Scalable Gradient-Based Attribution of LLM Behaviors

Anonymous Authors

Abstract

Large language models often exhibit unexpected behaviors arising from complex interactions between training data and optimization, making it challenging to diagnose and control model behavior. Existing data attribution methods, such as influence-function–based approaches, are often computationally and memory intensive, and are not easily scalable to multiple evaluation tasks. We propose an efficient representation-space data attribution framework that stores only activation-sized vectors per training example and supports attribution to new evaluation behaviors without recomputing training-set statistics. We apply our method to Llama-3.1-8b-Instruct on multiple instruction tuning training datasets, and evaluation tasks including hallucination, medical factuality, and personality-style behaviors. Across these settings, our method often identifies training data more strongly associated with target behaviors than comparably efficient alternatives such as persona vectors and weight-space dot product with dimension reduction.

1 Introduction

Finetuning large language models (LLMs) often induces unusual behaviors not immediately present in the training data, such as models exhibiting evil and antisocial traits after training on code [BTW+25] or models exhibiting an affinity for owls after training on strings of numbers [CLC+25]. As models become more capable, they increasingly exhibit unexpected behaviors [MTB+25], creating a need for efficient data attribution methods. In this paper, we develop our attribution framework, a framework that can efficiently select a behavior-inducing subset of a large training corpus, such as Ultrachat or OpenOrca, across a diverse collection of evaluation behaviors, such as laziness or hallucinations.

Conventional approaches for data attribution operate in weight-space, as they estimate how a model's parameters (and therefore its behaviors) change if a particular example is added to the training data. Such approaches calculate the second-order approximation to the gradient update by computing the dot products between the gradients of the training data and the gradients of the evaluation data. Unfortunately, this calculation requires significant space or time costs. To save space, [GBA+23] store low-rank approximations to the evaluation gradients, but this can only offer at most $\sim$1000$\times$ space savings, so the method would still require at least 10$\times$ model-size storage to scale to hundreds of evaluation behaviors. In response, [WMSJ25] use ghosted dot products to avoid materializing the gradients, but this approach requires recomputing the finetuning run for every new evaluation behavior.

To obtain a scalable data attribution procedure, we operate directly in activation space rather than parameter space. Consider a training example x and an evaluation example or behavior specification b, and let $\mathbf{z}_x^l$ and $\mathbf{z}_b^l$ denote their activations at a fixed residual block l. Our goal is to quantify how much training on x would influence the internal representations associated with behavior b, without performing explicit fine-tuning or full parameter-gradient comparisons.

Under a first-order approximation, the effect of a training example on downstream behavior can be expressed as an inner product between two activation-sized vectors: a training-side vector that captures how x would locally shift internal representations under a gradient update, and an evaluation-side vector that encodes the direction in activation space associated with the target behavior. Concretely, we approximate the attribution score by a dot product

$$s(x, b) \;\approx\; \langle \phi(x), \psi(b) \rangle,$$

where $\phi(x)$ depends on the activation and loss gradient induced by x, and $\psi(b)$ summarizes the corresponding behavior-specific activation direction.

Crucially, this formulation requires only gradients with respect to activations at a chosen interface, rather than gradients with respect to model parameters. As a result, training-side representations $\{\phi(x)\}$ can be computed once and reused across many evaluation behaviors, enabling attribution to scale to large training corpora and diverse behavioral queries. We explore several concrete instantiations of ϕ and ψ within this framework, and find that a gradient-based approximation to the residual-block activation change—our proposed *residual-change-treatment* method—yields the strongest empirical performance.

We quantitatively evaluate a concrete instantiation of our attribution method across multiple training corpora and behavioral benchmarks. First, we show that our attribution method consistently identifies problematic training data that degrades factual reliability and instruction-following behavior relative to other instances of our attribution framework. On UltraFeedback-based factual truthfulness and coding instruction-following tasks, our attribution method yields up to a 35% improvement in trait-eliciting score over strong baselines such as persona vectors. On MedHallu, a medical hallucination benchmark with explicitly provided evidence, our attribution method is also particularly effective at identifying corrupted training data: fine-tuning on data selected by our method reduces hallucination scores by approximately 150%–300% relative to the strongest baseline, depending on the training corpus. In the personality trait setting, where persona vectors are given an inherent advantage through direct access to the trait-defining system prompts, residual change treatment still achieves trait-eliciting scores that are 10–30% higher than alternative baselines.

2 Related Work

Approaches to data attribution. Data attribution is used to diagnose how training data affects model behavior. The core quantity is an *influence score*: the effect of a single training example on a particular model prediction or evaluation behavior. Broadly, existing approaches to estimating influence fall into three categories: retraining-based, parameter-based, and activation-based.

Retraining-based methods estimate the causal effect of a data point by retraining on different subsets of the training data (e.g., Shapley-style estimators and related data valuation methods) [GZ19, JDW$^+$19, IPE$^+$22, WMSJ25]. Because they directly measure behavioral change under counterfactual datasets, these methods are conceptually clean and often serve as a gold standard in small regimes. In practice, they have been used for dataset valuation and curation (e.g., identifying low-value or harmful examples) and for diagnosing model failures, but their cost typically makes them infeasible at modern scales.

Parameter-based methods instead approximate how the model parameters would change if a training point were upweighted or removed, using techniques such as representer points [YKYR18], gradient-based approximations such as TracIn [PLKS20], and influence functions [KL17]. Influence functions in particular have been extended to estimate the effect of entire subsets of data [KATL19] and to incorporate higher-order approximations [BYF20]. These advances have enabled a range of applications, including filtering noisy training data [KL17], constructing poisoned data [KSL22], augmenting training data [LPPY20, OKRK21], and improving downstream performance [HT21, DZMP25]. However, a recurring challenge is that parameter change is only an indirect proxy for the behavioral changes we ultimately care about.

Activation-based methods are a more recent alternative that operate in representation space rather than parameter space. Empirically, [LZLS24] evaluate several influence-function approaches and find that while they can perform well in simple regimes (e.g., highly templated datasets), a simple activation-similarity baseline is often surprisingly strong. One plausible explanation is that influence functions estimate *parameter change*, which may be a poor proxy for *behavioral change*—especially when behaviors are mediated by localized or distributed representations that do not correspond cleanly to small parameter perturbations. Since our goal is to understand and attribute model behaviors, this motivates activation-based attribution. Recent work on uncovering model personas [CAS$^+$25, WlTW$^+$25] similarly suggests that representation similarity can identify relevant training data, and [LZLS25] also considers gradients for representation-based approaches. These methods are attractive because they require substantially less computation than retraining-based or parameter-based attribution. Our work builds on these ideas by providing a framework for scaling activation-based data attribution to dozens of evaluation behaviors.

Scaling data attribution methods. Scaling data attribution introduces several computational bottlenecks. Recent work has proposed more efficient parameter-based estimators: TRAK [PGI+23] linearizes the model via the neural tangent kernel and computes attribution using random projections of per-example weight gradients; [GBA+23] introduces heuristics such as TF–IDF to restrict attribution to candidate sequences identified through keyword matching; and in-run data Shapley [WMSJ25] uses ghosted dot products to avoid explicitly materializing full gradient matrices. Despite these advances, most parameter-space methods still face an unfavorable trade-off between space and time: they either incur substantial storage costs (e.g., retaining projections or per-example statistics) or require repeated model passes for each new evaluation behavior. TRAK (when applied to a fixed base model) is the primary exception to this pattern; we include it as a baseline and find that it underperforms our approach.

Motivated by these limitations, we focus on activation-based methods, which are inherently more scalable: once representations are computed, many behaviors can be evaluated with lightweight similarity computations. In addition, our attribution method improves data attribution performance relative to persona vectors, the current SOTA activation-based approach, while naturally supporting attribution across dozens of evaluation behaviors.

3 Framework and Methods

In this section we introduce our framework and methods to compute attribution scores of a training sample for an evaluation dataset.

3.1 A Unified Framework for Scalable, Task-Agnostic Data Attribution

We aim to identify training examples most associated with a target evaluation behavior (e.g., factuality, sycophancy) across many behaviors without repeatedly recomputing training-set statistics. A key obstacle is that different attribution methods expose different "attribution objects," with pipelines often entangled with specific evaluation tasks.

We propose a unified framework separating (i) one-time, per-training-example statistic computation, (ii) lightweight, per-evaluation-behavior statistic computation, and (iii) a modular scoring rule. This decomposition is not tied to any particular attribution method and subsumes many existing methods as special cases.

Framework. Let $\mathcal{D}_{\text{train}} = \{x_i\}_{i=1}^{N}$ denote the training set and $\mathcal{E}$ an evaluation set or behavior specification. Our framework defines: (i) a *training-side representation* $\phi_i \triangleq \phi(x_i) \in \mathbb{R}^d$ cached for each training example; (ii) an *evaluation-side representation* $\psi_\mathcal{E} \triangleq \psi(\mathcal{E}) \in \mathbb{R}^d$ computed per behavior; and (iii) a *scoring functional* $g : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ yielding attribution score $s(i;\mathcal{E}) \triangleq g(\phi_i, \psi_\mathcal{E})$. The key property is that $\{\phi_i\}_{i=1}^{N}$ are computed once and reused across all behaviors, while $\psi_\mathcal{E}$ is computed cheaply for each new behavior. The scoring function g (e.g., dot product, cosine similarity) does not require recomputing training-side statistics.

Instantiation. Our method (Sections 3.2 and 3.3) instantiates ϕ_i as an activation-sized vector capturing how x_i would shift internal representations under a gradient update, and $\psi_\mathcal{E}$ as the corresponding evaluation direction. Attribution to a new behavior reduces to forming $\psi_\mathcal{E}$ and computing $s(i;\mathcal{E})$ for retrieval.

Relationship to prior work. Many existing methods fit this interface with different (ϕ, ψ, g) choices. Persona vectors [CAS+25] construct $\psi_\mathcal{E}$ in activation space and rank by similarity. Parameter-space methods like TRAK [PGI+23] define these objects via gradients, with different computational tradeoffs. Casting both families in this interface clarifies what is cached once versus computed per behavior, and makes comparing different attribution methods easier by isolating the effect of each component.

This decomposition provides two immediate benefits: (1) *scalability*—attribution to new behaviors requires only computing $\psi_{\mathcal{E}}$ and vector similarity over N examples; and (2) *modularity*—ϕ, ψ, and g can be varied independently without recomputing training statistics, supporting fair ablations.

3.2 Gradient-Based Residual-Change Attribution

Prior activation-difference baselines often use the residual-stream difference between a training example and a separately constructed *control* message (e.g., a natural model response) as a proxy for the effect of fine-tuning on that example [CAS+25]. This has two practical limitations for our setting: (i) it requires constructing control messages (and, in general, regenerating them for each base model), and (ii) the resulting residual difference is not explicitly derived as a first-order approximation to the activation *change* induced by a gradient update. A related alternative is to take the activation difference between a base model and a separately fine-tuned trait-exhibiting model [WlTW+25], which can be accurate but requires constructing such a trait-exhibiting model, and thus incurs additional training overhead.

We propose *residual-change-treatment*, which approximates the activation change at a specific residual block under a single local gradient step on the same training example, without using any control message. Because a residual block contains attention, MLPs, normalization, and residual addition, the exact mapping from inputs to outputs is architecturally complex and can vary across model families. To obtain a simple, model-agnostic estimator, we adopt a local linear approximation of the block input–output interface: we treat the block as if it were a linear layer acting on the block input activation. This approximation is motivated by the empirical observation that, within a fixed model, the transformation between sequential layer activations can often be well-approximated by a linear map [RMG+24]. Our estimator is therefore a first-order surrogate that depends only on quantities available at the chosen interface (activations and their loss gradients), and can be computed consistently across different base models.

Formally, let $z^l_{in,x} \in \mathbb{R}^d$ and $z^l_{out,x} \in \mathbb{R}^d$ denote the residual-block input and output activations at layer l when evaluating training example x, and let ℓ_x be its training loss. We approximate the block locally by a linear map

$$z^l_{out} \approx z^l_{in} W^l,$$

with surrogate weight matrix $W^l \in \mathbb{R}^{d\times d}$. Under one gradient step with learning rate η,

$$\Delta W^l \;=\; -\eta \frac{\partial \ell_x}{\partial W^l}.$$

For the linear surrogate, $\frac{\partial \ell_x}{\partial W^l} = (z^l_{in,x})^\top g^l_x$ where

$$g^l_x \;:=\; \frac{\partial \ell_x}{\partial z^l_{out}} \in \mathbb{R}^d,$$

so

$$\Delta W^l \;=\; -\eta (z^l_{in,x})^\top g^l_x.$$

Re-evaluating the same example x after this local update, the induced change in the block output is

$$\Delta z^l_{out,x} \;\approx\; z^l_{in,x} \Delta W^l \;=\; -\eta\, z^l_{in,x} (z^l_{in,x})^\top g^l_x.$$

To decouple attribution scores from fine-tuning hyperparameters, we drop the global scalar η and define the layer-l residual-change vector

$$t^l(x) \;:=\; -\, z^l_{in,x} (z^l_{in,x})^\top g^l_x \;=\; -\, z^l_{in,x}\, \langle z^l_{in,x}, g^l_x \rangle.$$

This produces an activation-sized vector per training example that targets a first-order approximation to the activation change at layer l induced by a single gradient step on that example, while avoiding any dependence on separately generated control messages.

3.3 Behavior Query Construction

Having defined training-side statistics ϕ_i via residual-change vectors (Section 3.2), we now describe how to construct the behavior query vector $\psi_{\mathcal{E}}$ at a fixed layer l. Let $\mathcal{E} = \{(x'_j, x_j)\}_{j=1}^{|\mathcal{E}|}$ be an evaluation set of matched treatment/control message pairs, where x'_j exhibits the target behavior and x_j does not, and let $\operatorname{norm}(v) \triangleq v/\|v\|_2$.

We study three query constructions:

Residual-change query. Using the gradient-based residual-change vector $t^l(\cdot)$ from Section 3.2, we define

$$\psi_{\mathcal{E}}^{\Delta,l} \triangleq \operatorname{norm}\left(\frac{1}{|\mathcal{E}|}\sum_{j=1}^{|\mathcal{E}|}\left(t^l(x'_j) - t^l(x_j)\right)\right),$$

which captures the difference in predicted activation change between treatment and control, isolating how the behavior modulates the first-order update-induced representation shift.

Residual-difference query. A simpler alternative uses raw activation differences:

$$\psi_{\mathcal{E}}^{\text{diff},l} \triangleq \operatorname{norm}\left(\frac{1}{|\mathcal{E}|}\sum_{j=1}^{|\mathcal{E}|}\left(z^l_{\text{out},x'_j} - z^l_{\text{out},x_j}\right)\right),$$

summarizing the behavior by the purely activation-based contrast between treatment and control.

Persona-vector query. Following [CAS+25], letting p' and p denote positive and neutral system prompts respectively:

$$\psi_{\mathcal{E}}^{\text{persona},l} \triangleq \operatorname{norm}\left(\frac{1}{|\mathcal{E}|}\sum_{j=1}^{|\mathcal{E}|}\left(z^l_{\text{out}}\,|p' - z^l_{\text{out}}\,|p\right)\right),$$

i.e., the average representation shift induced by swapping system prompts during generation.

Empirically, pairing our training-side statistic with $\psi_{\mathcal{E}}^{\Delta,l}$ or $\psi_{\mathcal{E}}^{\text{diff},l}$ improves attribution over baselines; the best choice is task-dependent (residual-change often performs well, while MedHallu favors residual-difference; see Section 4). A key advantage of our framework is that exploring multiple query constructions is cheap once training-side statistics are cached.

4 Experiments

We evaluate our data attribution framework across multiple training corpora and behavioral evaluation tasks, with the goal of assessing how well different attribution methods identify training examples that causally influence specific model behaviors. All experiments are conducted on `Llama-3.1-8B-Instruct`, using identical fine-tuning and evaluation protocols across methods unless otherwise stated. The details of our experimental findings can be found in Sections 4.1 to 4.3

Training datasets. We consider three widely used instruction-tuning datasets of varying scale and composition: *Dolly-10k*, *UltraChat-200k*, and *OpenOrca-200k*. These datasets differ substantially in size, domain coverage, and instruction style, allowing us to test whether attribution methods remain effective across heterogeneous training distributions. Unless otherwise specified, attribution scores are computed with respect to each individual training dataset independently.

Evaluation datasets. We evaluate attribution quality on four behavior-focused benchmarks: *MedHallu*, *UltraFeedback Coding Instruction Following*, *UltraFeedback Factual Hallucination*, and *Personality Traits*. These datasets are designed to probe concrete failure modes such as hallucination and instruction non-compliance, rather than generic language modeling performance.

Each evaluation example is constructed as a *contrastive pair* consisting of a treatment message and a control message that share the same user prompt but differ in the assistant response. The treatment response exhibits the behavioral trait of interest (e.g., factual hallucination or poor instruction following), while the control response does not, or exhibits the opposite behavior (e.g., a correct or faithful answer). This contrastive structure allows us to isolate internal representations associated with the trait under study by controlling for the user prompt and surface form.

Baselines. We compare against attribution methods operating in a comparable computational regime: *TRAK*, *persona vectors*, and *random selection*. TRAK computes weight-space influence via full backward passes, and in a naive implementation requires per-device batch size one. In our setup, this makes TRAK approximately 6× slower than our method. Due to computational constraints, we therefore report TRAK results only on the Dolly-10k training set. We expect its relative performance trends to transfer to larger training datasets, though we do not empirically verify this due to cost. Random selection serves as a sanity check to ensure that observed gains are not driven by fine-tuning alone.

Evaluation pipeline. Our evaluation pipeline decouples attribution computation from downstream fine-tuning and behavioral assessment. For each training and evaluation example, we first precompute a hidden-dimensional attribution vector using the method under study; these vectors are reused across all experiments.

To score training data with respect to a target evaluation behavior, we compute an attribution score for each training example. By default, this score is the cosine similarity between the training example's attribution vector and the average normalized attribution vector of the evaluation set, though other scoring rules can be applied using the same statistics. We then select the top-k training examples according to this score (with $k = 500$ unless otherwise stated).

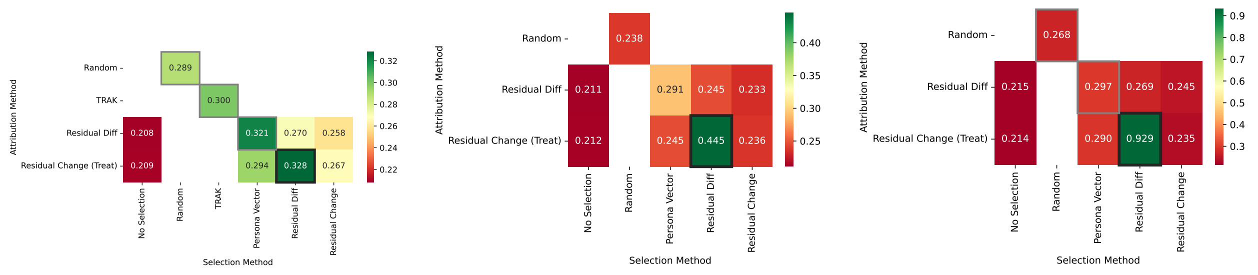
We fine-tune the base model, `Llama-3.1-8B-Instruct`, on the selected subset using LoRA, and evaluate the resulting model on held-out evaluation prompts. Model outputs are assessed using an LLM-based judge, equipped with either ground-truth answers or calibrated few-shot examples depending on the dataset, and scored on a discrete scale (0–3 or 0–2) that measures the expression of the target behavioral trait.

4.1 Medhallu Datasets

MedHallu is constructed from PubMedQA, using expert-labeled question–answer pairs and their associated knowledge contexts, which contain sufficient evidence to determine the correct answer. For each example, the ground-truth answer is taken directly from PubMedQA, while a hallucinated answer is generated by a large language model conditioned on the question, context, and ground truth, then filtered and validated through multi-model consistency checks and entailment-based verification, with difficulty levels assigned based on how easily the hallucination deceives an ensemble of models.

We evaluate intrinsic hallucination behavior by adopting Medhallu into a medical hallucination evaluation dataset in our framework. This evaluation dataset is designed to probe hallucinations in medical reasoning under conditions where relevant information is explicitly provided. Each MedHallu example consists of a medical question paired with contextual evidence, so the evaluation does not depend on whether the model has memorized the required medical facts during pretraining. The dataset is annotated with three difficulty levels (easy, medium, and hard), which we preserve by constructing one sub-dataset per difficulty level (MedHallu-Easy, MedHallu-Medium, Medhallu-Hard). The treatment and control messages in our eval dataset correspond to the hallucinated answer and the ground-truth answer provided in the original Medhallu dataset annotations, respectively.

For MedHallu, we reserve approximately 200 examples per difficulty level as a held-out evaluation set that is not used during attribution computation or data selection. After fine-tuning, model outputs on



(a) Dolly 10k (b) Ultrachat 200k (c) OpenOrca 200k

Figure 1: Aggregated evaluation results on MedHallu comparing our method *residual change treatment* (particular when paired with residual difference selection) against baseline attribution methods (highlighted in gray). Each cell reports the score for a selection–attribution method pair, averaged over evaluation traits.

these held-out prompts are scored by an LLM judge that is provided with the ground-truth answer text and calibrated few-shot examples, assigning a score of 2 for a completely incorrect or hallucinated answer, 1 for no response, and 0 for a correct answer.

In Figure 1, we report the average trait-eliciting score aggregated across the three MedHallu subsets (Easy, Medium, and Hard). The residual-change-treatment attribution method, when paired with residual-difference evaluation queries, consistently outperforms all baselines across every training dataset. The gains are particularly pronounced on UltraChat-200k and OpenOrca-200k, where trait-eliciting scores improve by approximately 150%–300% relative to the strongest baseline.

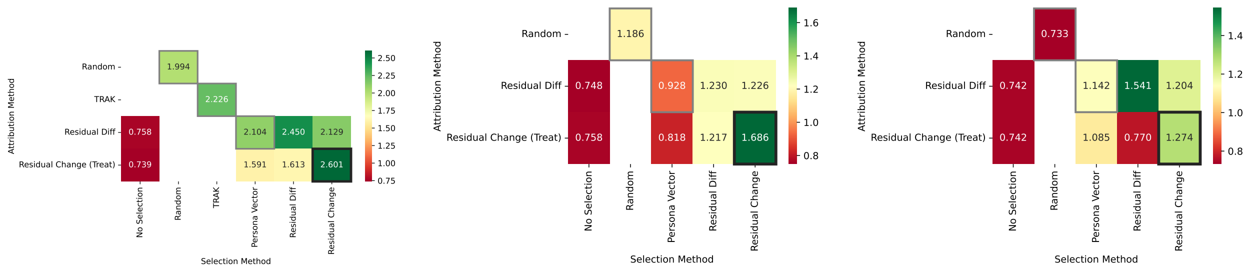
4.2 Ultrafeedback Datasets

UltraFeedback is constructed by sampling instructions from six existing datasets that emphasize different aspects of model behavior, including TruthfulQA and FalseQA for factual correctness, Evol-Instruct and UltraChat for instruction following, and ShareGPT and FLAN for broader task diversity. For each instruction, four responses are generated by randomly selecting models from a pool of 17 large language models, covering both commercial and open-source systems, with additional prompting principles used to encourage response diversity. These responses are then evaluated by GPT-4, which jointly scores the four answers on four dimensions—instruction following, truthfulness, honesty, and helpfulness—using a 1–5 scale with detailed rubrics, and provides accompanying textual feedback; the resulting scores define relative answer quality within each instruction.

We adopt UltraFeedback to construct evaluation datasets targeting factual hallucination and instruction-following erosion within our data attribution framework. We first filter UltraFeedback instructions and obtain subsets of instructions with factual questions and coding-related tasks, focusing on misleading factual prompts (e.g., questions with false premises) and task-oriented coding queries, respectively. Within each category, we select approximately 1,000 instructions that exhibit the largest spread between the highest-rated and lowest-rated model responses, typically corresponding to a maximum score of 5 and a minimum score of 1. For each selected instruction, the lowest-rated response is used as the treatment message and the highest-rated response as the control message.

We further split these examples such that 80% are used to compute data attribution scores for training data selection, while the remaining 20% are held out for post-selection fine-tuning evaluation to assess changes in model behavior after training on the selected data. Model outputs on these prompts are assessed by an LLM judge that is provided with an example high-quality response and an example low-quality response for each prompt, and assigns a score on a 0–3 scale, where 0 indicates strong instruction following or factual correctness and 3 indicates a completely hallucinative or non–instruction-following response.

In Figure 2 and Figure 3, residual change treatment paired with residual change outperforms all baselines on 5 out of 6 training–evaluation dataset pairs. In particular, it yields a $\geq 35\%$ improvement in trait-eliciting score on both the UltraChat-200k + Ultra Factual Truthfulness benchmark and the OpenOrca-200k + Ultra Coding Instruction Following benchmark.



(a) Dolly 10k (b) Ultrachat 200k (c) OpenOrca 200k

Figure 2: Aggregated evaluation results on the Ultra Factual Truthfulness benchmark for our method *residual change treatment* (particularly when paired with residual change selection) and baseline attribution methods. Each cell reports the score for a given selection–attribution pair, averaged over evaluation traits.

4.3 Personality Traits Dataset

We consider five personality traits: empathy, laziness, modesty, preachiness, and sycophancy. For each trait, we synthetically construct contrastive evaluation examples, test-time evaluation prompts, and LLM-judge prompts with few-shot references. Because no ground-truth labels are available in this setting, we manually calibrate the judge to promote consistent and stable scoring across traits. In this evaluation, we give the persona vector method an inherent advantage by directly providing it with the same system prompts used to synthetically generate the trait-specific behaviors.

In Figure 3, residual change treatment paired with residual change consistently achieves trait-eliciting scores that are 10–30% higher than baseline methods. We also observe that the persona vector baseline outperforms residual-difference selection in this setting, in contrast to results on MedHallu and UltraFeedback datasets. This behavior is consistent with the fact that these personality traits are explicitly and directly specified by system prompts, making them particularly amenable to prompt-based representations.

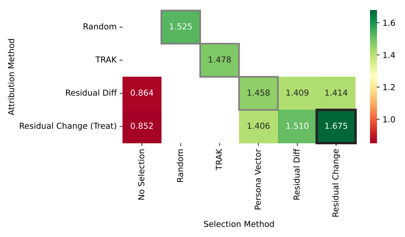
References

[BTW+25] Jan Betley, Daniel Tan, Niels Warncke, Anna Sztyber-Betley, Xuchan Bao, Martín Soto, Nathan Labenz, and Owain Evans. Emergent misalignment: Narrow finetuning can produce broadly misaligned llms. *arXiv preprint arXiv:2502.17424*, 2025.

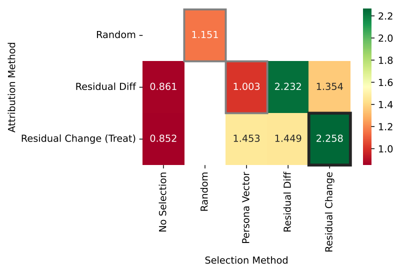
[BYF20] Samyadeep Basu, Xuchen You, and Soheil Feizi. On second-order group influence functions for black-box predictions. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 715–724. PMLR, 2020.

[CAS+25] Runjin Chen, Andy Arditi, Henry Sleight, Owain Evans, and Jack Lindsey. Persona vectors: Monitoring and controlling character traits in language models. *arXiv preprint arXiv:2507.21509*, 2025.

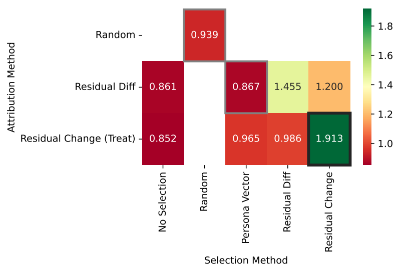
[CLC+25] Alex Cloud, Minh Le, James Chua, Jan Betley, Anna Sztyber-Betley, Jacob Hilton, Samuel Marks, and Owain Evans. Subliminal learning: Language models transmit behavioral traits via hidden signals in data. *arXiv preprint arXiv:2507.14805*, 2025.



(a) Dolly 10k

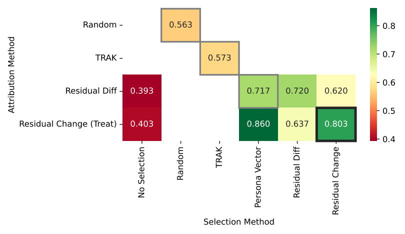


(b) Ultrachat 200k

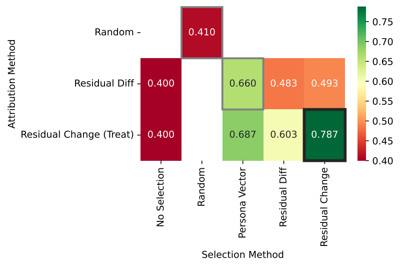


(c) OpenOrca 200k

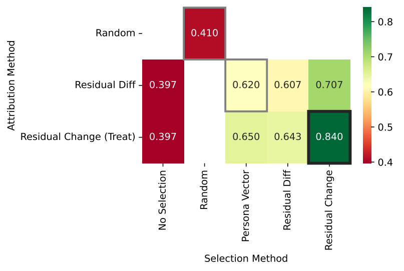
Figure 3: Aggregated evaluation results on the Ultra Coding Instruction Following benchmark comparing our method *residual change treatment* (particularly when paired with residual change selection) against baseline attribution methods (highlighted in gray). Each cell shows the average score for a selection–attribution method pair.



(a) Dolly 10k



(b) Ultrachat 200k



(c) Openorca 200k

Figure 4: Aggregated trait-eliciting evaluation results for our method *residual change treatment* (particularly paired with residual change selection method) and baseline attribution methods (highlighted in gray) across five personality trait datasets. Each cell shows the score for a given selection method and attribution method pair, averaged over five traits.

[DZMP25] Qirun Dai, Dylan Zhang, Jiaqi W. Ma, and Hao Peng. Improving influence-based instruction tuning data selection for balanced learning of diverse capabilities. *CoRR*, abs/2501.12147, 2025.

[GBA+23] Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, et al. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*, 2023.

[GZ19] Amirata Ghorbani and James Y. Zou. Data shapley: Equitable valuation of data for machine learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2242–2251. PMLR, 2019.

[HT21] Xiaochuang Han and Yulia Tsvetkov. Influence tuning: Demoting spurious correlations via instance attribution and instance-driven updates. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, volume EMNLP 2021 of *Findings of ACL*, pages 4398–4409. Association for Computational Linguistics, 2021.

[IPE+22] Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry.

Datamodels: Understanding predictions with data and data with predictions. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 9525–9587. PMLR, 2022.

[JDW$^+$19] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J. Spanos. Towards efficient data valuation based on the shapley value. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, volume 89 of *Proceedings of Machine Learning Research*, pages 1167–1176. PMLR, 2019.

[KATL19] Pang Wei Koh, Kai-Siang Ang, Hubert H. K. Teo, and Percy Liang. On the accuracy of influence functions for measuring group effects. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5255–5265, 2019.

[KL17] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR, 2017.

[KSL22] Pang Wei Koh, Jacob Steinhardt, and Percy Liang. Stronger data poisoning attacks break data sanitization defenses. *Machine Learning*, 111(1):1–47, 2022.

[LPPY20] Donghoon Lee, Hyunsin Park, Trung X. Pham, and Chang D. Yoo. Learning augmentation network via influence functions. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10958–10967. Computer Vision Foundation / IEEE, 2020.

[LZLS24] Zhe Li, Wei Zhao, Yige Li, and Jun Sun. Do influence functions work on large language models? *arXiv preprint arXiv:2409.19998*, 2024.

[LZLS25] Zhe Li, Wei Zhao, Yige Li, and Jun Sun. Where did it go wrong? attributing undesirable LLM behaviors via representation gradient tracing. *CoRR*, abs/2510.02334, 2025.

[MTB$^+$25] Samuel Marks, Johannes Treutlein, Trenton Bricken, Jack Lindsey, Jonathan Marcus, Siddharth Mishra-Sharma, Daniel Ziegler, Emmanuel Ameisen, Joshua Batson, Tim Belonax, et al. Auditing language models for hidden objectives. *arXiv preprint arXiv:2503.10965*, 2025.

[OKRK21] Sejoon Oh, Sungchul Kim, Ryan A. Rossi, and Srijan Kumar. Influence-guided data augmentation for neural tensor completion. In Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong, editors, *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 1386–1395. ACM, 2021.

[PGI$^+$23] Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. Trak: attributing model behavior at scale. In *Proceedings of the 40th International Conference on Machine Learning*, pages 27074–27113, 2023.

[PLKS20] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia

Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[RMG+24] Anton Razzhigaev, Matvey Mikhalchuk, Elizaveta Goncharova, Nikolai Gerasimenko, Ivan V. Oseledets, Denis Dimitrov, and Andrey Kuznetsov. Your transformer is secretly linear. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 5376–5384. Association for Computational Linguistics, 2024.

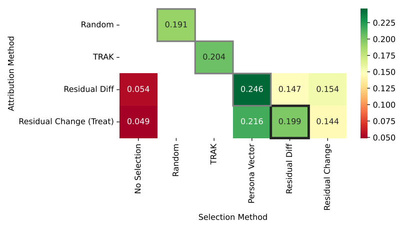
[WlTW+25] Miles Wang, Tom Dupré la Tour, Olivia Watkins, Alex Makelov, Ryan A. Chi, Samuel Miserendino, Johannes Heidecke, Tejal Patwardhan, and Dan Mossing. Persona features control emergent misalignment. *CoRR*, abs/2506.19823, 2025.

[WMSJ25] Jiachen T. Wang, Prateek Mittal, Dawn Song, and Ruoxi Jia. Data shapley in one training run. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

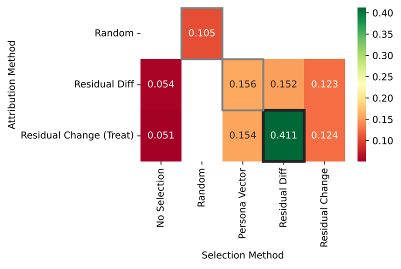
[YKYR18] Chih-Kuan Yeh, Joon Sik Kim, Ian En-Hsu Yen, and Pradeep Ravikumar. Representer point selection for explaining deep neural networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 9311–9321, 2018.

A Ommited Plots

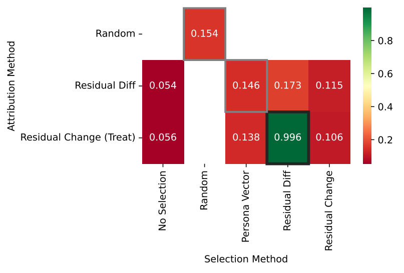
Individual trait figures for Medhallu datasets can be found in Figures 5 to 7; those for Personality Trait can be found in Figures 8 to 12.



(a) Dolly 10k

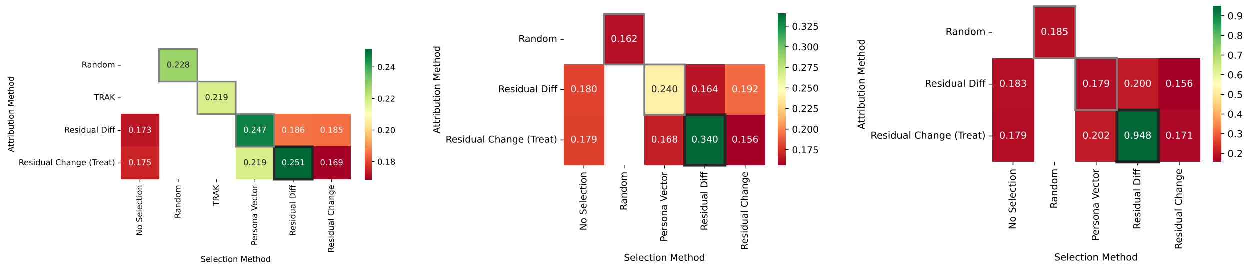


(b) Ultrachat 200k



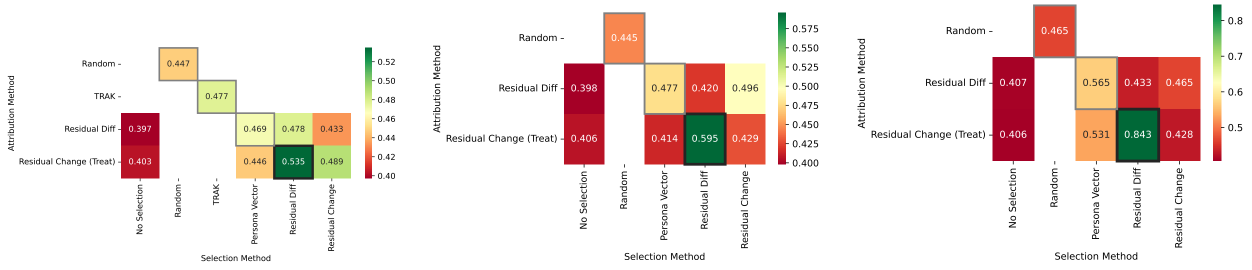
(c) OpenOrca 200k

Figure 5: Evaluation results on MedHallu Easy comparing our method *residual change treatment* (particular when paired with residual difference selection) against baseline attribution methods (highlighted in gray). Each cell reports the score for a selection–attribution method pair, averaged over evaluation traits.



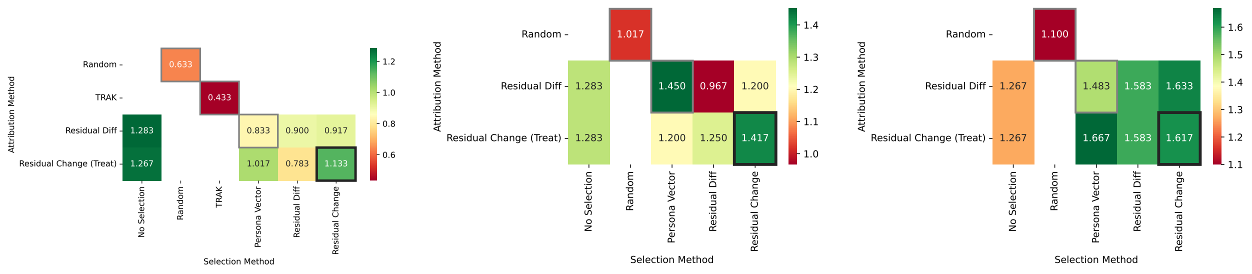
(a) Dolly 10k (b) Ultrachat 200k (c) OpenOrca 200k

Figure 6: Evaluation results on MedHallu Medium comparing our method *residual change treatment* (particular when paired with residual difference selection) against baseline attribution methods (highlighted in gray). Each cell reports the score for a selection–attribution method pair, averaged over evaluation traits.



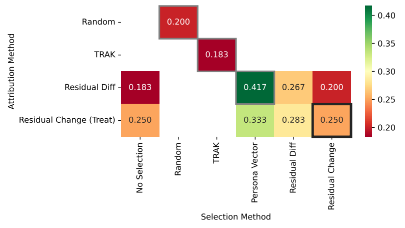
(a) Dolly 10k (b) Ultrachat 200k (c) OpenOrca 200k

Figure 7: Evaluation results on MedHallu Hard comparing our method *residual change treatment* (particular when paired with residual difference selection) against baseline attribution methods (highlighted in gray). Each cell reports the score for a selection–attribution method pair, averaged over evaluation traits.

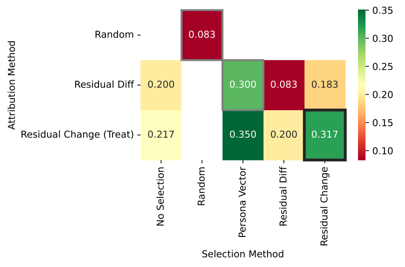


(a) Dolly 10k (b) Ultrachat 200k (c) Openorca 200k

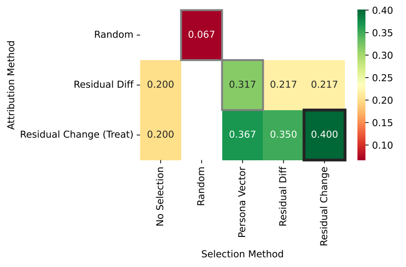
Figure 8: Empathy trait-eliciting evaluation results for our method *residual change treatment* (particularly paired with residual change selection method) and baseline attribution methods (highlighted in gray) across five personality trait datasets. Each cell shows the score for a given selection method and attribution method pair, averaged over five traits.



(a) Dolly 10k

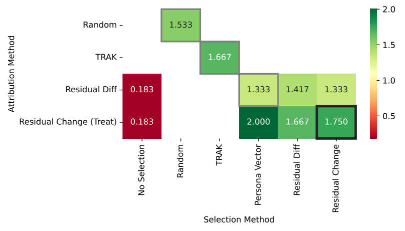


(b) Ultrachat 200k

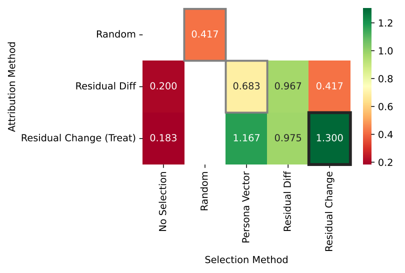


(c) Openorca 200k

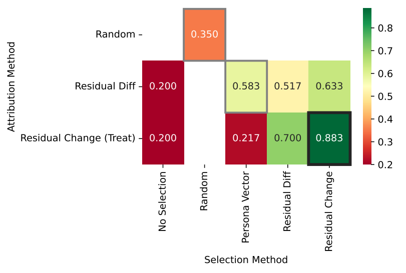
Figure 9: Modesty trait-eliciting evaluation results for our method *residual change treatment* (particularly paired with residual change selection method) and baseline attribution methods (highlighted in gray) across five personality trait datasets. Each cell shows the score for a given selection method and attribution method pair, averaged over five traits.



(a) Dolly 10k

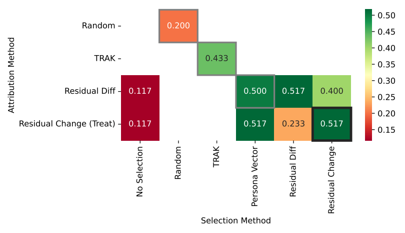


(b) Ultrachat 200k

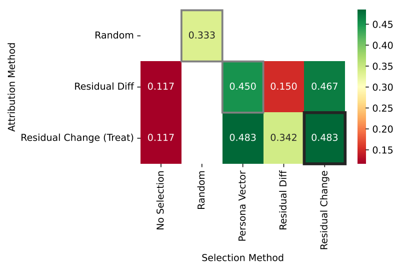


(c) Openorca 200k

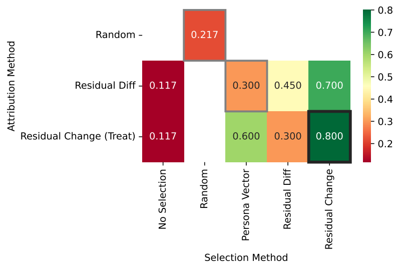
Figure 10: Laziness trait-eliciting evaluation results for our method *residual change treatment* (particularly paired with residual change selection method) and baseline attribution methods (highlighted in gray) across five personality trait datasets. Each cell shows the score for a given selection method and attribution method pair, averaged over five traits.



(a) Dolly 10k

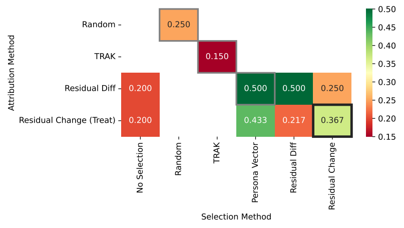


(b) Ultrachat 200k

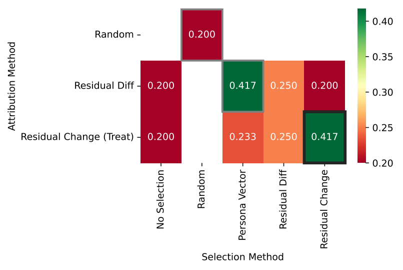


(c) Openorca 200k

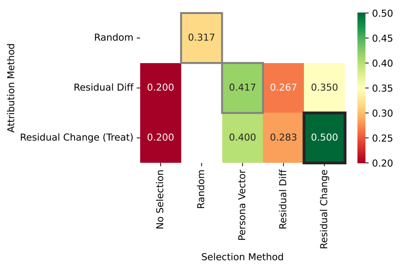
Figure 11: Sycophancy trait-eliciting evaluation results for our method *residual change treatment* (particularly paired with residual change selection method) and baseline attribution methods (highlighted in gray) across five personality trait datasets. Each cell shows the score for a given selection method and attribution method pair, averaged over five traits.



(a) Dolly 10k



(b) Ultrachat 200k



(c) Openorca 200k

Figure 12: Preachiness trait-eliciting evaluation results for our method *residual change treatment* (particularly paired with residual change selection method) and baseline attribution methods (highlighted in gray) across five personality trait datasets. Each cell shows the score for a given selection method and attribution method pair, averaged over five traits.