, S., Pakazad, S., and Neubig, G. (2024a).

Better instruction-following through minimum bayes risk.

<https://arxiv.org/abs/2410.02902>.

-  Wu, Y., Sun, Z., Li, S., Welleck, S., and Yang, Y. (2024b). **Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models.**
<https://arxiv.org/abs/2408.00724>.
-  Xia, H., Yang, Z., Dong, Q., Wang, P., Li, Y., Ge, T., Liu, T., Li, W., and Sui, Z. (2024). **Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding.**
-  Zaharia, M., Khattab, O., Chen, L., Davis, J. Q., Miller, H., Potts, C., Zou, J., Carbin, M., Frankle, J., Rao, N., and Ghodsi, A. (2024). **The shift from models to compound ai systems.**
<https://bair.berkeley.edu/blog/2024/02/18/compound-ai-systems/>.

-  Zhang, L., Hosseini, A., Bansal, H., Kazemi, M., Kumar, A., and Agarwal, R. (2024).
Generative verifiers: Reward modeling as next-token prediction.
-  Zheng, L., Yin, L., Xie, Z., Sun, C., Huang, J., Yu, C. H., Cao, S., Kozyrakis, C., Stoica, I., Gonzalez, J. E., Barrett, C., and Sheng, Y. (2024).
Sglang: Efficient execution of structured language model programs.