COMPOSING KNOWLEDGE AND COMPRESSION INTERVENTIONS FOR LANGUAGE MODELS

Arinbjörn Kolbeinsson[*][†]
University of Virginia

Tianjin Huang[*]
Eindhoven University of Technology

Shanghua Gao
Harvard University

Shiwei Liu
University of Oxford

Jonathan Richard Schwarz
Harvard University

Anurag Vaidya
MIT

Faisal Mahmood
Harvard University

Marinka Zitnik
Harvard University

Tianlong Chen
MIT

Thomas Hartvigsen
University of Virginia

ABSTRACT

Test-time interventions for language models aim to enhance factual accuracy, reduce harmful outputs, and improve model efficiency while avoiding excessive training costs. But existing interventions are developing independently. In practice, multiple interventions must be applied to the same model sequentially. We introduce *composable interventions*, a framework for studying the impacts of repeatedly intervening on the same language model. To showcase our framework, we compose interventions for two burgeoning interventions: *knowledge editing* and *model compression*. We find that compression undoes knowledge edits faster than it decays general model performance. We also find that compressing models makes them harder to edit and show that composing interventions impacts predicted logits.

1 INTRODUCTION

Large language models (LLMs) exhibit striking capabilities on important tasks like medical question–answering [35], finance [45], science [39], and entertainment [52]. But despite high performance, LLMs can still misbehave unpredictably, requiring correction. For example, LLMs notoriously generate *hallucinatory* [18], *harmful* [27, 10], and *factually incorrect* content [4]. Beyond unwanted behaviors, user requirements also changes over time. For example, regulations may arise [40], computational resources constrict, and copyrighted materials may be identified in pre-training data [8]. Without ways to quickly correct these issues, models can be left mis-calibrated, outdated, and biased, limiting their widespread responsible use [19].

To efficiently update LLMs, recent works have begun investigating *test-time interventions*. Such interventions aim to update pre-trained models to meet new requirements, while avoiding excessive or expensive training. Some popular interventions include model compression [54], detoxification [44, 51], and knowledge editing [26], However, interventions are largely advancing independently. In reality, we have many *simultaneous* requirements for our models (*e.g.,* factuality, harmlessness, privacy, efficiency), and each can shift over time. With the uptick in general-purpose models, we must treat each intervention as one of many options and develop methods that expect to be composed with others on the same model. Some works have begun studying how different training objectives interact, like privacy and fairness [25] or quantization and finetuning [48, 21]. However, to the best of our knowledge, we are the first to consider interactions between test-time interventions.

A key challenge in composing interventions is *ripple effects*: some interventions may hinder others. For example, quantization methods can alter activations [7], so knowledge editors trained on specific activations may suffer from distribution shift [31]. A second challenge when composing isolated

[*] equal contribution

[†] Correspondence: arinbjorn@virginia.edu

interventions is incompatible implementations alongside diverse evaluation settings. This challenge results in important, unanswered questions like *Will compression undo my knowledge edits?* and *Which debiasing strategy is best if I might compress my model later?* Answers require developing a unified evaluation framework.

We address these challenges by proposing a conceptual framework for *composable interventions*. Within this framework, we introduce two desirable features of successful composition: First, we consider the change in intervention success before and after applying *other* interventions. Second, we measure *order invariance*, considering interventions to be more composable if their success is unaffected by their order of application. We then showcase our framework with the first study composing *knowledge editing* and *model compression* interventions. We comprehensively compose a recent, popular editor with four compression methods on three datasets using two LLM architectures.

Our experiments elicit four findings. First, we compression consistently removes knowledge edits, while editing is less successful for compressed models. Second, compression undoes knowledge edits faster than it decays general model performance. Third, order invariance can depend heavily on the choice of compression algorithm.

Our contributions are as follows:

- We introduce *composable interventions*, a novel evaluation framework for language models that opens doors to studying important, practical questions.
- We showcase our framework with the first study of composability for knowledge editing and model compression.
- We discover new interactions between editing and compression, pointing to clear directions for future work in these areas.

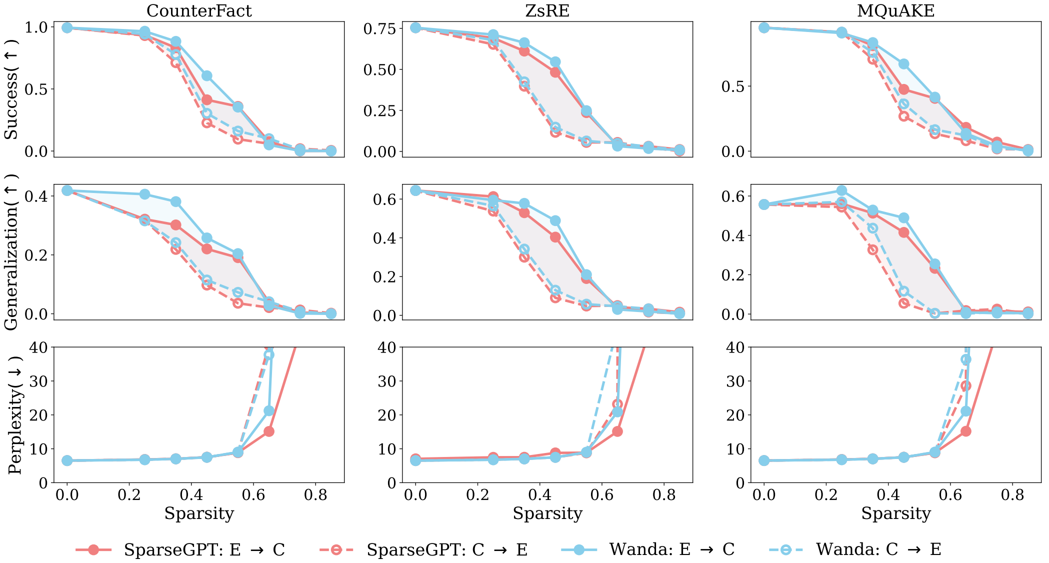

Figure 1: Composing editing with model pruning for Llama-2 on three editing datasets. E → C denotes editing followed by compression and vice versa. We plot interventions composed in each order and observe that pruning destroys edits and editing pruned models is less effective than pruning edited models.

2 Experimental Findings

In this section, we discuss our experiments that elucidate interactions between model compression and model editing. We first provide a brief overview of our experimental setting. Details on the framework itself are available in Appendix B.

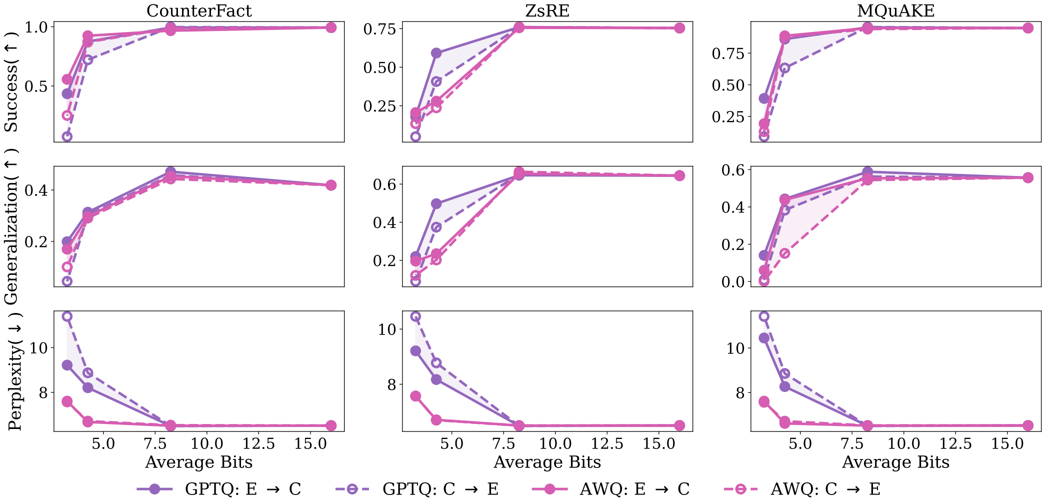

Figure 2: Composing editing with quantization for Llama-2 on three datasets. E → C indicates Editing followed by Compression and vice versa. Editing the quantized model is less effective than quantizing the edited model, though both struggle to make successful edits with fewer than 8-bit weights.

Datasets, Models, Interventions. We experiment with three widely-recognized model editing datasets, two public pre-trained LLMs, and five editing and compression interventions. The editing datasets include ZsRE[20], MQuAKE [53] and CounterFact [28]. Each is a question-answering dataset and the objective is to edit a model to output updated answers. We use the popular MEMIT [29] model editor, which applies batches of edits simultaneously. We use four state-of-the-art compression methods including two pruning methods: SparseGPT [6] and Wanda [37], and two quantization methods: GPTQ [7] and AWQ [23]. For pruning methods, we vary ratio of weights that are set to zero in the model, or the sparsity ratio. For quantization, we vary the number of bits. We intervene on the popular Llama-2-chat-7b [41] and the Pythia suite of models [1] to study the impacts of model size on intervention composability. In all experiments, we apply 50 randomly-selected edits, which we replicate 10 times to measure the variance in performance.

Intervention Evaluation Criteria. We evaluate the success of knowledge editing and model compression using standard metrics. For knowledge editing, we use three popular metrics. First, *edit success* measures whether the post-edit model successfully outputs the correct, edited response. Second, *edit generalization* measures how well the edit generalizes to other semantically-equivalent inputs. Following recent work, we compute generalization as the average edit success across a set of 10 holdout edit rephrasings. Third, *locality* measures the impact of editing on unrelated inputs. We compute locality as the average success on random, unedited samples. All editing metrics are computed as the F1 score on the correct output logits.

For model compression, we report the chosen sparsity ratio and bits for pruning and quantization, respectively. While compression styles are chosen based on practical requirements, we also facilitate their direct comparison by computing the average bits. We compute average bits for pruning by assuming weights that are zero are one bit, while unpruned weights are the original precision (16 in our experiments). We also report FLOPs and the compressed model's perplexity on the wikitext dataset [30].

2.1 Intervention Retention

We first study the impact of new interventions on the success of prior interventions. By measuring the edit metrics in both directions (editing followed by compression, and compression followed by editing), we can quantify their relative impact. For each combination of editor and model compression method, we report our three editing metrics alongside perplexity (PPL) on wikitext to quantify general model degradation. For each experiment, the only change to the model comes from the new intervention, so any changes in performance are directly due to intervention.

Our results on the CounterFact dataset are shown in Figure 1.

① **Compression decays previous edits.** We observe that compression significantly degrades the editing metrics significantly. For instance, in Figure 1 (E → C) we see that by the time model weights are 50% sparse, roughly half of the edits are unsuccessful. This trend occurs in both edit success and generalization, while perplexity remains relatively close to its original value. We also notice that Wanda composes with MEMIT better than SparseGPT does, as edit success and generalization achieve higher values across the range of sparsity ratios. This degradation is important because practical knowledge editing is conducted when critical knowledge *must* be updated. Therefore, compression in knowledge-intensive settings may pose risks when conducted on edited models.

② **Compression hinders editability.** We observe the same trend in the reverse order (C → E) as well: Editing compressed models is far less successful than editing non-compressed models, and editing sparser models leads to worse edit success, lower generalization, and lower perplexity upon heavy sparsification. Performance decays even more when composing in this order, which we observe for both pruning and quantization. This may imply that editors benefit from more free parameters.

Intervention	**CounterFact**				**zsRE**				**MQuAKE**			
	Edit Success		**Generalization**		**Edit Success**		**Generalization**		**Edit Success**		**Generalization**	
	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{A}$	$\mathcal{P}$
SparseGPT	**0.61**	3.87	**0.39**	5.70	**0.66**	4.94	**0.56**	5.27	0.74	3.66	0.81	4.95
Wanda	0.69	**3.58**	0.52	**5.49**	0.88	**4.37**	0.81	**4.72**	**0.68**	**3.51**	**0.79**	**4.80**
GPTQ	0.34	0.41	0.10	1.91	0.25	1.18	0.19	1.42	0.38	0.51	0.15	1.62

Table 1: Quantifying composability for each intervention. $\mathcal{A}$ measures the change in intervention success when applying interventions in each direction (edit → compress vs. compress → edit) as the area between the curves when varying the level of compression (see Figure 1). $\mathcal{P}$ measures the performance decay directly from composition using the area above the highest values in the curves from pairwise intervention comparisons. Lower values are better for both $\mathcal{A}$ and $\mathcal{P}$. GPTQ appears most composable across the board. SparseGPT is slightly more order invariant (lower $\mathcal{A}$) while overall Wanda decays model performance slightly less (lower $\mathcal{P}$), while outperforming SparseGPT on MQuAKE.

2.2 Edits decay faster than general LLM performance

Our results in Appendix Figure 5 show that the NLL for edits increases significantly faster than for general NLL. This indicates that edits are indeed degraded faster than general model performance, indicating a specific relationship between compression and editing performance. We find this relationship to be true for all three compression methods, though GPTQ decays NLL less than the other methods. An ideal composition would instead decay edits and general text at the same rate, being parallel to the gray identity line in the figure.

2.3 Interpreting intervention interactions

We have now established that editing and compression interventions hinder one another. Towards understanding why, we conduct a case study on how interventions alter Llama-2's behavior. Specifically, we study predicted logits from Llama-2 before and after each intervention in isolation and after their composition. We pick one real edit about the location of the next summer Olympic games. Before editing, Llama-2 consistently predicts "The next summer Olympics will be held in" → "Tokyo". But the correct answer is "Paris". As shown in Figure 3, we then plot Llama-2's top-30 predicted logits and observe that "Tokyo" is indeed the largest, while the correct "Paris" (in red) is the third largest. We then edit the model to confidently predict "Paris" and observe the "Paris" logit grows while all others drop substantially. We then compress this model using Wanda with 55% sparsity and observe that all logits shrink further, "Paris" included, which is surpassed by the special token in the eighth position. While "Paris" remains likely with respect to the other tokens, the compression has lowered its chance of generation. On the right side of Figure 3 we show the impacts of compressing first. Upon compression, Llama-2 still consistently predicts “Tokyo,” though all logits have shrunk. Still, “Tokyo” remains the most likely token. However efforts to edit this compressed model prove futile, and while the logits change and “Paris” becomes relatively more likely, the edit is unsuccessful. These observations hearken back to two of our main findings: 1) Knowledge edits often decay after compression, and 2) Compressed models are hard to edit.

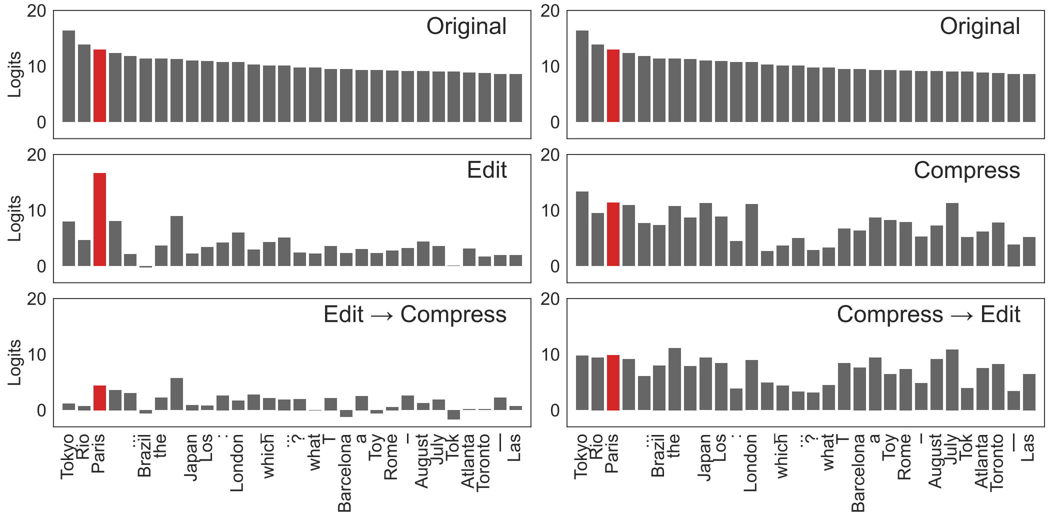

Figure 3: Tokens for Llama-2 prompted with “The next summer Olympics will be held in”. We edit and compress Llama-2 independently, then compose these interventions, and observe effects on Llama-2’s predictions. Editing spikes the correct token, though compression shrinks it again. Compression maintains similar tokens, though hinders editing success.

3 Limitations and Conclusion

We introduce a framework for composing test-time interventions for language models. Such interventions are an important step towards keeping expensively-trained models up-to-date over longer deployments with respect to factuality, efficiency, and social responsibility. Our framework lays the foundation for a new range of intervention methods that are *composable* with one another, proposing two metrics for measure composability of interventions.

However, there are inherent limitations due to the vast potential combinations of interventions, methods, and metrics unexplored within this study. We focus on a single model editor and a pair of interventions, acknowledging the existence of other strategies and the need for further research on their effectiveness across various settings. Additionally, our current research does not extend to the dynamics of massive models, highlighting the importance of future studies to understand the scalability and impact of interventions more comprehensively. We aim to inspire further exploration and community engagement in this emerging field.

Our composability metrics also pave the way for formal investigations and practical principles. In light of the rapid arrival of pre-compressed models, our findings also suggest tempering our expectations for direct composition of editing and compression. Finally, our framework is general and naturally extends to a broad range of test-time interventions. We hope to drive progress in machine learning by encouraging the study and development of test-time interventions that explicitly compose with one another.

REFERENCES

[1] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

[2] Davis Brown, Charles Godfrey, Cody Nizinski, Jonathan Tu, and Henry Kvinge. Robustness of edited neural networks. *arXiv preprint arXiv:2303.00046*, 2023.

[3] Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *arXiv preprint arXiv:2307.12976*, 2023.

[4] Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6491–6506, 2021.

[5] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022.

[6] Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, 2023.

[7] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations*, 2023.

[8] Michael Grynbaum and Ryan Mac. The times sues openai and microsoft over a.i. use of copyrighted work. *The New York Times*, 2023.

[9] Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. Model editing can hurt general abilities of large language models. *arXiv preprint arXiv:2401.04700*, 2024.

[10] Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. Toxigen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pp. 3309–3326. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.ACL-LONG.234. URL `https://doi.org/10.18653/v1/2022.acl-long.234`.

[11] Thomas Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with grace: Lifelong model editing with discrete key-value adaptors. In *Advances in Neural Information Processing Systems*, 2023.

[12] Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. In *Advances in Neural Information Processing Systems*, 2023.

[13] Rima Hazra, Sayan Layek, Somnath Banerjee, and Soujanya Poria. Sowing the wind, reaping the whirlwind: The impact of editing language models. *arXiv preprint arXiv:2401.10647*, 2024.

[14] Jason Hoelscher-Obermaier, Julia Persson, Esben Kran, Ioannis Konstas, and Fazl Barez. Detecting edit failures in large language models: An improved specificity benchmark. *arXiv preprint arXiv:2305.17553*, 2023.

[15] Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.

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

[17] Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-patcher: One mistake worth one neuron. In *International Conference on Learning Representations*, 2023.

[18] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

[19] Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. Challenges and applications of large language models. *arXiv preprint arXiv:2307.10169*, 2023.

[20] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pp. 333–342, 2017.

[21] Yixiao Li, Yifan Yu, Chen Liang, Pengcheng He, Nikos Karampatziakis, Weizhu Chen, and Tuo Zhao. Loftq: Lora-fine-tuning-aware quantization for large language models. In *International Conference on Learning Representations*, 2023.

[22] Zhoubo Li, Ningyu Zhang, Yunzhi Yao, Mengru Wang, Xi Chen, and Huajun Chen. Unveiling the pitfalls of knowledge editing for large language models. *arXiv preprint arXiv:2310.02129*, 2023.

[23] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration. *arXiv preprint arXiv:2306.00978*, 2023.

[24] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. *arXiv preprint arXiv:2305.11627*, 2023.

[25] Cleo Matzken, Steffen Eger, and Ivan Habernal. Trade-offs between fairness and privacy in language modeling. *arXiv preprint arXiv:2305.14936*, 2023.

[26] Vittorio Mazzia, Alessandro Pedrani, Andrea Caciolai, Kay Rottmann, and Davide Bernardi. A survey on knowledge editing of neural networks. *arXiv preprint arXiv:2310.19704*, 2023.

[27] Julia Mendelsohn, Ronan Le Bras, Yejin Choi, and Maarten Sap. From dogwhistles to bullhorns: Unveiling coded rhetoric with language models. *arXiv preprint arXiv:2305.17174*, 2023.

[28] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. In *Advances in Neural Information Processing Systems*, 2022.

[29] Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *International Conference on Learning Representations*, 2023.

[30] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2016.

[31] Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. In *International Conference on Learning Representations*, 2022.

[32] Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. Memory-based model editing at scale. In *International Conference on Machine Learning*. PMLR, 2022.

[33] Swarnadeep Saha, Peter Hase, and Mohit Bansal. Can language models teach weaker agents? teacher explanations improve students via theory of mind. *arXiv preprint arXiv:2306.09299*, 2023.

[34] Pratyusha Sharma, Jordan T Ash, and Dipendra Misra. The truth is in there: Improving reasoning in language models with layer-selective rank reduction. *arXiv preprint arXiv:2312.13558*, 2023.

[35] Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Le Hou, Kevin Clark, Stephen Pfohl, Heather Cole-Lewis, Darlene Neal, et al. Towards expert-level medical question answering with large language models. *arXiv preprint arXiv:2305.09617*, 2023.

[36] Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitry Pyrkin, Sergei Popov, and Artem Babenko. Editable neural networks. In *International Conference on Learning Representations*, 2020.

[37] Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023.

[38] Chenmien Tan, Ge Zhang, and Jie Fu. Massive editing for large language models via meta learning. In *International Conference on Learning Representations*, 2024.

[39] Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085*, 2022.

[40] The President of the United States. Executive order 14110 of october 30, 2023: Safe, secure, and trustworthy development and use of artificial intelligence. Federal Register, 2023. URL `https://www.govinfo.gov/content/pkg/FR-2023-11-01/pdf/2023-24283.pdf`. Published November 1, 2023.

[41] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[42] Peifeng Wang, Zhengyang Wang, Zheng Li, Yifan Gao, Bing Yin, and Xiang Ren. Scott: Self-consistent chain-of-thought distillation. *arXiv preprint arXiv:2305.01879*, 2023.

[43] Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. Knowledge editing for large language models: A survey. *arXiv preprint arXiv:2310.16218*, 2023.

[44] Johannes Welbl, Amelia Glaese, Jonathan Uesato, Sumanth Dathathri, John Mellor, Lisa Anne Hendricks, Kirsty Anderson, Pushmeet Kohli, Ben Coppin, and Po-Sen Huang. Challenges in detoxifying language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 2447–2469, 2021.

[45] Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*, 2023.

[46] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023.

[47] Mingxue Xu, Yao Lei Xu, and Danilo P Mandic. Tensorgpt: Efficient compression of the embedding layer in llms based on the tensor-train decomposition. *arXiv preprint arXiv:2307.00526*, 2023.

[48] 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 preprint arXiv:2305.11186*, 2023.

[49] 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. *Advances in Neural Information Processing Systems*, 35:27168–27183, 2022.

[50] Lu Yin, You Wu, Zhenyu Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Mykola Pechenizkiy, Yi Liang, Zhangyang Wang, and Shiwei Liu. Outlier weighed layerwise sparsity (owl): A missing secret sauce for pruning llms to high sparsity. *arXiv preprint arXiv:2310.05175*, 2023.

[51] Charles Yu, Sullam Jeoung, Anish Kasi, Pengfei Yu, and Heng Ji. Unlearning bias in language models by partitioning gradients. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 6032–6048, 2023.

[52] Shanshan Zhong, Zhongzhan Huang, Shanghua Gao, Wushao Wen, Liang Lin, Marinka Zitnik, and Pan Zhou. Let's think outside the box: Exploring leap-of-thought in large language models with creative humor generation. *arXiv preprint arXiv:2312.02439*, 2023.

[53] Zexuan Zhong, Zhengxuan Wu, Christopher D Manning, Christopher Potts, and Danqi Chen. Mquake: Assessing knowledge editing in language models via multi-hop questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[54] Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. A survey on model compression for large language models. *arXiv preprint arXiv:2308.07633*, 2023.

A Methods Background

We specifically compose knowledge editing and model compression interventions and introduce them each here.

Knowledge Editing. Knowledge editors aim to address factuality decay in LLMs: As the world changes, some facts an LLM learned during training become inaccurate [26, 43]. For example, a model trained before 2020 would still predict `The latest pandemic in the US is` → "`Swine Flu`" until updated to predict "`COVID`". There are four general approaches to model editing: 1) Memory-based methods that cache knowledge [32, 11], 2) Locate-then-Edit methods that selectively finetune parameter subsets [28, 29], 3) Hypernetwork methods that predict new model weights [36, 31, 38], and 4) Prompt editors that add facts to prompts [53, 3]. Most editors update singular facts [31, 4, 36, 28], though recent works apply many simultaneously [29, 32, 38]. Others works embrace the practical need for *sequential* edits [17, 11]. Recent works have also begun investigating unintended impacts of editing on pre-trained models [9, 14, 22, 13, 2, 12]. However, these works study impacts on model performance, not interactions with other types of interventions.

We specifically focus on MEMIT [29]. This editor represents locate-then-edit editors, which are popular because they are agnostic to LLM architecture and enable simple, in-place LLM updates by leaving the model's architecture unchanged.

Model Compression. Large Language Models (LLMs) have achieved remarkable performance across various tasks, but their deployment is often hampered by substantial computational and memory demands. To mitigate these challenges, plenty of strategies have been proposed, as outlined in recent work by Zhu et al. [54]. Notable among these are quantization [7, 23, 46], network pruning [6, 37, 50, 24], knowledge distillation [42, 33, 15], and low-rank factorization [16, 47, 34]. This study concentrates on network pruning and quantization, two prevalent post-training strategies that effectively reduce memory overhead and accelerate model inference.

Quantization involves lowering the bit-precision of model parameters, effectively shrinking model size and expediting inference. Various post-training quantization methods have been specifically tailored for LLMs. Early works, such as ZeroQuant [49] and LLM.int8 [5], focused on fine-tuning quantization granularity, showing promising results predominantly at higher bit-widths, like 8-bit. However, more recent innovations like GPTQ [7] and AWQ [23] have successfully pushed the boundaries by reducing bit-width to as low as 3 or 4 bits per weight, with negligible performance degradation.

Concurrently, network pruning aims to eliminate redundant components, such as weights or channels, thereby reducing the model's footprint. Several pruning techniques have been developed for LLMs, shifting from traditional methods that typically required re-training—a significant challenge given the size of LLMs—to more recent approaches that drop this step. Techniques like SparseGPT [6] and Wanda [37] have demonstrated that it is possible to maintain a significant proportion of an LLM's performance even when discarding approximately 50% of its weights. The OWL technique [50] further advances these techniques, achieving higher levels of sparsity by reallocating sparsities across layers based on the activation outlier ratio.

We choose to employ cutting-edge quantization methods QPTQ [7] and AWQ [23] alongside pruning methods SparseGPT [6] and Wanda [37].

B A FRAMEWORK FOR COMPOSABLE INTERVENTIONS

Let us assume we are interested in interventions on a class of functions $\mathcal{F}$ parameterized by neural networks, i.e., $\mathcal{F} = \{f_\theta : \theta \in \Theta\}$ where Θ is the parameter space. An intervention Ω is an operator $\Omega : \Theta \to \Theta'$ defined over the neural network parameter space and designed to instill desirable properties into the resulting function, e.g., factuality, compactness, or privacy. An operator's success may be defined by measuring its effect according to some criterion $\ell : \Theta \times \mathcal{D} \to \mathbb{R}$ computed over data $\mathcal{D}$ (e.g., to test biasedness or check an intervention's effect on the negative log-likelihood of the model). For ease of notation, we drop this explicit dependence on the data. In designing new interventions, researchers seek to find an Ω^* (e.g., the best possible compressor) as measured on a number of openly available models $\{f_{\theta_i}\}_{i=1}^N$:

$$\Omega^* = \arg\min_{\Omega} \frac{1}{N} \sum_{i=1}^{N} \ell(\Omega(f_{\theta_i})) \tag{1}$$

In practice, an intervention Ω may be controlled by a parameter γ_Ω to vary its strength (e.g., a sparsity level for network pruning), yielding $\Omega(\cdot, \gamma_\Omega)$, in which case Ω^* achieves the best result for each setting of γ_Ω. We can categorize the difference between the best possible intervention and an actual obtainable intervention on a specific model f_θ through a performance metric $\mathcal{P}(\Omega, f_\theta)$. Intuitively, a perfect intervention would achieve optimal performance for each setting of γ_Ω (e.g., we achieve 100% edit success for all compression levels). Assuming $\ell, \gamma_\Omega \in [0, 1]$ (appropriately normalized), the distance of an intervention Ω from the optimal intervention is the area between the two curves (i.e., the shaded red area in Figure 4):

$$\mathcal{P}(\Omega, f_\theta) := 1 - \int d\gamma_\Omega \ell(\Omega(f_\theta), \gamma_\Omega) \tag{2}$$

where Ω^* with achieve the optimal value $\mathcal{P}(\Omega^*, f_\theta)$ and we can compare two interventions $\mathcal{P}(\Omega_1, f_\theta), \mathcal{P}(\Omega_2, f_\theta)$ over a range of settings.

Note that our discussion so far only considered a single intervention in isolation. In this paper, we instead argue that there is a gap between theory and practice and hence propose the study of the case of interacting interventions. Hence, we are interested in the behavior of two or more interventions (Ω, Ψ). Apart from performing well according to their respective criteria $\mathcal{P}(\Omega, f_\theta), \mathcal{P}(\Psi, f_\theta)$, we characterize the effect of these interventions on each other. An intuitive way of doing so would be to compute the above performance metric for the composition of operators $\mathcal{P}(\Psi \circ \Omega, f_\theta)$ and $\mathcal{P}(\Omega \circ \Psi, f_\theta)$, where $\mathcal{P}$ is defined either wrt. to the objective ℓ_Ω or ℓ_Ψ. This gives us an intuitive measure of how applying one operator after the other affects performance.

An especially desirable property for any two operators is composability (order-invariance), in which case we apply two operators in either order and receive approximately the same result. We can define composability as follows:

Definition 1. (**ϵ_Ω-composable interventions**): Let Ω, Ψ be interventions on a function class $\mathcal{F}$. Ω, Ψ are set to be ϵ_Ω-composable for a specific function $f_\theta \in \mathcal{F}$ wrt. to the error tolerance ϵ_Ω associated with the success of Ω as evaluated by ℓ_Ω:

$$|\ell_\Omega(\Psi(\Omega(f_\theta))) - \ell_\Omega(\Omega(\Psi(f_\theta)))| \leq \epsilon_\Omega \tag{3}$$

In other words, we consider the operators composable if the difference of performance in ℓ_Ω is below a threshold. This definition allows for a trivial case when either operator is the identity, i.e., $\Psi : \theta \mapsto \theta$, in which the interventions are clearly composable. Another degenerate case would be an ϵ-order invariant intervention, which weakens Ψ's effectiveness wrt. to ℓ_Ψ to an extent which makes it near meaningless (e.g., compression reducing memory cost only marginally). To avoid such cases, we may thus define a constrained version:

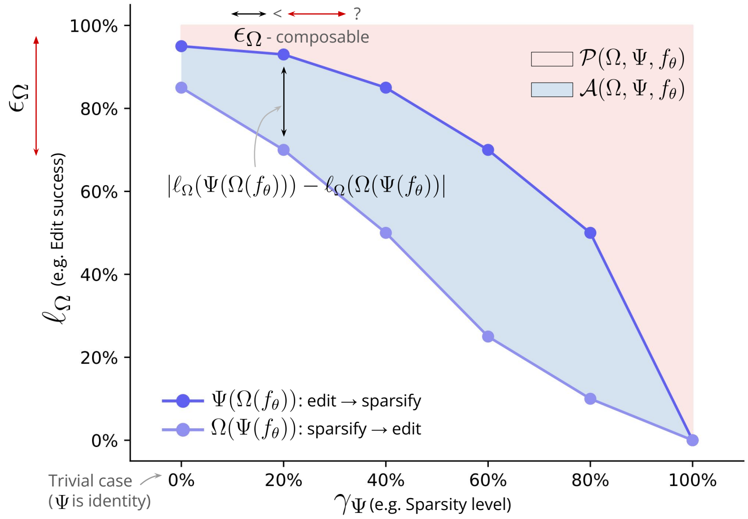

Figure 4: Composable intervention criteria on a hypothetical example measuring the composability of model editing and compression wrt. edit success. Ω : Model editing, Ψ : Network pruning/quantization.

Definition 2. (Constrained ϵ_Ω-composable interventions): Let ϵ_Ω be our error threshold threshold wrt. ℓ_Ω and let ϵ_Ψ be a second error threshold wrt. to ℓ_Ψ. A contained composable intervention satisfies:

$$|\ell_\Omega(\Psi(\Omega(f_\theta))) - \ell_\Omega(\Omega(\Psi(f_\theta)))| \leq \epsilon_\Omega \tag{4}$$

subject to $\ell_\Psi(\Omega(\Psi(f_\theta))) \leq \epsilon_\Psi$.

Two measures passing this test are set to be ϵ_Ω-composable interventions at ϵ_Ψ. Note that we can generalize this definition straightforwardly for more than two interventions by introducing additional constraints.

As before, Ψ may be controlled by a parameter γ_Ψ to vary its strength. Thus, if we vary γ_Ψ, we get a range of outcomes, some of which may be composable (i.e., pass the test when measured against ϵ_Ω) while others may fail. In order to provide a holistic characterization of the interventions behavior, we can define a measurement between the two curves obtained by varying γ_Ψ over N settings, i.e. $[\gamma_\Psi^{(i)}, \ell_\Omega(\Omega(\Psi(f_\theta, \gamma_\Psi)))^{(i)}]_{i=1}^N$ and $[\gamma_\Psi^{(i)}, \ell_\Omega(\Psi(\Omega(f_\theta), \gamma_\Psi))^{(i)}]_{i=1}^N$. Analogous to our definition of $\mathcal{P}$, a natural choice would be the area $\mathcal{A}(\Omega, \Psi, f_\theta)$ between the two curves (i.e. the shaded blue area in Figure 4):

$$\begin{aligned} &\mathcal{A}(\Omega, \Psi, f_\theta) := \\ &\quad \int d\gamma_\Psi |\ell_\Omega(\Psi(\Omega(f_\theta), \gamma_\Psi)) - \ell_\Omega(\Omega(\Psi(f_\theta, \gamma_\Psi)))| \end{aligned} \tag{5}$$

where $\mathcal{A}(\Omega, \Psi, f_\theta) = 0$ indicates perfect and larger values indicate worse composability. This allows the easy comparison of alternative interventions Ψ_1, Ψ_2 (e.g., two compressors) wrt. to Ω (model editing) by comparing their $\mathcal{A}$ scores, provided both methods have been evaluated across the same settings for γ_Ψ. In the case of more than two interventions, the respective lines (one for each unique permutation) form a hypercube, the volume of which characterizes their composability.

There is one remaining subtlety in the extension of our previous definition of $\mathcal{P}$ to the case of multiple interventions: In practice, for a specific γ_Ψ, a user will choose whatever order of intervention achieves better results, i.e., either $\ell_\Omega(\Psi(\Omega(f_\theta), \gamma_\Psi))$ or $\ell_\Omega(\Omega(\Psi(f_\theta, \gamma_\Psi)))$. For $\mathcal{P}(\Omega, \Psi, f_\theta)$ to take this into account, we can simply compute the area between the optimal intervention and the elementwise max (at each γ_Ψ):

$$\begin{aligned} &\mathcal{P}(\Omega, \Psi, f_\theta) := \\ &\quad 1 - \int d\gamma_\Psi \max(\ell_\Omega(\Psi(\Omega(f_\theta), \gamma_\Psi)), \ell_\Omega(\Omega(\Psi(f_\theta, \gamma_\Psi)))) \end{aligned} \tag{6}$$

In summary, the two metrics $\mathcal{A}$ & $\mathcal{P}$ give us a holistic view of both composability and absolute performance of two interventions, allowing for a principled and fair comparison. Figure 4 illustrates the various defined criteria on a hypothetical example.

C Extended Results

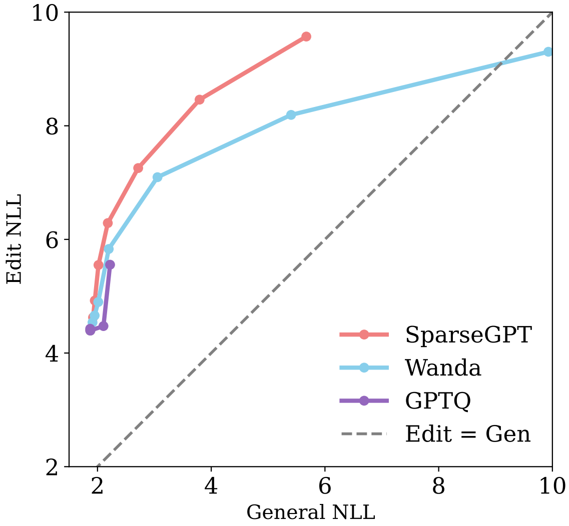

Figure 5: Comparing length-normalized Negative Log-Likelihood (NLL) decay in edits to NLL on general text at various levels of compression (editing → compression). The gray dashed line represents hypothetical equal decay in edit and general NLL. The sharp rise in edit NLL for each compression method indicates edits are forgotten first.