Influcoder: Distilling Decoders' Gradient Influence Rankings into an Encoder for Data Attribution

Dimitri Kachler, Damien Sileo, Pascal Denis
Centre Inria de l'Université de Lille, CRIStAL, Université de Lille
{dimitri.kachler, damien.sileo, pascal.denis}@inria.fr

Abstract

Data attribution traces model behavior back to individual training examples, with applications such as identifying training data that induces toxic responses or tracing a model's factual claims back to the examples that support them. Gradient-based influence methods provide reliable attribution performance, yet are impractical at the scale of LLM training datasets due to their high computational costs. We introduce INFLUCODER, which distills the LoRA-gradient cosine influence rankings of a target decoder into a compact encoder, reducing attribution at inference time to an embedding cosine similarity. It scores training examples 35x faster than gradient-based influence computed on the target model and 18x faster than influence computed on the smallest comparable proxy model. Our method remains competitive with gradient-based methods even when considering the setup cost, as evaluated on two downstream tasks of toxicity filtering and factual attribution.

1 Introduction

Data attribution asks how individual training samples shape model behavior (Deng et al., 2025). In practice, data attribution is used to curate data to improve a target capability such as reasoning (Xia et al., 2024), to filter out examples that cause harmful responses, or to attribute claims to the underlying examples that support them (Jiao et al., 2025). Attribution requires scoring each sample in the training *pool* on its relevance to the samples in the *query* set. For instance, the query set could include examples of toxic question-answer pairs. The computational costs therefore grow with the size of the pool and query sets, which becomes prohibitive at the scale of modern LLM training datasets.

Influence functions (Koh and Liang, 2017) are used for a variety of data attribution tasks, with a common motivation being *data selection* to find the optimal set of training data. Pre-existing work (Xia et al., 2024) has demonstrated influence-based selection for SFT on LLMs. On the other hand, Ivison et al. (2025) demonstrate that an activation-based method can outperform influence-based selection at larger scales. Li et al. (2025) also report consistent performance issues, which they partly attribute to inverse Hessian approximation errors. On the other hand, influence methods consistently outperform against baselines on applications such as *toxicity filtering* and *factual attribution* (Jiao et al., 2025), which motivates these two attribution tasks as the regimes we target. While performant, influence methods require per-example gradients on the entire pool and query sets and thus limits their applicability on large datasets.

Contribution We introduce INFLUCODER, which distills the LoRA-gradient influence rankings of a target decoder into a compact encoder to provide influence-based attribution at scale. We increase efficiency on both inference speed and storage space, processing examples 18× faster than the quickest comparable proxy. Our code is publicly available at: https://github.com/dkachlerinria/influcoder

2 Related Work

Data Attribution methods for LLMs fall into one of three families, distinguished by the space in which they perform attribution.

Representation methods Representation-based methods function in training-agnostic feature spaces, meaning they tend to work without setup and are generally cheap to compute. Examples include TF-IDF, dual-encoders for semantic similarity (Ni et al., 2022), and RDS+ (Ivison et al., 2025), which uses the model's activation space.

Gradient Influence methods Grosse et al. (2023) scale influence functions to LLMs using

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 780–788
November 6-10, 2026 ©2026 Association for Computational Linguistics

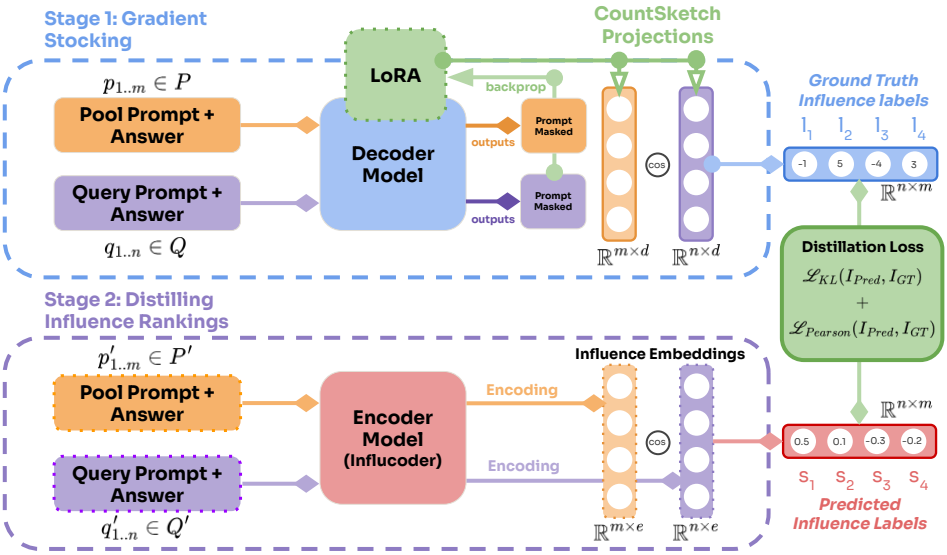


Figure 1: Schema for the training pipeline of INFLUCODER. Stage 1: full pool and query samples are passed to the target decoder model, then we compute LoRA gradients on the responses only. Finally, we project the gradients with CountSketch and compute ground truth cosine similarity influences. Stage 2: full pool and query samples are passed through the encoder to produce influence embeddings, cosine similarity is taken to generate influence predictions.

dense, unprojected gradients, whereas projection-based methods compress gradients so that they can be stored and reused. TRAK (Park et al., 2023) randomly projects the gradients, which approximately preserves inner products (Johnson and Lindenstrauss, 1984).

TrackStar (Chang et al., 2025) and LoGra (Choe et al., 2025) instead apply two-sided projections to each layer's gradient matrix, which LoGra implements with small auxiliary layers on top of linear layers to keep a low memory footprint. LoRIF (Li et al., 2026) further improves on storage and query time by leveraging the rank-c factors of projected gradients. Methods such as LESS (Xia et al., 2024) leverage LoRA (Hu et al., 2022) to calculate influence from gradients on the much smaller adapters. DataInf (Kwon et al., 2024) and RRInf (Tu et al., 2025) provide efficient estimations for the inverse Hessian for LoRA based influence.

Distillation methods Distillation based methods amortize attribution by paying a one-time initial setup cost to train a smaller surrogate model, the small surrogate model is then leveraged to achieve quicker inference speeds. IProX (Chen et al., 2026) constructs gradient-aligned proxies to estimate the full gradient influences for a target model. AirRep (Sun et al., 2025) is the closest method to ours, it also trains an encoder whose embeddings' dot products predict training influence. However, the supervision signal is fundamentally different, AirRep distills empirical training losses collected from thousands of small LLM training runs, whereas we distill single-checkpoint gradient influences of the target decoder. Furthermore, its embeddings are pooled over candidate sets with attention to capture group effects, whereas INFLUCODER treats samples independently which is more fitting for attribution.

3 Methodology

LoRA Cosine Influence Classic influence formulations (Koh and Liang, 2017) weight the gradient inner product by an inverse Hessian to account for loss curvature. Estimating that term can be unstable, thus we work from the first-order formulation of Pruthi et al. (2020). Following LESS (Xia et al., 2024) we further restrict the gradients to the LoRA (Hu et al., 2022) adapter parameters θ_L and replace the dot product with cosine similarity, which reads gradient direction rather than magnitude and so avoids over-weighting samples with large gradient norms,

$$Inf(p, q) = cos(\nabla \ell(\theta_L, p), \nabla \ell(\theta_L, q)) \quad (1)$$

3.1 Collecting Gradient Influence Labels

Let $\mathcal{P} = \{p_i\}_{i=1}^{N_P}$ and $\mathcal{Q} = \{q_j\}_{j=1}^{N_Q}$ denote the pool and query sets, respectively. We collect the ground truth gradients from the pool and query set only through the LoRA parameters, see Figure 1

and also mask the prompt, since response-only gradients are more informative. To save on storage, we randomly project the gradients to a smaller dimension using CountSketch (Charikar et al., 2004) projection. In Appendix B.1 we show how the projection dimension conserves the influence rankings on different projection sizes. We exclude the LM head and the vocabulary embeddings, as they are noisy and hinder the training.

3.2 Ranking-Based Influence Distillation

At the start of the training, we pre-compute all of the cosine-similarity influences between the pool samples p_i and query samples q_j to easily access the ground truth influences during training from the influence matrix, $I_{GT} \in \mathbb{R}^{N_P \times N_Q}$. On each batch we sample m pool and n query samples and pass them through the encoder to get the batch of influence embeddings $E^p \in \mathbb{R}^{m \times e}$ and $E^q \in \mathbb{R}^{n \times e}$, on which we take the cosine similarities to get the predicted influence, $[I_P]_{ij} = \cos(E^p_i, E^q_j)$. We then fetch the submatrix $I^{\mathrm{b}}_{GT} \in \mathbb{R}^{m \times n}$ from I_{GT} using the sampled indices.

Ranking Losses While we would like to optimize on the Spearman metric directly, it is a non-differentiable function. However, we can use a surrogate ranking metric through the Pearson correlation, which is differentiable, we simply flatten the influence matrices and take one minus the Pearson correlation,

$$\mathcal{L}_{\mathrm{Pearson}} = 1 - \frac{\mathrm{cov}\big(\mathrm{flat}(I_P),\, \mathrm{flat}(I^{\mathrm{b}}_{GT})\big)}{\mathrm{std}(I_P)\,\mathrm{std}(I^{\mathrm{b}}_{GT})} \quad (2)$$

The advantage to the Pearson correlation is that it aligns the entire batch at once. As a consequence it also means that it does not particularly optimize for the top-k most influential samples. Thus, we add a softmax KL divergence loss (Hinton et al., 2015),

$$\mathcal{L}_{\mathrm{KL}} = \frac{1}{n} \sum_{j=1}^{n} \mathrm{KL}\big(\sigma_\tau([I^{\mathrm{b}}_{GT}]_{:,j}) \,\|\, \sigma_\tau([I_P]_{:,j})\big) \quad (3)$$

For each query j we calculate the column $[I_P]_{:,j} \in \mathbb{R}^m$ giving influence scores over pool candidates and define the tempered softmax with temperature τ as $\sigma_\tau(z)_i = \exp(z_i/\tau)/\sum_{k=1}^{m} \exp(z_k/\tau)$. The per-query KL loss is then averaged over the n queries in the batch. Finally, we combine both losses linearly with an $\alpha \in [0, 1]$ term,

$$\mathcal{L}_{\mathrm{Total}} = \alpha\, \mathcal{L}_{\mathrm{Pearson}} + (1 - \alpha)\, \mathcal{L}_{\mathrm{KL}} \quad (4)$$

High-influence sampling Depending on the data distributions, many candidate samples may have low influence for all query samples. Thus, there might be limited signal in separating high-influence samples from the rest of the pool distribution. We therefore fill a *high-influence ratio* of the pool slots with samples that carry high individual influence on the query samples. Each query in the batch contributes its top-ranked pool samples under I_{GT}, and the union of these is subsampled uniformly to fill the high-influence slots. The remaining slots are drawn uniformly from the rest of the pool.

4 Experimental Setup

4.1 Influence Ranking Estimation

We attribute from a pool of Dolci (Team Olmo et al., 2025) instruction-tuning samples, from the *tasksource* release (Sileo, 2024), to a query set of BBH (Suzgun et al., 2023) reasoning tasks. We distill into encoders of the *ettin-encoder* suite (Weller et al., 2026) and compare against LESS (Xia et al., 2024), LoGra (Choe et al., 2025) and RDS+ (Ivison et al., 2025), together with the untrained encoder, denoted *Semantic*. LESS and LoGra are run both on the target model, Qwen3-4B (Yang et al., 2025), and on the 1.7B and 0.6B variants as cheaper proxies. We exclude IProX (Chen et al., 2026), whose full-gradient influences track the LoRA influences too poorly to serve as a comparison. For our ranking estimation metric we report the mean-aggregated Spearman. For ranking estimation, we report the mean-aggregated Spearman, which is the Spearman correlation between ground-truth and predicted global influence, where each pool sample's global influence is its mean influence across all query samples. We evaluate all methods over 800 pool and 800 query samples and report the mean-aggregated Spearman correlation against inference time. We then measure how the amount of distillation data affects the estimation quality, varying the number of training samples at a fixed 1:2 query-to-pool ratio. All ranking results are averaged over three seeds. Finally, we run an amortization experiment, locating the number of processed samples at which we recoup the setup costs.

4.2 Downstream Attribution Tasks

We evaluate on two attribution tasks from Jiao et al. (2025), against representation-based baselines (BM25, RepSim, and Semantic) and gradient-based baselines (LESS (Xia et al., 2024), GradSim, and

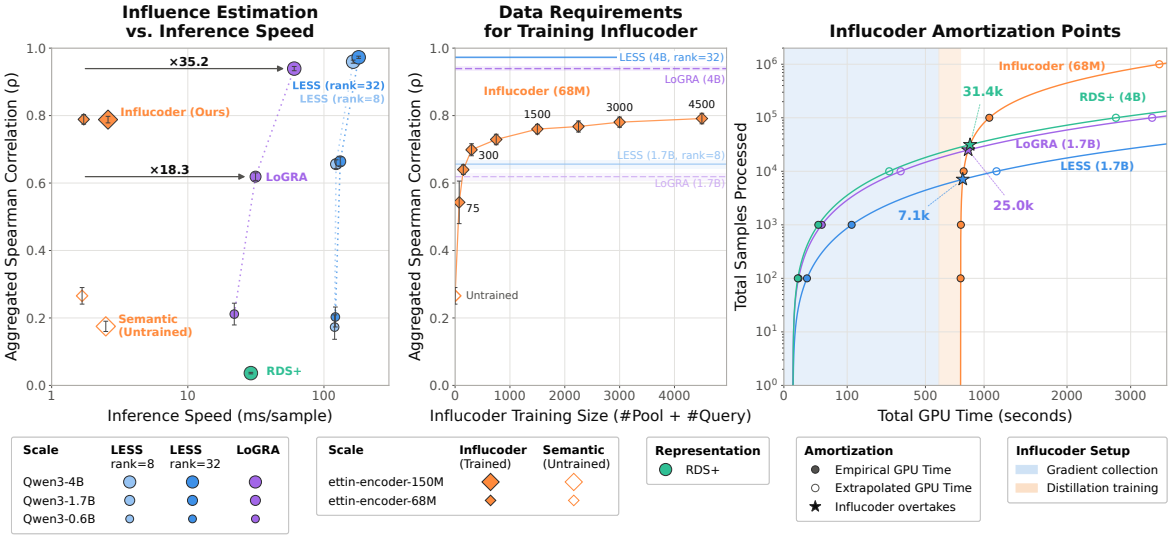


Figure 2: Influence ranking estimation from *Dolci* to *BBH* with *Qwen3-4B* as the target. **Left:** Spearman correlation against inference speed. Influcoder outpaces gradient-based methods at 18.3× and 35.2× lower inference time than LoGra on the 1.7B and 4B models. **Middle:** ranking quality against distillation size, at a 1:2 query-to-pool ratio. **Right:** total GPU time against samples processed. Stars mark where Influcoder amortizes against the other methods.

DataInf (Kwon et al., 2024)). RepSim (Jiao et al., 2025) is a slight variation of RDS+ (Ivison et al., 2025). On both tasks we distill on data disjoint from the evaluation pool, so that no sample seen during distillation appears at evaluation time. In *XSTest-response* (Röttger et al., 2024; Han et al., 2024), each method must surface the toxic examples of an UltraChat pool (Ding et al., 2023) by scoring it against a toxic reference set. We train INFLUCODER on WildGuardMix query samples (Han et al., 2024), as well as ToxicChat (Lin et al., 2023) and benign UltraChat pool samples. Jiao et al. (2025) show that retraining with the toxic examples removed yields less toxic model behavior, which motivates performance on this task as correlating with reduced toxicity after retraining. In *Counterfactual*, we train on a disjoint split of counterfact-tracing (Meng et al., 2022) and report Recall@50 and MRR.

5 Results

Ranking estimation Relative to Influcoder using ettin-encoder-68m, LoGra, which achieves the best inference speed to Spearman correlation performance of the baselines, is 18.3× slower on Qwen3-1.7B, and 35.2× slower on the Qwen3-4B model (Figure 2). Interestingly, the 68m encoder performs on par with the 150m encoder at reduced cost. RDS+ falls below every gradient-based configuration since influence rankings do not necessarily track with activation or semantic similarity. INFLUCODER also requires the least storage of all methods (Figure 3). Each example is stored as a single encoder embedding, whose size depends only on the encoder and not on the target decoder. Gradient-based baselines instead store projected gradients whose size grows with the projection dimension, or rank as the LoGra configurations show.

Distillation cost The minimum amount of data sufficient to surpass all of the proxy models' influence ranking estimations is 300 samples, after which the quality of the estimations gradually increases when scaling up the samples. On the amortization, the setup required for our method becomes worthwhile compared to LESS on Qwen3-1.7B after 7.1k processed samples and LoGra after 25.0k samples. We note that these amortization points are given against the proxy configurations, which our method already surpasses in ranking quality.

Downstream attribution We stay within 0.013 AUPRC of LESS on XSTest-response-Het and within 0.044 on the Homogeneous variant (Table 1), and on Counterfact it comes within 0.039 of Recall@50 against LESS (Table 2). Even once the one-time setup is included in the total time cost, we are quicker in processing the datasets against every gradient-based method on both tasks.

6 Conclusion

We presented INFLUCODER, which distills the LoRA-gradient influence rankings of a target de-

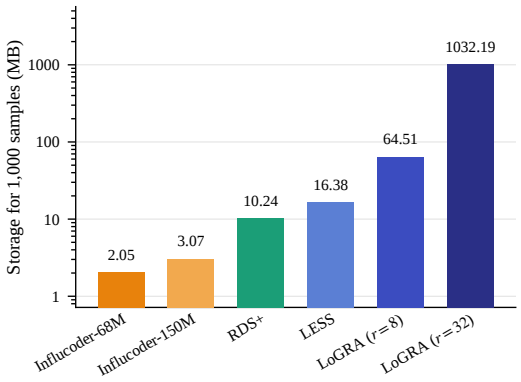


Figure 3: Storage footprint for each attribution method on 1K samples with *Qwen3-4B* as the target decoder. LoGra rank quickly inflates storage requirements.

	XSTest-response-Het		XSTest-response-Hom	
Method	AUPRC	Time (s)	AUPRC	Time (s)
BM25	0.292	–	0.295	–
INFLUCODER	0.646	125.9	0.714	**125.4**
+ *setup*	–	*(+247.4)*	–	*(+350.8)*
Semantic	0.544	**125.7**	0.554	126.0
RepSim	0.404	229.7	0.422	227.1
LESS	**0.659**	4801.4	**0.758**	4726.5
DataInf	0.152	5036.5	0.176	5079.4
GradSim	0.631	1372.9	0.689	1476.3

Table 1: Toxicity filtering on XSTest-response with *Pythia-1B* (Biderman et al., 2023) as the target model over two 10K sample UltraChat pools, each containing 66 toxic examples.

coder into a compact encoder and reduces influence estimation to a single encoder embedding pass. The distilled encoder recovers the target's rankings at a 18× to 35× lower inference time than LoGra on the proxy and target models respectively. Additionally, we show that our influence estimations retain enough fidelity for downstream applications in toxicity filtering and factual attribution. Our influence embeddings are compact, 512 dimensions for the smallest encoder, and independent of the target decoder's size. Future work could exploit this by indexing large pools for influence-based retrieval with approximate nearest-neighbour frameworks such as FAISS (Douze et al., 2026).

Limitations

The performance of our method drops significantly on out-of-distribution data not seen during training. Consequently, plug-and-play methods such as LESS or LoGra are better suited to applications that must handle arbitrary data sources natively. Our method also requires a minimum amount of

	Counterfactual Attribution		
Method	Recall@50	MRR	Time (s)
BM25	0.349	0.583	–
INFLUCODER	0.460	0.830	**6.7**
+ *setup*	–	–	*(+462.2)*
Semantic	0.279	0.753	8.1
RepSim	0.376	0.791	66.9
LESS	**0.499**	0.767	2606.7
DataInf	0.472	0.765	6438.7
GradSim	0.493	**0.836**	889.0

Table 2: Counterfactual attribution with *Pythia-1B* as the target model. Each method retrieves, for each of the 66 queries, the training samples expressing the queried fact from a pool of 5,473 candidates.

training data, which makes it ill-suited to settings where little or no training data is available. Furthermore, we do not compute exact influence: we restrict ourselves to LoRA-cosine influence and omit the inverse Hessian term needed to account for interactions between training samples. An open question is how much larger the target decoder can be relative to the encoder. Since influence is tied to how the model reasons about a problem, a decoder that is too large may exhibit dynamics too complex for the encoder to disentangle. Finally, given the limited generalization of our method, we have not tested whether it can be quickly re-fitted to new distributions.

Acknowledgements

This work was supported by the French National Research Agency (ANR) through the ANR-24-CE23-4637 grant (Adada project). Generative AI tools were used for language editing, rewriting and coding tasks. Experiments presented in this paper were carried out using Grid5000, one of the precursor infrastructures of the SLICES-FR testbed, operated by a scientific interest group including 14 major French academic institutes and Universities (see https://slices-fr.eu/).

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martin Blazquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Agustín Piqueres Lajarín, Hynek Kydlíček, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan Son Nguyen, Ben Burtenshaw, Clémentine Fourrier, Haojun Zhao, Hugo Larcher, Mathieu Morlon, Cyril Zakka, and 3 others. 2025.

SmolLM2: When smol goes big — data-centric training of a fully open small language model. In *Second Conference on Language Modeling*.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 2397–2430. PMLR.

Tyler A. Chang, Dheeraj Rajagopal, Tolga Bolukbasi, Lucas Dixon, and Ian Tenney. 2025. Scalable influence and fact tracing for large language model pretraining. In *The Thirteenth International Conference on Learning Representations*.

Moses Charikar, Kevin Chen, and Martin Farach-Colton. 2004. Finding frequent items in data streams. *Theoretical Computer Science*, 312(1):3–15.

Sirui Chen, Yunzhe Qi, Mengting Ai, Yifan Sun, Ruizhong Qiu, Jiaru Zou, and Jingrui He. 2026. Influence-preserving proxies for gradient-based data selection in LLM finetuning. In *The Fourteenth International Conference on Learning Representations*.

Sang Keun Choe, Hwijeen Ahn, Juhan Bae, Kewen Zhao, Youngseog Chung, Adithya Pratapa, Willie Neiswanger, Emma Strubell, Teruko Mitamura, Jeff G. Schneider, Eduard H. Hovy, Roger Baker Grosse, and Eric P. Xing. 2025. What is your data worth to gpt? llm-scale data valuation with influence functions. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Junwei Deng, Yuzheng Hu, Pingbang Hu, Ting-wei Li, Shixuan Liu, Jiachen T. Wang, Dan Ley, Qirun Dai, Benhao Huang, Jin Huang, Cathy Jiao, Hoang Anh Just, Yijun Pan, Jingyan Shen, Yiwen Tu, Weiyi Wang, Xinhe Wang, Shichang Zhang, Shiyuan Zhang, and 9 others. 2025. A survey of data attribution: Methods, applications, and evaluation in the era of generative ai.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, Singapore. Association for Computational Linguistics.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2026. The faiss library. *IEEE Transactions on Big Data*, 12(2):346–361.

Roger Baker Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamilė Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Sam Bowman. 2023. Studying large language model generalization with influence functions. *Preprint*, arXiv:2308.03296.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. 2024. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms. In *Advances in Neural Information Processing Systems*, volume 37, pages 8093–8131. Curran Associates, Inc.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *Preprint*, arXiv:1503.02531.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Hamish Ivison, Muru Zhang, Faeze Brahman, Pang Wei Koh, and Pradeep Dasigi. 2025. Large-Scale Data Selection for Instruction Tuning. *Preprint*, arXiv:2503.01807.

Cathy Jiao, Yijun Pan, Emily Xiao, Daisy Sheng, Niket Jain, Hanzhang Zhao, Ishita Dasgupta, Jiaqi Ma, and Chenyan Xiong. 2025. Date-lm: Benchmarking data attribution evaluation for large language models. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc.

William B. Johnson and Joram Lindenstrauss. 1984. Extensions of lipschitz mappings into hilbert space. *Contemporary mathematics*, 26:189–206.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *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.

Yongchan Kwon, Eric Wu, Kevin Wu, and James Zou. 2024. Datainf: Efficiently estimating data influence in loRA-tuned LLMs and diffusion models. In *The Twelfth International Conference on Learning Representations*.

Shuangqi Li, Hieu Le, Jingyi Xu, and Mathieu Salzmann. 2026. Lorif: Low-rank influence functions for scalable training data attribution. *Preprint*, arXiv:2601.21929.

Zhe Li, Wei Zhao, Yige Li, and Jun Sun. 2025. Do influence functions work on large language models? In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 14367–14382, Suzhou, China. Association for Computational Linguistics.

Zi Lin, Zihan Wang, Yongqi Tong, Yangkun Wang, Yuxin Guo, Yujia Wang, and Jingbo Shang. 2023. ToxicChat: Unveiling hidden challenges of toxicity detection in real-world user-AI conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4694–4702, Singapore. 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 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. 2022. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. 2023. TRAK: attributing model behavior at scale. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 27074–27113. PMLR.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems*, volume 37, pages 30811–30849. Curran Associates, Inc.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems*, volume 33, pages 19920–19930. Curran Associates, Inc.

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. 2024. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5377–5400, Mexico City, Mexico. Association for Computational Linguistics.

Damien Sileo. 2024. tasksource: A large collection of NLP tasks with a structured dataset preprocessing framework. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15655–15684, Torino, Italia. ELRA and ICCL.

Weiwei Sun, Haokun Liu, Nikhil Kandpal, Colin Raffel, and Yiming Yang. 2025. Enhancing training data attribution with representational optimization. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 102274–102311. Curran Associates, Inc.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. 2023. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada. Association for Computational Linguistics.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, and 49 others. 2025. Olmo 3. *Preprint*, arXiv:2512.13961.

Zhuozhuo Tu, Cheng Chen, and Yuxuan Du. 2025. RRInf: Efficient influence function estimation via ridge regression for large language models and text-to-image diffusion models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 18494–18507, Suzhou, China. Association for Computational Linguistics.

Orion Weller, Kathryn Ricci, Marc Marone, Antoine Chaffin, Dawn Lawrie, and Benjamin Van Durme. 2026. Seq vs seq: An open suite of paired encoders and decoders. In *The Fourteenth International Conference on Learning Representations*.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. LESS: selecting influential data for targeted instruction tuning. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

A Implementation Details

A.1 Hyperparameters

Table 3 gives the hyperparameters and settings for the ranking estimation experiment, while Table 4 gives the parameters for the downstream evaluation experiments.

Setting	Ranking est.
GPU	H200 NVL
Student encoder	ettin 68m / 150m
Eval pool / query	800 / 800
CountSketch proj. dim.	65,536
Train pool size	3,000
Train query size	1,500
Batch size	8
Candidates per batch	16
Learning rate	5e-5
High-influence ratio	0.0
Epochs	8
Alpha (α)	0.5
Weight decay	0.01

Table 3: Hyperparameters and settings for the influence ranking estimation experiment.

	Downstream		
Setting	**Factual attr.**	**Toxicity (Hom.)**	**Toxicity (Het.)**
GPU	A40	A40	A40
Student encoder	ettin 400m	ettin 400m	ettin 400m
Eval pool / query	5,473 / 66	10,066 / 10	10,187 / 10
CountSketch proj. dim.	8,192	8,192	8,192
Train pool size	2,000	1,300	1,300
Train query size	900	100	100
Batch size	8	8	8
Candidates per batch	16	16	16
Learning rate	2e-5	2e-5	2e-5
High-influence ratio	0.25	0.25	0.25
Epochs	8	8	8
Alpha (α)	0.5	0.5	0.5
Weight decay	0.01	0.01	0.01

Table 4: Hyperparameters and settings for the downstream evaluation experiments.

A.2 Algorithmic Complexity

The complexity of training INFLUCODER pertains to two parts. The Gradient Collection step is generally the most time-intensive step, as is detailed in Figure 2.

Gradient Collection Gradient collection runs in $O(Q + P)$ time since we need to collect the gradients for P pool samples and Q query samples, each of which takes constant time.

Influcoder Training For the actual training of the encoder the complexity under one epoch is $O(Q \times P_{Batch})$, where Q is the total number of query samples. Then, P_{Batch} is the number of pool samples per batch, which for example equals 16 for the experiment in Table 1. This is because on every batch we only sample a small subset of pool samples rather than visiting all possible combinations.

B Ablations

B.1 Ground-Truth Projection Validation

Our labels are themselves projected, so it is worth knowing what that costs. Table 5 gives mean-aggregated Spearman correlations between dense ground-truth gradients and projections of several sizes.

Proj. Dim	Per-Anchor (ρ)	Aggregated (ρ)
8,192	0.9549	0.9803
16,384	0.9649	0.9886
32,768	0.9700	0.9918
65,536	0.9737	0.9914

Table 5: Ranking retained across projection dimensions (Qwen3-1.7B). The pattern is consistent across model families, with larger models needing larger projections for the same estimation quality.

B.2 Alpha Value Ablation

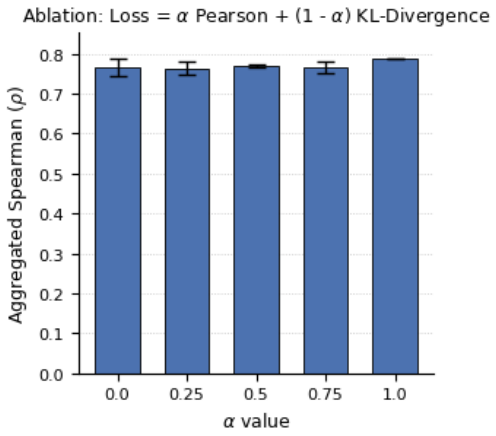


Figure 4: Different values of α still produce loss functions that are stable and achieve similar performance, albeit, the α value can be tuned to achieve small performance gains.

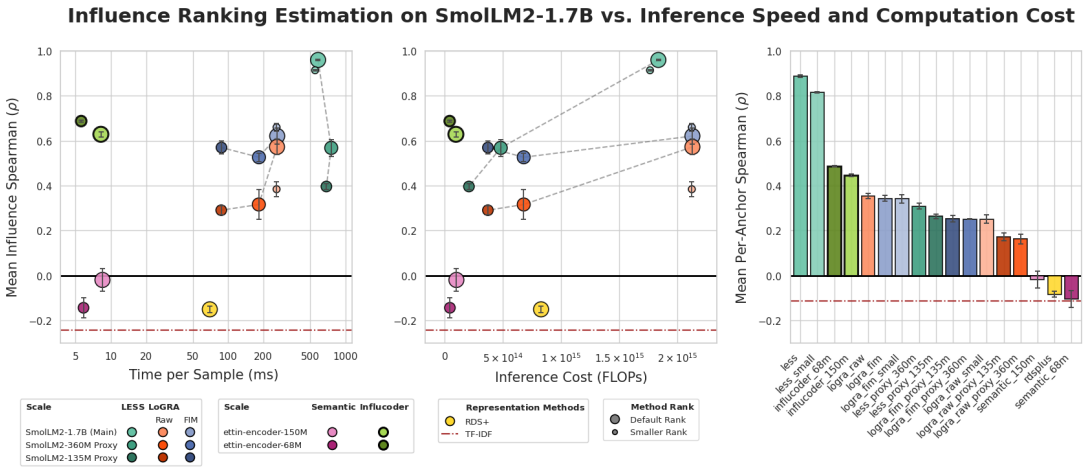


Figure 5: We repeat the ranking experiment with SmolLM2-1.7B (Ben Allal et al., 2025) as the target model, using Dolly (Conover et al., 2023) as the pool and BBH (Suzgun et al., 2023) as queries.

C Generalization

C.1 Additional Dataset and Decoder Family Reproduction

Our ranking experiment results remain consistent across changes in the pool dataset from *Dolci* to *Dolly* as well as a different model family, *SmolLM2* for the target decoder (Figure 5).

C.2 Out-of-Distribution Evaluation

We evaluate an INFLUCODER distilled on Dolly pool samples (SmolLM2-1.7B target, BBH queries, the setup is similar to Appendix C.1) on a pool drawn from Fineweb (Penedo et al., 2024), which it did not see during distillation. Table 6 reports the aggregated Spearman correlation on both pools.

Eval pool	Aggregated Spearman (ρ)
Dolly	0.60
Fineweb	0.06

Table 6: Ranking quality in and out of the distillation distribution.