Hold Onto That Thought: Assessing KV Cache Compression On Reasoning

Minghui Liu[†]*, Aadi Palnitkar[†]*, Tahseen Rabbani[‡]*, Kyle Rui Sang[†]*

Hyunwoo Jae[†], Dixi Yao[‡], Shayan Shabihi[†], Fuheng Zhao[♣]

Kunpeng Zhang [†], Tian Li[‡], Ce Zhang [‡], Furong Huang [†]

† University of Maryland, ‡ University of Chicago, ♣ University of Utah

Abstract

Large language models (LLMs) have demonstrated remarkable performance on long-context tasks, but are often bottlenecked by memory constraints. Namely, the KV cache, which is used to significantly speed up attention computations, grows linearly with context length. A suite of compression algorithms has been introduced to alleviate cache growth by evicting unimportant tokens. However, several popular strategies are targeted towards the prefill phase, i.e., processing long prompt context, and their performance is rarely assessed on reasoning tasks requiring long decoding. In particular, short but complex prompts, such as those in benchmarks like GSM8K and MATH500, often benefit from multi-step reasoning and self-reflection, resulting in thinking sequences thousands of tokens long. In this work, we benchmark the performance of several popular compression strategies on long-reasoning tasks. For the non-reasoning Llama-3.1-8B-Instruct, we determine that no singular strategy fits all, and that performance is heavily influenced by dataset type. However, we discover that H2O and our decoding-enabled variant of SnapKV are dominant strategies for reasoning models, indicating the utility of heavy-hitter tracking for reasoning traces. We also find that eviction strategies at low budgets can produce longer reasoning traces, revealing a tradeoff between cache size and inference costs.

1 Introduction

Large language models (LLMs) have demonstrated remarkable performance on complex NLP tasks that require multi-step reasoning. Unlike summarization tasks [Bai et al., 2023, Fabbri et al., 2021] and keyword tracking tasks [Hsieh et al., 2024], which scale task complexity with context length, reasoning benchmarks challenge models to generate answers that are not clearly contained in the prompt. Such tasks include reading comprehension [Dua et al., 2019, Yu et al., 2020], commonsense reasoning [Zellers et al., 2019, Talmor et al., 2018, Geva et al., 2021], first-order logic [Han et al., 2022, Kwon et al., 2025], and mathematical problem-solving [Cobbe et al., 2021].

Reasoning benchmarks differ from long-context tasks in that they normally compel the LLM to provide answers that are longer than the question itself. This can pose a serious resource problem for the LLM, as past token key and value embeddings are maintained in memory to avoid redundant attention calculations. This key-value (KV) cache grows linearly with sequence length, which

*denotes equal contributions.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

can result in memory blowup for older or single-GPU setups. Furthermore, specialized reasoning models such as DeepSeek-R1 [Guo et al., 2025] and the Llama-Nemotron series are known to output excessively long reasoning traces [Cai et al., 2025, Fatemi et al., 2025] which outnumber the length of the prompt itself by hundreds to thousands of tokens.

To address the memory demands of long sequences, numerous KV cache compression methods have been proposed. These techniques generally maintain a fixed KV cache size by selectively discarding tokens deemed "unimportant". However, defining token importance" is non-trivial, and different approaches rely on distinct heuristics: attention scores [Zhang et al., 2023, Liu et al., 2023, Li et al., 2024], cosine similarity [Liu et al., 2024a, Han et al., 2023], embedding norms [Devoto et al., 2024], and head-specific token-type preferences [Ge et al., 2023]. Despite this variety, most evaluations of cache compression have focused on long-context benchmarks such as LongBench [Bai et al., 2023] and RULER [Hsieh et al., 2024], or on heterogeneous batteries like LM Eval Harness ([Gao et al., 2024]), rather than tasks where the generation length, not the prompt, dominates memory usage.

In this work, we conduct a comprehensive assessment of the major state-of-the-art KV cache compression strategies across eight reasoning benchmarks: FOLIO [Han et al., 2022], DROP [Dua et al., 2019], GSM8K [Cobbe et al., 2021], MATH-500 [Lightman et al., 2023], ReClor [Yu et al., 2020], StrategyQA [Geva et al., 2021], CommonSenseQA [Talmor et al., 2018], and OpenBookQA [Mihaylov et al., 2018]. Together, these benchmarks span four critical reasoning categories: reading comprehension, common sense, logical reasoning, and mathematical reasoning. We evaluate these strategies on Llama-3.1-8B-Instruct as well as four reasoning models: Llama-3.1-Nemotron-Nano-8B-v1, DeepSeek-R1-Distill-Llama-8B, and DeepSeek-R1-Distill-Qwen-7B/14B. By focusing on long-generation rather than long-prompt scenarios, our study fills a notable gap in the existing literature. Our primary contributions are threefold:

A comprehensive benchmark: We conduct a comprehensive evaluation of major KV cache compression strategies, including StreamingLLM, H2O [Zhang et al., 2023], a decoding-enabled SnapKV [Li et al., 2024], R-KV [Cai et al., 2025], and KNorm [Devoto et al., 2024], across a suite of eight benchmarks spanning mathematical, logical, and commonsense reasoning. We evaluate over several realistic settings, cache, and max token budgets for a single-GPU system.

Renewed attention for attention-based compression: Our analysis reveals that classical attention-based "heavy-hitter" strategies, which evict tokens based on accumulated attention scores, significantly outperform other methods, even defeating full-cache reasoning occasionally. Namely, this includes H2O and our novel and simple extension of SnapKV (prompt-only compression method) to a decoding-enabled variant, SnapKV-**D**ecoding. Both methods, especially SnapKV-D, win over *all* budgets and datasets for reasoning models.

A library for analyzing decoding compression: We implement a fork of the NVIDIA `kvpress` library, which adds support for decoding phase compression. We extend decoding-phase support for all 25+ compression strategies present within `kvpress`, providing a open-source playground for analyzing end-to-end KV cache compression strategies. Our code is available at `https://github.com/minghui-liu/kvpress/tree/decode`

2 Experiments & Analysis

2.1 Setup

KV Compression Methods. We test **H2O**, **StreamingLLM**, **KNorm**, our own decoding-variant of SnapKV which we call **SnapKV-D**, and **ShadowKV** [Sun et al., 2024]. We note that ShadowKV uses the CPU to offload the cache and thus is not a true compression strategy. However, offloading strategies represent an important class of compression methods; thus, we include them as a baseline.

Models. For our core experiments, we test the base, non-reasoning Llama-3.1-8B-Instruct and three reasoning models: DeepSeek-R1-Distill-Qwen-7B, Nemotron-Nano-8B-v1 and DeepSeek-R1-Distill-Llama-8B.

Datasets. We divide our benchmark into 4 distinct groups: **(1) Reading Comprehension:** DROP, ReClor; **(2) Logical Reasoning:** StrategyQA, FOLIO; **(3) Commonsense Reasoning:** OpenBookQA (OBQA), CommonsenseQA (CSQA); **(4) Math Reasoning:** MATH-500, GSM8K. For each dataset, we randomly sample 100 questions for two different seeds.

Table 1: Strategy versus Accuracy. Cache budgets = [128, 256, 384, 512]. Llama-3.1-8B-Instruct=ML, Deepssek-R1-Distill-Qwen-7B=DQ,=DQ,-Nemotron-Nano-8B-v1==LN, DeepSeek-R1-Distill-Llama-8B=DL.

Benchmark	ML				DQ				LN				DL			
	128	256	384	512	128	256	384	512	128	256	384	512	128	256	384	512
GSM8K full	0.88				0.70				0.64				0.70			
shadowkv	0.32				0.47				0.44				0.51			
h2o	**0.63**	**0.77**	0.82	0.83	0.21	0.44	0.51	0.52	0.22	0.45	0.52	0.57	0.37	0.53	0.62	0.61
knorm	0.05	0.53	0.73	0.82	0.00	0.00	0.08	0.16	0.01	0.02	0.09	0.18	0.00	0.09	0.19	0.28
rkv	0.12	0.34	0.50	0.49	0.04	0.07	0.18	0.30	0.04	0.03	0.09	0.15	0.05	0.04	0.14	0.17
snapkv	0.53	0.55	0.56	0.53	**0.67**	**0.67**	**0.70**	**0.71**	**0.65**	**0.63**	**0.66**	**0.66**	**0.72**	**0.72**	**0.74**	**0.72**
streaming_llm	0.26	0.75	**0.84**	**0.87**	0.02	0.19	0.32	0.44	0.03	0.20	0.40	0.53	0.06	0.25	0.39	0.56
Math500 full	0.39				0.47				0.45				0.46			
shadowkv	0.22				0.33				0.28				0.34			
h2o	**0.30**	**0.33**	**0.33**	**0.36**	0.14	0.21	0.29	0.31	0.16	0.24	0.31	0.33	0.20	0.31	0.36	0.36
knorm	0.03	0.18	0.22	0.33	0.00	0.01	0.03	0.05	0.01	0.01	0.03	0.06	0.00	0.01	0.02	0.06
rkv	0.03	0.10	0.16	0.20	0.04	0.04	0.05	0.17	0.02	0.04	0.03	0.06	0.03	0.05	0.02	0.02
snapkv	0.20	0.21	0.19	0.20	**0.38**	**0.36**	**0.36**	**0.32**	**0.41**	**0.44**	**0.45**	**0.43**	**0.42**	**0.44**	**0.41**	**0.41**
streaming_llm	0.11	0.26	0.32	0.35	0.03	0.12	0.19	0.26	0.02	0.13	0.22	0.34	0.03	0.09	0.21	0.29
CSQA full	0.77				0.67				0.51				0.75			
shadowkv	0.20				0.20				0.20				0.20			
h2o	**0.74**	0.76	**0.77**	**0.77**	0.44	0.61	**0.60**	**0.64**	0.47	0.49	**0.52**	0.51	0.48	0.72	0.73	**0.73**
knorm	0.34	**0.77**	0.75	0.76	0.05	0.13	0.30	0.42	0.36	0.40	0.44	0.46	0.05	0.28	0.54	0.66
rkv	0.36	0.62	0.76	0.77	0.10	0.09	0.27	0.34	0.28	0.30	0.42	0.41	0.07	0.11	0.16	0.35
snapkv	0.70	0.64	0.71	0.72	**0.65**	**0.62**	0.59	0.61	**0.49**	**0.50**	0.51	**0.53**	**0.74**	**0.73**	**0.74**	**0.73**
streaming_llm	0.20	0.75	0.76	**0.77**	0.08	0.14	0.31	0.48	0.36	0.44	0.46	0.50	0.04	0.14	0.35	0.50
OBQA full	0.84				0.78				0.64				0.84			
shadowkv	0.31				0.31				0.31				0.31			
h2o	**0.83**	**0.86**	**0.86**	**0.86**	0.42	0.64	**0.69**	0.67	0.59	0.59	0.58	0.62	0.48	0.78	**0.83**	**0.84**
knorm	0.41	0.79	0.84	0.82	0.03	0.05	0.23	0.38	0.32	0.44	0.48	0.57	0.03	0.27	0.57	0.70
rkv	0.22	0.66	0.77	0.84	0.10	0.10	0.21	0.26	0.35	0.44	0.51	0.51	0.07	0.07	0.19	0.32
snapkv	0.73	0.77	0.72	0.76	**0.71**	**0.75**	0.68	**0.76**	**0.68**	**0.63**	**0.66**	**0.66**	**0.82**	**0.83**	**0.83**	0.81
streaming_llm	0.14	0.72	0.84	0.84	0.02	0.11	0.28	0.37	0.36	0.46	0.52	0.62	0.07	0.15	0.32	0.52
ReClor full	0.60				0.45				0.48				0.51			
shadowkv	0.27				0.27				0.27				0.27			
h2o	0.32	0.56	**0.60**	0.58	0.01	0.04	0.18	0.28	0.20	0.22	0.35	**0.40**	0.03	**0.08**	**0.23**	**0.38**
knorm	0.01	0.19	0.46	**0.59**	0.00	0.00	0.01	0.01	0.01	0.03	0.07	0.07	0.00	0.00	0.02	0.10
rkv	0.04	0.21	0.40	0.54	0.04	0.03	0.02	0.01	0.03	0.08	0.08	0.07	0.04	0.03	0.03	0.11
snapkv	**0.53**	**0.57**	0.58	0.55	**0.45**	**0.39**	**0.40**	**0.43**	**0.42**	**0.42**	**0.42**	0.37	**0.50**	**0.08**	0.05	0.05
streaming_llm	0.05	0.21	0.59	0.58	0.00	0.01	0.01	0.04	0.03	0.06	0.09	0.14	0.00	0.00	0.01	0.06
DROP full	0.15				0.16				0.11				0.14			
shadowkv	0.28				0.14				0.11				0.09			
h2o	0.12	**0.14**	**0.17**	**0.17**	0.04	0.07	0.10	0.10	0.05	0.06	0.10	0.09	0.06	0.07	0.10	0.11
knorm	0.01	0.08	0.13	0.13	0.00	0.01	0.01	0.03	0.01	0.01	0.02	0.03	0.00	0.01	0.01	0.05
rkv	0.06	0.07	0.14	0.11	0.04	0.04	0.03	0.04	0.02	0.06	0.05	0.03	0.03	0.03	0.05	0.07
snapkv	**0.15**	0.12	0.11	0.12	**0.13**	**0.11**	**0.12**	**0.16**	**0.11**	**0.11**	**0.12**	**0.10**	**0.17**	**0.15**	**0.15**	**0.16**
streaming_llm	0.09	0.11	0.15	0.16	0.04	0.05	0.08	0.13	0.03	0.02	0.06	0.08	0.02	0.02	0.09	0.13
StrategyQA full	0.83				0.67				0.89				0.74			
shadowkv	0.68				0.60				0.65				0.80			
h2o	**0.81**	**0.87**	0.88	**0.89**	0.33	**0.64**	**0.74**	**0.72**	0.76	0.84	**0.85**	**0.83**	0.25	**0.69**	**0.77**	**0.79**
knorm	0.47	0.85	0.88	0.87	0.00	0.12	0.44	0.59	0.38	0.55	0.68	0.76	0.06	0.36	0.57	0.70
rkv	0.60	0.79	0.77	0.79	0.05	0.14	0.34	0.42	0.42	0.45	0.64	0.71	0.08	0.35	0.50	0.63
snapkv	0.78	0.78	0.81	0.76	**0.60**	0.59	0.57	0.63	**0.83**	**0.85**	0.84	**0.84**	**0.68**	0.66	0.64	0.68
streaming_llm	0.11	0.76	**0.89**	0.85	0.00	0.05	0.22	0.42	0.24	0.39	0.52	0.69	0.03	0.11	0.36	0.56
FOLIO full	0.51				0.36				0.36				0.47			
shadowkv	0.33				0.33				0.33				0.33			
h2o	0.22	**0.43**	0.41	0.43	0.03	0.21	0.23	0.23	0.22	0.36	0.35	0.37	0.07	0.37	0.41	**0.46**
knorm	0.02	0.28	0.39	0.38	0.00	0.01	0.03	0.05	0.03	0.04	0.07	0.13	0.00	0.03	0.11	0.21
rkv	0.07	0.36	0.44	0.34	0.04	0.03	0.02	0.06	0.06	0.08	0.11	0.14	0.03	0.08	0.13	0.26
snapkv	**0.44**	0.40	**0.45**	**0.46**	**0.30**	**0.25**	**0.31**	**0.29**	**0.38**	**0.42**	**0.41**	**0.41**	**0.46**	**0.45**	**0.49**	**0.46**
streaming_llm	0.03	0.09	0.25	0.35	0.00	0.01	0.02	0.03	0.03	0.03	0.06	0.15	0.00	0.01	0.04	0.09

Performance. For benchmarking the individual compression strategies, we use the NVIDIA kvpress library, which natively provides most of the targeted algorithms. We provide each dataset to each model over the cache budgets {128, 256, 384, 512}. Each model is allowed to generate a maximum of 2048 new tokens via greedy decoding. This token limit is enforced to better simulate a resource-constrained setting for inference and also based on mean generation lengths reported in Table 2.. We use author-recommended hyperparameters for all methods. Accuracy benchmarks were performed on an HPC cluster using an NVIDIA RTX A6000 48GB GPU.

2.2 Results

In this section, we review the high-level performance trends of our selected KV cache compression strategies as reflected in Table 1 and additional material in the Appendix.

Attention is the most versatile estimator for reasoning models. SnapKV-D and H2O are the most dominant, significantly outcompeting nearly all compression strategies across all budget constraints and datasets for our reasoning models. These methods rely on accumulated attention scores to determine the most important tokens to retain. (i.e., “heavy hitters"). While both maintain a recency window, H2O is focused on heavy hitters with regard to the current token, while SnapKV (and consequently, SnapKV-D) finds heavy hitters with respect to an observation window at the end of the current sequence. The latter approach is more effective, routinely defeating H2O. The utility of the observation window was previously known to work well for prompt compression, but not for long decoding phases. *We find that this trend generalizes for model size (Section D), max token settings (Section D.1), and budgets.*

No singular strategy is dominant for the non-reasoning Llama-3.1-8B-Instruct. For models that do not produce reasoning traces, the optimal choice of strategy is dataset-dependent. For example, while StreamingLLM excels at GSM8K, it is less effective on all other task types. While SnapKV-D and H2O are capable of winning 2/4 settings for several datasets, other methods, such as R-KV and KNorm, can convincingly defeat them over the remaining budgets.

Eviction lags full cache performance for reasoning models. According to Table 1, all compression strategies can defeat the full cache performance of Llama-3.1-8B-Instruct on at least one setting (with H2O and SnapKV-D frequently achieving this). However, for reasoning models, this trend only holds true for SnapKV-D. While H2O is still second best compared to other strategies, it significantly lags full cache performance on nearly every dataset. As noted in Figure 3, H2O results in significantly longer reasoning traces than SnapKV-D, which occasionally do not terminate. It is possible that allowing a less restrictive constraint on the maximum number of new tokens can alleviate this performance drop, though this would result in longer inference.

Cache compression can cost more computation. Interestingly, according to Figure 3, eviction strategies can result in more “talkative" reasoning models, generating noticeably longer sequences compared to the full cache setting, while this does not occur for Llama-3.1-8B-Instruct. In Section E, we show this phenomenon at work, where KNorm results in long circular babble for Deepseek-R1-Distill-Llama-8B that never produces an answer. This is problematic for resource-constrained settings: while cache eviction can reduce peak memory usage, it can result in significantly longer auto-regression. At lower budgets, eviction occurs more frequently over shorter stretches of context, resulting in the eviction of critical reasoning tokens, which can result in longer reasoning.

3 Conclusion

In this work, we comprehensively assessed the performance of several popular KV cache compression strategies on reasoning tasks for the non-reasoning Llama-3.1-8B-Instruct and several popular reasoning models. For the non-reasoning model, we find that no singular method is dominant. However, for reasoning models, we demonstrate that attention-based eviction methods such as H2O and SnapKV-D perform extraordinarily well on a variety of reasoning tasks, even occasionally exceeding full cache performance. Furthermore, this generalizes to a larger model, R1-Distill-Qwen-14B (Section D). We also discovered that it is possible, especially at lower budgets, for compression strategies to produce longer reasoning traces, thus revealing an under-considered tradeoff between memory and inference costs. For future work, even larger models should be assessed, along with larger cache budgets, to fully assess the limits of the cache compression/performance tradeoff.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit inference in rotated llms. *Advances in Neural Information Processing Systems*, 37:100213–100240, 2024.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*, 2023.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Yucheng Li, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Junjie Hu, et al. Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling. *arXiv preprint arXiv:2406.02069*, 2024.

Zefan Cai, Wen Xiao, Hanshi Sun, Cheng Luo, Yikai Zhang, Ke Wan, Yucheng Li, Yeyang Zhou, Li-Wen Chang, Jiuxiang Gu, et al. R-kv: Redundancy-aware kv cache compression for training-free reasoning models acceleration. *arXiv preprint arXiv:2505.24133*, 2025.

Zhuoming Chen, Ranajoy Sadhukhan, Zihao Ye, Yang Zhou, Jianyu Zhang, Niklas Nolte, Yuandong Tian, Matthijs Douze, Leon Bottou, Zhihao Jia, et al. Magicpig: Lsh sampling for efficient llm generation. *arXiv preprint arXiv:2410.16179*, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Alessio Devoto, Yu Zhao, Simone Scardapane, and Pasquale Minervini. A simple and effective l_2 norm-based strategy for kv cache compression. *arXiv preprint arXiv:2406.11430*, 2024.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*, 2019.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409, 2021.

Mehdi Fatemi, Banafsheh Rafiee, Mingjie Tang, and Kartik Talamadupula. Concise reasoning via reinforcement learning. *arXiv preprint arXiv:2504.05185*, 2025.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL `https://zenodo.org/records/12608602`.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive kv cache compression for llms. *arXiv preprint arXiv:2310.01801*, 2023.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Insu Han, Rajesh Jayaram, Amin Karbasi, Vahab Mirrokni, David P Woodruff, and Amir Zandieh. Hyperattention: Long-context attention in near-linear time. *arXiv preprint arXiv:2310.05869*, 2023.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, et al. Folio: Natural language reasoning with first-order logic. *arXiv preprint arXiv:2209.00840*, 2022.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Yakun S Shao, Kurt Keutzer, and Amir Gholami. Kvquant: Towards 10 million context length llm inference with kv cache quantization. *Advances in Neural Information Processing Systems*, 37:1270–1303, 2024.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Yejin Kwon, Daeun Moon, Youngje Oh, and Hyunsoo Yoon. Logicqa: Logical anomaly detection with vision language model generated questions. *arXiv preprint arXiv:2503.20252*, 2025.

Seungpil Lee, Woochang Sim, Donghyeon Shin, Wongyu Seo, Jiwon Park, Seokki Lee, Sanha Hwang, Sejin Kim, and Sundong Kim. Reasoning abilities of large language models: In-depth analysis on the abstraction and reasoning corpus. *ACM Transactions on Intelligent Systems and Technology*, 2024.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. *Advances in Neural Information Processing Systems*, 37:22947–22970, 2024.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*, 2020.

Minghui Liu, Tahseen Rabbani, Tony O'Halloran, Ananth Sankaralingam, Mary-Anne Hartley, Furong Huang, Cornelia Fermüller, and Yiannis Aloimonos. Hashevict: A pre-attention kv cache eviction strategy using locality-sensitive hashing. *arXiv preprint arXiv:2412.16187*, 2024a.

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. *Advances in Neural Information Processing Systems*, 36:52342–52364, 2023.

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*, 2024b.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Hanshi Sun, Li-Wen Chang, Wenlei Bao, Size Zheng, Ningxin Zheng, Xin Liu, Harry Dong, Yuejie Chi, and Beidi Chen. Shadowkv: Kv cache in shadows for high-throughput long-context llm inference. *arXiv preprint arXiv:2410.21465*, 2024.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*, 2018.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context llm inference. *arXiv preprint arXiv:2406.10774*, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. Reclor: A reading comprehension dataset requiring logical reasoning. *arXiv preprint arXiv:2002.04326*, 2020.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Jinghan Zhang, Xiting Wang, Weijieying Ren, Lu Jiang, Dongjie Wang, and Kunpeng Liu. Ratt: A thought structure for coherent and correct llm reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 26733–26741, 2025.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023.

Appendix

A Related Work

A.1 KV Cache Compression

KV cache compression is a rich field of study composed of strategies ranging from quantization [Hooper et al., 2024, Ashkboos et al., 2024, Liu et al., 2024b] to offloading methods that move the entire cache to the CPU which is significantly less memory bound [Sun et al., 2024, Chen et al., 2024, Tang et al., 2024]. However, in this work, we are focused on strategies which maintain a constant cache size, thus permitting arbitrary generation length.

A.1.1 Token Eviction

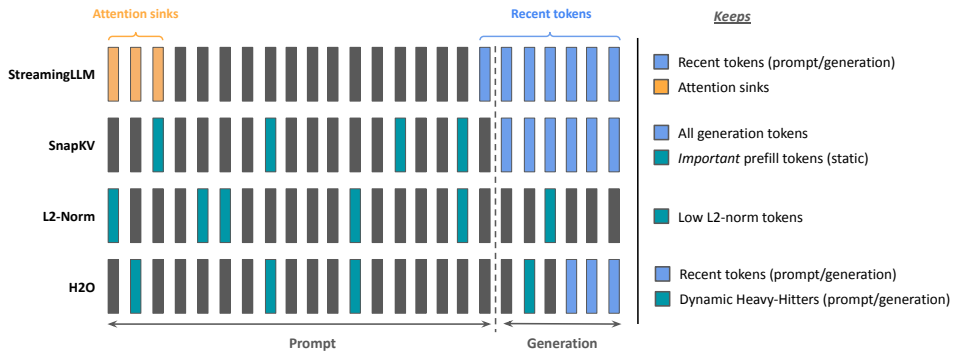

Figure 1: A Conceptual Comparison of Token Retention Strategies in Different KV Cache Compression Methods. Each row illustrates a method's logic for retaining tokens (colored) versus evicting them (gray) from the KV cache during a long sequence divided into a prefill and decoding phase.

A primary line of research for mitigating the memory burden of the KV cache involves *token eviction*. These methods aim to reduce the cache size by selectively removing or merging tokens deemed less important. To achieve this, multiple approaches have been developed, including recency-based approaches such as simple sliding window [Beltagy et al., 2020], importance-based methods that retain "attention sinks" or heavy-hitter tokens from the prompt [Xiao et al., 2023, Zhang et al., 2023, Li et al., 2024, Liu et al., 2023], dynamically adjustment of KV caches per layer for optimal efficiency-utility balancing [Cai et al., 2024], redundancy-aware techniques that merge semantically similar states [Cai et al., 2025]. Figure 1 provides a conceptual comparison of the most important approaches we cover in this work.

StreamingLLM [Xiao et al., 2023] is based on the critical observation that in autoregressive models, a small number of initial tokens act as "attention sinks," consistently receiving a large proportion of attention scores regardless of their semantic relevance. StreamingLLM's strategy is therefore to permanently cache the KV states of the first few (e.g., four) tokens, which serve as the attention sinks, and combine them with a sliding window of the most recent tokens. **H2O** [Zhang et al., 2023] dynamically identifies important or "heavy hitter" tokens based on their cumulative attention scores received during generation. The H2O cache is composed of two parts: a budget for the most recent tokens and a budget for the H2 tokens. **SnapKV** [Li et al., 2024] focuses primarily on compressing the KV cache of the initial prompt during the prefill stage. SnapKV uses a small "observation window" at the end of the prompt to predict importance. The attention scores from queries in this observation window are aggregated to "vote" for important positions (heavy hitters) in the prefix. A distinct and computationally efficient approach, which we refer to as the **KNorm** strategy [Devoto et al., 2024], bypasses the need for attention scores entirely. Specifically, the authors observe that tokens whose key vectors have a low L_2 norm consistently attract high attention scores from subsequent queries.

A.2 Benchmarking Reasoning

GSM8K (Grade School Math 8K) is a widely-used dataset of grade-school level math word problems that require a sequence of elementary arithmetic operations to solve [Cobbe et al., 2021]. More advanced challenges are drawn from the **MATH-500** dataset [Lightman et al., 2023], which contains competition-level problems across algebra, geometry, and number theory. **ReClor** [Yu et al., 2020] is a reading comprehension dataset built from GMAT and LSAT logical reasoning questions. Similarly, **LogiQA** [Liu et al., 2020] provides multiple-choice questions from civil service exams that require a deep understanding of logical puzzles and deductions. For evaluating capabilities in more formal systems, the **FOLIO** [Han et al., 2022] dataset assesses natural language reasoning in the context of First-Order Logic (FOL). Beyond formal and mathematical logic, a significant portion of research focuses on commonsense reasoning. **StrategyQA** [Geva et al., 2021] tests a model's ability to infer the implicit reasoning steps needed to answer a yes/no question by asking for the underlying strategy. Another tested benchmarks is **CommonsenseQA**, which tests a model's ability to reason with general world knowledge. Finally, the integration of textual understanding with quantitative skills is measured by benchmarks such as **DROP** [Dua et al., 2019]. This reading comprehension dataset is unique in that answering its questions requires performing discrete operations like counting, sorting, or simple arithmetic directly on the information presented in a context paragraph.

B Preliminaries

In this section, we briefly review the concepts of large language models, LLM inference and autoregressive generation, the KV cache, and the chain-of-thought (CoT) reasoning.

Transformer Architectures and Autoregressive Generation. Modern Large Language Models (LLMs) predominantly operate as autoregressive, decoder-only Transformers [Vaswani et al., 2017, Radford et al., 2019, Achiam et al., 2023, Touvron et al., 2023]. This architecture generates text sequentially, producing one token at a time by conditioning on the entire preceding sequence of tokens, which includes both the initial prompt and any previously generated output [Brown et al., 2020]. Importantly, the model's ability to maintain coherent and contextually relevant generation over time is crucial to its capabilities, especially in tasks requiring reasoning or narrative development [Lee et al., 2024, Zhang et al., 2025].

Self-Attention Mechanism and the KV Cache Bottleneck. During generation, a query (q) vector for the current token is *attends* to a series of Key (k) and Value (v) vectors corresponding to every token in the preceding context. In this process, notably, for the generation of every new token, the entire sequence of Key and Value vectors for *all* previous tokens should be accessed. To avoid recomputing these K-V pairs at each step, they are stored in the Key-Value (KV) cache, the size of which grows linearly with the sequence length (n), resulting in an $O(n)$ memory complexity that creates a significant bottleneck. Formally, for a sequence of n tokens, we denote the query cache $Q_l^h \in \mathbb{R}^{n\times d}$, key cache $K_l^h \in \mathbb{R}^{n\times d}$, and value cache $V_l^h \in \mathbb{R}^{n\times d}$, where d is the embedding dimension, l is the layer, and h denotes a head for multi-head attention layers [Vaswani et al., 2017]. The dot-product self-attention mechanism is defined as $A_l^h(Q_l^h, K_l^h, V_l^h) = \text{softmax}(Q_l^h(K_l^h)^\top/\sqrt{d})V_l^h$. To avoid linear scaling with sequence length, *token eviction* methods, the key focus of work, discard embeddings of previous tokens which are no longer "important" to the current decoding step.

Figure 2 (a) exhibits the Q, K, and V vectors along with the self-attention mechanism. Figure 2 (b) demonstrate the decoding KV cache bottleneck on memory.

Chain-of-Thought and Multi-Step Reasoning. While many long-context applications involve processing long prompts, a critical class of tasks requires long-form generation from short and complex prompts. Prompting strategies such as Chain-of-Thought (CoT) encourage models to externalize their reasoning process, generating intermediate "thinking" steps that can extend for hundreds or thousands of tokens to solve a problem [Wei et al., 2022, Wang et al., 2022]. Benchmarks like GSM8K [Cobbe et al., 2021] are representative of this domain, where the path to the correct answer necessitates a lengthy, self-generated chain of reasoning that stresses the models' decoding-phase memory limits.

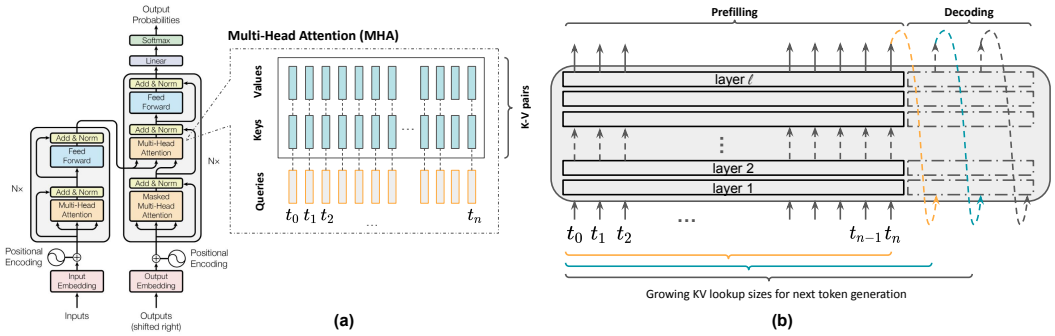

Figure 2: Overview of the Transformer Decoder Architecture and the Inference Bottleneck. (a) The standard Transformer decoder architecture (left) and the Multi-Head Attention (MHA) mechanism (right). In MHA, Query vectors representing the current context attend to a sequence of Key-Value (K-V) pairs from all previous tokens. Such K-V pairs form the basis of the KV cache. (b) The two-phase inference process in autoregressive generation. During Prefilling, the tokens in the input context are processed in parallel to populate the initial KV cache across all layers. During Decoding, each new token is generated sequentially. This requires recomputing the entire set of the preceding KV entries at each step, causing the lookup size to grow linearly with the sequence length.

C Cache Budget vs Output Length

We study the effects of cache budget on output generation lengths in Figure 3. Fascinatingly, lower budgets are capable of triggering longer reasoning traces, revealing a hidden tradeoff between cache budget and inference costs specifically for reasoning models. KNorm, arguably the lowest performing strategy, tends to cause the greatest elongation of outputs. In Section E, we examine one such non-terminating output that demonstrates repetitive, dead-end chain-of-thought.

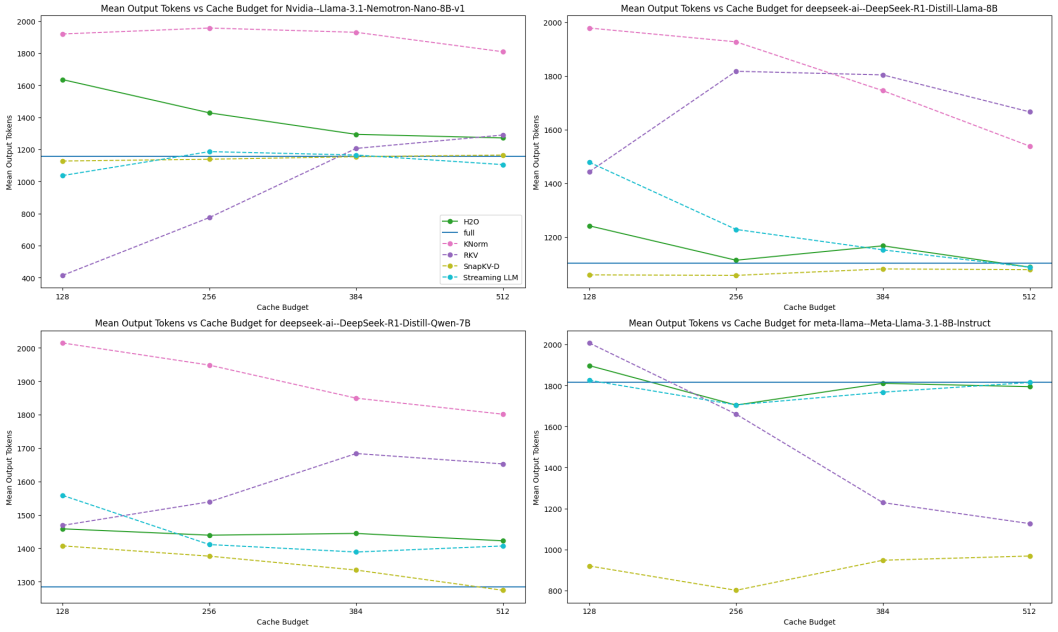

Figure 3: Budget vs Output Length. We observe that several compression methods, especially at lower budgets, ultimate produce longer outputs than the base full cache model.

C.1 Generation Lengths

In Table 2, we report the mean generation lengths for all methods across and models on MATH-500, the dataset which elicits the longest responses. To keep these tables concise, we averaged output lengths over all budgets.

Table 2: Mean output tokens generated by different models under various strategies for Math500.

Strategy	Model	Mean Output Tokens
full	`Nvidia-Llama-3.1-Nemotron-Nano-8B-v1`	1616.275
full	`deepseek-ai-DeepSeek-R1-Distill-Llama-8B`	1727.18
full	`deepseek-ai-DeepSeek-R1-Distill-Qwen-7B`	1728.84
h2o	`Nvidia-Llama-3.1-Nemotron-Nano-8B-v1`	1753.4075
h2o	`deepseek-ai-DeepSeek-R1-Distill-Llama-8B`	1763.805
h2o	`deepseek-ai-DeepSeek-R1-Distill-Qwen-7B`	1767.7275
knorm	`Nvidia-Llama-3.1-Nemotron-Nano-8B-v1`	1987.80875
knorm	`deepseek-ai-DeepSeek-R1-Distill-Llama-8B`	2001.07625
knorm	`deepseek-ai-DeepSeek-R1-Distill-Qwen-7B`	1967.4575
snapkv	`Nvidia-Llama-3.1-Nemotron-Nano-8B-v1`	1667.895
snapkv	`deepseek-ai-DeepSeek-R1-Distill-Llama-8B`	1790.165
snapkv	`deepseek-ai-DeepSeek-R1-Distill-Qwen-7B`	1794.0275
streaming_llm	`Nvidia-Llama-3.1-Nemotron-Nano-8B-v1`	1375.70375
streaming_llm	`deepseek-ai-DeepSeek-R1-Distill-Llama-8B`	1655.0375
streaming_llm	`deepseek-ai-DeepSeek-R1-Distill-Qwen-7B`	1698.25625

D Large Model Comparison

We determine whether our observed trends hold for a larger reasoning model, R1-Distill-Qwen-14B in Table 3. We examine the performance of all methods on the more challenging GSM8K and MATH500. Unsurprisingly, base accuracies do improve, but more importantly, we observe that again, the heavy-hitter methods H2O and SnapKV-D outperform their competitors by a significant margin indicating that larger reasoning models still benefit from attention-based eviction.

Table 3: R1-Distill-Qwen14B. Cache budgets = [128, 256, 384, 512]. We examine the performance of various compression methods for a larger reasoning model. Winner per budget in bold.

Method	GSM8K				MATH500			
	128	256	384	512	128	256	384	512
full	0.81				0.47			
shadowkv	0.53				0.38			
h2o	0.33	0.56	0.62	0.64	0.2	0.27	0.31	0.31
knorm	0	0.02	0.08	0.21	0	0	0	0.02
snapkv	**0.80**	**0.82**	**0.81**	**0.78**	**0.43**	**0.44**	**0.42**	**0.45**
streaming_llm	0.07	0.27	0.5	0.59	0.02	0.17	0.26	0.35

D.1 Max Token Ablation

As explained in setup, we chose a max token length of 2048 both because we find that the mean token length over datasets is under this budget and to better assess performance in a compute-bound setting. However, we study the effect of max token generation on method performance under a fixed budget of 1024 for MATH500 for R1-Distill-Qwen7B. We find that performance improves significantly for all methods initially, but then SnapKV-D overtakes all methods for all other max token limits.

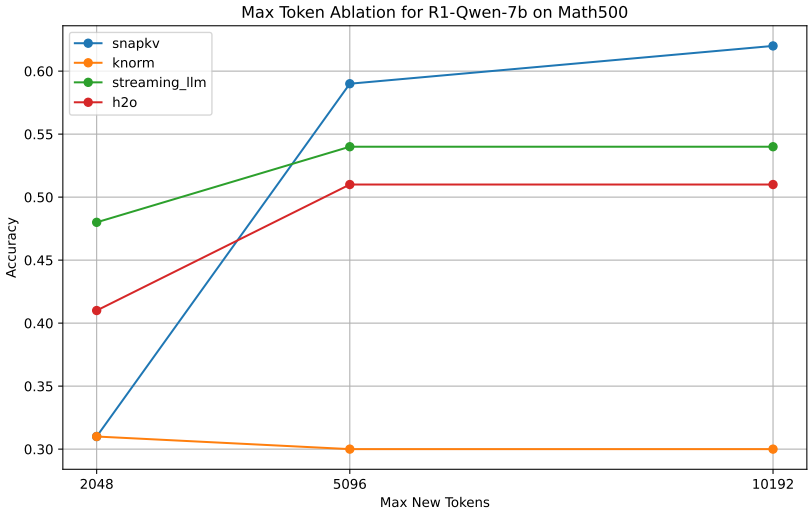

Figure 4: Max tokens allowed versus performance. We observe that SnapKV-D excels across most max token budgets.

E A Verbose, Wrong Answer

Below, we present a verbose, incorrect answer generated for Deepseek-R1-Distill-Llama-8B via KNorm on a budget of 256 tokens.

```
Define:

- Let a be the value of a small button (SB).
- Then, 1 SS = a
- 1 large button (LB) is worth 3 small buttons, so LB = 3a
- Alternatively, 1 LB = 3a
- 1 LS can be converted to 1 LB or 3 SS, so 1 LS = 1 LB or 3*(1a)? Wait, small
    stickers themselves are worth 1a.

Hold on, perhaps clarifying units.

Wait, sorry, the problem states:

"Lorraine has ... stickers and large buttons..."

Wait, maybe it's better to model it with equations.

Let me denote:

- Value of 1 small sticker = x
- Then, 1 medium sticker would be higher, but in our case, 1 large sticker (LS) is
    worth either 1 large button (LB) or 3 small buttons (SB).

But we don't know whether a small button is same as a small sticker? Let's see.

The problem says:

- A large button is worth three small stickers. So 1 LB = 3 SS.

So, 1 SS = 1/3 LB.

Or, alternatively, 1 LB = 3 SS.

So, yeah, 1 LB = 3 SS.
```

```
On the flip side, 1 LS can be either 1 LB or 3 SS.

But note that 1 LB is equal to 3 SS.

So, the question becomes:

Traders use sticker-based currencies.

Given that, let me
```

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: Section 2

   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: Section 3

   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: [NA]

Justification: No theory

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: Key libraries and setup mentioned

   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: [NA]

Justification: Workshop

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: Section 2, Experimental Setup

   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: [NA]

   Justification: 1 run

   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: Section 2, Experimental Setup

   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: No ethical concerns

   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: [Yes]

    Justification: Section 3

    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: NA

    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: All datasets and models cited.

    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: No 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: No crowdsourcing

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: No 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.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: NA

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.