# LLMSteer: Improving Long-Context LLM Inference by Steering Attention on Reused Contexts

**Zhuohan Gu*** **Jiayi Yao*** **Kuntai Du** **Junchen Jiang**
*University of Chicago*
*{zhuohan, jiayi3, kuntai, junchenj}@uchicago.edu*

## Abstract

As large language models (LLMs) show impressive performance on complex tasks, they still struggle with longer contextual understanding and high computational costs. To balance efficiency and quality, we introduce LLMSteer, a fine-tuning-free framework that enhances LLMs through query-independent *attention steering*. Tested on popular LLMs and datasets, LLMSteer narrows the performance gap with baselines by 65.9% and reduces the runtime delay by up to 4.8× compared to recent attention steering methods.

## 1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in complex tasks such as question answering, summarization, and reasoning [3, 1, 2]. To enhance their reliability, LLMs are often augmented with domain-specific or user-specific knowledge that extends beyond their inherent training data [15, 10, 5]. However, incorporating these supplemental contexts, which can exceed thousands of tokens [11, 9], presents two challenges: (1) models often struggle to comprehend *long context* (e.g., lost-in-the-middle problem [17, 13]) and (2) processing long context incurs substantial runtime costs [18, 16, 25].

Since the Key-Value (KV) cache of the same context text chunks is often reused [19, 23, 11] multiple times, many recent systems adopt prefix caching[11, 19, 20], which stores the KV caches for the frequently reused contexts such that LLMs no longer need to prefill these contexts repeatedly. However, the model persists in losing track of key information from the context as its KV pairs remain unchanged. **So, is there a way to simultaneously achieve high efficiency and high quality without fine-tuning models?**

We present LLMSteer, a pioneering, fine-tuning-free framework that efficiently improves the generation quality of models through *post-hoc attention steering*. More specifically, LLMSteer reweights attention scores to steer the model's attention toward selected tokens, thereby improving its contextual understanding. Our insight comes from the fact that storing and reusing the KV cache offers an opportunity to modify the KV cache offline to improve the model's understanding regarding the context. Thus, all queries could benefit from this improvement as the process of modifying the KV cache of the preceding context is independent of the queries. LLMSteer enables LLMs to read the same context in different ways by leveraging the idea that each pass through the same context with different prefix prompts leads the model to generate distinct KV caches, which represent different understandings of the context.

We implemented LLMSteer on top of LLaMA-3.1-8b-Instruct [8], and compared LLMSteer with LLaMA-3.1-8b-Instruct, LLaMA-3.1-70b-Instruct [8] and the state-of-the-art attention steering baseline. LLMSteer demonstrates that by steering tokens that are consistently important across the model's understandings, it not only achieves a significant increase in F1 scores (e.g., 72.9 -> 82.0), but also delivers up to 4.8× faster compared to the baselines. LLMSteer not only reduces runtime costs but also improves generation quality. This paper is the first effort to (1) improve model generation quality without fine-tuning and (2) do so in a way that is compatible with prefix caching.

*Equal contribution.

Machine Learning for Systems Workshop at (NeurIPS 2024).

## 2 Post-hoc attention steering

Attention steering didn't come out of nowhere. Methods like PASTA [24] and AutoPASTA [4] introduce techniques to boost the generation quality of LLMs by guiding their attention to specific parts of the input text. In PASTA, users manually specify tokens in the input and steer the LLM's attention toward these highlighted tokens through reweighting. However, annotating relevant input spans by humans is not always feasible, especially for lengthy contexts and context-specific tasks.

AutoPASTA enhances PASTA by automating the process of identifying key contextual information through iterative prompting. More specifically, AutoPASTA directly prompts the LLM to generate the key sentence in the context based on the query, and then upweight the attention scores of tokens in the key sentence. Despite its potential in improving model's generation quality, AutoPASTA has several limitations.

First, the iterative prompting approach necessitates two LLM calls, which increases runtime latency. Second, AutoPASTA is incompatible with prefix caching, a technique in which KV caches of the frequently used text chunks are stored and reused such that the prefill of the reused text chunks can be skipped. For example, Llama-8B [8] requires 2.04 seconds to process a 5000-token context using 2 A40 GPUs. However, if the KV cache of the text chunk is precomputed and stored, only the final user query needs to be prefilled, which only takes 0.039 seconds on the same hardware. Although prefix caching can still be applied to the initial call in AutoPASTA, the second call cannot benefit from prefix caching in that the attention of the context is steered in a query-dependent way. Consequently, the KV cache of the context will be altered and can no longer be reused across different queries.

## 3 LLMSteer design

We now present the design of LLMSteer, which not only reduces runtime costs but also improves inference quality. We begin with the basic insight, followed by how LLMSteer selects and steers tokens.

**Basic insight:** *When an LLM processes a context in a single pass, some tokens may not receive sufficient attention. By prompting the LLM to read the context multiple times, each with a different prefix prompt and thus a different KV cache, we hypothesize that tokens with consistently high attention scores across these KV caches are the ones requiring more attention.*

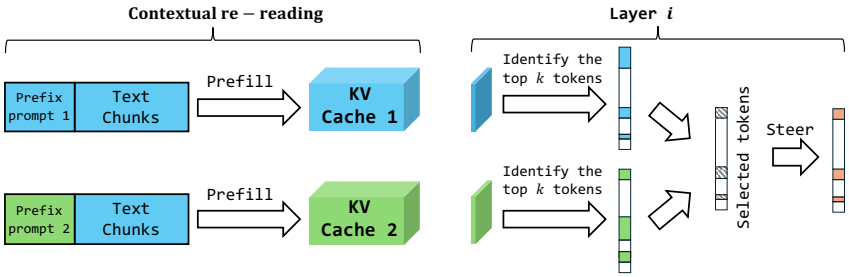


Figure 1: LLMSteer system in an LLM inference workflow processes the same context twice using different prefix prompts to generate key-value caches. It then reweights the selected tokens.

**Key components:** LLMSteer consists of three high-level steps. First, it prompts the LLM to process the context text chunks twice, each time using a different prefix prompt. This approach leverages the insight that varying the prefix prompt influences how the LLM interprets the context. Second, at each layer, it computes the cumulative attention scores for each head, selecting the top $k$ tokens based on these scores across all heads. It then identifies tokens that rank highly in both the first and second passes. Finally, it scales up the attention weight of selected tokens. An illustrative overview of LLMSteer is shown in Figure 1 and the corresponding algorithm can be found in §A.2.

**Contextual re-reading:** In this step, LLMSteer prompts the LLM to process the same context twice, each time using a distinct prefix prompt. The goal is to force the model to read the context in two different ways. The prefix prompts do not include the query itself because we want to ensure that the context is read in a non-query-dependent manner. This allows the model to capture general context-related information without bias toward a specific query. More importantly, given a fixed

context $C$ and a sequence of queries, our method makes the prefix cache reusable across all queries and reduces the need to call the LLM twice for each query during online inference.

**Token selection:** For each pass, we store the cumulative attention scores of the prefix prompt and context tokens at each layer and each head. For simplicity, we will ignore indices for layers and heads. Specifically, for each layer and each head, consider the input prompt tensor $\mathbf{X} \in \mathbb{R}^{L \times D}$, where $L$ denotes the input prompt length and $D$ represents the model's hidden dimension. The attention matrix is computed as:

$$\mathbf{A} = softmax\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D}}\right) \in \mathbb{R}^{L \times L},$$

where $\mathbf{Q} = \mathbf{X}\mathbf{W}^Q$ and $\mathbf{K} = \mathbf{X}\mathbf{W}^K$. Here, $\mathbf{W}^Q, \mathbf{W}^K \in \mathbb{R}^{D \times D}$ are learned weight matrices that project the input prompt $\mathbf{X}$ into query and key spaces. Let $L_p$ denote the combined length of the prefix prompt and context. The attention matrix for the prefix prompt and context tokens is:

$$\mathbf{A}_p = \mathbf{A}[: L_p, : L_p] \in \mathbb{R}^{L_p \times L_p}.$$

To compute the cumulative attention scores, we sum the attention weights across each column of the attention matrix. This summation gives us the total attention that each token has received from all preceding tokens:

$$\mathbf{A}_s = \sum_{i=0}^{L_p - 1} \mathbf{A}_p[i, :] \in \mathbb{R}^{L_p},$$

where $\mathbf{A}_s(i)$ is the cumulative attention score for token $i$, representing the total attention received by token $i$ from all preceding tokens.

We then store the cumulative attention scores from all heads and layers along with their corresponding indices for further processing. To determine which tokens to steer, we first sort the cumulative attention scores for each layer across all heads and identify the top $k$ tokens for that layer. Next, we find the intersection of these top-ranked tokens from both passes, selecting those that consistently rank high in both. These selected tokens are then upweighted.

**Steering:** In the steering phase, for each layer, LLMSTEER adjusts the attention mechanism by applying a weighting matrix, denoted as $\boldsymbol{M}$, which scales the attention scores for the selected tokens. Initially, we define $\boldsymbol{M} \in \mathbb{R}^{L \times L}$, where $L$ is the input prompt length, as a tensor filled with ones. For the selected tokens at each layer, we update the corresponding entries in $\boldsymbol{M}$ using a predefined scaling factor, $\alpha$. We then expand $\boldsymbol{M}$ across the batch and head dimensions to align with the shape of the attention weights, creating a tensor of shape $\mathbb{R}^{1 \times H \times L \times L}$, where $H$ represents the number of attention heads. Finally, we apply this weighting matrix element-wise to the original attention scores at each layer, effectively steering the attention toward the selected tokens:

$$\mathbf{A}_{\text{steered}} = \boldsymbol{M} \odot \mathbf{A},$$

where $\odot$ denotes element-wise multiplication and $\mathbf{A}$ represents the original attention matrix.

# 4 Evaluation

## 4.1 Setup

**Datasets:** Our evaluation covers three datasets: SQuAD [21], TriviaQA [12] and GSM8K [6]. SQuAD is a reading comprehension dataset with questions based on Wikipedia articles. TriviaQA is a large reading comprehension dataset, requiring reasoning across multiple sentences. GSM8K is a collection of diverse grade school math word problems designed for multi-step reasoning. For each dataset, we randomly sample 100 test cases. We use F1-score as the quality metric and request delay (in seconds) as the efficiency metric across all datasets.

**LLMSTEER setting:** We apply LLMSTEER to Llama-8B, using an attention scaling factor $\alpha$. The two prefix prompts we used to generate different KV cache are provided in §A.1. Following the AutoPASTA paper, we adopt coarse-to-fine model profiling on LLMSTEER to search for the optimal set of layers and heads to steer that yields the most quality improvement.

**Baselines:** We compare LLMSTEER with three baselines: Llama-8B, Llama-70B (8-bit quantized) and Llama-8B with AutoPASTA. We also adopt coarse-to-fine model profiling on AutoPASTA. We

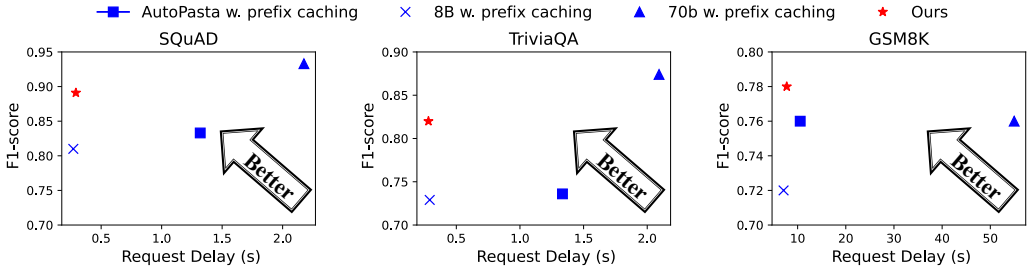


Figure 2: End-to-end Delay vs. Generation Quality. We assume KV cache is already in GPU memory when the request is being served.

assume prefix caching is enabled. That is, the KV cache of the prefix instruction and the context is pre-computed and has already been loaded to GPU memory when the request arrives.

**Implementation:** We build LLMSteer and all the baselines on top of Huggingface Transformers [22]. We implement AutoPASTA based on their paper as the code is not public yet. All experiments are conducted on two NVIDIA A40 GPUs.

### 4.2 Results

**Reduced request delay:** As shown in Figure 2, LLMSteer consistently achieves request delays close to the 8B baseline across all datasets. On the SQuAD, TriviaQA, and GSM8K datasets, LLMSteer's request delay is nearly negligible compared to the 8B baseline. Compared to the 70B model, LLMSteer is able to reduce request delay by 7.1–7.5×. Compared to AutoPASTA, LLMSteer can still reduce the request delay by 1.4–4.8×.

**Higher quality:** Figure 2 demonstrates the high generation quality of LLMSteer across all three datasets. On the SQuAD dataset, LLMSteer increases the F1 score by approximately 10% over the LLaMA-8B model with prefix caching and by about 7% over AutoPASTA. This improvement continues on the TriviaQA dataset, where LLMSteer achieves a similar F1 score increase, surpassing the 8B model by 12.5% and AutoPASTA by 11.4%. While our method does not outperform the 70B model on the SQuAD and TriviaQA datasets, it closely approaches the performance of the larger model without the substantial delay associated with the larger model. Notably, on the GSM8K dataset, our method even exceeds the performance of the 70B model.

**Understanding LLMSteer's improvement:** Unlike AutoPASTA, LLMSteer steers attention without relying on query information, operating with less information yet achieving superior performance. We test a query-dependent version of LLMSteer and observe smaller improvements compared to our query-independent method, suggesting that effective attention steering can be achieved without incorporating query information.

## 5 Limitations and Future work

We plan to extend to longer context lengths (e.g., >10k tokens) to fully explore LLMSteer's capabilities and to test it on models beyond Llama-8B to assess generalizability. We will also conducting an ablation study to quantify the contributions of the steering mechanism versus contextual re-reading. We will explore the limitations and capabilities of our method compared to traditional fine-tuning, especially considering context window length and model reasoning capabilities. Additionally, PagedAttention [14] and FlashAttention [7] could enhance efficiency. Investigating more fine-grained steering at the granularity of individual token pairs, beyond our current token-level attention upweighting, may further improve generation quality and will be explored in future work.

## 6 Conclusion

We introduce LLMSteer, a novel attention steering method aimed to improve the generation quality of LLMs by allowing the model to review the context multiple times. Specifically, LLMSteer narrows the response quality gap between small and large LLMs by 65.9% and deliver the responses faster by 4.8× compared to the state-of-the-art attention steering baseline.

## References


[1] 12 Practical Large Language Model (LLM) Applications - Techopedia. https://www.techopedia.com/12-practical-large-language-model-llm-applications. (Accessed on 09/21/2023).

[2] 7 top large language model use cases and applications. https://www.projectpro.io/article/large-language-model-use-cases-and-applications/887. (Accessed on 09/21/2023).

[3] Applications of large language models - indata labs. https://indatalabs.com/blog/large-language-model-apps. (Accessed on 09/21/2023).

[4] Anonymous. Model tells itself where to attend: Faithfulness meets automatic attention steering. In *Submitted to ACL Rolling Review - June 2024*, 2024. under review.

[5] Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762, 2024.

[6] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

[7] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

[8] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[9] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[10] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*, 2023.

[11] Chao Jin, Zili Zhang, Xuanlin Jiang, Fangyue Liu, Xin Liu, Xuanzhe Liu, and Xin Jin. Ragcache: Efficient knowledge caching for retrieval-augmented generation. *arXiv preprint arXiv:2404.12457*, 2024.

[12] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension, 2017.

[13] He Junqing, Pan Kunhao, Dong Xiaoqun, Song Zhuoyang, Liu Yibo, Liang Yuxin, Wang Hao, Sun Qianguo, Zhang Songxin, Xie Zejian, et al. Never lost in the middle: Improving large language models via attention strengthening question answering. *arXiv preprint arXiv:2311.09198*, 2023.

[14] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626, 2023.

[15] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[16] Bin Lin, Tao Peng, Chen Zhang, Minmin Sun, Lanbo Li, Hanyu Zhao, Wencong Xiao, Qi Xu, Xiafei Qiu, Shen Li, et al. Infinite-llm: Efficient llm service for long context with distattention and distributed kvcache. *arXiv preprint arXiv:2401.02669*, 2024.

[17] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts, 2023.

[18] Xiaoran Liu, Qipeng Guo, Yuerong Song, Zhigeng Liu, Kai Lv, Hang Yan, Linlin Li, Qun Liu, and Xipeng Qiu. Farewell to length extrapolation, a training-free infinite context with finite attention scope. *arXiv preprint arXiv:2407.15176*, 2024.

[19] Yuhan Liu, Hanchen Li, Kuntai Du, Jiayi Yao, Yihua Cheng, Yuyang Huang, Shan Lu, Michael Maire, Henry Hoffmann, Ari Holtzman, et al. Cachegen: Fast context loading for language model applications. *arXiv preprint arXiv:2310.07240*, 2023.

[20] Ruoyu Qin, Zheming Li, Weiran He, Mingxing Zhang, Yongwei Wu, Weimin Zheng, and Xinran Xu. Mooncake: Kimi's kvcache-centric architecture for llm serving. *arXiv preprint arXiv:2407.00079*, 2024.

[21] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, 2016.

[22] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[23] Jiayi Yao, Hanchen Li, Yuhan Liu, Siddhant Ray, Yihua Cheng, Qizheng Zhang, Kuntai Du, Shan Lu, and Junchen Jiang. Cacheblend: Fast large language model serving with cached knowledge fusion. *arXiv preprint arXiv:2405.16444*, 2024.

[24] Qingru Zhang, Chandan Singh, Liyuan Liu, Xiaodong Liu, Bin Yu, Jianfeng Gao, and Tuo Zhao. Tell your model where to attend: Post-hoc attention steering for llms, 2023.

[25] Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. Distserve: Disaggregating prefill and decoding for goodput-optimized large language model serving. *arXiv preprint arXiv:2401.09670*, 2024.

# A Appendix / supplemental material

## A.1 Prefix prompts

Prefix Prompts:

```
Prefix prompt 1:  Answer the question based on the given passages. Only
give me the answer and do not output any other words. The following are
given passages.

Prefix prompt 2: Respond to the query using only the provided
texts. Only return the answer. DO NOT return any extra words.
Below are the provided texts.
```

Figure 3: The two prefix prompts used in LLMSTEER to generate different key-value caches. By processing the same context with these varied prefixes, LLMSTEER encourages the model to interpret the context differently in each pass.

## A.2 LLMSTEER Algorithm

**Algorithm 1** LLMSTEER

**Input** A context $C$, a set of queries $\{q_1, q_2, \ldots, q_n\}$, an LLM $\mathcal{L}$, prefix prompts $P_1, P_2$, and a scaling factor $\alpha$.

1: Contextual re-reading: $A_i = \text{Attention}_{\mathcal{L}}(P_i \oplus C)$ for $i \in \{1, 2\}$;

2: Token selection: $T(l) = \bigcap_{i=1}^{2} \text{TopK}(\sum_h A_i^{l,h}, k)$, where $l$ denotes the layer and $h$ denotes the attention head;

3: Steering: $A_{\text{steered}} = \mathbf{M} \odot A$, where $\mathbf{M}$ denotes a weighting matrix;

**Output** Modified attention weights for context $C$: $A_{\text{steered}}$

## NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

**The checklist answers are an integral part of your paper submission.** They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: §1 discusses the paper's scope and contribution.

   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: Limitations are discussed in §5.

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: This paper contains no theoretical proof.

   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: We describe our design in §3 and algorithm in §A.2, which are sufficient to reproduce our main results.

   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: [No]

   Justification: We are using open datasets, and we will public our github repository upon acceptance.

   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: §4.1 discusses the choice of hyperparameters for this paper.

   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: While we don't include error bars, our F1 score remains significant, as we approach and even exceed the performance of a 70B model using an 8B model for inference.

   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: Hardware requirements are specified in §4.1.

   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: This paper raises no ethical issues.

   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: §1 discusses how our work allows people to save computation resources while still achieving the desired accuracy, with no negative 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: This paper only uses open datasets.

    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 assets used in this paper, including public datasets and open-source models, are properly cited in the References, and their licenses and terms of use have been fully respected.

    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: This 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: This paper does not involve crowdsourcing or 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: This paper does not involve crowdsourcing or 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.