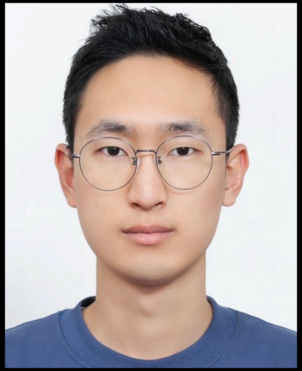
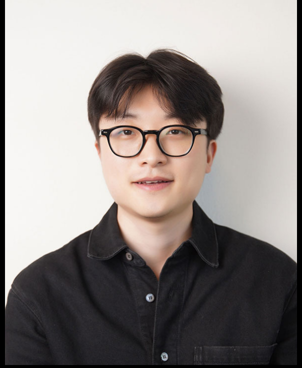
Predictive Pipelined Decoding:
A compute-Latency Trade-off for Exact LLM Decoding

Seongjun Yang*

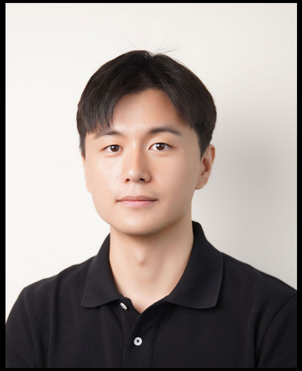


Gibbeum Lee*



Jaewoong Cho

Dimitris Papailiopoulos



Kangwook Lee

TRANSACTIONS tmlr on ML RESEARCH

KRAFTON

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Motivation

- To improve inference-time efficiency of transformer, many methods are proposed.
  - Model pruning techniques [1-8]
  - Knowledge distillation [9-10]
  - Quantization procedure [11-18]
  - Early-exiting algorithm [19-20]

Motivation

- To improve inference-time efficiency of transformer, many methods are proposed.
  - Model pruning techniques [1-8]
  - Knowledge distillation [9-10]
  - Quantization procedure [11-18]
  - Early-exiting algorithm [19-20]

However, It does not ensure ***the exact same output*** as the original decoding.

Motivation

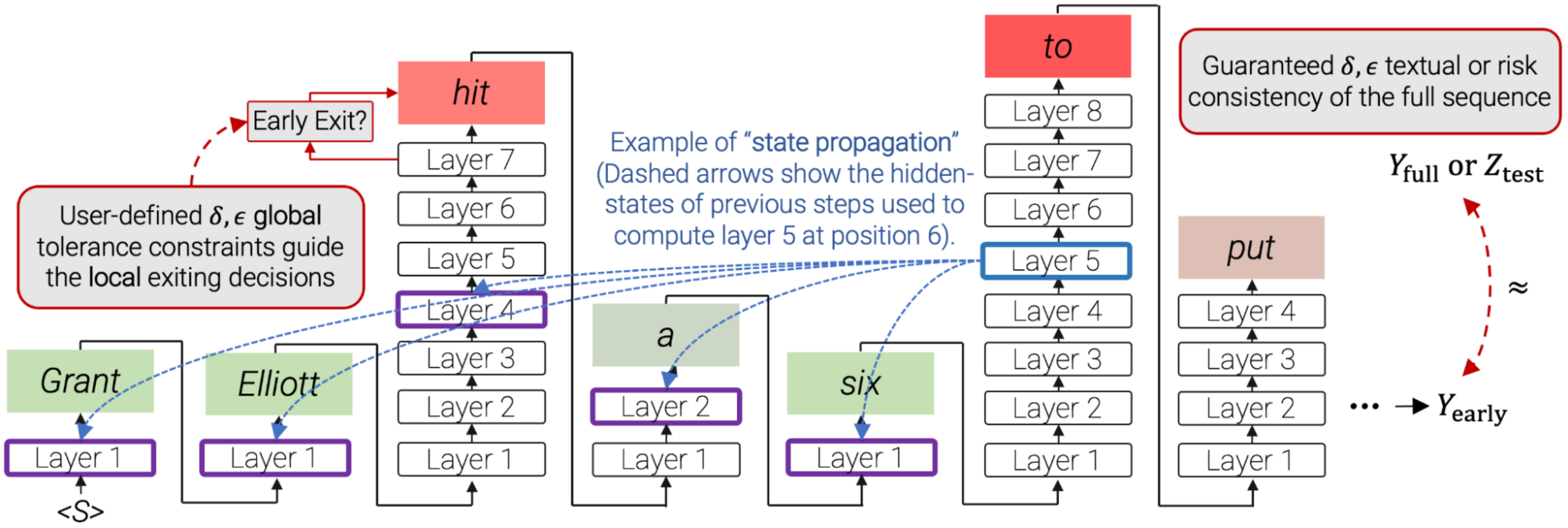
- To improve inference-time efficiency of transformer, many methods are proposed.
  - Model pruning techniques [1-8]
  - Knowledge distillation [9-10]
  - Quantization procedure [11-18]
  - Early-exiting algorithm [19-20]

However, It does not ensure ***the exact same output*** as the original decoding.

We propose Predictive Pipelined Decoding (*PPD*) to reduce latency **while preserving decoding results.**

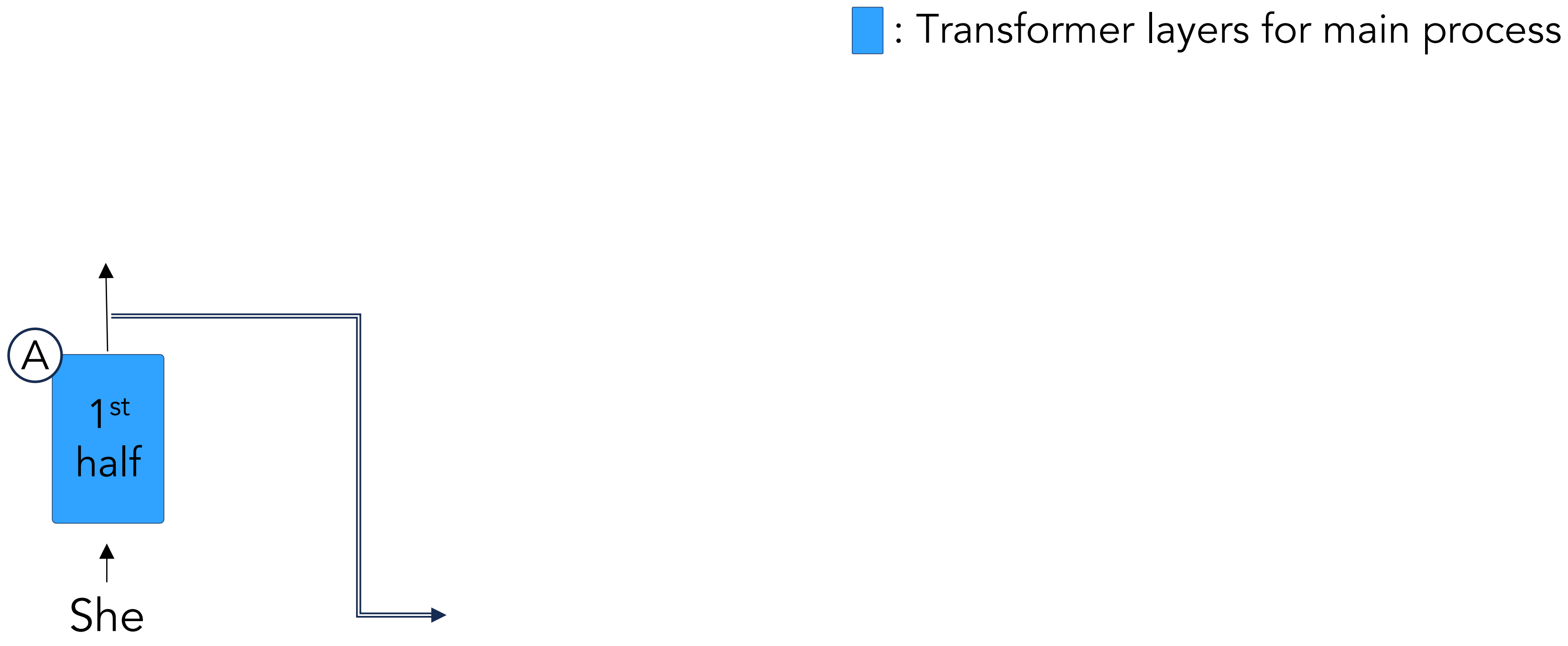
Early-exiting algorithm

- Inspired by early-exiting algorithm [19]
  - Decide when to stop computing based on the confidence of predictions



Predictive Pipelined Decoding

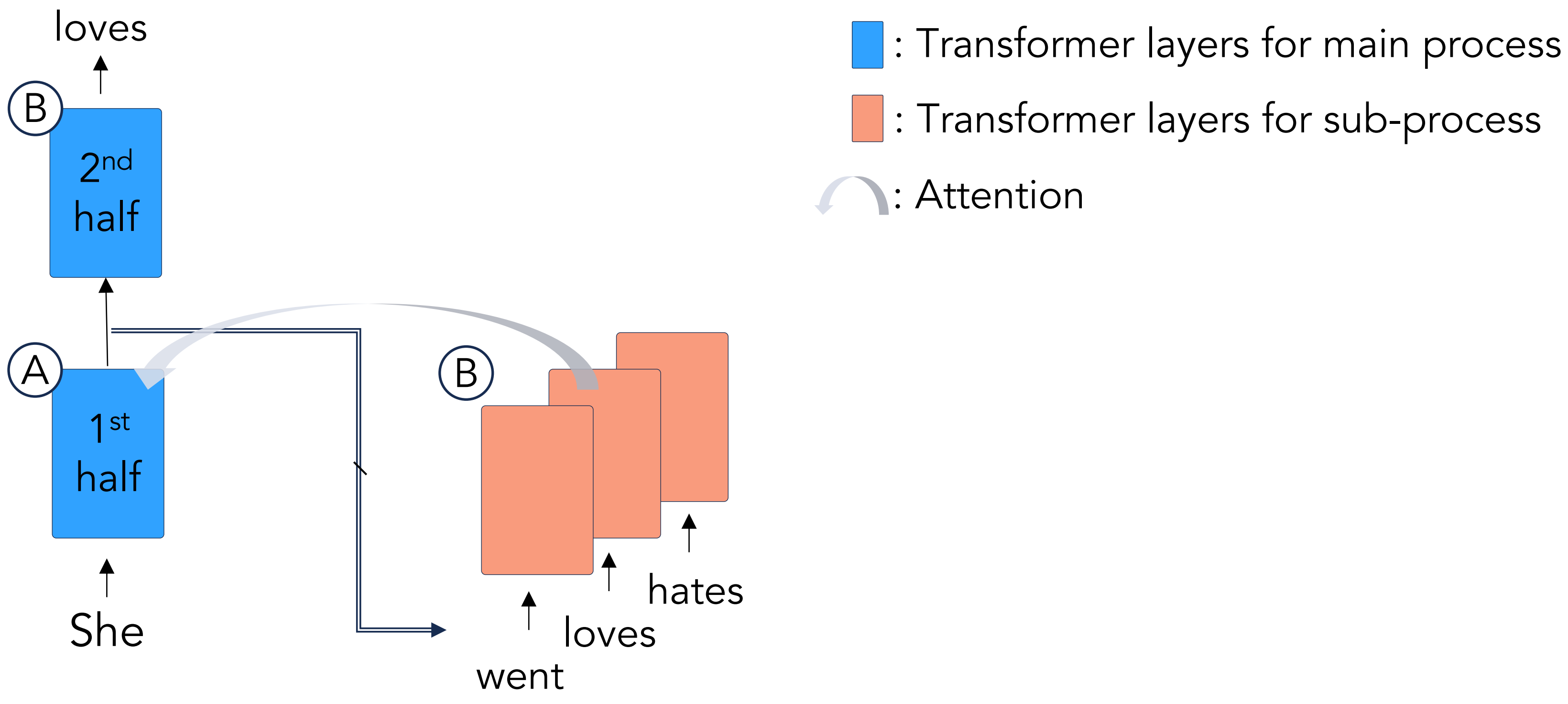
Predict future top-k tokens based on specific transformer layer outputs



Predictive Pipelined Decoding

Predict future top-k tokens based on specific transformer layer outputs

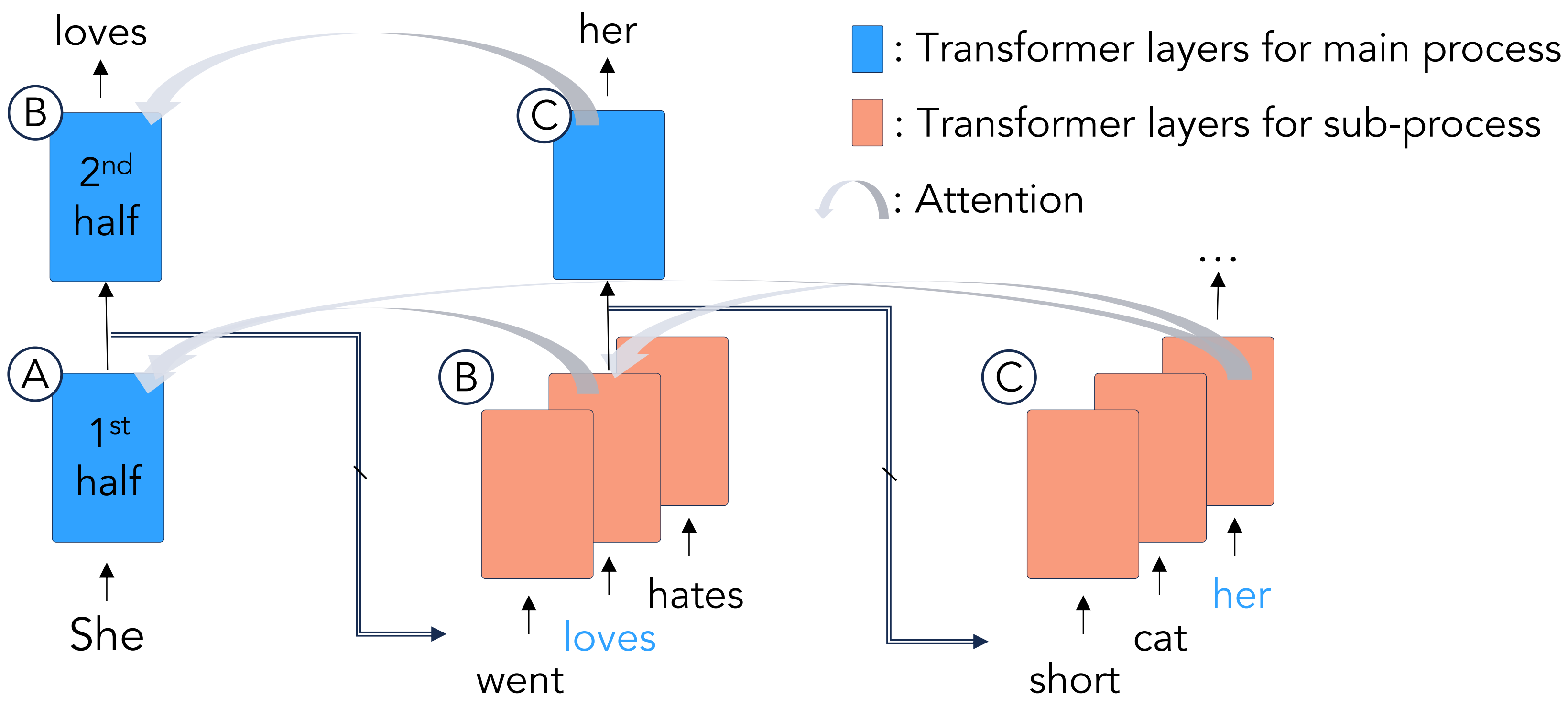
- with additional compute resources
- by parallelizing token decoding process



Predictive Pipelined Decoding

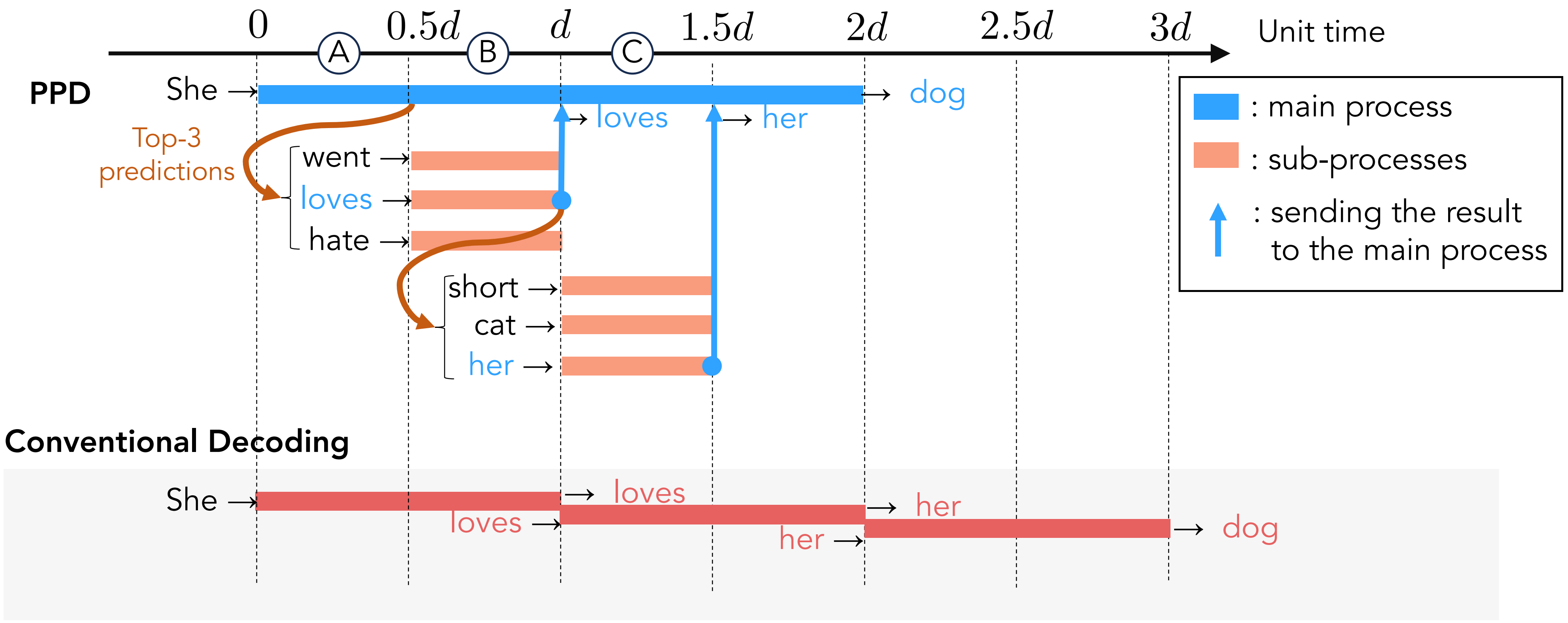
Predict future top-k tokens based on specific transformer layer outputs

- with additional compute resources
- by parallelizing token decoding process



Predictive Pipelined Decoding

- Initiate three sub-processes, predict next tokens at intermediate layer $d/2$
- latency reduction (:= $1d$) compared to conventional decoding



Theoretical Analysis

Assumption: i.i.d. matching events w/ the probability that the early prediction matches the final ouptut, denoted by p_{correct}

When PPD makes an early prediction at the $\bar{d}$-th layer out of d for generating ℓ tokens :

Total latency: $d\ell \underbrace{- (d - \bar{d})(\ell - 1)p_{\text{correct}}}_{\text{saving}}$

Total compute units: $d\ell - (d - \bar{d})(\ell - 1)p_{\text{correct}} + k(d - \bar{d})\ell$

Simulations

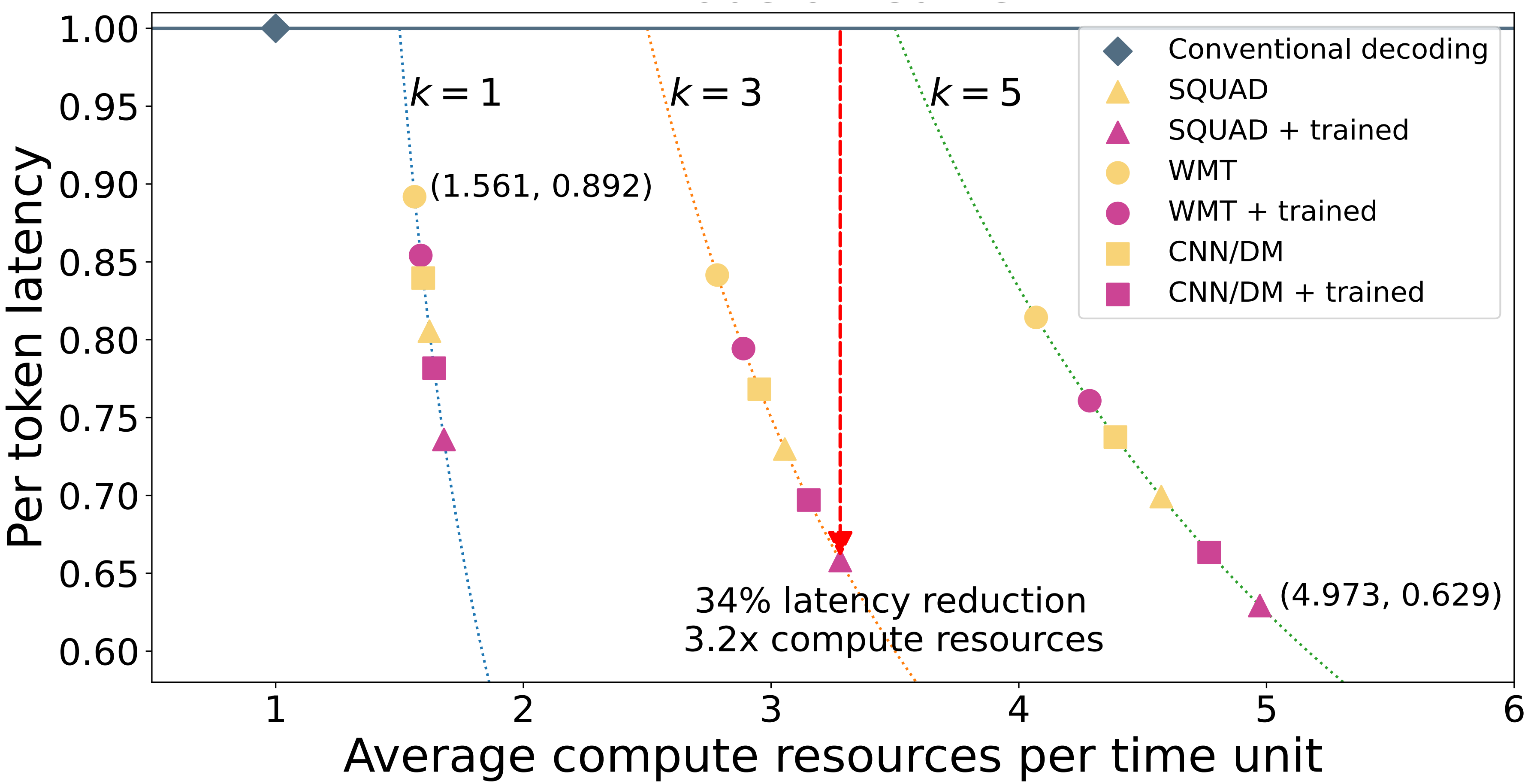
The result of match rate $\hat{p}_{\text{correct}}$

			Layers				
dataset	k	trained	10	20	30	35	37
SQUAD	1	N	5.88%	38.90%	62.90%	79.77%	88.01%
		Y	15.45%	52.81%	72.34%	87.68%	91.67%
	3	N	9.25%	54.04%	77.92%	92.64%	97.67%
		Y	23.48%	68.37%	87.49%	97.33%	98.91%
	5	N	11.04%	60.15%	83.84%	95.85%	99.08%
		Y	27.90%	74.15%	92.29%	98.81%	99.62%
WMT	1	N	2.40%	21.63%	39.69%	68.64%	78.15%
		Y	11.06%	29.17%	48.20%	74.84%	82.69%
	3	N	4.38%	31.69%	61.71%	85.03%	93.53%
		Y	14.83%	41.14%	68.50%	89.84%	95.48%
	5	N	5.57%	37.13%	68.84%	89.54%	96.41%
		Y	16.82%	47.84%	75.46%	93.36%	97.67%
CNN/DM	1	N	7.23%	32.08%	53.07%	68.90%	78.82%
		Y	19.02%	43.65%	61.45%	78.46%	84.42%
	3	N	12.84%	46.36%	68.14%	85.07%	93.81%
		Y	27.57%	60.60%	78.55%	93.07%	96.62%
	5	N	15.21%	52.51%	74.22%	90.04%	96.88%
		Y	31.33%	67.33%	84.83%	96.06%	98.40%

Model: Vicuna-13B

Simulations

Trade-off curve of compute resources per token and latency



Implementation

PPD can operate faster compared to the original greedy decoding.

		CNN/DM			SQUAD 1.1		
Method	k	$\hat{p}_{\text{correct}}$	Latency $\downarrow$	Throughput $\uparrow$	$\hat{p}_{\text{correct}}$	Latency $\downarrow$	Throughput $\uparrow$
greedy		-	18.171	7.044	-	14.994	8.537
greedy (w/ *PPD*)	1	25.72 %	17.019	7.521	30.97 %	13.711	9.336
greedy (w/ *PPD*)	3	41.99 %	16.685	7.671	47.24 %	13.712	9.335

Model: LLaMA2-13B

Summary

- We propose PPD aimed at reducing decoding latency, without compromising the original decoding outcomes
- We identify the efficacy of PPD by theoretical analysis and implementation.
- However, we acknowledge increased computational requirements despite the potential for latency improvements.

Reference

[1] Angela Fan et al. Reducing transformer depth on demand with structured dropout. In International Conference on Learning Representations, 2019.
[2] Trevor Gale et al. The state of sparsity in deep neural networks. arXiv preprint arXiv:1902.09574, 2019.
[3] Paul Michel et al. Are sixteen heads really better than one? Advances in neural information processing systems, 32, 2019.
[4] Elena Voita et al. Analyzing multi-head selfattention: Specialized heads do the heavy lifting, the rest can be pruned. arXiv preprint arXiv:1905.09418, 2019.
[5] Victor Sanh et al. Movement pruning: Adaptive sparsity by fine-tuning. Advances in Neural Information Processing Systems, 33:20378–20389, 2020.
[6] Eldar Kurtic et al. The optimal bert surgeon: Scalable and accurate second-order pruning for large language models. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pp. 4163–4181, 2022.
[7] Woosuk Kwon et al. A fast post-training pruning framework for transformers. Advances in Neural Information Processing Systems, 35:24101–24116, 2022.
[8] Daniel Campos et al. To asymmetry and beyond: Structured pruning of sequence to sequence models for improved inference efficiency. Proceedings of The Fourth Workshop on Simple and Efficient Natural Language Processing (SustaiNLP), pp. 91–109, Toronto, Canada (Hybrid), July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.sustainlp-1.6. URL https://aclanthology.org/2023.sustainlp-1.6.

Reference

[9] Xiaoqi Jiao et al. TinyBERT: Distilling BERT for natural language understanding. In Trevor Cohn, Yulan He, and Yang Liu (eds.), Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 4163–4174, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.372. URL https://aclanthology.org/2020.findings-emnlp.372.
[10] Victor SANH et al. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.
[11] Ofir Zafrir et al. Q8bert: Quantized 8bit bert. In 2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2- NIPS), pp. 36–39. IEEE, 2019.
[12] Sheng Shen et al. Q-bert: Hessian based ultra low precision quantization of bert. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pp. 8815–8821, 2020.
[13] Ali Hadi Zadeh et al. Gobo: Quantizing attention-based nlp models for low latency and energy efficient inference. In 2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), pp. 811–824. IEEE, 2020.
[14] Sehoon Kim et al. I-bert: Integer-only bert quantization. In International conference on machine learning, pp. 5506–5518. PMLR, 2021.
[15] Tim Dettmers et al. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. Advances in Neural Information Processing Systems, 35:30318–30332, 2022.

Reference

[16] Xiaoxia Wu et al. Xtc: Extreme compression for pre-trained transformers made simple and efficient. Advances in Neural Information Processing Systems, 35:3217–3231, 2022.
[17] Zhewei Yao et al. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. Advances in Neural Information Processing Systems, 35:27168–27183, 2022.
[18] Elias Frantar et al. OPTQ: Accurate quantization for generative pre-trained transformers. In The Eleventh International Conference on Learning Representations, 2023. URL https://openreview.net/forum?id=tcbBPnfwxS.
[19] Tal Schuster et al. Confident adaptive language modeling. Advances in Neural Information Processing Systems, 35: 17456–17472, 2022.
[20] Shengkun Tang et al. You need multiple exiting: Dynamic early exiting for accelerating unified vision language model. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10781–10791, 2023.

Thank you for listening!

TRANSACTIONS tmlr on ML RESEARCH

KRAFTON

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON