CSKV: Training-Efficient Channel Shrinking for KV Cache in Long-Context Scenarios

Luning Wang[1,2], **Shiyao Li**[1,2], **Xuefei Ning**[1], **Zhihang Yuan**[2],
Shengen Yan[2], **Guohao Dai**[3,2], **Yu Wang**[1]

[1]Tsinghua University
[2]Infinigence-AI
[3]Shanghai Jiao Tong University

Abstract

Large Language Models (LLMs) have been widely adopted to process long-context tasks. However, the large memory overhead of the key-value (KV) cache poses significant challenges in long-context scenarios. Existing training-free KV cache compression methods typically focus on quantization and token pruning, which have compression limits, and excessive sparsity can lead to severe performance degradation. Other methods design new architectures with le ss KV overhead but require significant training overhead. To address the above two drawbacks, we further explore the redundancy in the channel dimension and apply an architecture-level design with minor training costs. Therefore, we introduce **CSKV**, a training-efficient **C**hannel **S**hrinking technique for **KV** cache compression: (1) We first analyze the singular value distribution of the KV cache, revealing significant redundancy and compression potential along the channel dimension. Based on this observation, we propose using low-rank decomposition for key and value layers and storing the low-dimension features. (2) To preserve model performance, we introduce a bi-branch KV cache, including a window-based full-precision KV cache and a low-precision compressed KV cache. (3) To reduce the training costs, we minimize the layer-wise reconstruction loss for the compressed KV cache instead of retraining the entire LLMs. Extensive experiments show that CSKV can reduce the memory overhead of the KV cache by 80% while maintaining the model's long-context capability. Moreover, we show that our method can be seamlessly combined with quantization to further reduce the memory overhead, achieving a compression ratio of up to 95%. Code is available at `https://github.com/wln20/CSKV`.

1 Introduction

Large Language Models (LLMs) have been widely adopted in various natural language processing tasks, particularly those requiring long-context capabilities, such as document analysis and fact retrieval [6]. However, the key-value (KV) cache mechanism used in transformer-based LLMs poses significant efficiency challenges as its memory overhead grows linearly with the sequence length, often replacing the weights to be the memory bottleneck in long-context scenarios. For instance, processing a sequence with 200K tokens using LLaMA-2-7B [17] results in a KV cache occupying around 100GB, compared to 14GB required for model weights. Compressing the KV cache by over 10× is necessary to fit such a sequence on a single NVIDIA RTX 4090 GPU with 24GB of memory.

Existing KV cache compression methods, mainly training-free techniques like token pruning [22, 12, 18, 8] and quantization [10, 13, 11, 15], struggle to maintain model performance at high compression ratios, particularly in long-context tasks. Alternatively, training-required techniques, such as MLA [3]

4th NeurIPS Efficient Natural Language and Speech Processing Workshop (ENLSP-IV 2024).

and cache sharing [16, 2], offer higher compression ratios but at the cost of significant retraining and are typically unable to be integrated with existing pre-trained models.

Inspired by MLA, we observe significant redundancy in the large channel dimensions of the KV cache, evidenced by the long-tailed distribution of singular values in the key and value caches (Details in Appendix). Experiments reveal that removing the smallest 50% of these singular values results in less than 1% average accuracy loss on the MMLU [5] benchmark (from 0.458 to 0.449).

Given this redundancy, we propose **CSKV**, a training-efficient **C**hannel **S**hrinking technique for the **KV** cache, designed to balances high compression ratios with low training costs. To sum up, we have the following contributions:

- To reduce the memory overhead of the KV cache while maintaining the performance, we design a **bi-branch KV cache** by preserving the recently used KV cache with original dimensions and reducing the dimension of the historical KV cache.
- To further improve the performance without significant training overhead, we propose an effective **SVD-based initialization** technique and train LLMs in a layer-wise manner by minimizing the reconstruction loss.
- Extensive experimental results demonstrate that our method can achieve an 80% KV cache compression ratio while maintaining the model's long-context capability. We further demonstrate that our method can be seamlessly combined with 4-bit quantization, showcasing its power in achieving a total compression ratio of 95%.

2 Method

2.1 Inference with Bi-Branch KV Cache

To reduce the memory overhead, we design to reduce the memory overhead of the KV cache by using low-rank decomposition for both the Key and Value weight matrix. Without loss of generality, we will detail the workflow of compressing the key cache, as the process is identical to that of the value cache.

As shown in Figure 1, we use two matrices, $A_K \in R^{h_{in} \times h_{comp}}$ and $B_K \in R^{h_{comp} \times h_{out}}$, to approximate the weight matrix of $W_K \in R^{h_{in} \times h_{out}}$. Here the $h_{in}, h_{out}, h_{comp}$ are the input dimension of W_K, the output dimension of W_K, and the intermediate dimension of the low-rank decompression. Keeping the h_{comp} smaller than h_{out} and **storing the intermediate features as the compressed Key cache**, we can significantly save the memory overhead, especially in the long context scenario.

To maintain the high performance, we propose to follow the prior research by preserving the recently used tokens [1, 18] because they are crucial for accurate next-token prediction. To prevent the degradation of this local information during inference, we propose **the bi-branch KV cache** that preserves the recently used tokens effectively during both the prefilling and decoding stages. With a pre-defined window size l_w, we compress the KV cache only after the tokens fill a complete window while retaining the residual tokens in their original hidden dimensions.

Specifically, for the prefilling stage, as shown in Figure 1(a), given an input sequence with n tokens, we first use the A_K to generate the compressed Key matrix and store it in the Compressed Key Cache K_C. In this case, the Compressed Key Cache contains all of the historical information of the given sentence. On the other branch, we use the original W_K to generate the full-precision Key matrix K for computation, which can guarantee that the computation results of the prefilling stage are the same as the original LLMs. Then, we only store the full-precision Key activation of the last m tokens K_{local} to preserve the local information for the decoding stage.

Moreover, during the decoding stage, as shown in Figure 1(b), we only process one token during each forward pass. We take the process of the $(n+1)$-th token as an example. For the cache update, we compute both the compressed Key activation K_C and full-precision Key activation K and update both Key caches with the new activations. In this case, the compressed Key cache has $(n+1)$ tokens, and the full-precision Key cache has $(m+1)$ tokens. To get the $(n+1)$ tokens' Key matrix, we use the $(m+1)$ tokens from the full-precision Key cache as K_{local} and use the B_K to process the oldest $(n-m)$ tokens in the compressed Key cache as $\hat{K}$. By concatenating the $\hat{K}$ and K_{local}, we

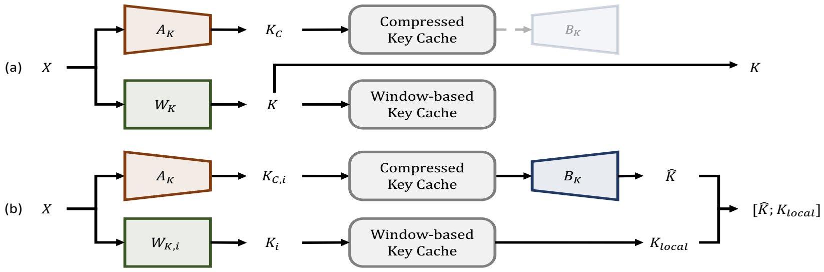

Figure 1: The overview of the inference process. (a) The prefilling stage. (b) The decoding stage.

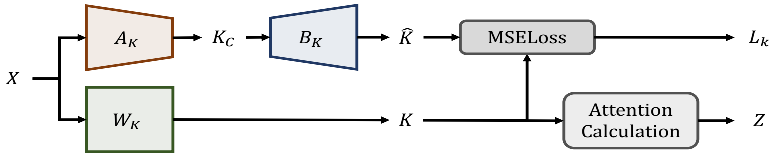

Figure 2: The overview of the efficient layer-wise reconstruction fine-tuning.

can get the target Key matrix for attention computation. Finally, we remove the oldest token from the full-precision Key cache to keep the window size as m.

2.2 Efficient Fine-tuning by SVD-based Initialization

Directly applying low-rank decomposed weight matrices for KV cache compression would result in the degradation of model performance when the compression ratio becomes high. To further enhance the model performance, we propose to introduce an efficient training process. We find that the initialization method to the proposed A_K and B_K is of great importance for convergence and final performance. In this case, we proposed to use the ASVD-based decomposition results for initialization. As shown in Figure 2, we train LLMs in a layer-wise manner by minimizing the layer-wise reconstruction loss for the compressed keys and values.

Specifically, for each layer, we can use the W_K to generate the full-precision Key matrix $K = XW_K$ and use A_K, B_K to generate the lossy key matrix $\hat{K} = XA_KB_K$. The local reconstruction loss of this layer could be defined as Equation 1:

$$L_K = \text{MSELoss}(K, \hat{K}) \tag{1}$$

where L_K denotes the loss of keys in this layer, and $\text{MSELoss}(\cdot, \cdot)$ is the Mean Square Error (MSE) loss function. Finally, define the loss of keys and values in the i-th layer as $L_{K,i}, L_{V,i}$, the loss for the whole model is shown in Equantion 2:

$$\mathcal{L}_{all} = \sum_{j=0}^{n_l} (L_{K,j} + L_{V,j}) \tag{2}$$

where $\mathcal{L}_{all}$ denotes the loss for the whole model, and n_l denotes the number of layers.

3 Experiment

3.1 Experimental Setup

We evaluate our method on LongChat-7B-v1.5-32k [9] and Mistral-7B-Instruct-v0.2 [7]. We evaluate our method on three widely-used long-context benchmarks: LongEval [9], LongBench [21], and LVEval [19]. For comparison, we include results from StreamingLLM [18], H_2O [22] [1], and ASVD [20]. The first two are token pruning methods, while the latter is a SOTA channel-shrinking method. More details can be found in the Appendix.

[1] Here we only compare the effect of H_2O on Longchat-7b-v1.5-32k, as it only supports LLaMA architecture in its official implementation.

Table 1: Performance of models with CSKV on long-context benchmarks.

Model	C. Ratio	Method	LongEval ↑				LongBench ↑			LV-Eval ↑
			4k	6k	8k	10k	0-4k	4-8k	8k+	16k
Longchat-7b-v1.5-32k	0%	-	1.00	1.00	0.98	0.98	0.46	0.43	0.46	0.13
	50%	StreamingLLM	0.12	0.16	0.06	0.20	0.37	0.39	0.40	0.09
		H_2O	0.62	0.56	0.52	0.50	0.40	0.38	0.38	0.09
		ASVD	0.92	**0.96**	0.92	**0.94**	0.44	0.41	0.43	0.11
		CSKV (Ours)	**0.98**	0.94	**0.96**	**0.94**	**0.46**	**0.42**	**0.45**	**0.12**
	80%	StreamingLLM	0.06	0.06	0.02	0.02	0.31	0.35	0.39	0.06
		H_2O	0.18	0.24	0.26	0.10	0.34	0.30	0.32	0.05
		ASVD	0.26	0.12	0.06	0.04	0.36	0.31	0.32	0.04
		CSKV (Ours)	**0.92**	**0.94**	**0.94**	**0.90**	**0.43**	**0.40**	**0.41**	**0.10**
Mistral-7b-instruct-v0.2	0%	-	1.00	1.00	0.98	0.94	0.50	0.47	0.45	0.20
	50%	StreamingLLM	0.06	0.12	0.04	0.14	0.39	0.38	0.37	0.12
		ASVD	**1.00**	0.98	0.92	**0.94**	0.49	0.45	0.44	0.17
		CSKV (Ours)	**1.00**	**1.00**	**0.96**	**0.94**	**0.50**	**0.47**	**0.47**	**0.20**
	80%	StreamingLLM	0.06	0.04	0.00	0.04	0.34	0.34	0.33	0.06
		ASVD	0.04	0.00	0.04	0.00	0.33	0.29	0.29	0.05
		CSKV (Ours)	**0.98**	**0.96**	**0.90**	**0.92**	**0.45**	**0.42**	**0.41**	**0.17**

3.2 Main Results

We apply compression ratios of 50% and 80% consistently for both keys and values. The results are presented in Table 1.

According to the evaluation results in Table 1, the token pruning methods are especially not skilled in retrieval tasks like LongEval, even at a 50% compression ratio, when ASVD and CSKV only incur minor performance loss. As the compression ratio reaches 80%, all methods except for CSKV suffer great performance degradation on all three tasks. To dive deeper, we examine the failure cases of token pruning methods, and found that although the model could generate coherent sentences based on instructions, a great deal of the retrieved answers deviate from the ground truth by a small portion, like answering "4244" when the label is "42440", or give an irrelevant answer such as "1386". This might be caused by their token eviction mechanisms which inherently have to discard the information of some tokens completely, facing great risk of losing the ground truth information. In contrast, the abundant failure cases of ASVD at 80% compression are mainly caused by the loss of the model's language modeling capabilities, like responding with dozens of tokens that could hardly form a sentence. Different from the aforementioned methods, CSKV consistently enables the model to generate instruction-following responses and give accurate answers on either retrieval tasks or QA tasks, showing its superior capability of keeping the model's long-context abilities even at high compression ratios.

3.3 Ablation Studies

We conduct several ablation studies to further explore the potential of our method, and the main conclusions include: 1) The SVD-based initialization methods is crucial to the success of training; 2) The model performance is positively correlated with the window size, while the benefit would become less significant after it reaches a certain level; 3) In most cases, it would be better to compress the key cache more than the value cache given a certain budget; 4) CSKV could be seamlessly integrated with 4-bit QAT with very small performance loss. See Appendix for details.

4 Limitation and Future Directions

While demonstrating competitive performance, the proposed method's compression ratio assignment is user-defined and might not be optimal, offering the potential to achieve higher compression ratios. Future work could explore the application of automated search algorithms to dynamically assign compression ratios to individual layers, accounting for their varying sensitivity to compression. Similarly, automated strategies could optimize memory budget allocation for keys and values, maximizing performance within a given constraint. We leave those directions for future works to explore.

References

[1] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020.

[2] William Brandon, Mayank Mishra, Aniruddha Nrusimha, Rameswar Panda, and Jonathan Ragan Kelly. Reducing transformer key-value cache size with cross-layer attention, 2024.

[3] DeepSeek-AI et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024.

[4] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

[5] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[6] Yunpeng Huang, Jingwei Xu, Junyu Lai, Zixu Jiang, Taolue Chen, Zenan Li, Yuan Yao, Xiaoxing Ma, Lijuan Yang, Hao Chen, Shupeng Li, and Penghao Zhao. Advancing transformer architecture in long-context large language models: A comprehensive survey, 2024.

[7] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[8] Sehoon Kim, Sheng Shen, David Thorsley, Amir Gholami, Woosuk Kwon, Joseph Hassoun, and Kurt Keutzer. Learned token pruning for transformers. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 784–794, 2022.

[9] Dacheng Li*, Rulin Shao*, Anze Xie, Ying Sheng, Lianmin Zheng, Joseph E. Gonzalez, Ion Stoica, Xuezhe Ma, and Hao Zhang. How long can open-source llms truly promise on context length?, June 2023.

[10] Shiyao Li, Xuefei Ning, Luning Wang, Tengxuan Liu, Xiangsheng Shi, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. Evaluating quantized large language models, 2024.

[11] Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. Qserve: W4a8kv4 quantization and system co-design for efficient llm serving. *arXiv preprint arXiv:2405.04532*, 2024.

[12] Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. Scissorhands: Exploiting the persistence of importance hypothesis for llm kv cache compression at test time, 2023.

[13] Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*, 2024.

[14] Aleksandra Piktus. https://huggingface.co/datasets/ola13/small-the_pile, 2022.

[15] Ying Sheng, Lianmin Zheng, Binhang Yuan, Zhuohan Li, Max Ryabinin, Beidi Chen, Percy Liang, Christopher Ré, Ion Stoica, and Ce Zhang. Flexgen: High-throughput generative inference of large language models with a single gpu. In *International Conference on Machine Learning*, pages 31094–31116. PMLR, 2023.

[16] Yutao Sun, Li Dong, Yi Zhu, Shaohan Huang, Wenhui Wang, Shuming Ma, Quanlu Zhang, Jianyong Wang, and Furu Wei. You only cache once: Decoder-decoder architectures for language models, 2024.

[17] Hugo Touvron et al. Llama 2: Open foundation and fine-tuned chat models, 2023.

[18] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks, 2024.

[19] Tao Yuan, Xuefei Ning, Dong Zhou, Zhijie Yang, Shiyao Li, Minghui Zhuang, Zheyue Tan, Zhuyu Yao, Dahua Lin, Boxun Li, Guohao Dai, Shengen Yan, and Yu Wang. Lv-eval: A balanced long-context benchmark with 5 length levels up to 256k, 2024.

[20] Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. Asvd: Activation-aware singular value decomposition for compressing large language models, 2024.

[21] Bai Yushi, Lv Xin, Zhang Jiajie, Lyu Hongchang, Tang Jiankai, Huang Zhidian, Du Zhengxiao, Liu Xiao, Zeng Aohan, Hou Lei, Dong Yuxiao, Tang Jie, and Li Juanzi. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*, 2023.

[22] Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2o: Heavy-hitter oracle for efficient generative inference of large language models, 2023.

Appendix

A. Distribution of Singular Values of key cache

We visualize the distribution of singular values of key cache in the 14-th layer of LLaMA-2-7B-chat model, using data randomly sampled from the Pile [4] dataset. We find that the singular value of the key cache has a significant long-tailed distribution, and a similar phenomenon also appears in the value cache. In this case, only a tiny fraction of singular values have large magnitudes, while the vast majority are around zero, which can be removed without significant degradation of model performance.

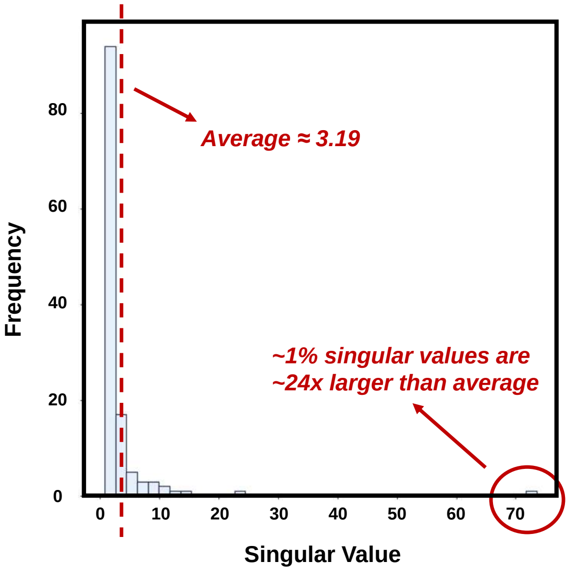

Figure 3: Distribution of of Singular Values of key cache.

B. Details of Experimental Setup

We evaluate our method on widely used long-context models, including LongChat-7B-v1.5-32k [9] and Mistral-7B-Instruct-v0.2 [7]. For fine-tuning, we use a scaled-down version of the Pile [4] dataset [14] and is conducted with both the epoch and batch size set to 1, using the AdamW optimizer with an initial learning rate of 5e-5. The entire fine-tuning process for each 7B model is completed within 90 minutes on a single NVIDIA A100-80G GPU, resulting in minimal training costs. We initialize the model with ASVD [20], selecting 256 samples from the fine-tuning dataset as calibration data. We set $\alpha = 0.5$ and use the Absolute Mean Value method for configuring the scaling matrix S.

The evaluation of our method is performed on three widely-used long-context benchmarks, including LongEval [9], LongBench [21] and LVEval [19]. Specifically, we choose the 200,300,400,500 lines subsets in LongEval (with an average length of 4k,6k,8k,10k), the qasper, hotpotqa, multifieldqa_en, gov_report, triviaqa subset of LongBench-E, along with the 16K subset of LVEval. To compare the results with other methods, we choose StreamingLLM[18], H_2O[22] and ASVD[20], in which the first two are token pruning methods and the last one could be regarded as a channel shrinking method[2]. We select compression ratios of 50% and 80% for the experiments, with the same compression ratios for keys and values.

[2] While the standard ASVD perform low-rank decomposition on all weights, here we merely decompose the W_K, W_V in each layer.

C. Ablation Study

Without loss of generality, we perform an ablation study on LongEval with the Longchat-7b-v1.5-32k model. The window size is set to 32 and the compression ratio is evenly distributed on keys and values by default. The "Avg.Acc" column in the following tables indicates the average accuracy on the four chosen subsets of LongEval.

C.1 Effect of Initialization Methods

We test three initialization methods for the low-rank decomposed matrices: 1) random initialization, 2) standard SVD initialization, and 3) ASVD initialization. We keep their fine-tuning settings the same as mentioned in the Experimental Setups. The loss curves of 80% compression are shown in Figure 4, and the evaluation results for the trained models with a bi-branch strategy are shown in Table 2.

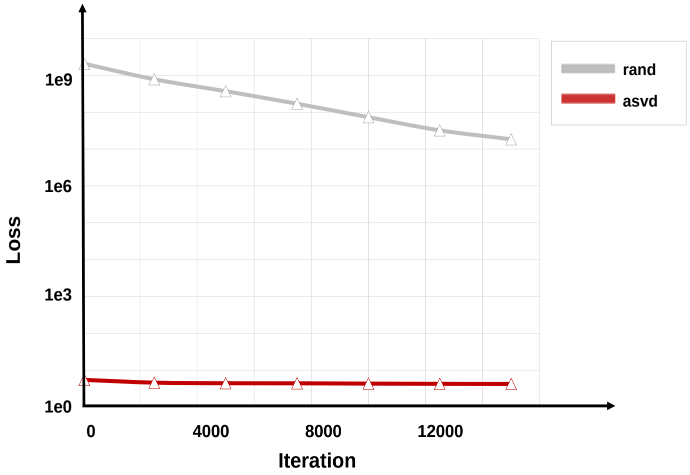

Figure 4: Loss curve with 80% compression ratio. "asvd" means initialize with ASVD, "rand" means random initialization. We drop the curve for standard SVD initialization as it almost overlaps with the ASVD one in the figure.

Table 2: Results of different initialization methods

C. Ratio	Init. Method	Avg. Acc
0%	-	0.99
50%	Random	0.00
	SVD	0.94
	ASVD	**0.95**
60%	Random	0.00
	SVD	0.93
	ASVD	**0.94**
70%	Random	0.00
	SVD	0.89
	ASVD	**0.93**
80%	Random	0.00
	SVD	0.87
	ASVD	**0.92**

It could be found that the loss of the random initialization method remains extremely high (∼1e9) and is very hard to converge in a reasonable time, leading to the deterioration of model performance.

This is quite intuitive as the information stored in the initial W_K, W_V are completely destroyed and their information cannot be utilized. In contrast, the SVD-based initialization methods' loss could converge quickly from approximately 5.5 to 4.0, leading to superior model performance. **Therefore, the SVD-based initialization methods is crucial to the success of training**. Specifically, the ASVD-initialized model performs slightly better than the SVD-initialized one after training, so we choose ASVD as the default initialization method.

C.2 Effect of Window Size

The window size determines how much local information could be preserved, which is of vital importance to the quality of generated content. We fix the compression ratio to 80% and evaluate the performance of the bi-branch trained model with multiple window size settings. The results are shown in Table 3.

Table 3: Results of different window sizes.

C. Ratio	Window Size	Avg. Acc
0%	-	0.99
80%	2	0.77
	4	0.83
	8	0.85
	16	0.88
	32	**0.92**
	64	0.93
	128	0.94
	256	0.94
	512	0.94
	1024	0.95
	2048	0.96
	4096	0.96

The accuracy of the model shows a positive correlation with the window size, which is quite intuitive. Specifically, as the window size increases from 2 to 32, the accuracy improves relatively rapidly. However, when the window size exceeds 32, the rate of accuracy improvement notably decreases. This might indicate that a window size around 32 would be enough for local information preservation, while greater window sizes could not bring obvious improvement. **Therefore, we may conclude that the model performance is positively correlated with the window size, while the benefit would become less significant after it reaches a certain level**. Considering that an excessively large window size incurs non-negligible memory overhead, practitioners should carefully balance the trade-off between memory budget and accuracy when selecting the optimal window size for real-world applications.

C.3 Effect of Compression Ratio Allocation for KV

Different from the token pruning methods that have to keep or discard a certain token's keys and values simultaneously, our channel shrinking method allows for the key cache and value cache to have different compression ratios. To investigate the impact of allocating a certain compression ratio to the key cache and value cache in different proportions, we conduct experiments by fixing the total compression rate at 50% and 75%, respectively. We then evaluate the model's performance under various combinations of compression ratios for keys and values. The results are shown in Table 4.

It could be found from the evaluation results that among the selected combinations, the optimal configuration consistently occurs when the compression ratio for the key cache exceeds that of the value cache, showing that **it would be better to compress the key cache more than the value cache given a certain budget, in most cases**. This potentially reveals that the sensitivity of keys towards compression is weaker than that of values, making the key cache much easier to compress.

Table 4: Results of different compression ratio assignments

C. Ratio	KV C. Ratio	Avg. Acc
0%	-	0.99
50%	K(87.5%) V(12.5%)	0.97
	K(75.0%) V(25.0%)	**0.98**
	K(62.5%) V(37.5%)	0.96
	K(50.0%) V(50.0%)	0.95
	K(37.5%) V(62.5%)	0.95
	K(25.0%) V(75.0%)	0.94
	K(12.5%) V(87.5%)	0.80
75%	K(43.75%) V(6.25%)	0.73
	K(37.50%) V(12.50%)	0.89
	K(31.25%) V(18.75%)	**0.95**
	K(25.00%) V(25.00%)	0.93
	K(18.75%) V(31.25%)	0.88
	K(12.59%) V(37.50%)	0.80
	K(6.25%) V(43.75%)	0.43

C.4 Compatibility with Quantization

As the low-bit quantization methods are orthogonal with our method, we further demonstrate that quantization could be seamlessly combined with our method. Specifically, we apply KIVI [13] with 4-bit quantization on the compressed keys and values, using per-channel quantization for the former and per-token quantization for the latter. Both the window size and the residual size are set to 32. We separately perform the experiments with two quantization manners: PTQ (Post-Training Quantization) and QAT (Quantization-Aware Training). The results are shown in Table 5, where the "None" rows are the referenced results from the full-precision model.

Table 5: Results of integration with quantization

C. Ratio (origin)	C. Ratio (4-bit)	Q. Mode	Avg. Acc
0%	0%	-	0.99
50%	87.5%	None	0.95
		PTQ	0.00
		QAT	**0.96**
60%	90.0%	None	0.94
		PTQ	0.00
		QAT	**0.94**
70%	92.5%	**None**	**0.93**
		PTQ	0.00
		QAT	0.92
80%	95.0%	**None**	**0.92**
		PTQ	0.00
		QAT	0.90

According to the results in Table 5, directly applying PTQ would completely deteriorate the model's performance, while the QAT results show minor degradation compared with their full-precision counterparts. The failure of PTQ might be a result of the significant density of the compressed representations, which are a lot more intact and difficult to directly quantize. In contrast, the QAT method includes the quantization loss during the optimization process and shows great compatibility with our channel shrinking method, where a total of 95% compression would still keep more than 90% of the model's long-context capability. Therefore, it could be concluded that **it would be better to compress the key cache more than the value cache given a certain budget, in most cases**.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made in the abstract shows the paper's main ideas and contributions.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitation section of the paper points out the limitations of this paper and possible future directions. (See Sec. 4)

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The formulas in the paper are enough for deriving the results. (See Sec. 2)

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The details of experiments are shown in Sec. 3 and the corresponding sections in Appendix, like the choice of optimizer and learning rate.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code is submitted according to the guidelines, and the data used for fine-tuning is open-source and its link is shown in references.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The details of experiments are shown in Sec. 3 and the corresponding sections in Appendix, like the choice of optimizer and learning rate.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The reported accuracies are all from models that use deterministic generation strategy (i.e. greedy search), so the results are deterministic and can be precisely reproduced.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper provide the information of computer resources in Sec. 3.1 and the corresponding sections in Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The method proposed in this paper is purely for academic research and has no obvious social impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The information of the existing assets is properly shown in the reference section.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.