Data-scarce Behavior Editing of Language Models

Anonymous ACL submission

Abstract

Large Language Models trained on web-scale text acquire language generation abilities that can solve a wide range of tasks, particularly when task knowledge is refined into the generative prior using in-context examples. However, spurious features learned from noisy data hinder their generalizability. Supervised finetuning can enhance task specificity but may lead to data inefficiency. Prior studies indicate that (i) noisy neural circuitries coexist with generalizable ones within LLMs, and (ii) finetuning typically enhances (or suppresses) existing abilities without introducing newer ones. Building upon these, we propose `TaRot`, a novel method for task adaptation. `TaRot` intervenes in the neural circuitries using learnable rotation matrices that are optimized using Bayesian optimization, on labelled samples in the order of standard few-shot prompting examples. Experiments on multiple classification and generation tasks using LLMs of varying sizes reveal the efficacy of `TaRot`, improving upon both zero- as well as few-shot performance, with average improvements (across models and tasks) of 15.6% and 14%, respectively.

1 Introduction

Large Language Models (LLMs) acquire the ability to associate different language concepts presented in a sequential context by optimizing the prediction probability of the next token given a context. Despite its apparent simplicity, when scaled across web-sized text corpora, such a learning strategy introduces the ability to solve a wide range of tasks presented in natural language. However, the web contains almost everything humankind has written, and therefore, it introduces spurious token associations that are irrelevant or even counter-productive to the model to become generalized task-solvers. We observe phenomena like brittle few-shot performance (Sclar et al., 2024), hallucination (Huang et al., 2023), harmful text generation (Wen et al., 2023), etc. as evidence of learning noisy patterns. Remedial interventions like instruction tuning (Zhang et al., 2024), alignment tuning (Shen et al., 2023), etc. have been proposed. Recent research has shown that such mediation only acts on a superficial level — out-of-distribution inputs can reinforce noisy behavior and break the model (Ghosh et al., 2024). Without an in-depth understanding of the inner workings, remedial strategies become wild goose chase.

Mechanistic disentangling of Transformer-based language models has shed some light on this direction (Elhage et al., 2021; Olsson et al., 2022; Wang et al., 2023). Two recent investigations (Jain et al., 2024; Prakash et al., 2024) on the effects of fine-tuning confirm the inability of supervised fine-tuning to alter fundamental abilities acquired via pretraining. On a tangential investigation, Dutta et al. (2024) recently confirmed the existence of multiple parallel neural pathways of answer processing within LLMs. Bhaskar et al. (2024) echoed similar findings in the case of syntactic generalization while pointing out that different components acquire different generalization behaviors. These findings lead us to the central research question of this work: *is it possible to edit the model behavior by editing internal representations in a generalizable manner?* Prior work in this direction has heavily relied on careful manual effort to localize task-specific neural components and design intervention techniques (Meng et al., 2022; Li et al., 2024a). Two key shortcomings limit the scalability of such methods: (i) Localization complexity grows polynomially with model size, making it difficult to identify task-relevant components and design effective ablations; (ii) Redundant components performing similar neural computations hinder the generalizability of any single intervention.

Our contributions. To this end, we propose a novel intervention technique, `TaRot` – **T**ask-**a**ware

Rotation of token-association (see Figure 1 for a representative depiction)[1]. We establish the conceptual prior from Transformers's implicit gradient descent bias in next token prediction. Specifically, we first show that attention-weighted averaging of value vectors facilitates the memorization of token association from pertaining data in individual attention heads, in the sense that each attention head acts as a mini-language model. Due to the vast number of token associations present in the pretraining corpus compared to the number of attention heads in even the largest of the models, we hypothesize that individual directions of these memorized associations remain in superposition, and removal or downscaling of a head can counteract model performance. Instead, we construct parametrized rotations to align head outputs for task-adaptation. The rotation parameters are then optimized using Bayesian optimization. Furthermore, `TaRot` is *extremely data- and compute-efficient*: we use 6-20 supervised examples for each task and $\frac{dL}{4}$ rotation parameters (where d is the model dimension and L is the number of layers) for each different task. This renders `TaRot` at par with standard few-shot prompting in labeled data-efficiency.

We experiment with four different classification tasks and two natural language generation tasks; the choice of tasks seeks to investigate general world knowledge (news topic classification) as well as the ability to generalize beyond imitation (BIG Bench tasks (BIG-bench authors, 2023)). `TaRot` demonstrates consistent improvements over six different language models of varying sizes: Qwen2-1.5B-Instruct, Phi-3-mini-4k-instruct, Mistral-7B-Instruct-v0.1, Meta-Llama-3-8B-Instruct, Qwen2.5-14B-Instruct, and Qwen2.5-32B-Instruct, in both zero-shot as well as few-shot settings. Furthermore, we analyze the changes in neural representation introduced by `TaRot` to uncover useful insights.

2 Related Work

Our work is primarily relevant to two broad areas of existing literature: adaptation of pretrained language models to downstream tasks, and mechanistic understanding and intervention techniques.

Task adaptation of pretrained language models. The *pretrain-finetune* regime for adapting language models to downstream tasks dates back to the early approaches like BERT (Devlin et al., 2019) — pretrain a language model (LM) on large unstructured text corpora using self-supervised objective, followed by supervised fine-tuning on task-specific, relatively smaller datasets. Despite the apparent simplicity, the pitfalls of this regime have been pointed out in terms of *distribution shift* (Kumar et al., 2022). With the development of large-scale, autoregressive Transformer-based language models and their ability to learn from in-context examples (Brown et al., 2020), a definitive shift has happened in the more recent past. Current practices of using these models for downstream tasks primarily rely on designing suitable prompt templates and labeled example retrieval for in-context learning (ICL) (Liu et al., 2022; Rubin et al., 2022; Tanwar et al., 2023); traditional techniques of fine-tuning have taken a back seat due to the computational cost and catastrophic forgetting introduced by small-scale task-specific data that hurts the pretrained abilities (Zhai et al., 2024). Instead, fine-tuning to follow task instructions, *aka* instruction-tuning (Zhang et al., 2024), has gained popularity. Instruction-tuning has been shown to introduce zero-shot task adaptation abilities in LLMs (Wei et al., 2022). Additionally, different methods of alignment tuning have been proposed with the primary goal being aligning the generative distribution of the language models with human values and preferences (Shen et al., 2023; Wang et al., 2024b). Despite the popularity of instruction and alignment tuning, their ability to alter fundamental information processing has been put in question in recent literature. Jain et al. (2024) investigated the effects of fine-tuning in toy models trained with formal languages as well as precompiled ones; their findings suggest that supervised fine-tuning does not introduce any new ability into pretrained models but only reinforces (or suppresses) existing ones. Similar concerns have been raised upon investigating entity tracking in the neural representation space (Prakash et al., 2024). Ghosh et al. (2024) identified multiple limitations of instruction tuning, including the inability to introduce new knowledge and deterioration of performance due to over-reliance on pattern matching.

Mechanistic understanding and interventions. The umbrella of mechanistic interpretability broadly encompasses methods to disentangle model behavior via reverse engineering the underlying neural algorithm (Elhage et al., 2021; Ferrando et al., 2024). Endeavors to mechanistically under-

[1]The source code of `TaRot` is attached with the supplementary and will be made public upon acceptance of the paper.

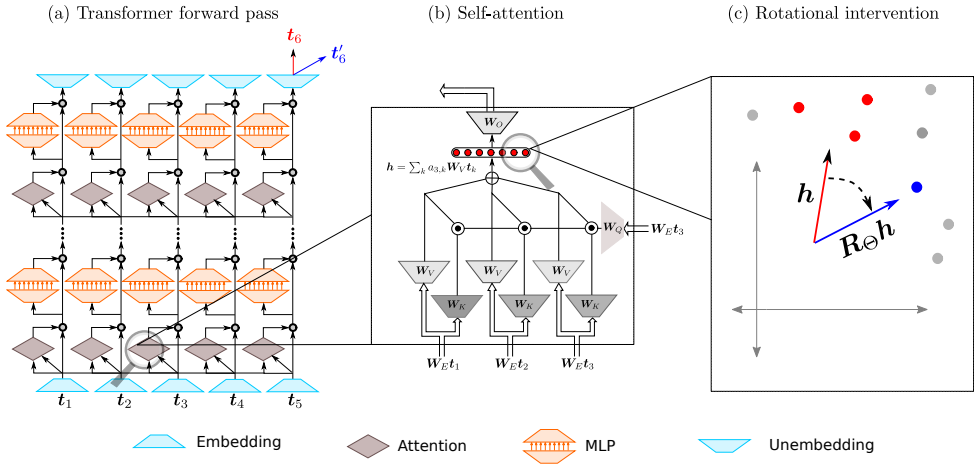

Figure 1: **A conceptual illustration of TaRot.** (a) Model generates an undesired next token t_6 upon an input token sequence. (b) A certain attention head is responsible for associating certain input tokens with the undesired output. (c) TaRot learns a parametrized rotation operator $\boldsymbol{R}_\Theta$ that rotates $\boldsymbol{h}$ to the direction of the desired token (red to blue). The intervention results in a change in the forward pass in (a) that outputs the desired token t'_6.

stand Transformer-based language models trace back to the seminal work by Elhage et al. (2021). Their framework established attention heads as one of the fundamental building blocks of language model interpretation. Subsequent studies have identified the functional roles of different attention heads in pretrained models: induction heads as a primary mechanism of prefix matching (Olsson et al., 2022), circuitries of attention heads responsible for indirect object identification (Wang et al., 2023), neural pathways that implement chain-of-thought reasoning (Dutta et al., 2024), etc. Much relevant to our analysis, Lv et al. (2024) found that certain attention heads memorize the association between country names and their capitals. On a tangential line of investigation, Geiger et al. (2024) introduced the Distributed Alignment Search (DAS) framework for localizing interpretable features in subspaces of the neural representations. Mechanistic methods provide actionable insights that have led to non-traditional techniques to edit model behavior. Elhage et al. (2021) experimented with key propagation to elicit induction heads (and thereby, prefix-matching ability) in single-layer attention-only Transformers. Meng et al. (2022) used causal tracing to locate factual associations in MLP neurons and proposed a gradient-free approach to edit factual recall patterns in pretrained language models. Li et al. (2024a) identified attention head circuitry that elicits toxic text generation in GPT-2; mean-ablation of these circuits is shown to reduce toxicity. Self-detoxification (Leong et al., 2023) identifies toxic generation direction in the internal representation using trigger prompts and then rewrites in the opposite direction to reduce toxicity. Wang et al. (2024a) formulated toxicity reduction as a knowledge editing task that can permanently alter toxic behaviors instead of suppressive interventions like supervised fine-tuning or RLHF-based alignment. Lamparth and Reuel (2024) localized backdoor mechanisms (i.e., vulnerabilities against adversarial prompt injections) in early-layer MLPs and proposed a low-rank substitution to improve robustness against such injections. Vergara-Browne et al. (2024) employed attribution patching techniques to identify and remove certain singular values in the parameter matrices to improve performance.

In comparison with prior intervention approaches, our work bears two fundamental differences: (i) TaRot does not necessitate task-specific localization of neural behaviors; this significantly reduces intense manual effort and risk of over-localization, eliciting efficient, generalizable interventions; (ii) TaRot is gradient-free, parameter-efficient, and requires supervised samples in the order of standard ICL; this poses TaRot as a practical alternative to intense prompt-engineering.

3 Methodology

In this section, we demonstrate the role of attention heads in memorizing token associations. Next, we lay out the working principles of TaRot.

3.1 Attention heads as token-token maps

Inspired by Elhage et al. (2021), we dissect the Transformer-based language models with the following assumptions: (i) each attention head reads

from and writes to the residual stream independently in a linear fashion, and (ii) given that the attention heads utilize hidden representation of dimensionality much smaller than the residual stream (i.e., for a model with 16 attention heads, each attention head uses 1/16-th of the dimension of the residual stream), they typical operate on small subspaces of the residual stream. This way, two attention heads can operate on two distinct subspaces and never interact with each other. These two assumptions allow us to interpret the working of the attention heads meaningfully even while treating each head in isolation. We start with identifying what a single-head attention operation tends to learn in isolation.

Following the standard terminology (Elhage et al., 2021), we represent the embedding and unembedding matrices as $\boldsymbol{W}_E \in \mathbb{R}^{d\times V}$ and $\boldsymbol{W}_U \in \mathbb{R}^{V\times d}$, where d and V are the dimensionality of the residual stream and the token space, respectively, the query, key, value, and output projection matrices denoted as $\boldsymbol{W}_Q, \boldsymbol{W}_K, \boldsymbol{W}_V, \boldsymbol{W}_O \in \mathbb{R}^{d\times d}$, respectively. Given a sequence of input tokens as one-hot column vectors $\boldsymbol{T} = \{\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n\}$, the forward pass for single-layer attention-only Transformer can be written as:

$$\hat{\boldsymbol{t}}_{n+1} = \boldsymbol{W}_U \left(\mathbf{W}_E \boldsymbol{t}_n + \boldsymbol{W}_O \sum_i a_{n,i} \boldsymbol{W}_V \boldsymbol{W}_E \boldsymbol{t}_i \right) \tag{1}$$

where $a_{n,i} = \frac{\exp\left(\boldsymbol{t}_n^\top \boldsymbol{W}_E^\top \boldsymbol{W}_Q^\top \mathbf{R}_{\Theta,n-i} \boldsymbol{W}_K \boldsymbol{W}_E \boldsymbol{t}_i\right)}{\sum_j \exp\left(\boldsymbol{t}_n^\top \boldsymbol{W}_E^\top \boldsymbol{W}_Q^\top \mathbf{R}_{\Theta,n-j} \boldsymbol{W}_K \boldsymbol{W}_E \boldsymbol{t}_j\right)}$ is the softmax-attention probability from source token $\boldsymbol{t}_i$ to destination token $\boldsymbol{t}_n$, and $\hat{\boldsymbol{t}}_{n+1} \in \mathbb{R}^V$ is the logit of the predicted next token. Upon reparametrization of $\boldsymbol{W}_U \boldsymbol{W}_O \boldsymbol{W}_V \boldsymbol{W}_E$ as $\boldsymbol{W}_{OV}$, we can rewrite Equation 1 as

$$\hat{\boldsymbol{t}}_{n+1} = \boldsymbol{W}_U \mathbf{W}_E \boldsymbol{t}_n + \sum_i \boldsymbol{W}_{OV} \boldsymbol{t}_i \tag{2}$$

Note that $\boldsymbol{W}_{OV} \in \mathbb{R}^{V\times V}$, denoted as OV-circuits by Elhage et al. (2021), maps a distribution over tokens to another distribution over tokens. If the true token is $\boldsymbol{t}_{n+1}$ with $I(\boldsymbol{t}_{n+1})$ donating its index (i.e., index of 1 in $\boldsymbol{t}_{n+1}$), then the typical language modeling loss can be calculated as:

$$\mathcal{L}(\hat{\boldsymbol{t}}_{n+1}, \boldsymbol{t}_{n+1}) = -\log \left(\frac{\exp\left(\hat{\boldsymbol{t}}_{n+1}^{(I(\boldsymbol{t}_{n+1}))}\right)}{\sum_k \exp\left(\hat{\boldsymbol{t}}_{n+1}^{(k)}\right)} \right) \tag{3}$$

We can compute the gradient dynamics of the OV-circuit (with unit batch size and zero momentum) using Equations 2 and 3 as follows:

$$\boldsymbol{W}_{OV}^{(s+1)} = \boldsymbol{W}_{OV}^{(s)} + \eta \boldsymbol{t}_{n+1} \left(\sum_i a_{n,i} \boldsymbol{t}_i \right)^\top - \eta \, \mathrm{SoftMax}\left(\boldsymbol{t}_{n+1}\right) \left(\sum_i a_{n,i} \boldsymbol{t}_i \right)^\top \tag{4}$$

where $\boldsymbol{W}_{OV}^{(s)}$ and $\boldsymbol{W}_{OV}^{(s+1)}$ are the OV-circuit parameters before and after the s-th gradient update step, respectively and η is the learning rate. The positive incremental component in the right-hand side of Equation 4 dictates that, when applied on a attention-weighted linear combination of the context tokens, OV-circuits learn to memorize a linear combination of possible next tokens.

However, in a deep Transformer model with several attention heads, MLP blocks and layer normalization, we can not determine the exact token-token map for the OV-circuits of attention head. Moreover, as Elhage et al. (2021) suggested, multiple attention heads across different layers can construct compositions, where the deeper heads use the output of the shallower heads. Alternatively, we can view each head as memorizing how to write in a specific direction in the residual stream, given a sequence of residual vectors—effectively acting as a *mini-LM*. When pretrained on web-scale corpora, these heads may memorize spurious token-token associations that harm downstream performance or introduce unsafe behaviors.

3.2 Editing model behavior via attention rotation

A natural conclusion from the prior discussion would be that, by suppressing undesired associations for certain attention heads, we can improve task performance. However, multiple token associations are expected to be memorized in each attention head in superposition since the number of attention heads is way smaller than the potential token associations present in the pretraining data — one cannot selectively switch off one certain association. Prior research in mechanistic interpretability has shown that, although we can often localize attention heads responsible for particular task, removing the non-dominant attention heads does not deliver the performance of the full model (Wang et al., 2023; Dutta et al., 2024).

Instead, one can *rotate* the output of the attention heads in order to maximize its alignment with rows of $\boldsymbol{W}_U$ corresponding to certain tokens while near-orthogonalizing with certain undesired tokens. This way, the model behaviour can be edited without destroying the superposed associations. Defining the complete space of $d \times d$ rotation matrices and optimizing them can become computationally challenging. Instead, we utilize the fact that any $d \times d$ orthonormal matrix is similar to a block-diagonal matrix $\boldsymbol{R}_\Theta$, where $\Theta = \{\theta_1, \cdots, \theta_{d/2}\} \subset [0, 2\pi)^{\frac{d}{2}}$, defined as:

$$\boldsymbol{R}^d_\Theta = \begin{bmatrix} \boldsymbol{B}(\theta_1) & 0 & \cdots & 0 \\ 0 & \boldsymbol{B}(\theta_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{B}(\theta_{d/2}) \end{bmatrix} \quad (5)$$

where

$$\boldsymbol{B}(\theta_i) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix}$$

Given the multi-head attention with H heads at layer $l \in [L]$, where L is the total number of layers in the Transformer, defined as:

$$\text{Attn}_l(\boldsymbol{x}^{(l)}_n | [\boldsymbol{x}^{(l)}_1, \cdots, \boldsymbol{x}^{(l)}_n]) = \boldsymbol{W}_O \Big\|_{h=1}^{H} \sum_i a^{(h,l)}_{n,i} \boldsymbol{W}^{(h,l)}_V \boldsymbol{x}^{(l)}_i$$

where $\|$ is the concatenation operator, $a^{(h,l)}_{n,i}$ and $\boldsymbol{W}^{(h,l)}_V$ denote the attention probability between source and destination residual streams at layer l $\boldsymbol{x}^{(l)}_i$ and $\boldsymbol{x}^{(l)}_n$ and the value projection matrix corresponding to the attention head with index $h \in [H]$ at layer l, respectively; we define the rotated attention as:

$$\text{RotAttn}_l(\boldsymbol{x}^{(l)}_n | [\boldsymbol{x}^{(l)}_1, \cdots, \boldsymbol{x}^{(l)}_n]) = \boldsymbol{W}_O \boldsymbol{R}^d_{\Theta_l} \Big\|_{h=1}^{H} \sum_i a^{(h)}_{n,i} \boldsymbol{W}^{(h)}_V \boldsymbol{x}^{(l)}_i \quad (6)$$

Note that the block-diagonal definition of $\boldsymbol{R}^d_\Theta$ in Equation 5 implies that applying $\boldsymbol{R}^d_\Theta$ on the concatenated head outputs is equivalent to applying H-distinct $\boldsymbol{R}^{d/H}_\Theta$ on each of the head outputs.

Without prior knowledge of which attention heads are responsible for memorizing undesired token associations, we need to apply the intervention defined in Equation 6 on a set of attention blocks at layers $l \in \hat{\mathbb{L}}$ (see Section 4 for the choice of the set $\hat{\mathbb{L}}$). Then, the intervened forward pass is denoted as:

$$\hat{\boldsymbol{t}}_{n+1} = \mathcal{M}_{\text{Rotated}}\Big(\{\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n\} | \Theta_{\text{O}}, \Theta_{\text{R}}\{\Theta_l | l \in \hat{\mathbb{L}}\}\Big) \quad (7)$$

where Θ_{O} is the set of pretrained model parameters, and Θ_{R} are the parameters of rotations, and $\mathcal{M}_{\text{Rotated}}$ denote the function representing the language model upon the designed intervention.

3.3 Optimization of rotation parameters

With the rotational interventions defined, all that we are left with is to optimize the rotational parameters. Let $\mathcal{D} := \{\boldsymbol{T}_j, \boldsymbol{Y}_j | j \in [D]\}$ be a set of D supervised examples for a given task, with $\boldsymbol{T}_j$, $\boldsymbol{Y}_j$ referring to the sequence of tokens corresponding to the input and gold output, respectively. If $\boldsymbol{Y}_j = \{\boldsymbol{y}_j\}$ is a single label token, the cost function to optimize becomes straightforward:

$$\max_{\Theta_{\text{R}}} \sum_j p\Big(\mathcal{M}_{\text{Rotated}}\Big(\boldsymbol{T}_j | \Theta_{\text{O}}, \Theta_{\text{R}}\{\Theta_l | l \in \hat{\mathbb{L}}\}\Big) = \boldsymbol{y}_j\Big) \quad (8)$$

where $\Theta \subset [0, 2\pi)$. For NLG tasks, maximizing the aggregate probability of all the generated tokens can be a solution. However, the goal of our rewiring method is to minimize undesired behaviors. When a model demonstrates such behaviors, depending upon the task, not all tokens equally correspond to the behavior under inspection. The pretrained model is trained using teacher-forcing and is generally able to generate grammatically correct responses. Hence, trying to align the model generation to a single reference response does not make much sense. Instead, we opt for a surrogate scoring function $s : \{\boldsymbol{Y}_j\} \to \mathbb{R}$ that scores the "desirability" of a generated response. We let the model with rotation intervention to generate a complete response given an input, compute the score for the generated response, and seek to minimize the aggregate score across $\mathcal{D}$. We implement Bayesian optimization (Snoek et al., 2012) to solve the optimization problem depending upon the task. However, standard Gaussian Process with Matern kernel fails to scale to high dimension input space (Li et al., 2024b). Instead, Infinite-width Bayesian Neiral Networks (I-BNN), proposed by Lee et al. (2017), has shown to scale effectively with high-dimensional parameter space[2]. The I-

[2]Here the term "high dimension" is relatively used. Our method seeks to optimize only the rotation configurations that

BNN covariance function does not rely on Euclidean distance, enabling the Gaussian Process to model non-stationary functions, an advantage since rotational effects may vary across the configuration space.

4 Experiment Setup

Training setting. Dutta et al. (2024) previously found that token associations corresponding to pretrained knowledge primarily resides in the initial half of the model. Since the rotational intervention designed in Equations 6 and 7 are primarily targeted towards undesired token associations acquired through pretraining, we restrict $\mathbb{L}$ to the initial half only. Therefore, the total number of parameters to optimise becomes $\frac{dL}{4}$. Since we want to optimise the rotation matrix for a particular task, only a small subset of training samples is required, i.e, $6 \leq D_{training} \leq 20$.

Models. Six different instruction-tuned models with varying size are used for all experiments: Qwen2-1.5B-Instruct (Yang et al., 2024), Phi-3-mini-4k-instruct (Abdin et al., 2024) (2.8 billion parameter), Mistral-7B-Instruct-v0.1 (Jiang et al., 2023), Meta-Llama-3-8B-Instruct (Dubey et al., 2024), Qwen2.5-14B-Instruct (Team, 2024), and Qwen2.5-32B-Instruct (Team, 2024); we refer to these models as `Qwen2-1.5B`, `Phi-3-mini`, `Mistral-7B`, `Llama-3-8B`, `Qwen2.5-14B`, and `Qwen2.5-32B`, respectively.

Tasks. We experiment with four different classification (i.e., single token generation) tasks and two NLG tasks. Classification tasks used are as follows: AG News (Zhang et al., 2015), Entailed Polarity (Srivastava et al., 2022), Navigate (Srivastava et al., 2022), and Winowhy (BIG-bench authors, 2023). The generation tasks used include Imdb Positive Review (Maas et al., 2011), and-Detoxify (Gehman et al., 2020) Further details and examples of tasks are available in Appendix A.1

Baselines. We compare `TaRot` with four different baselines: *Base model,* Eigen Pruning (Vergara-Browne et al., 2024), *RED (Representation EDiting) (Wu et al., 2024), and* Rescaling (additional Details in Appendix A.2).

Evaluation metrics. For NLG tasks, Imdb and Detoxify, two different types of reward models are used. To calculate the fluency of the generated text, GPT4 (Achiam et al., 2023) is used as an oracle. For both the tasks, the average of fluency and the score from the reward models are reported. Further details are present in Appendix A.3

5 Results

Tables 1 and 2 summarize the performance of various methods across classification tasks in zero- and 6-shot settings, respectively. Eigen Pruning is included only in zero-shot comparisons, per its original design. Table 3 presents results for NLG tasks.

Consistent improvement with `TaRot`. Across LLMs of varying sizes, `TaRot` consistently ranks as the best or second-best method across all tasks. Notably, it achieves relative gains in task-wise average F1 scores of 13.7%, 1.1%, 8.9%, 13%, 3.2%, and 1.3% over the base versions of `Qwen2-1.5B`, `Phi-3-mini`, `Mistral-7B`, `Llama-3-8B`, `Qwen2.5-14B`, and `Qwen2.5-32B`, respectively, in the zero-shot setting (see Table 1). The only exceptions are the Entailed polarity task with `Qwen2-1.5B` and Winowhy with `Qwen2.5-32B`, where `TaRot` slightly underperforms (e.g., 0.98 F1 vs. perfect score). In contrast, baseline methods like Eigen Pruning and Rescaling suffer from inconsistency—while they may improve performance in some cases, they often cause severe drops without any clear task- or model-specific patterns. For instance, Eigen Pruning improves `Qwen2-1.5B` on all but one task, yet fails on all tasks with `Phi-3-mini`.

In-context examples vs. `TaRot`. Unlike Eigen Pruning (or even, traditional fine-tuning), `TaRot` is optimized with a mixture of M-shot inference to avoid zero-shot bias. Consequently, we can observe the improvement over the base model achieved via `TaRot` while provided with in-context examples, except with `Mistral-7B` on AG News and Navigate (c.f. Table 2). Specifically, we observe improvements with `Qwen2-1.5B` on AG News, Entailed Polarity, and Winowhy; and with `Phi-3-mini`, `Llama-3-8B`, Qwen 2.5 14B, and Qwen 2.5 32B across all tasks. For `Mistral-7B`, gains are limited to Entailed Polarity and Winowhy.

Importance of rotation over rescaling attention heads. Comparing `TaRot` with the rotation-free Rescaling approach highlights key differences in intervention effectiveness. Rescaling is often brittle, with no consistent performance pattern. For

scales as $\mathcal{O}(Ld)$, which is substantially low-dimensional if compared to the parameter space of the LM itself.

Method		AG News	Entailed polarity	Navigate	Winowhy	Avg.
Qwen2-1.5B	Base	0.691	**1.000**	0.173	0.389	0.563
	Eigen Pruning	0.720	0.919	0.290	0.415	0.586
	Rescaling	**0.796**	0.719	0.214	0.458	0.547
	TaRot	0.778	0.980	**0.515**	**0.547**	**0.705**
Phi-3-mini	Base	0.729	**1.000**	0.470	0.588	0.697
	Eigen Pruning	0.519	0.878	0.392	0.099	0.472
	Rescaling	0.739	0.921	0.273	**0.629**	0.641
	TaRot	**0.740**	**1.000**	**0.491**	0.600	**0.708**
Mistral-7B	Base	0.653	0.762	0.140	0.618	0.543
	Rescaling	0.437	**0.896**	**0.550**	0.683	**0.642**
	TaRot	**0.721**	0.823	0.216	**0.767**	0.632
Llama-3-8B	Base	0.662	0.980	0.155	0.568	0.591
	RED	0.688	**0.980**	**0.957**	0.236	0.715
	Rescaling	0.636	0.544	0.550	0.255	0.496
	TaRot	**0.718**	**1.000**	0.464	**0.701**	**0.721**
Qwen 2.5 14B	Base	0.753	0.763	0.424	0.723	0.666
	Rescaling	0.738	0.517	0.463	0.506	0.556
	TaRot	**0.754**	**0.826**	**0.480**	**0.732**	**0.698**
Qwen 2.5 32B	Base	0.808	0.901	0.717	**0.788**	0.803
	Rescaling	0.803	0.892	0.625	0.593	0.728
	TaRot	**0.824**	**0.927**	**0.734**	0.767	**0.813**

Table 1: **Overall performance in zero-shot regime.** Performance of methods with different LLMs in terms of F1 scores are presented across different tasks and on average. **Bold-faced** number denote the best method. For Mistral-7B, Llama-3-8B, Qwen 2.5 14B and Qwen 2.5 32B, Eigen Pruning resulted in OOM and RED codebase is only compatible with LLaMA architecture. Further details in Appendix A.5

Method		AG News	Entailed polarity	Navigate	Winowhy	Avg.
Qwen2-1.5B	Base	0.680	**0.902**	0.173	0.393	0.537
	Rescaling	0.662	0.765	0.314	**0.576**	0.579
	TaRot	**0.695**	**0.902**	**0.494**	0.544	**0.659**
Phi-3-mini	Base	0.745	0.974	0.440	0.604	0.691
	Rescaling	0.732	0.980	0.196	0.562	0.618
	TaRot	**0.764**	**0.991**	**0.494**	**0.647**	**0.724**
Mistral-7B	Base	0.691	0.921	**0.236**	**0.790**	0.660
	Rescaling	**0.746**	0.698	0.196	0.580	0.555
	TaRot	0.684	**0.960**	0.196	**0.790**	**0.658**
Llama-3-8B	Base	0.524	0.950	0.645	0.651	0.693
	Rescaling	0.444	0.702	0.196	0.577	0.480
	TaRot	**0.638**	**1.000**	**0.727**	**0.761**	**0.782**
Qwen 2.5 14B	Base	0.749	0.868	0.527	0.691	0.709
	Rescaling	0.739	0.807	0.362	0.422	0.583
	TaRot	**0.752**	**0.888**	**0.605**	**0.759**	**0.751**
Qwen 2.5 32B	Base	0.877	0.950	0.791	0.647	0.816
	Rescaling	0.844	0.941	0.715	0.674	0.793
	TaRot	**0.882**	**0.966**	**0.802**	**0.688**	**0.835**

Table 2: **Overall performance in few-shot regime.** Performance of methods with different LLMs in terms of F1 scores are presented across different tasks (and on average). **Bold-faced** numbers denote the best methods.

Method		Imdb	Toxicity
Qwen2-1.5B	Base	0.677	0.566
	Rescale	0.252	0.161
	TaRot	0.708	0.581
Phi-3-mini	Base	0.707	0.536
	Rescale	0.686	0.416
	TaRot	0.749	0.564
Llama-3-8B	Base	0.708	0.571
	Rescale	0.669	0.566
	TaRot	0.729	0.579

Table 3: **Performance comparison on NLG tasks.** Performance of Imdb review and toxicity task. The reported score are the average of the fluency and reward scores. A higher score indicates better performance on both NLG tasks.

instance, in zero-shot Entailed Polarity prediction, Rescaling significantly outperforms both the base model and TaRot on Mistral-7B (Table 1), but fails to scale in the few-shot setting (Table 2) and deteriorates performance across most other models. Two factors explain this: (1) As discussed in Section 3.2, attention head token associations exist in superposed states, making direct scaling or ablation unreliable; (2) large fluctuations introduced by Rescaling hinder optimization. While Rescaling requires fewer parameters—H per layer vs. $\frac{d}{2}$ in TaRot —the difficulty arises from the polysemantic nature of OV-circuits. In some cases, downscaling all associations in a head helps, likely

Method		AG News	Average
Qwen2-1.5B	SFT	0.603	0.362
	TaRot	0.655	0.447
Phi-3-mini	SFT	0.677	0.745
	TaRot	0.738	0.614
Llama-3-8B	SFT	0.693	0.661
	TaRot	0.744	0.520

Table 4: **Generalizaibility of TaRot.** Performance of supervised fine-tuning (SFT) and TaRot when trained on the AG News dataset and evaluated on both AG News and the average of two other tasks (Winowhy and Navigate).

due to non-interacting associations, but this varies unpredictably across tasks and models. In contrast, TaRot 's rotational alignment offers fine-grained control and robust, consistent performance. Future work can develop a formal theoretical framework to directly compare rotation-based (TaRot) and rescaling-based interventions, potentially by analyzing their effects on the residual stream. In case of NLG tasks, the combined score of the individual task specific reward model and fluency, is higher for both the tasks across the models.We believe that combining reward model and fluency scores provides a more comprehensive evaluation – the reward model captures task alignment, and fluency ensures the outputs remain coherent and natural. This combination better reflects overall performance (details in Appendix A.4). Table 3 presents TaRot's results on IMDb and toxicity tasks, where it consistently outperforms both the base model and the rescaling approach. The reported scores reflect the combined metric, with higher values indicating better performance. On average, the performance of TaRot is improved on IMDb by 3.1% over the base model and 1.7% on toxicity tasks. However, on observing fluency and reward score separately. we see that solely in terms of reward values, Rescaling performs better than TaRot, and both interventions perform better than the original model. However, TaRot delivers more fluent response in terms of evaluation by GPT-4, pointing towards the more drastic edits of rescaling as compared to TaRot (see Appendix A.4 for complete results).

Generalizability of TaRot. TaRot applies fine grained intervention to the model attention heads, without altering the performance on remaining tasks. To show this, we perform supervised fine tuning(SFT), keeping the size of the train set similar to that of TaRot. We choose AG news as the train task as this is the only multi-class classification problem. Table 4 compares the performance of SFT and TaRot trained on the AG News dataset across three different models: Qwen2-1.5B, Phi-3-mini, and Llama-3-8B. The results indicate that TaRot outperforms SFT on the trained tasks. We observe strong generalization in smaller models but weaker gains in larger ones (e.g., LLaMA-3-8B), due to two main factors: (1) TaRot removes spurious features but cannot inject new task-specific or syntactic knowledge; (2) the high-dimensional rotation space in larger models makes optimization harder. Moreover, since all tasks are classification-based, SFT, being explicitly task-driven, offers stronger supervision, reducing TaRot's relative impact at scale. This explains the performance boost from SFT even on unseen tasks. Future work can focus on enhancing TaRot 's robustness for larger models.

Ablation studies. To assess the robustness of our approach, we conduct two ablation studies. **(1) Hyperparameter $\mathbb{L}$:** Using the Qwen 2.5-14B model, we apply the rotation transformation on varying layers of the model by changing the hyperparameter $\mathbb{L}$. Results show that applying Bayesian optimization to the initial layers yields the best performance with minimal parameters (see Appendix A.6.1 for details). **(2) System Prompt:** To guide the model towards accurate outputs, we use a fixed system prompt per task. We evaluate TaRot's robustness to prompt variation on AG News using Qwen 2.5-14B with three different prompts. Despite fluctuations in base model performance, TaRot consistently outperforms it and reduces variance. Full results are in Appendix A.6.2.

6 Conclusion

In this work, we proposed TaRot, a novel, gradient-free, mechanistic intervention method for editing language models. TaRot builds on observations from implicit gradient descent bias of causal attention and applies parametrized rotation on the attention output to minimize the effects of undesired memorizations, doing away with effort-intensive localization steps and task-specificity of prior intervention techniques. Using Bayesian optimization of the rotational parameters, TaRot renders as data-efficient as in-context learning; yet, across a variety of tasks and language models of different sizes and families, robust improvement is observed. In a nutshell, TaRot can pave the path for general-purpose model editing methods in the future beyond supervised fine-tuning.

7 Limitations and Ethical Considerations

TaRot is designed to perform when the model has a generalization ability that is suppressed by noisy memorization. In that sense, it is limited by the boundaries of pretraining and cannot be used for domain adaptation. Fundamentally, it is not applicable to proprietary models. Finally, similar to any intervention technique, TaRot can be used in reverse to bypass alignment tuning and reinforce undesired behaviors. Future work can be to address the potential misuse of TaRot for bypassing alignment. One can develop detection mechanisms to identify whether TaRot or similar transformations have been applied to manipulate a model's behavior. Incorporating regularization strategies that penalize rotations leading to toxic, biased, or otherwise misaligned generations would further ensure that the optimization process remains consistent with ethical AI principles. These directions can help mitigate potential misuse of TaRot and similar model editing techniques.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

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

Adithya Bhaskar, Dan Friedman, and Danqi Chen. 2024. The heuristic core: Understanding subnetwork generalization in pretrained language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14351–14368, Bangkok, Thailand. Association for Computational Linguistics.

BIG-bench authors. 2023. Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Subhabrata Dutta, Joykirat Singh, Soumen Chakrabarti, and Tanmoy Chakraborty. 2024. How to think step-by-step: A mechanistic understanding of chain-of-thought reasoning. *arXiv preprint arXiv:2402.18312*.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12.

Javier Ferrando, Gabriele Sarti, Arianna Bisazza, and Marta R. Costa-jussà. 2024. A primer on the inner workings of transformer-based language models. *Preprint*, arXiv:2405.00208.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*.

Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah Goodman. 2024. Finding alignments between interpretable causal variables and distributed neural representations. In *Proceedings of the Third Conference on Causal Learning and Reasoning*, volume 236 of *Proceedings of Machine Learning Research*, pages 160–187. PMLR.

Sreyan Ghosh, Chandra Kiran Reddy Evuru, Sonal Kumar, Ramaneswaran S, Deepali Aneja, Zeyu Jin, Ramani Duraiswami, and Dinesh Manocha. 2024. A closer look at the limitations of instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 15559–15589. PMLR.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *Preprint*, arXiv:2311.05232.

Samyak Jain, Robert Kirk, Ekdeep Singh Lubana, Robert P. Dick, Hidenori Tanaka, Edward Grefenstette, Tim Rocktäschel, and David Scott Krueger. 2024. Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks. *Preprint*, arXiv:2311.12786.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2. Minneapolis, Minnesota.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. 2022. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*.

Max Lamparth and Anka Reuel. 2024. Analyzing and editing inner mechanisms of backdoored language models. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '24, page 2362–2373, New York, NY, USA. Association for Computing Machinery.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. 2017. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*.

Chak Tou Leong, Yi Cheng, Jiashuo Wang, Jian Wang, and Wenjie Li. 2023. Self-detoxifying language models via toxification reversal. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4433–4449, Singapore. Association for Computational Linguistics.

Maximilian Li, Xander Davies, and Max Nadeau. 2024a. Circuit breaking: Removing model behaviors with targeted ablation. *Preprint*, arXiv:2309.05973.

Yucen Lily Li, Tim G. J. Rudner, and Andrew Gordon Wilson. 2024b. A study of bayesian neural network surrogates for bayesian optimization. In *The Twelfth International Conference on Learning Representations*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Ang Lv, Yuhan Chen, Kaiyi Zhang, Yulong Wang, Lifeng Liu, Ji-Rong Wen, Jian Xie, and Rui Yan. 2024. Interpreting key mechanisms of factual recall in transformer-based language models. *Preprint*, arXiv:2403.19521.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. In *Advances in Neural Information Processing Systems*, volume 35, pages 17359–17372. Curran Associates, Inc.

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. *Preprint*, arXiv:2209.11895.

Nikhil Prakash, Tamar Rott Shaham, Tal Haklay, Yonatan Belinkov, and David Bau. 2024. Fine-tuning enhances existing mechanisms: A case study on entity tracking. In *The Twelfth International Conference on Learning Representations*.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671, Seattle, United States. Association for Computational Linguistics.

V Sanh. 2019. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *The Twelfth International Conference on Learning Representations*.

Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. 2023. Large language model alignment: A survey. *Preprint*, arXiv:2309.15025.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Eshaan Tanwar, Subhabrata Dutta, Manish Borthakur, and Tanmoy Chakraborty. 2023. Multilingual LLMs are better cross-lingual in-context learners with alignment. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6292–6307, Toronto, Canada. Association for Computational Linguistics.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Tomás Vergara-Browne, Álvaro Soto, and Akiko Aizawa. 2024. Eigenpruning: an interpretability-inspired peft method. *Preprint*, arXiv:2404.03147.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2023. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*.

Mengru Wang, Ningyu Zhang, Ziwen Xu, Zekun Xi, Shumin Deng, Yunzhi Yao, Qishen Zhang, Linyi Yang, Jindong Wang, and Huajun Chen. 2024a. Detoxifying large language models via knowledge editing. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3093–3118, Bangkok, Thailand. Association for Computational Linguistics.

Zhichao Wang, Bin Bi, Shiva Kumar Pentyala, Kiran Ramnath, Sougata Chaudhuri, Shubham Mehrotra, Zixu, Zhu, Xiang-Bo Mao, Sitaram Asur, Na, and Cheng. 2024b. A comprehensive survey of llm alignment techniques: Rlhf, rlaif, ppo, dpo and more. *Preprint*, arXiv:2407.16216.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Jiaxin Wen, Pei Ke, Hao Sun, Zhexin Zhang, Chengfei Li, Jinfeng Bai, and Minlie Huang. 2023. Unveiling the implicit toxicity in large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

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. *Preprint*, arXiv:2402.15179.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Yuexiang Zhai, Shengbang Tong, Xiao Li, Mu Cai, Qing Qu, Yong Jae Lee, and Yi Ma. 2024. Investigating the catastrophic forgetting in multimodal large language model fine-tuning. In *Conference on Parsimony and Learning*, volume 234 of *Proceedings of Machine Learning Research*, pages 202–227. PMLR.

Hongming Zhang, Xinran Zhao, and Yangqiu Song. 2020. WinoWhy: A deep diagnosis of essential commonsense knowledge for answering Winograd schema challenge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5736–5745, Online. Association for Computational Linguistics.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2024. Instruction tuning for large language models: A survey. *Preprint*, arXiv:2308.10792.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

A Appendix

A.1 Task Details

We experimented with five different classification (i.e., single token generation) tasks and two NLG tasks. Below are the details of the tasks with their prompt templates used:

AG News: The goal of the task is to categories new articles into one of the four predefined categories.

- World – News about global events, international politics, and worldwide issues.
- Sports – News related to sporting events, athletes, competitions, and sports industry developments.
- Business - News focusing on the economy, financial markets, companies, and business trends.
- Science & Technology – News about technological advancements, scientific discoveries, and research.

System prompt used for AG News task: *You are a news classification model. Your task is to classify news articles into one of the following four categories: World, Sports, Business, or Science. You should respond with only the category name and no other characters.*

Entailed polarity: The Entailed Polarity task is a yes/no question-answering task (Srivastava et al., 2022). Given a fact and a question, the goal is to determine whether the fact entails a yes or no answer to the question. The task tests the model's ability to infer whether the factual statement logically supports the answer in terms of polarity (positive or negative). Example:

- Fact: "Ed remembered to go."
- Question: "Did Ed go?"
- Answer: "Yes"

System prompt used for Entailed Polarity task: *Follow the instructions below and answer with Yes / No.*

Navigate: The objective is to follow a set of directional or spatial instructions and determine if, after following those steps, the entity returns to the starting point. The answer is either True or False, depending on whether the instructions guide the entity back to where they started. Example:

- Instruction: "If you follow these instructions, do you return to the starting point?"
- Steps: "Always face forward.", "Take 7 steps left.", "Take 2 steps backward.", "Take 7 steps backward.", "Take 7 steps backward.", "Take 3 steps forward."
- Question: "Do you return to the starting point?"
- Answer: False

System prompt used for the task: *Answer the following question and output only True/False.*

Winowhy: This task (Srivastava et al., 2022) requires models to identify the correct reasons behind the answers to the Winograd Schema Challenges(Zhang et al., 2020).

This task is based on the original Winograd Schema Challenge (WSC) dataset and 4095 WinoWhy reasons (15 for each WSC question) that could justify the pronoun coreference choices in WSC. The model is presented with a passage that contains a pronoun and an explanation of which word or entity the pronoun refers to. The model's job is to assess whether the explanation given is correct or incorrect based on the context of the passage.

- Text: "Fred is the only man alive who still remembers my father as an infant. When Fred first saw my father, he was twelve years old. The 'he' refers to Fred because, in his own words, he is 'a very odd man'."
- Question: "The above reasoning is:"
- Answer: "Incorrect".

System prompt used for Winowhy task: *Follow the instructions and output Correct/Incorrect.*

Imdb: Tune model to generate positive movie reviews using a BERT (Kenton and Toutanova, 2019) sentiment classifier as a reward function. The reward model evaluates the sentiment of the generated reviews, and the goal is to maximize the likelihood of generating reviews classified as positive.

- Dataset Used: imdb (Maas et al., 2011)
- Reward Model: lvwerra/distilbert-imdb, a fine-tuned version of distilbert-base-uncased (Sanh, 2019) on the imdb dataset.

Detoxify: Involves reducing the toxicity of language model outputs. The toxicity evaluation is done using a classifier, such as facebook/roberta-hate-speech-dynabench-r4-target, which distinguishes between "neutral" and "toxic" text. The classifier provides feedback (reward or penalty) based on the toxicity of the model's output, guiding the model to produce less toxic text. The dataset used is allenai/real-toxicity-prompts (Gehman et al., 2020).

A.2 Experimental Setup Details

Bayesian optimization. We use I-BNN with 12 hidden layers, and LogExpectedImprovement as the acquisition function. We use a mixture of M-shots generation to avoid biasing the intervention, with M chosen randomly from 0 to 6. Each task was optimized for 150 iterations.

Baselines. We compare `TaRot` with four different baselines: (1) *Base model* denotes the pretrained LLM (zero-shot or few-shot) without any interventions. (2) *Eigen Pruning* (Vergara-Browne et al., 2024) removes singular values from weight matrices in an LLM to improve its performance in a particular task. (3) *RED (Representation EDiting)* (Wu et al., 2024), which modifies the representations generated at some layers through the application of scaling and biasing operations. To

have a fair comparison, we also use a maximum of 20 prompts in its training phase. (4) *Rescaling* ablates attention heads by scaling their output in the unit interval instead of rotating their outputs; we use the same optimization technique to figure out the optimal scaling configuration.

Evaluation metrics. For Imdb positive review tasks, a sentiment analysis reward model, `lvwerra/distilbert-imdb`[3] is used. `Roberta-hate-speech-dynabench-r4-target`[4] is used for detoxification. For fluency GPT4 (Achiam et al., 2023) is used as an oracle to assign a value between 1 and 5, 1 being the least and 5 being the highest.

A.3 Fluency

To evaluate the fluency of a given text, the following prompt was used with GPT4 (Achiam et al., 2023):

System prompt used: *Please rate the fluency of the following text on a scale of 1 to 5, where 1 is least fluent and 5 is most fluent: text. Provide only the number.*

where text is the output from the model.

A.4 NLG tasks performance

Table 3 presents the performance of TaRot on IMDB sentiment classification and toxicity detection, where it consistently outperforms both the base model and rescaling-based methods. The table reports a combined score of fluency and reward model outputs, where a higher score indicates better performance for both tasks.

A.4.1 Combing score of reward mode and fluency

Evaluation Setup of the two NLG tasks and fluency is described below:

- **Fluency:** Assessed using GPT-4, which assigns a score from 1 to 5 (where 1 = least fluent and 5 = most fluent).
- **IMDB Sentiment Reward Model:** We use lvwerra/distilbert-imdb, where higher scores indicate better sentiment classification.
- Toxicity Reward Model: We use RoBERTa-hate-speech-dynabench-r4-target, where higher scores indicate higher toxicity.

[3] https://huggingface.co/lvwerra/distilbert-imdb

[4] https://huggingface.co/facebook/roberta-hate-speech-dynabench-r4-target

Method	Imdb		Detoxify	
	Reward	Fluency	Reward	Fluency
`Qwen2-1.5B`				
Base	-0.80	—	4.50	—
Rescaling	**0.72**	1.25	**2.29**	1.26
`TaRot`	-0.25	**2.24**	4.01	**4.56**
`Mistral-7B`				
Base	-0.05	—	4.31	—
Rescaling	**0.19**	2.12	**3.18**	4.12
`TaRot`	0.16	**2.5**	4.01	**4.30**
`Llama-3-8B`				
Base	-0.31	—	4.05	—
Rescaling	**0.28**	**2.56**	**3.18**	**4.76**
`TaRot`	.002	2.38	3.90	4.24

Table 5: Performance comparison on NLG tasks.

The scoring methodology of the NLG tasks and fluency combined is described below:

- **IMDB + Fluency:** Both scores were normalized to [0,1] and summed to obtain the final score.
- **Toxicity + Fluency:** The toxicity score was normalized and inverted (so that lower toxicity results in a higher score), then combined with fluency.

Thus, in both cases, a higher final score reflects improved overall performance (i.e., better fluency and alignment with task objectives). The complete breakdown of the toxicity and fleuncy of the NLG tasks is shown in Table 5.

A.5 Additional baselines

Eigenpruning and RED were used as baselines in our study. Below, we outline the key challenges that prevented us from incorporating additional baselines:

Eigenpruning:

- Eigenpruning requires fine-tuning the model before identifying circuits, which is a computationally expensive process.
- Given our resource constraints, we were unable to perform the necessary fine-tuning steps required for a fair comparison.

RED:

- The methodology and code for RED have only been demonstrated on GPT-2 and LLaMA models.
- The publicly available codebase lacks implementation details for extending RED to other model architectures.

Method	Layer	AG News	Navigate	Winowhy
Base	NA	0.753	0.424	0.723
TaRot	0 - 6	0.752	0.45	0.723
TaRot	0 - 12	**0.757**	0.432	0.715
TaRot	0 - 24	0.754	**0.480**	**0.732**
TaRot	0 - 32	0.743	0.439	0.712

Table 6: Zero Shot Performance with TaRot applied on different layers.

Method	Layer	AG News	Navigate	Winowhy
Base	NA	0.749	0.527	0.691
TaRot	0 - 6	0.755	0.549	0.733
TaRot	0 - 12	0.75	0.556	0.73
TaRot	0 - 24	0.752	**0.605**	**0.759**
TaRot	0 - 36	**0.757**	0.601	0.735

Table 7: Few Shot Performance with TaRot applied on different layers.

As a result, we were only able to run RED on the LLaMA model for comparison.

A.6 Ablation Study

A.6.1 Hyperparameter $\mathbb{L}$

Previous studies indicate that token associations related to pretrained knowledge primarily reside in the first half of the model. Based on this insight, we applied the rotation transformation only to the first half of the model. However, we acknowledge that an ablation study on this hyperparameter is necessary to fully assess the robustness of our approach.

To address this, we conducted an ablation study on the Qwen 2.5-14B model, which has 48 layers. We tested different layer ranges for applying the rotation matrix: 0-6, 0-12, 0-24, and 0-36. The tables below report zero-shot and few-shot F1 scores across three tasks: AG News, Navigate, and Winowhy. Table 6, 7 shows the performance zero shot and few shot performance respectively.

Key Observations

- For Navigate and Winowhy, the best performance was achieved when applying TaRot to the first 24 layers (0-24).
- Ideally, 0-32 layers should also perform well, but the increased parameter space dimensionality makes it harder for Bayesian optimization to converge effectively.
- For task AG news we see comparable performance of TaRot optimized on the first half of the model with the best performing setting.

Method	Prompt 1	Prompt 2	Prompt 3
Base (Zero Shot)	0.724	0.815	0.73
TaRot (Zero Shot)	**0.772**	**0.832**	**0.768**
Base (Few Shot)	0.67	0.792	0.697
TaRot (Few Shot)	**0.7**	0.792	**0.715**

Table 8: Ablation of different prompt used for Qwen/Qwen2-1.5B-Instruct on AG News Tasks.

Therefore we see that applying optimization on the first half provides with us with the best performance.

A.6.2 System Prompt

We tested the model with three semantically equivalent but syntactically different prompts. TaRot was optimized on each of these prompts separately to evaluate its effectiveness in mitigating performance fluctuations.

Prompt 1:
System prompt: You specialize in classifying news articles into distinct categories. Given a news article, determine whether it belongs to World, Sports, Business, or Science. Only provide the category name as a response.
Question: News Content: <review>
Query: What is the most suitable category for this news piece?

Prompt 2:
System prompt: You are an expert in news topic classification. Your role is to analyze articles and assign them to one of these four categories: Sports, World, Business, or Science. Do not add extra text—respond with just the category name.
Question: Text: <review>
Question: Which of the four categories (World, Sports, Business, or Science) does this article belong to?

Prompt 3:
System prompt: Your function is to categorize news articles into one of four groups: World, Sports, Business, or Science. Given a news article, determine its category and respond using only the category name.
Question: News Article: <review>
Task: Identify the correct category for this article.

Table 8 shows the performance of TaRot in zero and few shot settings compared with the base model. The model used is Qwen/Qwen2-1.5B-Instruct and the dataset is Ag News. The results demonstrate that the base model exhibits considerable fluctuation across different prompts, indicating a high sensitivity to prompt phrasing. In

contrast, TaRot consistently outperforms the base model across all prompt settings, showcasing its reliability and robustness. This consistency highlights TaRot's ability to generalize better and remain stable despite variations in input structure. Notably, while the base model suffers a significant absolute drop of 0.145 in few-shot performance between prompts (from 0.815 to 0.67), TaRot substantially minimizes such performance degradation. This suggests that TaRot enhances the model's resilience to prompt perturbations, reducing brittleness and improving reliability. Furthermore, by demonstrating improved performance across varied prompt templates, the experiment effectively addresses the reviewer's concern—confirming that TaRot's improvements are not limited to a single prompt instance but instead generalize across different prompt structures.

A.7 System Prompt Used

For each system custom system prompts were used to help guide the model to output the final answer directly. We ensured that system prompts were only used when necessary—for instance, they were not applied in tasks like Entailed Polarity, where the model naturally follows the desired output structure, i.e output the final answer directly. The system prompt used for each tasks are as follows:

- **Task Navigate:** Prompt: "If you follow these instructions, do you return to the starting point?"
- **Task Entailed Polarity:** Prompt: "Given a fact, answer the following question with a yes or a no."
- **Task Winowhy:** Prompt: "Please answer the following questions about which words certain pronouns refer to."
- **Task AG News:** Prompt: "News Article: review: Question: What category does this news article belong to?"