Simple LLM Compression Recovery Using Dynamic Prompting with Theoretical Analysis

Duc Hoang
Apple
dhoang2@apple.com

Minsik Cho
Apple
minsik@apple.com

Thomas Merth
Apple
tmerth@apple.com

Mohammad Rastegari
Apple
mrastegari@apple.com

Zhangyang Wang*
The University of Texas at Austin
atlaswang@utexas.edu*

Abstract

Large Language Models (LLMs) need compression to be serviceable on hardware-limited devices, with the tradeoff being a reduction in performance, especially in natural language comprehension. As a direct consequence, parameter-efficient fine-tuning (PEFT) methods, previously used in task adaptation, are increasingly being utilized for post-compression performance recovery; however, the overall cost-benefit of these methods in this area is still unclear. In this work, we perform a comprehensive experimental study on various PEFT methods on Llama and OPT models with different compression approaches on a dedicated test suite aimed at measuring a model's performance, particularly in English comprehension. To analyze our results, we propose two conjectures that differentiate the nature of the compression damage on LLMs: one is that certain knowledge is **forgotten (or erased)** after LLM compression; the other presumes that knowledge is **internally displaced**. We found that the often-overlooked prompting holds a competitive advantage against more advanced approaches such as LoRA. Furthermore, we show we can extend prompting at minimal cost to latency by allowing multiple prompts to be dynamically allocated to different inputs at inference time, leading to even better or comparable post-compression performance recovery.

1 Introduction

Model compression techniques, such as quantization and sparsification, have since become increasingly popular for reducing the size of LLMs without significantly compromising their performance. Traditional approaches often involve post-compression re-training to mitigate performance losses [6]. More recent 'training-free' compression methods, like GPTQ [5] and SparseGPT [4], promise minimal impact on perplexity and standard task benchmarks. Nevertheless, recent studies [11] reveal that these compressed models still suffer from reduced effectiveness in *knowledge-intensive* language comprehension and generation tasks.

Parameter Efficient Fine Tuning (PEFT) methods [8, 7, 14] existed as a way to quickly and efficiently adapt pre-trained models for domain-specific tasks by training on task-specific calibration data. We note that it is quite trivial to extend such methods to performance recovery on pre-trained compressed models with the purpose of on device deployment. What is not clear is the cost-effectiveness of these approaches in this new setting, particularly when it is typical to have multiple adapters on-device. This inhibits many PEFT methods, particularly Low-rank Adapters (LoRA), from being merged with the underlying model's weights due to on-device memory constraints, thus suffering from increasing latency times.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

We also investigate into what transpires within a compressed model that leads to diminished performance on tasks? Is this knowledge permanently lost in the compression process, or is it merely obscured? Addressing these questions is not solely of theoretical interest; it has tangible implications for devising strategies to effectively counteract the impacts of compression on model knowledge. We hypothesize regarding the root cause of this performance degradation: the first posits that key knowledge is **forgotten (or erased)** as a consequence of LLM compression, necessitating a re-learning process with the addition of extra parameters [8]; the second hypothesis suggests that the knowledge is merely **internally displaced** within the LLM. This implies that strategic redirection of knowledge flow, potentially through input-side enhancements like prompting [23], could efficiently recover model accuracy. A more comprehensive exploration of these ideas is presented in Section 2.2.

Showcasing these two hypotheses, we conducted extensive experiments to test two central hypotheses of compressed LLM performance loss: “knowledge displaced” versus “knowledge forgotten” to validate these hypotheses effectively. Furthermore, we recognize the potential of prompting for model customization on the fly (simply pairing different inputs with different prompts within the same batch). We, therefore, introduce an easy trick to utilize multiple prompts by allowing dynamic prompt selection by input during inference. We called this trick Inference-time Dynamic Prompting or IDP.

We show that our trick with IDP is robust even at fairly short prompt lengths (Figure 6). Our investigation into layer-wise cosine similarity (Figure 5) further revealed that, compared to baseline attention patterns, prompt-tuning leads to significant divergences, whereas re-trained models tend to align more closely with the baseline, despite achieving similar outcomes.

In summary, our contributions are:

- We critically examine the impact of compression on LLMs’ knowledge, formally raising the conjectures of knowledge ’displacement’ versus ’forgetfulness’.
- We design experiments to endorse the hypothesis of **“knowledge displaced"** over **“knowledge forgotten"**. We also reveal a number of insights, including two different regimes of performance recovery.
- By extending on existing prompting approach, IDP achieves similar performance recovery to LoRA, at orders-of-magnitude lower parameter and latency overheads,

2 Background

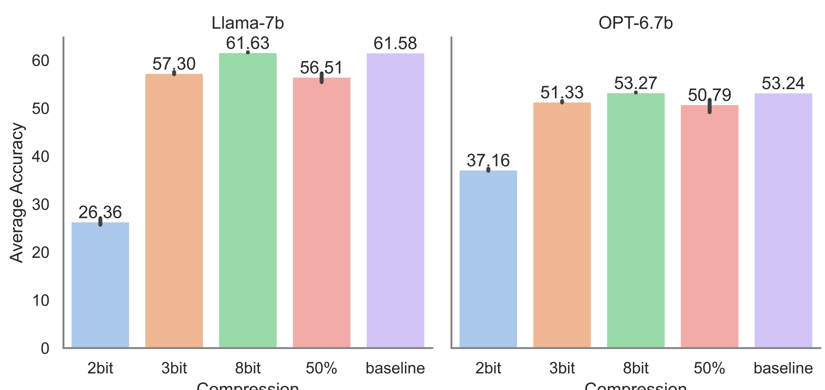

Figure 1: This figure presents a comparative analysis of the performance of compressed models using GPTQ for quantization and SparseGPT for pruning. The models were compressed leveraging either C4 or Wikitext datasets. Their average performance is depicted across a spectrum of nine tasks, each representing diverse knowledge domains.

To address the size and latency challenges of LLMs, we focus on compressing model parameters. Compressive techniques are generally divided into two categories: compression-aware training and post-training compression. We concentrate on post-training compression, especially relevant for extremely large models where full training or fine-tuning is prohibitively expensive.

Quantization reduces the model’s footprint by lowering the bit precision of its weights [5, 24, 22], which also accelerates inference due to less demanding computations. Sparsification, or pruning, involves selectively removing weight elements or masking activation values [4, 9, 10] to eliminate less important parts, thereby reducing computational overhead or enhancing throughput.

Using GPTQ and SparseGPT for model compression, Figure 1 shows a performance drop when lowering bit counts or parameters, except for int8 quantization. This aligns with claims that these methods are optimized for the largest LLMs [5, 4]. The limitations observed in smaller—but still substantial—LLMs underscore the need for additional post-compression performance improvements beyond parameter adjustment. We provided further details on our hypothesis in Appendix section B.

3 Testing Baseline methods

3.1 Basic settings

We utilize OPT-6.7b [26] and Llama-7b [20] as foundational models, both featuring an embedding size (denoted as **e**) of 4096. For compression, we apply **GPTQ** [5] and **SparseGPT** [4] to achieve 3-bit quantization and 50% pruning, respectively. In our discussion, we will primarily focus on the quantization approach, as the pruning process exhibits a very similar pattern. For more details regarding fine-tuning of LoRA and/or Prompt / Prefix-tuning, please see our Appendix.

3.2 Cost-effectiveness comparison between baseline methods

Table 1: This table summarizes the results for 3-bit GPTQ across all nine tasks for multiple fine-tuning baselines. World, Common, and Language are performance averages across tasks within those knowledge domains. Average is the average performance across all nine tasks.

Model	Type	Param	arcE	arcC	sciq	webqs	triviaqa	**World**	piqa	**Common**	hellaswag	lambada	winogrande	**Language**	**Average**
Llama-7b	—	—	71.46	37.71	92.60	17.96	33.02	50.55	76.01	76.01	53.11	68.58	67.48	63.06	57.55
Llama-7b	lora	4.4M	70.08	37.12	93.50	17.67	34.11	50.50	77.04	77.04	54.47	70.48	67.40	**64.12**	57.99
Llama-7b	lora	6.7M	71.09	36.69	93.00	17.47	34.73	50.60	76.44	76.44	54.55	70.23	67.09	63.96	57.92
Llama-7b	lora	8.9M	70.62	37.12	93.30	17.86	34.86	50.75	76.77	**76.77**	54.27	70.33	67.40	64.00	58.06
Llama-7b	prompt	0.1M	71.97	38.40	92.90	20.47	33.20	51.39	75.84	75.84	53.75	69.45	67.17	63.46	58.13
Llama-7b	prompt	0.2M	71.51	38.31	92.10	21.11	34.56	**51.52**	75.84	75.84	53.92	69.69	68.75	64.12	**58.42**
Llama-7b	prompt	0.4M	72.01	39.16	91.80	21.60	34.43	51.80	75.95	75.95	54.33	69.49	67.01	63.61	**58.42**
Llama-7b	ptune	3.1M	70.24	36.77	91.40	14.42	30.42	48.65	75.73	75.73	53.40	66.49	63.77	61.22	55.85
Llama-7b	ptune	6.5M	69.57	34.81	91.30	15.55	30.65	48.38	75.30	75.30	52.98	64.84	63.22	60.35	55.36
Llama-7b	ptune	13.1M	69.32	34.73	88.70	16.14	27.84	47.35	74.59	74.59	52.01	64.35	64.17	60.18	54.65
OPT-6.7b	—	—	64.77	29.01	89.40	9.50	17.90	42.12	75.24	75.24	48.57	65.34	63.54	59.15	51.47
OPT-6.7b	lora	4.7M	63.55	28.75	88.50	11.42	18.84	42.21	76.22	**76.22**	49.14	66.16	63.46	59.59	51.78
OPT-6.7b	lora	7.1M	64.27	29.01	89.20	11.07	18.95	42.50	75.90	75.90	48.89	66.50	64.40	**59.93**	52.02
OPT-6.7b	lora	9.4M	64.06	29.35	88.20	13.24	18.90	**42.75**	76.01	76.01	49.12	66.64	63.93	59.90	**52.16**
OPT-6.7b	prompt	0.1M	64.27	28.41	89.80	10.73	18.22	42.50	76.01	76.01	49.05	65.34	63.22	59.20	51.79
OPT-6.7b	prompt	0.2M	64.94	28.84	89.90	10.88	18.80	42.67	75.63	75.63	49.13	65.96	63.77	59.62	51.98
OPT-6.7b	prompt	0.4M	64.60	28.50	89.70	11.52	18.76	42.62	76.12	76.12	48.82	65.90	63.54	59.42	51.94
OPT-6.7b	ptune	3.1M	63.05	28.84	89.00	10.73	18.39	42.00	75.95	75.95	48.38	64.68	60.85	57.97	51.10
OPT-6.7b	ptune	6.5M	62.88	28.58	88.80	10.43	18.34	41.81	75.79	75.79	48.54	65.17	60.93	58.21	51.05
OPT-6.7b	ptune	13.1M	62.54	29.18	88.60	10.43	18.37	41.82	75.52	75.52	48.72	65.32	63.38	59.14	51.34

Table 2: This table summarizes the results for 50% unstructured sprase using SparseGPT across all nine tasks for multiple fine-tuning baselines. World, Common, and Language are performance averages across tasks within those knowledge domains. Average is the average performance across all nine tasks.

Model	Type	Param	arcE	arcC	sciq	webqs	triviaqa	**World**	piqa	**Common**	hellaswag	lambada	winogrande	**Language**	**Average**
Llama-7b	—	—	70.33	37.03	93.50	14.07	28.88	48.76	77.04	77.04	51.68	74.54	68.03	64.75	57.23
Llama-7b	lora	4.4M	71.04	37.63	91.90	14.47	33.28	49.66	76.99	76.99	53.98	70.95	67.17	64.03	57.49
Llama-7b	lora	6.7M	70.79	36.69	92.40	15.85	33.02	49.75	76.71	76.71	53.91	71.03	68.03	64.32	57.60
Llama-7b	lora	8.9M	71.04	37.88	92.10	14.86	32.85	**49.75**	77.20	77.20	54.01	70.70	68.03	64.25	**57.63**
Llama-7b	prompt	0.1M	71.59	38.74	93.10	15.21	29.66	49.66	77.04	77.04	53.48	71.24	67.48	64.07	57.50
Llama-7b	prompt	0.2M	71.38	38.57	92.20	14.86	30.48	49.50	77.15	77.15	53.75	71.76	67.09	64.20	57.47
Llama-7b	prompt	0.4M	71.38	38.31	92.60	14.86	30.86	49.60	77.31	77.31	53.97	70.99	67.17	64.04	57.49
Llama-7b	ptune	3.1M	63.17	32.59	88.20	11.81	24.60	44.07	72.63	72.63	50.18	64.97	56.91	57.35	51.67
Llama-7b	ptune	6.5M	67.17	34.90	88.70	12.11	24.74	45.52	74.76	74.76	50.36	65.59	59.12	58.36	53.05
Llama-7b	ptune	13.1M	65.78	31.40	87.20	11.61	21.97	43.59	74.21	74.21	49.77	63.87	59.43	57.69	51.69
OPT-6.7b	—	—	63.01	28.41	89.40	9.69	17.79	41.66	75.19	75.19	47.67	70.56	63.93	60.72	51.74
OPT-6.7b	lora	4.7M	64.06	29.61	88.60	10.58	18.26	42.22	75.57	75.57	48.52	66.60	64.33	59.82	51.79
OPT-6.7b	lora	7.1M	63.93	29.78	88.20	10.14	18.48	42.11	75.90	75.90	48.58	66.45	64.56	59.86	51.78
OPT-6.7b	lora	9.4M	62.84	29.86	88.30	10.33	18.79	42.02	75.41	75.41	48.76	66.49	65.19	60.15	51.77
OPT-6.7b	prompt	0.1M	63.09	28.58	90.70	12.30	18.75	42.68	75.14	75.14	48.40	68.78	63.69	60.29	52.16
OPT-6.7b	prompt	0.2M	63.68	29.44	90.60	12.40	18.36	42.90	75.24	75.24	48.58	67.86	63.22	59.89	52.15
OPT-6.7b	prompt	0.4M	64.06	29.27	89.60	12.80	19.12	**42.97**	75.19	75.19	48.49	67.49	63.61	59.86	**52.18**
OPT-6.7b	ptune	3.1M	61.03	28.50	86.90	13.09	19.46	41.80	72.74	72.74	46.44	62.08	59.67	56.06	49.99
OPT-6.7b	ptune	6.5M	63.01	29.86	88.00	9.40	17.10	41.47	75.08	75.08	47.84	64.89	61.80	58.18	50.78
OPT-6.7b	ptune	13.1M	60.94	29.10	88.60	13.53	19.95	42.42	73.39	73.39	46.93	62.68	62.19	57.27	50.81

In Table 1 and Table 2, we compare the performance and efficiency (in parameters) of our baseline methods. From the results, especially those highlighted in green, we draw several conclusions:

Performance Recovery Our tests show that most techniques provide modest performance gains in both quantization and pruning scenarios, except for prefix-tuning ("ptune"), which decreased performance across all tasks. Quantization generally recovers performance better than pruning. GPTQ shows an average improvement of 1%, while SparseGPT sees a smaller gain of 0.37%, likely due to the stricter limitations of parameter removal in pruning. Notably, prompting excels in both scenarios, offering higher-than-average recovery even with minimal parameters.

Knowledge Domain Adaptation Categorizing tasks into world knowledge, common reasoning, and language understanding, we find that redirection methods like prompting outperform integrated approaches like LoRA for world knowledge tasks. This suggests that input redirection effectively restores factual knowledge in compressed models. In contrast, tasks requiring nuanced understanding, such as language comprehension, benefit more from LoRA's additional parameters and external knowledge, though the performance difference remains small—under 0.2

4 Extending Prompting With Dynamic Prompting

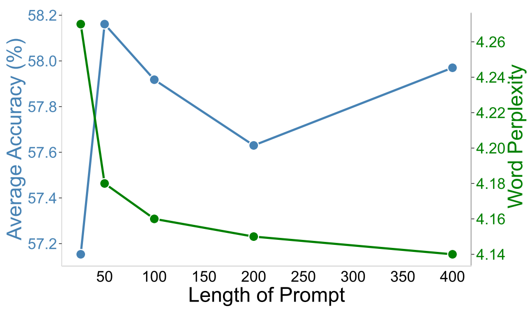

Figure 2: Using a 3-bit quantized Llama-7b model fine-tuned on C4 dataset, we contrast the average accuracy across nine tasks against its word's perplexity score across various prompt lengths. A longer sequence length improves perplexity but does not always sustain better performance.

Figure 2 reveals a growing perplexity-to-performance gap as prompts lengthen, reinforcing findings from [11] and highlighting the limitations of using perplexity alone as a performance measure. We find that longer prompts struggle to scale performance, suggesting that effective prompting depends more on aligning the right prompt to the right input than simply extending a single prompt. This mirrors ensemble methods but avoids the training-heavy approaches seen in [13] and [17], increasing training time and inference costs.

To address these issues, we propose Inference-time Dynamic Prompting (IDP), a technique for one-shot input-to-prompt matching that minimizes latency impact. IDP aligns prompts more accurately to inputs with minimal computational overhead, offering a notable boost in performance recovery for compressed models.

4.1 The IDP Methodology

In prompt tuning, we introduce an additional token sequence, termed as P, preceding the input sequence to improve the predicted output likelihood, $Pr_\theta(Y|[P;X])$, where θ are the static parameters. The sequence $P = p_1, p_2, ...p_n$ is defined by its learnable parameters, $\theta_p \in \mathbb{R}^{n \times e}$, with n being the prompt tokens count and e as their embedding size.

When we extend to a collection of m prompts, represented as $Z = P_1, P_2, ..., P_m$, each prompt has distinct trained parameters. Thus, the modified likelihood of Y becomes $Pr(Y|[Z;X])$. Let's consider the layer-wise token attention as $A \in \mathbb{R}^{b \times h \times tk \times tk}$, where tk stands for the combined token count of Z and X. For simplicity, we'll take b and h as one.

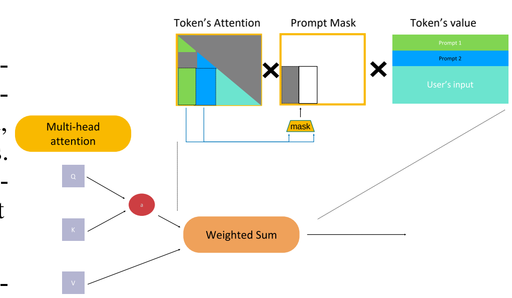

Figure 3: IDP is a straightforward alterations to the existing weighted sum operation. Using the existing attention matrix for prompt selection, IDP accomplishes its objectives without incurring any additional parameter costs.

To facilitate Inference-time Dynamic Prompting, we introduce two modifications to A: **Firstly**, we prevent interactions among the prompts in Z by setting their inter-attention, $A_{[Z_i:Z_j]}$, to $-\infty$. This constraint is twofold: Individual prompts have distinct training and do not share contextual relevance. Mixing them during inference can alter their inherent definitions, affecting the performance. Additionally, by eliminating inter-prompt attention, we can pre-cache the KV (Key, Value) for the prompts; this enables us to amortize the cost of processing. **Secondly**, for dynamic prompt selection, we measure the mean attention from input-to-prompt and select the prompt attracting the maximum overall input attention: $(\{\overline{A}_{[Z_i:X]} | \forall i \in [1, m])$. In the final phase of the self-attention mechanism, we use an attention mask to discard any unintended prompts, ensuring they do not modify the main input sequence and improve our inference latency. The process is depicted in Figure 3.

4.2 Comparing results with baseline prompting

Table 3: This table summarizes the results for 3-bit GPTQ across all nine tasks for multiple fine-tuning for IDP. For IDP we allow input to select between two different prompts one is 50 tokens and the other 100 tokens long. Note that the prompts used for IDP are fine-tuned independently with identical settings as previously described.

Model	Type	Param	arcE	arcC	sciq	webqs	triviaqa	**World**	piqa	**Common**	hellaswag	lambada	winogrande	**Language**	**Average**
Llama-7b	prompt	0.1M	71.97	38.40	92.90	20.47	33.20	51.39	75.84	75.84	53.75	69.45	67.17	63.46	58.13
Llama-7b	prompt	0.2M	71.51	38.31	92.10	21.11	34.56	51.52	75.84	75.84	53.92	69.69	68.75	64.12	58.42
Llama-7b	prompt	0.4M	72.01	39.16	91.80	21.60	34.43	51.80	75.95	75.95	54.33	69.49	67.01	63.61	58.42
Llama-7b	IDP	0.6M	72.43	39.76	92.50	19.83	36.39	**52.18**	76.44	76.44	53.96	70.25	67.56	63.92	**58.79**
OPT-6.7b	prompt	0.1M	64.27	28.41	89.80	10.73	18.22	42.50	76.01	76.01	49.05	65.34	63.22	59.20	51.79
OPT-6.7b	prompt	0.2M	64.94	28.84	89.90	10.88	18.80	42.67	75.63	75.63	49.13	65.96	63.77	59.62	51.98
OPT-6.7b	prompt	0.4M	64.60	28.50	89.70	11.52	18.76	42.62	76.12	76.12	48.82	65.90	63.54	59.42	51.94
OPT-6.7b	IDP	0.6M	64.18	28.67	90.40	11.96	19.05	**42.85**	76.17	76.17	49.03	66.82	63.22	59.69	**52.17**

In Table 3 we compare the performance of IDP with standard prompting in quantization settings. As observed with green highlight, this simple extension outperforms standard prompting in nearly all settings. Additionally, when we examined the link between the method's parameter size and performance, as detailed in Figure 4. Our findings show that IDP is much more efficient than LoRA for compression recovery. For example, when fine-tuning the Llama-7b model with QPTQ settings, LoRA's parameters range between 4.4 to 8.9 million, while IDP uses only around 0.8 million, leading to substantial space savings of 81% to 91% — a notable **20-fold** reduction.

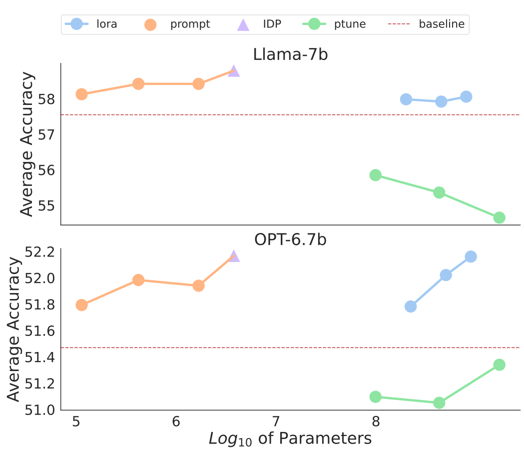

Figure 4: GPTQ models' average accuracy across nine tasks vs. number of trainable parameters.

Additionally, prompting tends to have a faster inference speed. Basic inference testing shows prompting incurs at most 0.37s versus LoRA's 0.62s for an input batch of 16 and a sequence length of 1024 – this is a substantial **60% improvement in speed**. Despite the smaller size, IDP generally sees a modest average improvement of 1% across the nine tasks evaluated. For further details on the performance and parameter size, refer to Table 1 and Figure 4, and our appendix provides a detailed explanation of how the total number of parameters was calculated for both LoRA and IDP.

Finally, we underscore the robustness of IDP's performance, irrespective of prompt in Figure 6. This figure reveals a variance of less than 1% in average accuracy performance, yet with a 5-fold reduction in token size. Notably, even with a modest average of 20 tokens, IDP adeptly facilitates performance recovery, surpassing the compressed baseline. This evidence positions IDP as not only efficient in parameter utilization but also as a resilient mechanism for enhancing performance in the wake of model compression. For further ablation studies on IDP, please refer to our Appendix.

5 Conjectures Analysis and Abalation Studies

5.1 Evaluating Knowledge Forgetfulness and Displacement

We employed a detailed visualization of the layer-wise attention and activation matrices to validate our hypothesis. Opting for cosine similarity over magnitude differences as our analytical tool, we

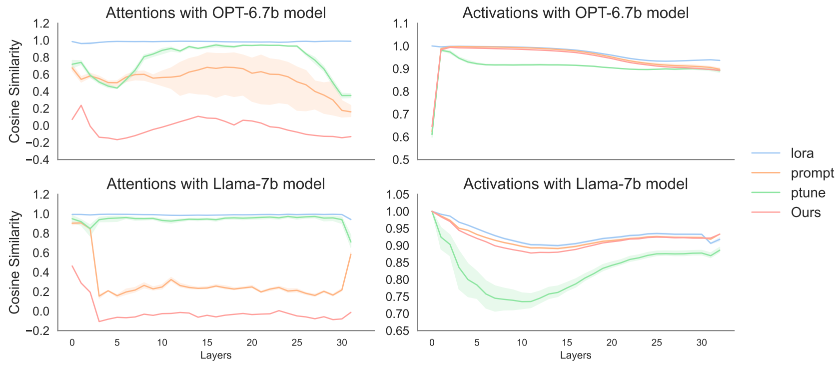

Figure 5: Cosine similarity compares the self-attention and token activation at each layer to an uncompressed baseline using different fine-tuning techniques. A higher cosine score means it's closer to the baseline.

aim to understand the distribution differences rather than magnitude. Our findings are presented in Figures 5, and 6, leading to several key observations:

① When compared to LoRA, the attention mechanism of both prompting/IDP markedly diverges from the baseline, hinting at a potential contextual redirection. Conversely, the activation patterns echo similarities with LoRA. Given that LoRA incorporates a residual network at every layer to maintain congruity and prompting only at the self-attention, this semblance is unexpected.

② These observations imply that prompting/IDP can tap into latent knowledge within the model. This is further supported by the data in Table 1 and Table 2, which show a propensity of prompting/IDP for tasks involving world knowledge. These tasks rely on the model's internal knowledge base, reinforcing our conclusion about the efficacy of prompting/IDP in accessing embedded information.

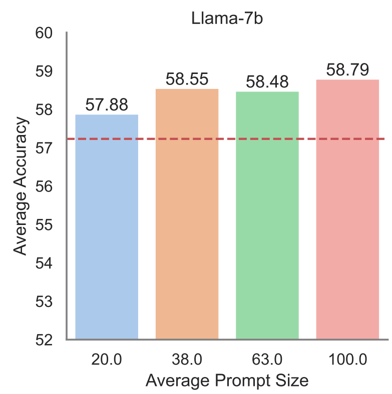

Figure 6: This figure illustrates the average performance over nine tasks using IDP. Results show IDP maintains relatively stable performance working with various average prompt sizes.

③ Additionally, IDP demonstrates remarkable consistency in information retrieval. As evidenced in Figure 6, it maintains stable performance across a range of prompt sizes. This suggests that even with fewer tokens, knowledge rerouting via IDP remains effective, opening avenues for future optimizations and refinements in its application.

④ Finally, our analysis of prefix-tuning indicates its tendency to align with the original attention patterns of the model. However, as shown in Figure 5, its activation patterns significantly deviate, hinting at a potential shortfall in redirecting knowledge.

These insights strongly endorse the notion of "redirection" as the more effective mechanism for recovering performance in compressed models.

6 Conclusion and Limitations

This study examines the impact of compression on LLMs and explores mitigation strategies through two hypotheses: knowledge forgotten and knowledge displaced. We focus on parameter-efficient methods like LoRA and introduce Inference-time Dynamic Prompting (IDP), a lightweight enhancement to traditional prompting. Our results show that IDP and prompting perform on par or better than LoRA while being smaller and faster. Visualization of embeddings indicates that instruction-based redirection effectively recovers lost knowledge. However, IDP requires pre-generated KV caches and is limited to prompts with high initial performance.

References

[1] Jonathan Berant, Andrew K. Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Conference on Empirical Methods in Natural Language Processing*, 2013.

[2] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language, 2019.

[3] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *ArXiv*, abs/1803.05457, 2018.

[4] Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. *ArXiv*, abs/2301.00774, 2023.

[5] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *ArXiv*, abs/2210.17323, 2022.

[6] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[7] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, 2019.

[8] J. Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685, 2021.

[9] Itay Hubara, Brian Chmiel, Moshe Island, Ron Banner, Seffi Naor, and Daniel Soudry. Accelerated sparse neural training: A provable and efficient method to find n: M transposable masks. *ArXiv*, abs/2102.08124, 2021.

[10] Itay Hubara, Yury Nahshan, Yair Hanani, Ron Banner, and Daniel Soudry. Accurate post training quantization with small calibration sets. In *International Conference on Machine Learning*, 2021.

[11] Ajay Jaiswal, Zhe Gan, Xianzhi Du, Bowen Zhang, Zhangyang Wang, and Yinfei Yang. Compressing llms: The truth is rarely pure and never simple, 2023.

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

[13] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Conference on Empirical Methods in Natural Language Processing*, 2021.

[14] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, abs/2101.00190, 2021.

[15] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

[16] Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset, August 2016.

[17] XIANGYU PENG, Chen Xing, Prafulla Kumar Choubey, Chien-Sheng Wu, and Caiming Xiong. Model ensemble instead of prompt fusion: a sample-specific knowledge transfer method for few-shot prompt tuning. In *The Eleventh International Conference on Learning Representations*, 2023.

[18] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[19] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019.

[20] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971, 2023.

[21] Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. *ArXiv*, abs/1707.06209, 2017.

[22] Guangxuan Xiao, Ji Lin, Mickael Seznec, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. *ArXiv*, abs/2211.10438, 2022.

[23] Zhaozhuo Xu, Zirui Liu, Beidi Chen, Yuxin Tang, Jue Wang, Kaixiong Zhou, Xia Hu, and Anshumali Shrivastava. Compress, then prompt: Improving accuracy-efficiency trade-off of llm inference with transferable prompt. *ArXiv*, abs/2305.11186, 2023.

[24] Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers, 2022.

[25] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019.

[26] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068, 2022.

A Fine-tuning and Hyper parameter settings

Baseline Methods: To recover performance, we employed three methodologies: prompt-tuning [13], prefix-tuning [14], and LoRA [8]. For consistent benchmarks across these techniques, we establish the following **fine-tuning criteria**: 1) The aggregate count of training tokens is limited to 40,960,000 tokens. Our decision on the total token count draws inspiration from [23]. 2) In alignment with [5], we use AdamW as our optimization algorithm. We choose a learning rate of 2e-4 with a weight decay set at 1e-5. All three methods are then fine-tuned using compressed LLM following the described settings with LLama-7b and OPT-6.7b. When fine-tune, we aim to keep the number of parameters as low as possible to be more suitable for on-device deployment. This means for input-side augmentation methods like Prompt/Prefix-tuning, we test with 26, 50 and 100 tokens prompts; LoRA we tested with 2, 3 and 4 feature dimensions.

Fine-tuning Dataset: We calibrate each baseline methods on C4 [18] and Wikitext [15] and select the results which maximize the test results. To maintain a controlled experimental space, our fine-tuning of various baseline techniques is restricted to the identical dataset used initially to calibrate our model compression.

Validation Tasks: To gauge the model's ability in English comprehension , we identify a suite of evaluation tasks that encapsulate three fundamental domains of cognition: world knowledge, common reasoning, and language understanding. Among the many available tasks, we distilled our focus to a curated list of nine that we deemed most representative.

For the domain of world knowledge, our chosen evaluative tasks were ARC-challenge & ARC-easy [3], SCIQ [21], WebQS [1], and TriviaQA [12]. Tapping into the breadth of language understanding benchmarks, we centered our attention on Hellaswag [25], Lambada [16], and WinoGrande [19]. Lastly, for common reasoning, we identified PIQA [2] as our touchstone. Notably, all the tasks we adopted are structured in a multiple-choice format.

B Our Conjectures

The primary drawback (of most) current advancements in LLM compression is their heavy reliance on **perplexity** as their primary metric to evaluate performance claims. Perplexity is a statistical measure of how confident a model is at predicting a text sample by quantifying the model's uncertainty, where lower perplexity is better. Recent work by **(author?)** [11] has demonstrated this strategy's flaw by showcasing significant performance degeneration on various LLMs at 50% sparsity yet having relatively good perplexity measures. This performance-to-perplexity gap necessitates comprehensive downstream tasks to validate the model's performance.

B.1 Forgotten, or Displaced? A Two-Way Argument

- Forgetfulness implies that the compression process irrevocably eliminates certain knowledge. Integrating an external knowledge source becomes essential to recuperate performance, as this process essentially replenishes the lost information.
- Displacement posits that the inherent knowledge within these models is not irrevocably erased but instead shifted internally, leading to the inefficacy of the established inference pathways. In this context, input-side augmentation or instructions are needed to "redirect" the internal self-attention. This enables the re-engagement of the pre-existing, albeit repositioned, knowledge in the compressed LLM, thereby aiding in the recuperation of its performance.

We position LoRA [8] and prompting to correlate respectively with our hypothesis on "**knowledge forgotten**" and "**knowledge displaced**." LoRA tackles "forgetfulness" by fundamentally altering the model's structure, specifically the weights in the self-attention and feedforward neural network (FFN) layers, thereby reintegrating knowledge lost due to compression. Prompting, in contrast, operates by subtly influencing the self-attention mechanism without changing the underlying weights, thus redirecting the model's existing but less accessible knowledge.

C Abalation Studies

Table 4: This table includes results for our Inference-time Dynamic Prompting strategy. To illustrate its effectiveness, we also include the results of the individual prompts used along with naive soft-prompts concatenation. 26 and 100 refers to the number of tokens in our prompts.

Model	arcE	arcC	sciq	webqs	triviaqa	**World**	piqa	**Common**	hellaswag	lambada	winogrande	**Language**	**Average**
OPT-6.7b/26	64.94	28.84	89.90	10.88	18.80	42.67	75.63	75.63	49.13	65.96	63.77	59.62	51.98
OPT-6.7b/100	64.02	27.90	89.50	11.32	18.37	42.22	76.39	**76.39**	48.81	65.42	63.22	59.15	51.66
OPT-6.7b/Concat	63.80	28.50	89.40	12.30	19.55	42.71	75.79	75.79	48.92	64.72	63.85	59.16	51.87
OPT-6.7b/IDP	64.18	28.67	90.40	11.96	19.05	**42.85**	76.17	76.17	49.03	66.82	63.22	**59.69**	**52.17**
Llama-7b/26	71.97	38.40	92.90	20.47	33.20	51.39	75.84	75.84	53.75	69.45	67.17	63.46	58.13
Llama-7b/100	71.51	38.31	92.10	21.11	34.56	51.52	75.84	75.84	53.92	69.69	68.75	64.12	58.42
Llama-7b/Concat	71.17	37.80	92.30	16.88	33.84	50.40	74.92	74.92	53.34	67.18	66.46	62.33	57.10
Llama-7b/IDP	71.63	38.65	92.60	21.60	33.84	**51.66**	76.01	**76.01**	53.97	69.67	68.98	**64.21**	**58.55**

We used IDP strategy with two distinct prompts of differing lengths, both trained using the same dataset to streamline our experimental parameters. We subsequently evaluated against our task benchmark, with the comprehensive findings cataloged in Table 4. In a complementary visual aid, Figure 7 highlights the percentage differences in performance against the baseline quantized models, providing an at-a-glance understanding of the performance gains across individual tasks.

Our analysis showed that IDP subtly enhances average accuracy. This is evident in our results with OPT and Llama models, where IDP showed a modest improvement of 0.5% and 0.42%, respectively. This contrasted with the outcomes of basic prompt concatenation, which yielded only a 0.16% increase and even a decrease of -1.03%. While these findings, detailed in Table 4, might not be groundbreaking, they highlight the potential of zero-shot input-to-prompt matching for compression recovery for various knowledge domains.

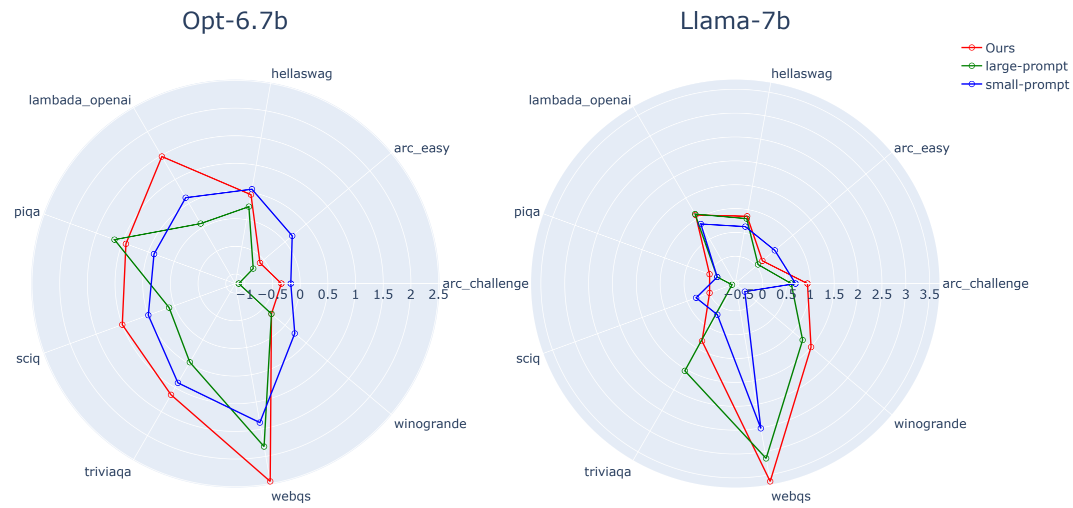

Figure 7: This graph shows the percentage performance improvement using two prompts at various lengths compared to a 3-bit quantized baseline for the OPT and LLama models. We've also showcased results from our IDP method, which selects prompts dynamically using the same two prompts. Small and Large correspond to 26 and 100 tokens respectively.

Further, in our examination of quantized foundation models, as shown in Figure 7, we noted areas where IDP demonstrated a slight but consistent superiority. Specifically, OPT models showed this incremental benefit in tasks such as Sciq, Triviqa, and Webqs, all falling within the world knowledge domain. Similarly, the Llama models exhibited slight improvements in tasks like Webqs, Arc, and Winogrand, with gains ranging between 1%-1.5%.

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:

   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:

   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:

   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:

   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:

   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:

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: [Yes]

Justification:

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:

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:

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: [No]

Justification: This paper poses no societal risk.

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:

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:

    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:

    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:

    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:

    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.