Compressed Context Memory For Online Language Model Interaction

Jang-Hyun Kim[1], Junyoung Yeom[1], Sangdoo Yun[2†], Hyun Oh Song[1†]

[1]Seoul National University [2]NAVER AI Lab [†]equal supervision

ICLR 2024

Why and what

- Transformers require linearly **increasing memory and FLOPS** for attention keys/values. (1GB for 1k tokens even for 7B LLaMA!)

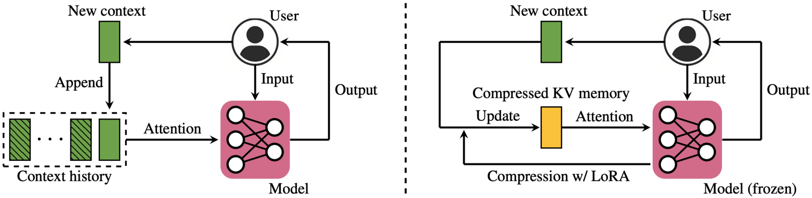


Figure: **Left**: Conventional online inference approach. **Right**: The proposed system with compressed context memory. The colored boxes represent attention keys/values (or input tokens).

Advantages of the proposed method

- Memory efficient inference by recurrent key/value (KV) compression
- Increases throughput of LM
- Only requires lightweight LoRA for compression
- Fully parallelized training strategy

	A100 PCIe 80GB		
	Full context	CCM-concat	CCM-merge
Throughput (sample/sec)	5.3	24.4	**69.9**
Maximum batch size	60	300	**950**
Context KV length	800	128	**8**
Performance (Accuracy %)	**70.8**	70.0	69.6

Figure: Inference throughput on the MetaICL dataset with LLaMA-7B, FP16.

Method: Inference (compression)

Given a context $c(t)$, we obtain the compressed key/value $h(t)$ as

$$h(t) = g_{\text{comp}}(\text{Mem}(t-1), c(t)),$$

where g_{comp} is a language model's forward pass with conditional adapter.

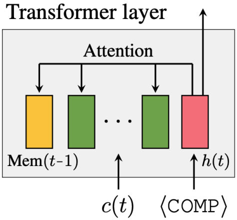


Figure: The compression process at time step t. Each colored box symbolizes attention hidden states.

Method: Inference (memory update)

The compressed context memory $\text{Mem}(t)$ is then updated via an update function g_{update} as

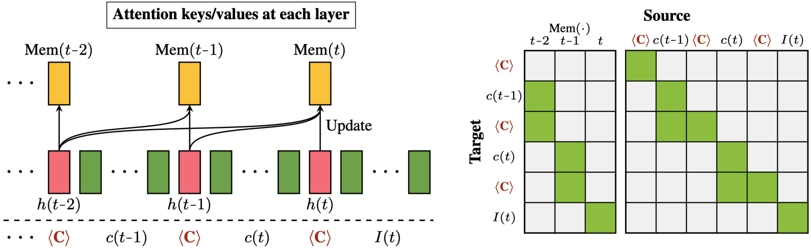
$$\text{Mem}(t) = g_{\text{update}}(\text{Mem}(t{-}1), h(t)).$$

- **CCM-concat**: For a scalable memory, we employ the *concatenation* function as g_{update}.
- **CCM-merge**: For a fixed-size memory like an RNN, we propose a *merging* function:

$$\text{Mem}(t) = (1 - a_t)\text{Mem}(t{-}1) + a_t h(t),$$

where $a_1 = 1$ and $a_t \in [0, 1]$ for $t \geqslant 2$.

Method: Parallelized training



(a) Parallel memory update

(b) Attention mask

Figure: In (a), each colored box symbolizes attention keys/values of memory, compression tokens, and normal text tokens. In (b), gray indicates that attention is blocked. ⟨C⟩ stands for ⟨COMP⟩. At each layer, after the parallel updates of compressed context memory, the attention operation occurs with the mask in (b).

Method: Conditional LoRA

For a parameterized language model f_θ, naive finetuning might lead to overfitting on inputs $I(t)$:

$$\min_{\theta} \mathbb{E}_t[-\log f_{\theta}(O(t) \mid \mathsf{Mem}(t;\theta), I(t))].$$

- Our proposal:

$$\min_{\Delta\theta} \; \mathbb{E}_t \left[-\log f_\theta(O(t) \mid \mathsf{Mem}(t;\theta + \Delta\theta), I(t))\right].$$

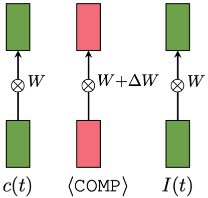


Figure: Forward operations of conditional LoRA.

Qualitative result sample

Context:
A: What's the problem, Nada? You look down in the dumps. ⟨COMP⟩
B: I don't know. My life is a big mess. Everything is so complicated. ⟨COMP⟩
A: Come on, nothing can be that bad. ⟨COMP⟩
B: But promise me, you'll keep it a secret. ⟨COMP⟩
A: Ok, I promise. So what's troubling you so much? ⟨COMP⟩
B: I've fallen in love with my boss. ⟨COMP⟩
A: Really? Is he married? ⟨COMP⟩
⇒ Total **103** tokens. Context compression ratios are **7/103** (CCM-concat) and **1/103** (CCM-merge).

Input: No, of course not. He is still single.
Output generated w/o context: I'm sorry, I'm not sure what you mean.
Output generated by CCM-concat: So what's the problem?
Output generated by CCM-merge: What's his problem?
Ground truth output: Then what's your problem?

Figure: An example result using our method with LLaMA-7B on a DailyDialog test sample.

Experiment: Compression performance

- Check out the conversation and personalization benchmark results in our paper.

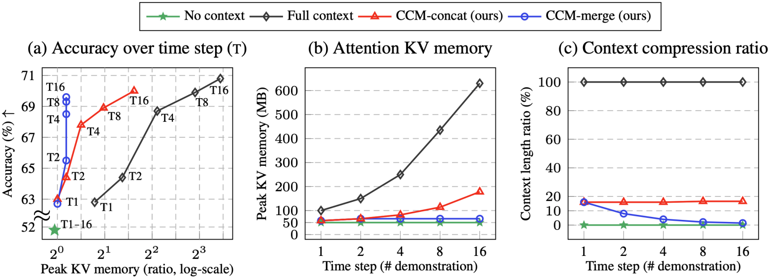


Figure: Comparison to full context approach on MetaICL test tasks with LLaMA-7B. *Peak KV memory* refers to the peak memory space occupied by attention keys/values at each time step.

Experiment: Comparison to compression baselines

- Compression factor = 8.

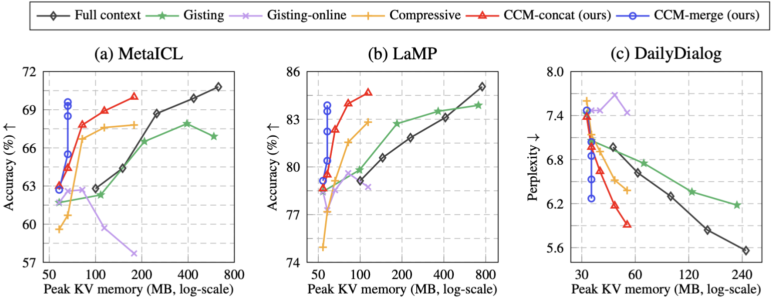


Figure: Test performance in online inference scenario with LLaMA-7B. All compression methods have the **identical compression factor around 8**, except for CCM-merge, which has a higher compression factor.

Experiment: Streaming setting

- Streaming with sliding window.

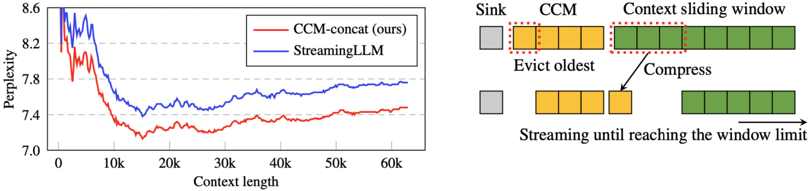


Figure: Streaming evaluation on PG19 validation set using sliding window with LLaMA-7B.

Experiment: Training data source analysis

Training dataset	# training data	Evaluation dataset Pretrain	SODA	DailyDialog	MetaICL
Pretrain (= RedPajama + LmSys-Chat)	500k	**-0.55**	-0.22	-0.74	-4.9%
Pretrain + MetaICL	500k	-0.59	-0.26	-0.82	-1.2%
Pretrain + MetaICL + SODA	500k	-0.61	-0.10	-0.54	-1.3%
Pretrain + MetaICL + SODA	750k	-0.57	**-0.09**	**-0.53**	**-1.1%**

Figure: Compression performance gap across different data sources used to train compression adapter.

Summary

- Our approach dynamically creates **compressed KV memory** during LLM interactions.
- Our approach only requires training a **conditional LoRA** for compression.
- We use a **fully parallelized training** strategy for recurrent compression procedures.
- We conduct evaluations on diverse applications: conversation, multi-task ICL, and personalization, achieving the performance level of a full context model with $5\times$ smaller context memory space.