LoFiT: Localized Fine-tuning on LLM Representations

Fangcong Yin
The University of Texas at Austin
fangcongyin@utexas.edu

Xi Ye
Princeton University
xi.ye@princeton.edu

Greg Durrett
The University of Texas at Austin
gdurrett@cs.utexas.edu

Abstract

Recent work in interpretability shows that large language models (LLMs) can be adapted for new tasks in a learning-free way: it is possible to intervene on LLM representations to elicit desired behaviors for alignment. For instance, adding certain bias vectors to the outputs of certain attention heads is reported to boost the truthfulness of models. In this work, we show that localized fine-tuning serves as an effective alternative to such representation intervention methods. We introduce a framework called **Lo**calized **Fi**ne-**T**uning on LLM Representations (LoFiT), which identifies a subset of attention heads that are most important for learning a specific task, then trains offset vectors to add to the model's hidden representations at those selected heads. LoFiT localizes to a sparse set of heads ($3\% - 10\%$) and learns the offset vectors from limited training data, comparable to the settings used for representation intervention. For truthfulness and reasoning tasks, we find that LoFiT's intervention vectors are more effective for LLM adaptation than vectors from representation intervention methods such as Inference-time Intervention. We also find that the localization step is important: selecting a task-specific set of attention heads can lead to higher performance than intervening on heads selected for a different task. Finally, across 7 tasks we study, LoFiT achieves comparable performance to other parameter-efficient fine-tuning methods such as LoRA, despite modifying 20x-200x fewer parameters than these methods.[1]

1 Introduction

A significant body of work has studied how to localize model behavior within pre-trained Transformer language models [5, 22, 21, 17, 33]. The localized modules can be used with lightweight interventions to modify that behavior [21, 35, 31, 8, 27, 15, 45]. These approaches are unified in recent work on *representation intervention* [45, 41], which adds offset vectors into various layer representations of a model to achieve desired behaviors. Computing these vectors from model activations is reported to require less data and compute than fine-tuning approaches [15].

At the same time, a distinct line of work has established the effectiveness of *parameter-efficient fine-tuning (PEFT)* methods [11, 1, 9, 16] by updating only parts of the pre-trained weights. Very recently, new PEFT methods have been proposed to change models' representations rather than

[1]An extended version of this paper is available at https://arxiv.org/abs/2406.01563. Our code is available at https://github.com/fc2869/lo-fit.

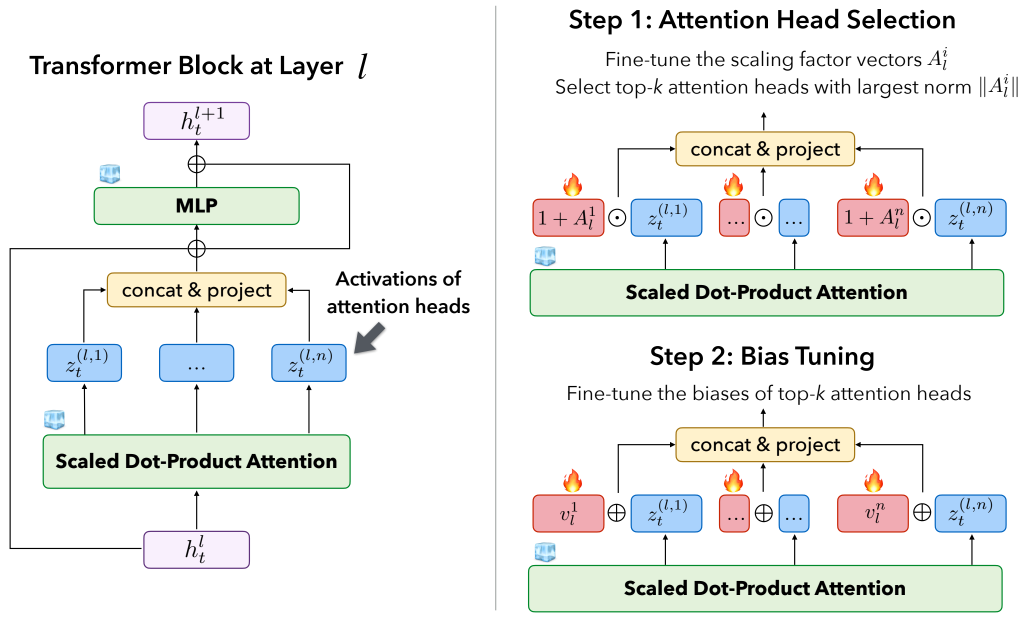
Transformer Block at Layer l
h_t^{l+1}
MLP
concat & project
Activations of attention heads
$z_t^{(l,1)}$
...
$z_t^{(l,n)}$
Scaled Dot-Product Attention
h_t^l
Step 1: Attention Head Selection
Fine-tune the scaling factor vectors A_l^i
Select top-k attention heads with largest norm $\|A_l^i\|$
$1+A_l^1$
$1+A_l^n$
Step 2: Bias Tuning
Fine-tune the biases of top-k attention heads
v_l^1
v_l^n

Figure 1: LoFiT methodology. LoFiT freezes all pre-trained weights of a transformer language model and appends two sets of lightweight parameters to the LLM representations with two steps: Attention Head Selection and Bias Tuning. Only the tuned biases are used in the final model.

weights [39, 41], in line with the motivation of representation intervention. However, these methods typically do not use any explicit localization step.

In this work, we investigate whether the idea of localization from representation intervention can be useful for fine-tuning LLMs. We propose **Lo**calized **Fi**ne-**T**uning on LLM Representations (LoFiT; Figure 1). LoFiT first selects a subset of attention heads to modify for the target task by fine-tuning scaling factors on the model's attention head outputs and selecting the heads with the largest norm of learned scaling weights. Then, we perform a localized fine-tuning step to learn offset vectors added to these heads' representations, which gives our final model.

We compare LoFiT with representation intervention methods on truthfulness and reasoning tasks. We show that LoFiT is substantially more effective than representation intervention baselines, including Inference-Time Intervention [15, ITI] and Representation Engineering [45, RepE]. Moreover, localization is important for LoFiT. Across tasks and models at different scales, using the set of heads specialized to a particular task improves the final fine-tuning step.

Finally, we compare LoFiT against existing PEFT methods, specifically LoRA [11], RED [39], and ReFT [41]. LoFiT is on par with these across various settings for different LLMs, despite using 20x-200x fewer learned parameters.

The main contributions of this work are the following: (1) We introduce LoFiT, a localized fine-tuning method that achieves competitive downstream performance on truthfulness and reasoning tasks by modifying the representations of a small number of attention heads. (2) We show the benefits of localization to particular sets of heads across tasks and models, suggesting that interpretability methods can be combined with PEFT for strong performance.

2 LoFiT: Localized Representation Fine-Tuning

Preliminaries: Localized Representation Intervention Consider a decoder-only Transformer model of L layers and a hidden size of d, following notations from [36, 6]. At time step t, a Transformer block of layer $l \in [1, L]$ takes as input the hidden vectors of all previous time steps $h_{\leq t}^l$ (where $h_i^l \in \mathbb{R}^d$) and outputs a hidden representation h_t^{l+1} for the next layer $l+1$: $h_t^{l+1} = h_t^l + \text{MultiHead}(h_{\leq t}^l) + \text{MLP}(h_t^l + \text{MultiHead}(h_{\leq t}^l))$. MultiHead represents the multi-head attention outputs with H attention heads of head dimension d_{head} after a linear projection W^O into the residual stream: $\text{MultiHead}(h^l_{\leq t}) = \text{concat}(z_t^{(l,1)}, ..., z_t^{(l,i)})W^O$. Here, $z_t^{(l,i)} \in \mathbb{R}^{d_{head}}$ for $i \in [1, H]$ represents the activations output by the ith attention head at layer l. $z_t^{(l,i)}$ is essentially the output representation of a single attention head.

We define localized intervention $I = \langle T, V \rangle$. Here, $T = \{(l_1, i_1) \dots (l_K, i_K)\}$ is a set of K attention heads to intervene on selected by some scoring function $\mathcal{S} : (\mathbb{Z}^+, \mathbb{Z}^+) \to \mathbb{R}$, where l_j denotes the target layer and i_j denotes the target head at the target layer. $V = \{v^i_l\}_{(l,i) \in T}$ is the set of offset vectors, where $v^i_l \in \mathbb{R}^{d_{head}}$. During *inference time*, V will be added to offset the targeted attention activations. That is, $z_t^{(l,i)}$ will be overwritten as a linear combination with the offset vectors $z_t^{(l,i)} \leftarrow z_t^{(l,i)} + \alpha v^i_l$ if $(l, i) \in T$ where α is a constant. Several representation intervention methods in literature can be cast in this framework [15, 42, 27]; details can be found in Appendix A.1.

LoFiT: Localized Representation Fine-Tuning Localized Representation Fine-tuning (LoFiT) also aims to learn a set of the offset vectors V in the representation space targeted at a localized set of heads in T. As illustrated in Figure 1, our approach also follows a two-step framework.

Step 1. Attention Head Selection: The first step of LoFiT incorporates a learnable vector of scaling factors $A^i_l \in \mathbb{R}^{d_{head}}$ for any head at layer $l \in [1, L]$ and index $i \in [1, H]$ of the pre-trained LLM. A^i_l can be viewed as scaling each neuron in the activations $z_t^{(l,i)}$ of the head i at layer l.

With the scaling factors, during a forward pass, the activation $z_t^{(l,i)}$ is rescaled by $z_t^{(l,i)} \leftarrow (1 + A^i_l) \odot z_t^{(l,i)}$. We freeze all pre-trained weights and learn A^i_l end-to-end with the cross-entropy loss on a small amount of labeled data from the task of interest. During training, A^i_l is initialized from $\mathcal{N}(0, \sigma_A)$. We regularize the optimization with an L1 normalization with a scaling hyperparameter λ to encourage sparsity for better head selection. We score each head using the norm of the learned A, i.e., $\mathcal{S}(i, j) = \|A^i_l\|$. A large score $\mathcal{S}(i, j)$ indicates a stronger intervention is needed for a particular head. Therefore, we select the set of top-K attention heads as T. K and σ_A are adjustable hyperparameters. [2]

Step 2. Bias Tuning: The second step of LoFiT learns the offset vectors V added to the hidden representations of each attention head in T. We freeze all pre-trained weights and add learnable parameters V $= \{v^i_l \,|(l, i) \in T\}$ such that during a forward pass, the activation $z_t^{(l,i)}$ will have an added offset bias vector: $z_t^{(l,i)} \leftarrow v^i_l \oplus z_t^{(l,i)}$. During training, we learn V with the cross-entropy loss on the same training data. v^i_l is initialized from $N(0, \sigma_v)$ and σ_v is an adjustable hyperparameter.

At inference time, the learned biases V are added to the hidden representations of the target attention heads T in the same way as a forward pass during training.

3 Experimental Setup

We evaluate LoFiT on truthfulness, multi-hop reasoning, and counterfactual reasoning tasks, which are common settings for evaluating interpretability-motivated methods [15, 45], with the following datasets: **TruthfulQA** [18], **CLUTRR** [30], and **MQuAKE** [43]. We focus on the common low-data condition of representation intervention methods: for each dataset, we sample 500 training points or fewer. Details of datasets and evaluation metrics can be found in Appendix B.1.

Baselines: We compare LoFiT with representation intervention methods, including Inference-time Intervention [15, ITI], and Representation Engineering [45, RepE]. Details are discussed in Appendix A.1.

Implementation: We use Llama 2-7B [34], Llama 2-13B, and Gemma-7B [32] for experiments. For all experiments in Section 4, we select 3% attention heads for each model for LoFiT and ITI and use the top-1 layer for RepE.[3] Details can be found in Appendices C.

[2]Further discussion on the hyperparameters of LoFiT can be found in Appendix D.

[3]We discuss the optimal choice of the number of heads to use for LoFiT in Appendix F.

Table 1: Test accuracy of LoFiT against representation intervention baselines. LoFiT beats baselines by a large margin across all settings on all models.

		TruthfulQA MC1	TruthfulQA MC2	MQuAKE EM	CLUTRR EM	*Average*
Gemma-7B	0-shot	31.5	48.1	23.3	60.2	40.8
	ITI	29.7	50.0	49.5	67.3	49.1
	RepE	38.5	53.7	54.2	66.9	53.3
	LoFiT	**60.5**	**79.4**	**69.4**	**86.7**	**74.0**
Llama 2-7B	0-shot	28.4	43.4	18.9	64.7	38.8
	ITI	33.4	49.5	34.5	68.2	46.4
	RepE	46.8	64.4	37.9	66.2	53.8
	LoFiT	**58.1**	**75.8**	**73.4**	**89.7**	**74.3**
Llama 2-13B	0-shot	29.1	44.3	25.4	64.7	40.9
	ITI	32.7	47.7	40.3	72.7	48.3
	RepE	43.7	66.4	40.6	64.1	55.1
	LoFiT	**56.7**	**76.0**	**76.2**	**89.7**	**74.6**

Table 2: Bias tuning accuracy using attention heads from LoFiT against other head selection methods. For TruthfulQA, we report MC1 accuracy. Best results are bolded. Fine-tuning the representations of LoFiT heads leads to consistently better performance than other head selection methods.

		Probe-layers	Random	Bias-based	ITI-heads	LoFiT
Gemma-7B	TruthfulQA	46.7	55.2	56.7	56.7	**60.5**
	MQuAKE	**71.2**	65.2	71.0	69.2	69.4
	CLUTRR	83.3	86.0	86.7	84.8	**86.7**
	Average	67.1	68.8	71.5	70.2	**72.2**
Llama 2-7B	TruthfulQA	52.6	46.6	52.3	57.1	**58.1**
	MQuAKE	72.8	71.9	72.2	**73.8**	73.4
	CLUTRR	86.7	88.0	88.2	86.7	**89.7**
	Average	70.7	68.8	70.9	72.5	**73.7**
Llama 2-13B	TruthfulQA	31.5	54.3	40.1	56.5	**56.7**
	MQuAKE	74.1	67.1	71.1	74.6	**76.2**
	CLUTRR	85.6	**90.7**	90.9	87.6	89.7
	Average	63.7	70.7	67.4	72.9	**74.2**
Average		67.2	69.4	69.9	71.9	**73.4**

4 Results: Effectiveness of Localization

As shown in Table 1, LoFiT outperforms the representation intervention baselines by a large margin across all settings. Our learning-based localized intervention can be much more effective than learning-free alternatives, even with limited training data.

Importance of LoFiT Heads: To validate the effectiveness of our localization method, we compare it with other head selection methods by tuning the biases V for a set of heads T selected by the following baseline methods: random sampling; probing for layers; bias-based selection; ITI head selection. Details of each baseline can be found in Appendix A.2. Table 2 shows that selecting attention heads based on LoFiT outperforms all other head localization methods in most settings. This shows that LoFiT is a strong localization method for finding some optimal sets of attention heads for learning the interventions.

Effectiveness of Localization: We are also interested in the localization across tasks: is localizing task-specific attention heads important for learning downstream tasks in the LLM representation space? To answer this question, we change the sets of head to tune, specifically using the selected heads from different tasks, and compare the accuracy of LoFiT using different-task heads with the same-task accuracy. Figure 2 shows that localized interventions by LoFiT are task-specific for TruthfulQA and MQuAKE: tuning biases on same-task heads is consistently better than tuning on different-task heads by a significant margin across all models. For TruthfulQA, using different-task heads can lead to as large as 10% absolute performance degradation, suggesting that the selected heads might have very specific functions. On the contrary, CLUTRR does not require a task-specific set of heads to achieve good performance, possibly because LLMs can be easily adapted without task-specific localization for this relatively easy task.

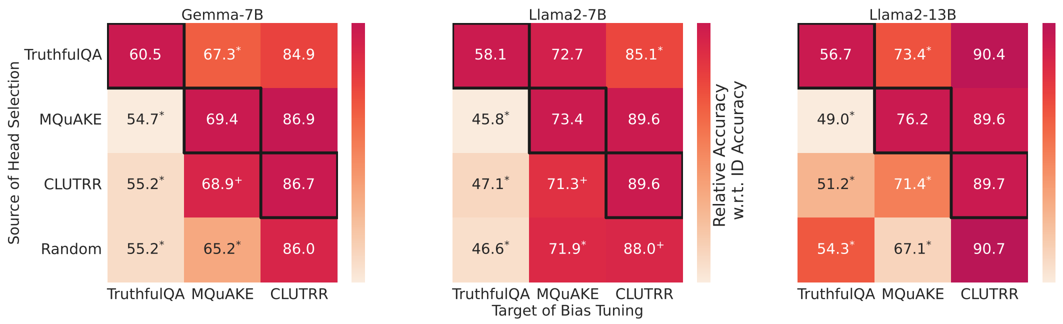

Figure 2: Test accuracy of using LoFiT heads learned from a different task. Colors reflect relative accuracy with respect to using same-task heads, with same-task heads (diagonals) representing 100% relative accuracy. Different-task results with $*$ are significantly lower than the same-task result at the significance level of 0.05 with a paired bootstrap test and results with $+$ are significantly lower at the level of 0.1. For TruthfulQA, we report MC1 accuracy. Across models, task-specific heads consistently outperform different-task heads for TruthfulQA and MQuAKE.

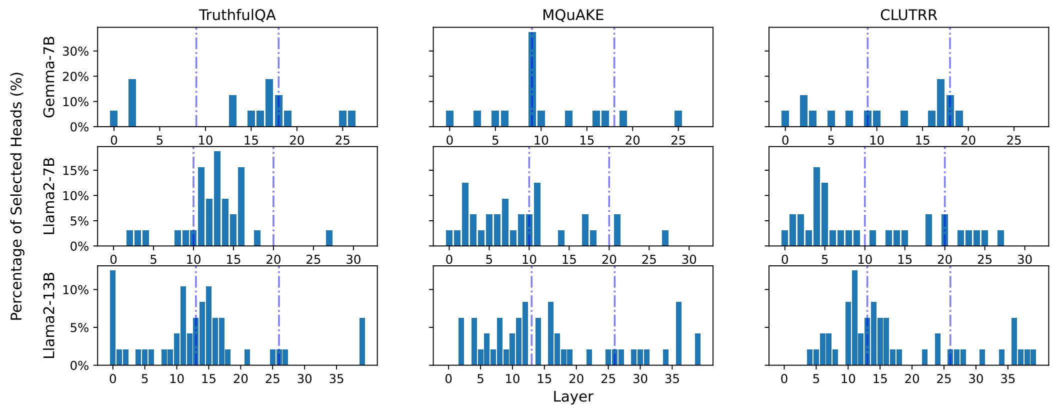

Figure 3: Distribution of LoFiT heads over layers for different tasks. Across tasks, LoFiT heads are often located in different parts of the model, and layer selection differs between Llama2 and Gemma.

Distribution of LoFiT Heads: We further examine where the task-specific heads reside in LLMs: do the localized sets overlap, or tend to select heads from similar layers? Figure 3 shows that task-specific distribution of LoFiT heads over layers peak in some contiguous sets of layers within the same model rather than being widely spread across layers or concentrated in a single layer, indicating there is not a single set of heads best for fine-tuning across all tasks.

5 Comparison with PEFT Methods

We also compare LoFiT with existing PEFT methods, **LoRA** [11], **RED** [39], and **ReFT** [41]. Details of the baselines and implementations can be found in Appendix A.3.

Our focus is on datasets where models can be steered to have the right behavior, e.g., by interpretability-motivated methods from Section 3. However, for completeness, we include a representative selection of datasets from [12] for PEFT methods that cover commonsense reasoning, open-book/closed-book QA, and mathematical reasoning: SIQA [28], ARC-Challenge [3], BoolQ [2], and SVAMP [24]. We stick to the low-data setting by sampling 500 training examples or fewer; details can be found in Appendix B.2 and Appendix C. For all the following experiments, we select 10% attention heads for each model: $K = 48$ for Gemma-7B, $K = 96$ for Llama 2-7B, and $K = 160$ for Llama 2-13B.

Table 3 compares LoFiT results with PEFT methods. The results show that across settings from Section 3, LoFiT outperforms ReFT on average, and gives results comparable to LoRA with 200x

Table 3: Test accuracy of LoFiT and state-of-the-art PEFT methods. Results are averaged over 2 random seeds and the best results are bolded. For LoFiT, we select 10% attention heads. With 20x - 200x fewer learned parameters, LoFiT is comparable to PEFT models across models and even outperforms them in some settings.

Evaluation Datasets for Interpretability-motivated Methods

		Learned Params		TruthfulQA		MQuAKE	CLUTRR	*Average*
		# Params	% Params	MC1	MC2	EM	EM	
Gemma-7B	RED	229K	0.003%	60.5	78.2	71.9	90.2	75.2
	LoRA	3.21M	0.04%	**60.8**	**78.9**	**75.7**	**92.9**	**77.1**
	ReFT	2.10M	0.03%	46.8	69.5	75.4	90.2	70.5
	LoFiT	12K	0.0001%	60.2	78.3	73.7	91.2	75.9
Llama 2-7B	RED	262K	0.004%	**56.3**	**76.8**	75.9	89.1	**74.5**
	LoRA	4.19M	0.06%	52.6	71.5	75.0	91.1	72.6
	ReFT	2.10M	0.03%	48.6	68.8	**77.2**	87.6	70.5
	LoFiT	12K	0.0002%	56.3	74.5	73.7	**92.0**	74.1
Llama 2-13B	RED	409K	0.003%	53.7	74.1	76.4	**93.8**	74.5
	LoRA	6.55M	0.05%	56.3	75.4	**76.4**	92.2	75.1
	ReFT	3.28M	0.03%	53.2	72.9	75.4	93.6	73.8
	LoFiT	20K	0.0002%	**57.0**	**76.8**	76.3	92.2	**75.6**

Evaluation Datasets for PEFT Methods

		Learned Params		Commonsense and QA			Math	*Average*
		# Params	% Params	SIQA	ARC-c	BoolQ	SVAMP	
Llama 2-7B	0-shot	-	-	48.3	45.9	77.4	36.7	52.1
	RED	262K	0.004%	**60.4**	**51.5**	**82.2**	55.3	**62.4**
	RED (Half)	131K	0.002%	55.5	49.7	77.6	52.7	58.9
	LoRA	4.19M	0.06%	59.8	50.6	80.8	**56.7**	62.0
	ReFT	2.10M	0.03%	58.1	51.0	80.1	52.7	60.5
	LoFiT	12K	0.0002%	59.3	50.3	80.9	52.3	60.7
Llama 2-13B	0-shot	-	-	50.3	49.4	81.7	46.7	57.0
	RED	409K	0.003%	65.3	64.2	82.5	63.0	68.7
	RED (Half)	204K	0.0015%	59.1	65.5	82.3	60.7	66.9
	LoRA	6.55M	0.05%	**66.4**	**67.6**	84.1	**63.7**	**70.4**
	ReFT	3.28M	0.03%	62.5	64.1	84.3	60.7	67.9
	LoFiT	20K	0.0002%	63.4	65.4	**85.4**	62.3	69.1

fewer learned parameters and to RED with 20x fewer learned parameters. On commonsense and mathematical reasoning datasets, LoFiT falls slightly short of LoRA and RED, but outperforms ReFT and the parameter-matched version of RED. This is probably because these tasks require resurfacing of memorized world knowledge from the pre-trained model and the benefit of having more parameter updates outweighs the benefit of localization. In addition, we also analyze the data efficiency (Appendix G), performance of open-ended generation (Appendix H), and out-of-domain generalization (Appendix I) of LoFiT with results in the Appendices. These results show that localization can enable further targeting of parameter updates without impacting task accuracy.[4]

6 Conclusion

In this work, we introduce LoFiT, a two-step localized fine-tuning method for LLMs that selects a subset of attention heads and learns task-specific offset vectors to be added to the hidden representations of the targeted attention heads. We show the strong downstream performance of LoFiT on tasks involving truthfulness and reasoning, outperforming representation intervention methods (ITI and RepE) and matching strong PEFT methods (LoRA) with fewer learned parameters. We also show that LoFiT is effective at localizing task-specific attention heads for learning downstream tasks, showing that interpretability insights have a part to play in improving LLM fine-tuning processes.

[4]Our two-step procedure requires fitting additional parameters in the first stage. However, even accounting for these parameters, which are not used in the final model, we still modify fewer parameters than PEFT methods: half the parameters of RED and 3% of LoRA even if counting both sets.

Acknowledgments

Thanks to Eunsol Choi, Chenglei Si, Zeyu Leo Liu, and other members of the TAUR lab for helpful discussion and suggestions. This work was partially supported by NSF CAREER Award IIS-2145280, the NSF AI Institute for Foundations of Machine Learning (IFML), the Alfred P. Sloan Foundation, a gift from Amazon, and a grant from Open Philanthropy.

References

[1] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

[2] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

[3] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *ArXiv*, abs/1803.05457.

[4] Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. 2024. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298.

[5] Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge Neurons in Pretrained Transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

[6] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A Mathematical Framework for Transformer Circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

[7] Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

[8] Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. 2024. Patchscopes: A Unifying Framework for Inspecting Hidden Representations of Language Models. *arXiv preprint arXiv:2401.06102*.

[9] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a Unified View of Parameter-Efficient Transfer Learning. In *International Conference on Learning Representations*.

[10] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

[11] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. *Proceedings of the International Conference on Learning Representations (ICLR)*.

[12] Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023. LLM-Adapters: An Adapter Family for Parameter-Efficient Fine-Tuning of Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, Singapore. Association for Computational Linguistics.

[13] Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

[14] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

[15] Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-Time Intervention: Eliciting Truthful Answers from a Language Model . *Advances in Neural Information Processing Systems*. ArXiv:2306.03341.

[16] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *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)*, pages 4582–4597, Online. Association for Computational Linguistics.

[17] Tom Lieberum, Matthew Rahtz, János Kramár, Neel Nanda, Geoffrey Irving, Rohin Shah, and Vladimir Mikulik. 2023. Does Circuit Analysis Interpretability Scale? Evidence from Multiple Choice Capabilities in Chinchilla. *arxiv preprint arXiv:2307.09458*.

[18] Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring How Models Mimic Human Falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

[19] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *The Seventh International Conference on Learning Representations*.

[20] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods. `https://github.com/huggingface/peft`.

[21] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and Editing Factual Associations in GPT. *Advances in Neural Information Processing Systems*, 36. ArXiv:2202.05262.

[22] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2022. In-context Learning and Induction Heads. *arXiv preprint arXiv:2209.11895*.

[23] Abhishek Panigrahi, Nikunj Saunshi, Haoyu Zhao, and Sanjeev Arora. 2023. Task-Specific Skill Localization in Fine-Tuned Language Models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

[24] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

[25] Marcin Pietron and Maciej Wielgosz. 2020. Retrain or not Retrain? – Efficient Pruning Methods of Deep CNN Networks. *arXiv preprint arXiv:2002.07051*.

[26] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

[27] Mansi Sakarvadia, Aswathy Ajith, Arham Khan, Daniel Grzenda, Nathaniel Hudson, André Bauer, Kyle Chard, and Ian Foster. 2023. Memory Injections: Correcting Multi-Hop Reasoning Failures During Inference in Transformer-Based Language Models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 342–356, Singapore. Association for Computational Linguistics.

[28] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

[29] Pratyusha Sharma, Jordan T. Ash, and Dipendra Misra. 2023. The Truth is in There: Improving Reasoning in Language Models with Layer-Selective Rank Reduction. *arXiv preprint arXiv:312.13558*.

[30] Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L. Hamilton. 2019. CLUTRR: A Diagnostic Benchmark for Inductive Reasoning from Text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4506–4515, Hong Kong, China. Association for Computational Linguistics.

[31] Nishant Subramani, Nivedita Suresh, and Matthew Peters. 2022. Extracting Latent Steering Vectors from Pretrained Language Models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 566–581, Dublin, Ireland. Association for Computational Linguistics.

[32] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. 2024. Gemma: Open Models Based on Gemini Research and Technology. *arXiv preprint arXiv:2403.08295*.

[33] Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. 2024. Function Vectors in Large Language Models. In *Proceedings of the 2024 International Conference on Learning Representations*.

[34] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes,

Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288*.

[35] Alexander Matt Turner, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid. 2023. Activation Addition: Steering Language Models Without Optimization. *arxiv preprint arXiv:2308.10248*.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. *Advances in Neural Information Processing Systems*.

[37] Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. 2020. TRL: Transformer Reinforcement Learning. `https://github.com/huggingface/trl`.

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

[39] Muling Wu, Wenhao Liu, Xiaohua Wang, Tianlong Li, Changze Lv, Zixuan Ling, Jianhao Zhu, Cenyuan Zhang, Xiaoqing Zheng, and Xuanjing Huang. 2024. Advancing Parameter Efficiency in Fine-tuning via Representation Editing. *arxiv preprint arXiv:2402.15179*.

[40] Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. 2024. Retrieval Head Mechanistically Explains Long-Context Factuality. *arXiv preprint arXiv:2404.15574*.

[41] Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. 2024. ReFT: Representation Finetuning for Language Models. *arxiv preprint arXiv:2404.03592*.

[42] Qingru Zhang, Chandan Singh, Liyuan Liu, Xiaodong Liu, Bin Yu, Jianfeng Gao, and Tuo Zhao. 2023. Tell Your Model Where to Attend: Post-hoc Attention Steering for LLMs. *arxiv preprint arXiv:2311.02262*.

[43] Zexuan Zhong, Zhengxuan Wu, Christopher Manning, Christopher Potts, and Danqi Chen. 2023. MQuAKE: Assessing Knowledge Editing in Language Models via Multi-Hop Questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15686–15702, Singapore. Association for Computational Linguistics.

[44] Max Zimmer, Megi Andoni, Christoph Spiegel, and Sebastian Pokutta. 2024. PERP: Rethinking the Prune-Retrain Paradigm in the Era of LLMs. *arXiv preprint arXiv:2312.15230*.

[45] Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Troy Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, Zico Kolter, and Dan Hendrycks. 2023. Representation Engineering: A Top-Down Approach to AI Transparency. *ArXiv*, abs/2310.01405.

A Details of Baselines

A.1 Representation Intervention Baselines

We compare LoFiT against two main representation intervention methods: Inference-time Intervention [15, ITI], and Representation Engineering [45, RepE].

ITI is a representation intervention method to improve LLM truthfulness. ITI localizes T by training a logistic regression classifier to predict the truthfulness of responses using $z_{t=-1}^{l,i}$ from the last timestep as the input features. It then selects top-K heads where the scoring function $\mathcal{S}(l, i)$ is the probe accuracy on the validation set. Finally, ITI extracts the difference in representations of heads in T between truthful and untruthful responses as V through a forward pass, and adds V to the pre-trained representations of heads in T to improve truthfulness. For ITI, we used the original paper implementations.

RepE steers LLMs towards certain behaviors, including honesty and counterfactual knowledge, by using representations extracted from contrastive prompts. RepE extracts the difference vectors between two contrastive prompts as V and adds them to the representations of selected MLP layers. Note that RepE intervenes on MLP layers instead of attention heads as ITI and LoFiT do. [5] While RepE proposes three different representation engineering methods for different tasks, we selected the method that works best for each task in the original paper: specifically, we used RepE with contrast vector for MQuAKE and CLUTRR, and RepE with reading vector for TruthfulQA. We refer readers to [45] for further details.

A.2 Head Selection Baselines

We use the following head selection baselines to compare with the scaling-factor-based head selection of LoFiT.

Random sampling: We randomly sample K heads from the uniform distribution.[6]

Probing for layers: Along with other mechanistic interpretability works [21, 29], RepE focuses on layers as the subject matter for localizing functionality in LLMs. We examine if localizing important layers is better than important attention heads. Given a prompt in training data, we concatenate the gold response with it and a sampled incorrect response with it to create a pair of contrastive responses. We extract the pre-trained representations of all attention heads at each layer at the last token of both responses through a forward pass and concatenate them. We then train a logistic regression classifier for each layer to predict the correctness of responses using the concatenated representations as the input features, i.e., $z_{t=-1}^{l} = \text{concat}(z_{t=-1}^{(l,1)}, ..., z_{t=-1}^{(l,i)})$ for $i \in [1, H]$. We define the scoring function over heads as a scoring function over layers $\mathcal{S}(*, l)$, which is the probe accuracy on the validation set.

Bias-based selection: Prune-then-retrain is a common paradigm in neural network sparsification literature [44, 25] by re-training models on a sparse set of fine-tuned weights. We adapt this paradigm as the bias-based head selection baseline: we fine-tune the biases for the hidden representations of all attention heads v_l^i, and select the top-K attention heads where the scoring function $\mathcal{S}(i, l) = \|v_l^i\|$.

ITI head selection: ITI shows that training a linear probe using the hidden representations of each attention head can help identify important heads for truthful generation. We select the top-K heads based on ITI head selection method as described in Section 2.

A.3 Parameter-efficient Fine-Tuning Baselines

LoRA: LoRA learns offsets to weight parameters (rather than representations, as we do) via a product of two low-rank matrices. We use the huggingface PEFT library [20] implementation of LoRA.

RED: RED fine-tunes scaling factors and biases on the hidden representations of LLMs, similar to combining Step 1 and Step 2 of LoFiT. We also include a half-parameter version of RED (*RED*

[5]We choose to focus on attention heads, as recent interpretability research indicates that attention heads are semantically interpretable units for many key functionalities of LLMs such as induction [22] and retrieval [40]

[6]We considered other random selection schemes, including sampling the same number of heads from each layer and biased sampling towards more important layers, but they worked worse than uniform random sampling.

(Half)) as an additional baseline where only the layers in the second half of the network are tuned. We used our replication of RED as the official implementation was not available at the time of writing this paper.

ReFT: Concurrent to our work, ReFT learns to edit the hidden representations with a linear projection in a lower-dimensional subspace. Unlike LoFiT, RED and ReFT involve no localization. For ReFT, we used the official implementation of the most performant variant of ReFT, called LoReFT, from [41]. LoReFT edits the representations of a model in a lower-dimensional subspace by learning a linear projection from the residual hidden states to the subspace.

B Details of Datasets for Evaluation

We use the following datasets to evaluate LoFiT. Licensing details of each dataset can be found in Appendix J.

B.1 Main Evaluation

TruthfulQA [18] is a QA dataset with questions where humans are likely to give false answers because of common misconceptions. Representation interventions such as ITI [15] have shown success in eliciting truthful responses from LLMs without tuning. We follow the setup in [15] to split TruthfulQA into train/dev/test sets into 326/82/407 questions and use two-fold cross validation. We report MC1 and MC2 accuracy in the results; details of these metrics can be found in Appendix B.3.

CLUTRR [30] is a deductive reasoning dataset requiring multi-hop reasoning over family relationships. We use the subset of 2-hop questions and randomly split the data into train/dev/test sets of 300/450/450 QA pairs. Evaluation is exact match (EM).

MQuAKE [43] is a knowledge editing benchmark for evaluating the propagation of edited knowledge to related facts. We convert the subset of 2-hop knowledge propagation into an in-context knowledge editing setup by prepending "*Imagine that <Edited Knowledge>*" to the question of related facts [4]; we use this setup to evaluate if our method can learn simple reasoning over counterfactuals. Data is randomly split into train/dev/test sets of 134/95/864 QA pairs. Evaluation is exact match (EM).

B.2 Evaluation Setup of Comparison with PEFT methods

Although our main focus is to evaluate methods on datasets that are commonly used to benchmark interpretability-motivated methods, including representation intervention methods and LoFiT, we include additional results of LoFiT on common benchmarks for PEFT methods in Section 5. To be consistent with our low-data setting in Section 3, we only sampled 100 to 350 training examples rather than using the entire training data. We use the single-task learning setting from [12] where each fine-tuned model only learns to do *one* task. Details of these datasets are below.

SIQA [28] SIQA is a commonsense QA dataset that evaluates the model's understanding of social scenarios. We sampled 100 training examples and 100 validation examples for training, and used the test split of 1954 examples for evaluation.

ARC-c [3] ARC is an open-book QA dataset. We used the challenging set, ARC-c, and sampled 100 training examples and 299 validation examples for training. We used the test split of 1172 examples for evaluation.

BoolQ [2] BoolQ is a closed-book QA dataset. We sampled 100 training examples and 100 validation examples for training. We used the test split of 3270 examples for evaluation.

SVAMP [24] SVAMP is a math word problem dataset. We split the dataset into train/dev/test splits of 350/350/300 examples, and we trained the models using the gold equation and the final answer as the label (rather than just using the final answer).

We converted SIQA, ARC-c, and BoolQ into a unified multiple-choice QA format using prompts from [12]. Following [12], we evaluate the models based on whether they can correctly generate the option (rather than using the log-likelihood scoring of the sequence of each option). The prompts can be found in Appendix E.

B.3 Evaluation of TruthfulQA

We would like to provide some additional clarifications on the evaluation metrics for TrutfhulQA. The multiple-choice setting of TruthfulQA evaluates whether a model is able to select the true responses to a question out of several false responses that are related to common misconceptions. The two standard metrics used for this setting, as proposed in the original paper implementation of TruthfulQA [18], are the following:

MC1 (Single-true): Given a question, there is a single correct response out of 4-5 responses. The model evaluates each response independently by computing the log probability as the continuation of the question, and the model gets a score of 1 for a question only if it assigns the highest log probability to the single correct answer and 0 otherwise. The MC1 accuracy is the average score across all questions.

MC2 (Multi-true) Given a question, there are multiple correct responses provided. The score for a question is the sum of the normalized probability assigned to each true response. The MC2 accuracy is the average score across all questions.

C Experiment Configurations

When training with LoFiT, we learn the same scalars and biases for every token position of the input sequence. When performing inference with LoFiT, we use greedy decoding and we add the bias terms at the targeted attention heads to every decoding step. Prompt templates can be found in Appendix E. Hyperparameters for LoFiT and the baselines can be found in D.

For CLUTRR and MQuAKE, we use cross-entropy loss with gold responses for fine-tuning. For TruthfulQA, we use direct preference optimization [26] by pairing the gold truthful responses and untruthful responses as preference data for fine-tuning, as SFT has been shown ineffective in [15].

We fine-tune LoFiT and baselines using a single NVIDIA-RTX A6000 GPU with 48G memory. We use the huggingface implementation of Transformers [38] in PyTorch for all fine-tuning, and the TRL [37] implementation of direct preference optimization [26] for fine-tuning on TruthfulQA. We use AdamW optimizer for fine-tuning [19] with $\epsilon = 1e{-}8$ an a weight decay of factor 0.01. For both fine-tuning and inference, we use full precision for Llama 2-7B and bfloat16 mixed-precision for Llama 2-13B and Gemma-7B to fit on a single GPU.

D Hyperparameters

For all experiments on TruthfulQA, MQuAKE, CLUTRR, and SVAMP, we fine-tuned for 5 epochs with a batch size of 8 for all methods except ReFT; we will discuss the specific hyperparameters for ReFT in later sections. For all experiments on SIQA, and ARC-c, we fine-tuned for 3 epochs with a batch size of 8 to prevent memorization of world knowledge. For BoolQ, we fine-tuned for 3 epochs with a batch size of 4 for Llama 2-7B and of 2 for Llama 2-13B to fit the long passage context into a single GPU. We used the same implicit reward hyperparameter of direct preference optimization $\beta = 0.5$ for TruthfulQA experiments. Method-specific hyperparameters can be found in the following subsections.

D.1 LoFiT

Hyperparameters of LoFiT used in each experiment are summarized in Table 4. Because LoFiT for different models involves different numbers of learned parameters, we did not use the same set of hyperparameters throughout; instead, we tuned the following hyperparameters with grid search. Specifically, we found that a small L1 regularization term is good for enforcing the learning of a sparse set of effective attention heads, which is also suggested by model sparsification literature [23]. We also found that when using fewer heads, a larger learning rate is needed to stabilize training for LoFiT.

Table 4: The hyperparameters used for different tasks and models for LoFiT. BT = Bias Tuning.

		Attention Head Selection		BT (3% heads)	BT (10% heads)
		λ	Learning Rate	Learning Rate	Learning Rate
Gemma-7B	TruthfulQA	5e-4	5e-3	2e-2	8e-3
	MQuAKE	5e-4	5e-3	8e-3	8e-3
	CLUTRR	5e-3	5e-4	1e-2	1e-2
Llama 2-7B	TruthfulQA	5e-4	5e-3	1e-2	5e-3
	MQuAKE	1e-3	5e-3	1e-2	5e-3
	CLUTRR	5e-3	5e-4	1e-2	5e-3
Llama 2-13B	TruthfulQA	1e-3	1e-3	2e-2	5e-3
	MQuAKE	1e-3	1e-3	8e-3	5e-3
	CLUTRR	1e-3	1e-3	1e-2	8e-3
Llama 2-7B	SIQA	5e-3	5e-4	-	1e-2
	ARC-c	1e-3	1e-3	-	5e-3
	BoolQ	5e-3	5e-4	-	8e-3
	SVAMP	1e-3	1e-3	-	1e-2
Llama 2-13B	SIQA	1e-3	1e-3	-	5e-3
	ARC-c	1e-3	1e-3	-	5e-3
	BoolQ	1e-3	1e-3	-	5e-3
	SVAMP	1e-3	1e-3	-	1e-2

In addition to hyperparameters listed in Table 4, we used the following hyperparameters uniformly across all experiments for LoFiT: $\sigma_A = \sigma_v = 0.001$ for the initialization of LoFiT. We found that this set of initialization is robust to random seeds.

D.2 Representation Intervention Baselines

ITI In preliminary studies, we tuned the scaling factor α of the offset vectors over a range of values suggested in the original paper [15]. We tuned α on a validation set of each dataset and we found that $\alpha = 15$ worked robustly across all settings.

RepE In preliminary studies, we tuned the scaling factor α of the offset vectors on a validation set of each dataset and found that $\alpha = 5$ worked best across settings.

D.3 PEFT Baselines

LoRA In preliminary studies, we experimented with different configurations of LoRA, including applying LoRA weights to MLP layers and changing the rank and α. We found that the following configuration strikes the best balance across settings between overfitting with too many parameters and under-tuning with too low rank: we fine-tuned the Q projection and V projection matrices of all attention heads, used $\text{rank} = 8, \alpha = 8$, and applied a dropout rate of 0.1 to prevent overfitting. We performed a hyperparameter sweep for the learning rate on the validation set of each dataset and the optimal configurations for each model and dataset can be found in Table 5.

RED Suggested by the original RED paper [39] and confirmed in our preliminary studies, fine-tuning all attention heads with RED is better than other alternative RED configurations, including tuning MLP layers and tuning all modules, across settings. We performed a hyperparameter sweep for the learning rate on the validation set of each dataset and the optimal configurations for each model and dataset can be found in Table 5.

ReFT The ReFT paper [41] indicates that ReFT has the following important hyperparameters to tune: layers to apply interventions, the rank of the subspace, token positions in the input where the interventions are applied, and learning rate. In addition, in our preliminary experiments, we found that ReFT is very sensitive to hyperparameter choices and batch size also has an impact on the ReFT

Table 5: The hyperparameters used for different tasks and models for LoRA and RED.

		LoRA Learning rate	RED Learning rate
Gemma-7B	TruthfulQA	1e-3	1e-3
	MQuAKE	1e-3	1e-3
	CLUTRR	1e-3	1e-4
Llama 2-7B	TruthfulQA	1e-4	5e-4
	MQuAKE	1e-3	1e-3
	CLUTRR	1e-3	5e-3
Llama 2-13B	TruthfulQA	1e-3	1e-3
	MQuAKE	1e-3	1e-3
	CLUTRR	1e-3	1e-3
Llama 2-7B	SIQA	1e-3	1e-3
	ARC-c	1e-3	8e-4
	BoolQ	5e-4	5e-4
	SVAMP	1e-3	8e-4
Llama 2-13B	SIQA	1e-3	1e-3
	ARC-c	8e-4	1e-3
	BoolQ	1e-4	1e-3
	SVAMP	8e-4	8e-4

performance. Therefore, we performed grid search for the aforementioned hyperparameters to select the best combinations on a validation set for each dataset. The optimal configurations can be found in Table 6. We ran all experiments with ReFT for 5 epochs.

E Prompt Templates for Experiments

We use the following prompt templates for fine-tuning and evaluating LoFiT and baselines.

E.1 TruthfulQA

We follow the prompt strategy in ITI [15]: at the fine-tuning steps, we simply concatenate the question with the gold response or the incorrect response in the preference pair of the training and validation data as:

Prompt E.1: TruthfulQA

Prompt:
"Q: `{Question}` A: `{Response}` "

For evaluation, we prepend the standard "QA prompt", which can be found in the original implementation of TruthfulQA [18] and are later adopted by others [15, 34, 32], to the aforementioned prompt as the standard way of evaluating models on TruthfulQA in literature.

E.2 MQuAKE

Each example in MQuAKE comes with a piece of edited knowledge and a multi-hop reasoning question that uses the edited knowledge. We used the following prompt for MQuAKE.

Prompt E.2: MQuAKE

Prompt:
"Q: Imagine that `{edited_knowledge}`. `{question}` A:"

Table 6: The hyperparameters used for different tasks and models for ReFT. “Layers” indicates the layers where the interventions are applied. “Token Positions” indicates the token positions in the inputs where the interventions are applied, and “fa+lb” means the first a tokens and the last b tokens are intervened.

		Layers	Rank	Token Positions	Learning rate	Batch Size
Gemma-7B	TruthfulQA	All	4	f1+l1	1e-3	16
	MQuAKE	All	4	f1+l1	1e-3	8
	CLUTRR	All	8	f3+l3	1e-3	8
Llama 2-7B	TruthfulQA	All	4	f1+l1	1e-3	16
	MQuAKE	All	8	f3+l3	9e-4	16
	CLUTRR	All	8	f3+l3	9e-4	16
Llama 2-13B	TruthfulQA	All	4	f1+l1	1e-3	16
	MQuAKE	All	8	f3+l3	9e-4	8
	CLUTRR	All	8	f3+l3	9e-4	8
Llama 2-7B	SIQA	All	8	f3+l3	9e-4	8
	ARC-c	All	8	f5+l5	5e-4	8
	BoolQ	All	8	f3+l3	1e-4	2
	SVAMP	All	8	f3+l3	9e-4	8
Llama 2-13B	SIQA	All	16	f3+l3	1e-4	8
	ARC-c	All	16	f3+l3	1e-4	8
	BoolQ	All	8	f3+l3	1e-4	2
	SVAMP	All	4	f3+l3	9e-4	8

E.3 CLUTRR

Each example in CLUTRR comes with a few-sentence story that describes relations among fictional characters and a pair of characters whose relationship needs to be inferred from the story and answered by the model. We use the following prompt for CLUTRR. In preliminary experiments, we found that LLMs sometimes refuse to answer and indicate that relationships in the provided story are incorrect based on names and relations of known people in the real world, so we include further clarifying instructions in the prompt in addition to basic formatting.

Prompt E.3: CLUTRR

Prompt:
“Read the following story about a family. `{Story}` Assume the relations described in the story are all true. Based on relations between the fictional characters in the story (assume the relations are all true) and your commonsense knowledge about family relationship, how is `{Character2}` related to `{Character1}` ? Answer: `{Character2}` is `{Character1}` ’s”

E.4 SIQA

We use the prompt from [12] for SIQA.

Prompt E.4: SIQA

Prompt:
“

Please choose the correct answer to the question.

Question: `{context}` `{question}`

A. `{answerA}`

B. `{answerB}`

C. `{answerC}`

Answer:"

E.5 ARC-c

We use the prompt from [12] for ARC-c.

Prompt E.5: ARC-c

Prompt:
"

Please choose the correct answer to the question.

Question: `{question}`

A. `{answerA}`

B. `{answerB}`

C. `{answerC}`

Answer:"

E.6 BoolQ

We use the prompt from [12] for BoolQ. BoolQ does not have answer options as the answers can only be True/False.

Prompt E.6: BoolQ

Prompt:
"

`{Passage}`

Question: `{question}`

Answer:"

E.7 SVAMP

In the prompt, we instruct the models to generate an equation for the math word problem, and we only evaluate the correctness of the final answer derived from the equation rather than the entire equation.

Prompt E.7: SVAMP

Prompt:
"

Question: `{Context}` `{question}`

Equation:"

F Effects of the Number of Heads on LoFiT performance

The number of heads K used for the bias tuning stage has an impact on LoFiT performance. We conduct an analysis on the percentage of attention heads used for LoFiT bias tuning versus the accuracy on MQuAKE and CLUTRR with Llama 2-7B and Llama 2-13B.

Results can be found in Figure 4. We observe that the performance plateaus when reaches 10% - 20% of the total number of attention heads and continues to increase as K gets larger before it reaches the above threshold. This is likely because the number of learned parameters is too small to be expressive when is smaller than 10% of attention heads (<10K parameters). Note that the results in

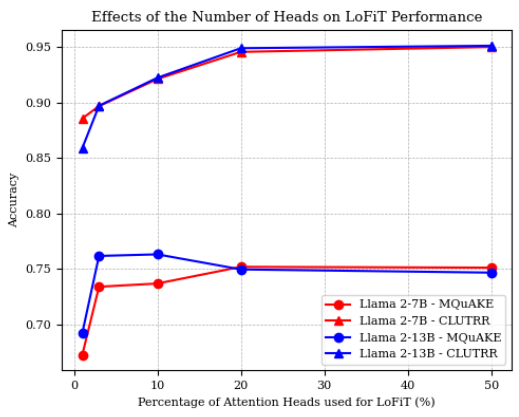

Figure 4: The effects of the percentage of attention heads K used for LoFiT Bias Tuning on LoFiT performance. Results are averaged over two runs. The test accuracy increases with K when K$< 10\%$ and plateaus when K reaches $10\% - 20\%$.

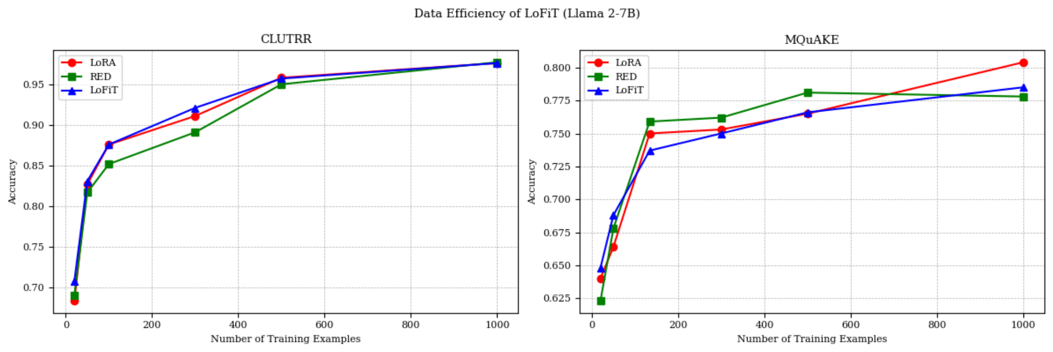

Figure 5: LoFiT performance using different numbers of training examples n on CLUTRR and MQuAKE with Llama 2-7B. For LoFiT, we tune 10% of the attention heads. Results are averaged over two runs. In the low data settings ($n \leq 100$), LoFiT is more data efficient than LoRA and RED. For $n \geq 300$, LoFiT is still comparable to LoRA and RED with fewer parameters.

the paper used either 3% (as in Table 1) for extremely parameter-efficient scenarios to achieve a fair comparison with representation intervention methods, or 10% (as in Table 3) for the best balance between parameter counts and performance.

G Data Efficiency

We further compare LoFiT against PEFT methods using different numbers of training examples n. We analyze the data efficiency of LoFiT on CLUTRR and MQuAKE with Llama 2-7B. Figure 5 shows that in the extremely low data settings ($n \leq 100$), LoFiT performs better than LoRA and RED, showing that LoFiT is very data efficient. For $300 \leq n \leq 1000$, LoFiT is still comparable to LoRA and RED with fewer parameters.

H Open-ended Generation

While being fine-tuned for discriminative tasks in the above experiments, LoFiT also shows good performance on the open-ended generation task of TruthfulQA. We fine-tune with LoFiT and other methods on TruthfulQA with the same setup in Section 3 and evaluate its open-ended generation on TruthfulQA test questions with GPT-4. We prompt GPT-4 to evaluate the informativeness (*Info*) and truthfulness (*True*) of model responses given the question and the gold labels. Table 7 shows that

Table 7: GPT-4 evaluation of open-ended generation quality to TruthfulQA. LoFiT is comparable to PEFT methods in terms of truthfulness and informativeness, and outperforms 0-shot and ITI baselines by a large margin.

		True	Info	True × Info
Llama 2-7B	0-shot	35.7	**92.7**	33.1
	ITI	52.3	81.4	42.6
	LoRA	64.3	88.0	56.6
	RED	66.5	91.4	**60.8**
	LoFiT	**67.2**	87.5	58.9
Llama 2-13B	0-shot	38.6	93.9	36.3
	ITI	46.9	78.5	36.8
	LoRA	67.2	90.5	60.8
	RED	67.0	**94.6**	**63.4**
	LoFiT	**67.5**	90.7	61.2

Table 8: Out-of-domain generalization performance of LoFiT on Llama 2-7B-Chat after fine-tuning on TruthfulQA. 0-shot prompts are used for OOD evaluation. "No-FT" represents the base model without being fine-tuned on TruthfulQA. In-domain evaluation results on TruthfulQA are also included for reference. Compared to PEFT methods, LoFiT better preserves the existing capabilities of the base model after being fine-tuned across all settings.

	TruthfulQA		Out-of-Domain			
	MC1	MC2	TriviaQA	NQ	MMLU	*Average*
No-FT	33.7	51.3	76.5	62.9	40.3	60.0
ITI	40.0	59.1	72.7	60.6	37.3	56.9
RED	54.0	73.1	74.9	63.3	35.4	57.9
LoRA	51.3	73.3	73.5	62.0	**41.0**	58.8
LoFiT	**54.5**	**74.9**	**76.7**	**64.4**	40.5	**60.5**

LoFiT leads to truthful and informative responses that are comparable to PEFT methods and are better than ITI.

I Out-of-domain Generalization

Benefiting from the small number of parameter updates, LoFiT can potentially generalize well to out-of-domain tasks after fine-tuning. We show a case study on TruthfulQA: we first fine-tune Llama 2-7B-chat on TruthfulQA and then evaluate 0-shot on three out-of-domain question-answering benchmarks: TriviaQA [13], Natural Questions [14, NQ], and MMLU [10]. We follow the same evaluation scheme in [15] to convert TriviaQA and Natural Questions into multiple-choice questions and report accuracy.[7] Table 8 shows that LoFiT suffers less from overfitting on TruthfulQA as its performance on TriviaQA and MMLU does not drop from the non-fine-tuned base model, and it even improves on NQ while OOD performance degradation can be observed in ITI and PEFT baselines.

J Licensing

We use the following publicly available datasets from prior works with open licenses.

TruthfulQA [18] uses the Apache-2.0 license and data is available at: `https://github.com/sylinrl/TruthfulQA`.

[7]For TriviaQA and NQ, we use the two evaluation splits curated by [15] that are publicly accessible in the honest_llama repository. We refer readers to [15] for details. For MMLU, we use the evaluation scheme of open-source package lm-evaluation-harness [7].

CLUTRR [30] uses CC-BY-NC 4.0 (Attr Non-Commercial Inter.) license and data is available at `https://github.com/facebookresearch/clutrr`. Our data splits of CLUTRR are available at `https://github.com/fc2869/lo-fit`.

MQuAKE [43] uses the MIT license as per `https://github.com/princeton-nlp/MQuAKE`. Our data splits of MQuAKE are available at `https://github.com/fc2869/lo-fit`.

SIQA, ARC-c, BoolQ, and SVAMP For these datasets, we follow [12] who use the open data commons attribution license. The data and licenses are available at `https://github.com/AGI-Edgerunners/LLM-Adapters`.