DATA ATTRIBUTION VIA SKETCHED METADIFFERENTIATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Data attribution seeks to quantify how individual training examples shape a model's predictions and underpins problems including data valuation, machine unlearning, and model interpretability. Despite having a long line of work, computationally scalable methods often struggle to predict the effect of removing training data in neural networks due to their non-convex nature. To overcome this challenge, metagradient-based methods such as MAGIC (Ilyas & Engstrom, 2025) differentiate each prediction through the entire training run and compute its exact influence with respect to the training data, but require a separate run for every prediction. To reduce this cost, we cast budgeted attribution as estimating a large influence matrix from a small number of measurements. We show that the measurements most appropriate for recovering this matrix differ from those best suited for attribution itself. We then present two algorithms, MAGE and SPELL, suited for reconstruction and attribution respectively, that run on existing metagradient machinery at no extra cost. Empirical studies demonstrate strong performance over existing baselines across training scales and measurement budgets.

1 INTRODUCTION

With the success of modern deep learning, understanding how training data shapes model behavior has become increasingly important. Previous studies have traced many undesirable model behaviors back to the training data, showing that neural networks can memorize private records (Carlini et al., 2021), inherit targeted vulnerabilities from a few poisoned examples (Wan et al., 2023), and learn shortcuts by relying on signals that are predictive yet brittle (Ilyas et al., 2019). At the same time, data selection allows smaller datasets to match or surpass the performance of larger counterparts for vision and language model training (Sorscher et al., 2022; Li et al., 2024). Similar effects appear during post-training, where a curated fraction of the data can elicit strong instruction following and reasoning (Zhou et al., 2023; Xia et al., 2024; Muennighoff et al., 2025). Together, these findings testify to the central role of training data in shaping model behavior.

One way to understand the impact of training data is to ask how a given metric of interest (e.g., held-out test loss) would have changed had the same model been trained on a different dataset. To answer this question naively would require one to modify the training dataset, retrain the new model, and observe the resulting outputs. Instead, predictive data attribution seeks to predict this counterfactual quantity without retraining (Ilyas et al., 2022) by estimating the contribution of every training example to the prediction of interest. Previous works have studied this problem using tools like approximate influence functions (Koh & Liang, 2017), Shapley values (Ghorbani & Zou, 2019), trajectory tracing (Pruthi et al., 2020), and scalable approximations of influence functions (Park et al., 2023). Despite their computational appeal, these methods have often produced estimates that align poorly with the true outcome of retraining in neural networks (Bae et al., 2022; 2024).

Overcoming these hurdles, MAGIC (Ilyas & Engstrom, 2025) differentiates the loss from a test prediction backward through the entire optimization trajectory, computing its gradient with respect to the training data weights. This gradient—also known as the *exact influence function*—near-perfectly predicts the effects of modifying the training dataset on the given test prediction's loss. However, computing the exact influence function with MAGIC would require a separate reverse differentiation through the full training run for every prediction of interest. This makes MAGIC expensive to scale to settings where we wish to study many held-out examples individually.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

In this work, we ask whether the cost of data attribution can be reduced when estimating the influence scores across many predictions. Our contributions are as follows:

1. We cast the problem of budgeted data attribution across many predictions as that of estimating a Jacobian matrix of influence scores from a limited number of left and right vector products.
2. We formalize two distinct objectives, reconstruction and attribution, and show that the two are optimized using different measurements, since reconstruction weights each prediction by its squared row norm whereas attribution weights them equally.
3. We introduce MAGE (**M**etagradient **A**pproximation by **G**ram **E**igendirections) for reconstruction and SPELL (**SP**herical **E**stimation for the **L**inear datamodeling score at **L**ow budgets) for attribution, both of which run on existing metagradient machinery.
4. Across a ResNet-9 on CIFAR-10 and a 92M-parameter language model on TinyStories, MAGE and SPELL each outperform existing computable alternatives on their intended objective under various measurement budgets.

2 PREDICTIVE DATA ATTRIBUTION AND INFLUENCE MATRIX ESTIMATION

We begin by formalizing both the predictive data attribution framework and the underlying computational primitives. The central idea is to parameterize the training dataset through continuous importance weights, after which our goal becomes computing derivatives of the trained model behavior with respect to these weights. Finally, we state the problem of estimating the influence matrix under a limited budget and define two objectives, reconstruction and attribution, which emphasize distinct aspects of any estimate.

2.1 SETUP AND NOTATION

We follow the datamodeling framework of Ilyas et al. (2022). Let $\mathcal{D} = \{x_i\}_{i=1}^n$ be a dataset of n training points. We parameterize the training set by an importance-weight vector $w \in \mathbb{R}^n$, where w_i scales the contribution of example i to the training loss. With this representation, a standard training pipeline corresponds to $w = \mathbf{1}_n$, where $\mathbf{1}_n \in \mathbb{R}^n$ is the all-ones vector, and setting $w_i = 0$ amounts to removing example i from the training set.

Next, let $\Theta \subseteq \mathbb{R}^p$ denote the parameter space and let $\mathcal{A}$ be a learning algorithm mapping data weights to trained model parameters, $\mathcal{A} : \mathbb{R}^n \to \Theta$ and $\theta(w) = \mathcal{A}(w)$. The learning algorithm $\mathcal{A}$ encompasses every aspect of training beyond the data weights, including the model architecture, optimizer, and learning rate schedule.

Following the single-model setting of Ilyas & Engstrom (2025), we also fix every source of training randomness, whereby the same weighting w always generates the same parameters $\theta(w)$. We further assume that $\mathcal{A}$ is iterative and smooth, meaning that $\theta(w)$ is produced by a fixed sequence of differentiable training updates, and small perturbations to w produce controlled changes in the resulting model output and its derivative.

Finally, we are given k test queries that we wish to attribute to the training data. Throughout, we write $[k] = \{1, \ldots, k\}$. For each test query $q \in [k]$, we let $\phi_q : \Theta \to \mathbb{R}$ be a differentiable scalar measurement of the trained model (e.g., the loss of the model with parameters θ on test query q). We define the model output function

$$f_q(w) = \phi_q(\theta(w)) = \phi_q(\mathcal{A}(w)) \tag{1}$$

as the composition function mapping data weights directly to the measurement for query q, so that $f_q(w)$ is the value that would be observed on query q if the model were trained using data weights w. We also define the vector-valued model output function $F(w) = (f_1(w), \ldots, f_k(w)) \in \mathbb{R}^k$.

2.2 METAGRADIENTS AND MAGIC

The composition in Equation 1 makes the model output function $f_q(w)$ differentiable with respect to the training weights, and its derivative $\nabla_w f_q(\mathbf{1}_n) = \nabla_w \phi_q(\mathcal{A}(w))\big|_{w=\mathbf{1}_n} \in \mathbb{R}^n$ is called the *metagradient* of query q, a gradient taken through the entire training procedure. Its ith entry measures

how the observed value on query q would respond to an infinitesimal change in the weight of training example i. Since computing this derivative naively would be computationally prohibitive, Engstrom et al. (2025) introduce REPLAY, which computes metagradients for large-scale training runs at a cost comparable to that of training itself. Hereinafter, we use the word "replay" synonymously with the REPLAY algorithm.

To leverage metagradients for data attribution, MAGIC (Ilyas & Engstrom, 2025) linearizes the model output function f_q around $w = \mathbf{1}_n$ with a first-order Taylor expansion, $f_q(w) \approx f_q(\mathbf{1}_n) + \langle \nabla_w f_q(\mathbf{1}_n),\, w - \mathbf{1}_n \rangle$, which defines a linear predictor for studying the effect of removing any subset of training examples. Let $S \subseteq [n]$ be a subset of examples to remove and let $\mathbf{1}_S \in \{0,1\}^n$ be its indicator vector, so that training without the examples in S corresponds to the weighting $\mathbf{1}_n - \mathbf{1}_S$. Writing $\Delta_q(S)$ for the resulting change in query q, MAGIC predicts

$$\Delta_q(S) = f_q(\mathbf{1}_n - \mathbf{1}_S) - f_q(\mathbf{1}_n) \approx -\langle \nabla_w f_q(\mathbf{1}_n),\, \mathbf{1}_S \rangle = -\sum_{i \in S} \left[\nabla_w f_q(\mathbf{1}_n)\right]_i .$$

Since the metagradient is evaluated at $\mathbf{1}_n$ and does not depend on $\mathbf{1}_S$, a single metagradient allows us to generate counterfactual predictions for all subsets of $\mathcal{D}$ simultaneously, and these predictions have been shown to near-perfectly estimate the effects of removing training data (Ilyas & Engstrom, 2025).

Finally, we collect the metagradients across all k queries and define the *influence matrix* $Y = \partial F(w)/\partial w|_{w=\mathbf{1}_n} \in \mathbb{R}^{k \times n}$, which records the influence of every training example on every test query. We write Y_q for its qth row—the metagradient of query q—so that recovering Y would allow us to generate counterfactual predictions for all k queries simultaneously.

2.3 Influence Matrix Estimation

MAGIC recovers row Y_q by differentiating f_q backward through the entire training trajectory. Thus, recovering all k rows would require k replays of training. In this paper, we ask:

Given a budget of only $B < k$ replays, how well can we approximate the influence matrix?

We consider two ways of evaluating the quality of any estimate $\widehat{Y}$ of the full influence matrix under this setting, which emphasize distinct yet equally important aspects of the problem.

Reconstruction. The reconstruction error is given by $\|Y - \widehat{Y}\|_F^2 = \sum_{q=1}^{k} \|Y_q - \widehat{Y}_q\|^2$, which is the standard measure of estimation quality for matrix recovery. This is an appropriate objective in settings where we wish to identify the training data driving model behavior, as is the case with data valuation (Ghorbani & Zou, 2019; Jia et al., 2019).

Attribution. Instead of optimizing directly for the proximity between $\widehat{Y}$ and Y in ℓ_2-norm, we can also measure how well the induced predictions track the behavior of models retrained on the modified data. One objective that captures this notion of counterfactual accuracy is the linear datamodeling score (LDS) (Ilyas et al., 2022). At a removal fraction p, the LDS for query q is the (Spearman) correlation, taken over a uniformly random subset $S \subseteq [n]$ of size np, between the change in the model's output on q predicted by the estimate when S is removed and the true change observed after retraining on $[n] \setminus S$. We report the LDS averaged over all queries. This is an appropriate objective in settings where we wish to predict counterfactual model behaviors under interventions on the training dataset, as in the case of machine unlearning (Guo et al., 2020; Bourtoule et al., 2021) and model interpretability (Koh & Liang, 2017).

3 Mage and Spell

In this section, we present Mage and Spell, our procedures for estimating the influence matrix Y from a budget of $B < k$ calls to the metagradient machinery. The two methods share a conceptually straightforward skeleton of measuring B well-chosen linear combinations of the rows of Y, and estimating every row by its projection onto the span of the measurements. They differ primarily in the rule selecting the next combination. We present both procedures in Algorithm 1.

3.1 One Replay Can Measure Combinations of Queries

Recall that MAGIC calculates the attribution scores of a single query using reverse-mode differentiation through the training run. However, since differentiation is linear, MAGIC's computation is not constrained to a single query. To make this precise, let $J = \partial\theta(w)/\partial w|_{w=\mathbf{1}_n} \in \mathbb{R}^{p\times n}$ denote the Jacobian of the trained model parameters with respect to the n training weights. Next, for each query q, let $v_q = \nabla_\theta\, \phi_q(\theta(\mathbf{1}_n)) \in \mathbb{R}^p$ denote the gradient of its measurement with respect to the final parameters. By the chain rule, the influence matrix factors as

$$Y = V^\top J \in \mathbb{R}^{k\times n}, \qquad \text{where} \qquad V = [v_1, \ldots, v_k] \in \mathbb{R}^{p\times k}.$$

Ordinarily, we would recover row Y_q by running REPLAY with v_q as the input to reverse-mode differentiation. More generally, for any coefficient vector z, we can send the combined query gradient Vz through REPLAY, which gives $(Vz)^\top J = z^\top V^\top J = z^\top Y \in \mathbb{R}^n$. Therefore, a single replay can measure any desired linear combination of the rows of Y, and a budget of B replays allows for B such chosen combinations.

3.2 A Projection Estimator via Forward-Mode Differentiation

Having laid out the above, we can estimate every row of Y by its projection onto the span of the acquired measurements. Writing $\mathcal{U} = \operatorname{span}(z_1^\top Y, \ldots, z_B^\top Y) \subseteq \mathbb{R}^n$ for the subspace derived from B vectors $\{z_i\}_{i\in[B]}$, we return the estimate $Y\Pi_{\mathcal{U}}$, where $\Pi_{\mathcal{U}}$ denotes the orthogonal projection onto $\mathcal{U}$.

However, we note that replays themselves cannot determine the projection onto $\mathcal{U}$. This is because projecting row Y_q onto $\mathcal{U}$ also requires its inner product with every measurement, that is,

$$\langle Y_q,\, z^\top Y\rangle = v_q^\top J(z^\top Y)^\top,$$

which involves the unknown row itself. In spite of this, we note that the factor $J(z^\top Y)^\top \in \mathbb{R}^p$ is common to every query, and can be calculated using standard forward-mode differentiation, as observed by Gunn (2026). We then pair each replay with an extra forward-mode pass to obtain these inner products. Having this information allows us to update the estimate along the part of the new measurement orthogonal to all previous replays (Golub & Van Loan, 2013), a step we call Extend-Projection in Algorithm 1.

3.3 Reducing the Two Objectives

To design our measurement strategy, we first note that the nature of orthogonal projections paves the way for powerful reductions of both objectives, which we derive below and make use of throughout our procedures.

Reconstruction. For any query q, we can expand the reconstruction error as

$$\left\|Y_q - \widehat{Y}_q\right\|^2 = \|Y_q\|^2 - 2\,\langle Y_q,\, Y_q\Pi_{\mathcal{U}}\rangle + \|Y_q\Pi_{\mathcal{U}}\|^2 = \|Y_q\|^2 - \|Y_q\Pi_{\mathcal{U}}\|^2\,,$$

where the second equality holds since $\Pi_{\mathcal{U}}$ is idempotent and symmetric. Summing over the k queries, we see that minimizing the reconstruction error is equivalent to maximizing the captured energy.

Attribution. To relate the linear datamodeling score to the orthogonal projection, we first reduce the objective through a series of approximations. Fixing a query q, we can write

$$\begin{aligned}
\mathrm{LDS}_q &= \text{Spearman-}\rho_S\left(-\langle\widehat{Y}_q, \mathbf{1}_S\rangle,\ f_q(\mathbf{1}_n - \mathbf{1}_S) - f_q(\mathbf{1}_n)\right)\\
&\overset{(1)}{\approx} \text{Pearson-}\rho_S\left(-\langle\widehat{Y}_q, \mathbf{1}_S\rangle,\ f_q(\mathbf{1}_n - \mathbf{1}_S) - f_q(\mathbf{1}_n)\right)\\
&\overset{(2)}{\approx} \text{Pearson-}\rho_S\left(-\langle\widehat{Y}_q, \mathbf{1}_S\rangle,\ -\langle Y_q, \mathbf{1}_S\rangle\right) \overset{(3)}{=} \text{Pearson-}\rho\left(\widehat{Y}_q, Y_q\right) \overset{(4)}{\approx} \frac{\langle Y_q\Pi_{\mathcal{U}},\, Y_q\rangle}{\|Y_q\Pi_{\mathcal{U}}\|\,\|Y_q\|}\\
&\overset{(5)}{=} \frac{\langle Y_q\Pi_{\mathcal{U}},\, Y_q\Pi_{\mathcal{U}}\rangle}{\|Y_q\Pi_{\mathcal{U}}\|\,\|Y_q\|} = \frac{\|Y_q\Pi_{\mathcal{U}}\|^2}{\|Y_q\Pi_{\mathcal{U}}\|\,\|Y_q\|} = \frac{\|Y_q\Pi_{\mathcal{U}}\|}{\|Y_q\|}.
\end{aligned}$$

In (1), we replace Spearman correlation with Pearson correlation, since under approximate joint normality of the two sums the two quantities are related monotonically (Kruskal, 1958), so improving

Algorithm 1 MAGE and SPELL.

Input: query gradients V, replay budget B, warm-up length a_0 (SPELL only)
Requires: $\textsc{Replay}(s) = s^\top J$, $\textsc{Forward}(u) = Ju^\top$
$\widehat{Y} \leftarrow 0, \mathcal{U} \leftarrow \{0\}$
for $a = 1, \dots, B$ **do**
  $z_a \leftarrow \textsc{Mage-Select}(a)$ or $\textsc{Spell-Select}(a, \widehat{Y})$
  $u \leftarrow \textsc{Replay}(Vz_a)$ ▷ One replay: $u = z_a^\top Y$
  $c \leftarrow V^\top \textsc{Forward}(u)$ ▷ One forward-mode pass: $c = Yu^\top$
  $\widehat{Y} \leftarrow \textsc{Extend-Projection}(\widehat{Y}, \mathcal{U}, u, c)$
  $\mathcal{U} \leftarrow \mathcal{U} + \mathrm{span}(u)$
return $\widehat{Y}$ ▷ $\widehat{Y} = Y\Pi_{\mathcal{U}}$

procedure $\textsc{Mage-Select}(a)$
  return the ath leading eigenvector of $G_V = V^\top V$

procedure $\textsc{Spell-Select}(a, \widehat{Y})$
  if $a \le a_0$ **then return** $\textsc{Mage-Select}(a)$ ▷ Warm up
  $\widehat{D} \leftarrow \mathrm{diag}(\max\{\|\widehat{Y}_q\|, \tau\})$ ▷ Estimate row norms
  $G_{\mathrm{res}} \leftarrow \textsc{Residual-Gram}(V, [z_1, \dots, z_{a-1}])$ ▷ Project out replayed gradients
  return $\widehat{D}^{-1} v_{\max}(\widehat{D}^{-1} G_{\mathrm{res}} \widehat{D}^{-1})$ ▷ $v_{\max}$: leading eigenvector

one improves the other. In (2), we replace the ground truth by its MAGIC prediction. We defer the calculation of (3) to Appendix A. The approximation in (4) is due to the empirical observation that the rows of Y are nearly mean-zero and centering has a negligible effect on the correlation. Finally, (5) holds since $\Pi_{\mathcal{U}}$ is idempotent and symmetric.

Summing over the k queries and writing $D = \mathrm{diag}\left(\|Y_1\|, \dots, \|Y_k\|\right)$ and $\widetilde{Y} = D^{-1}Y$,

$$\sum_{q=1}^{k} \mathrm{LDS}_q^2 \propto \sum_{q=1}^{k} \frac{\|Y_q \Pi_{\mathcal{U}}\|^2}{\|Y_q\|^2} = \left\|D^{-1} Y \Pi_{\mathcal{U}}\right\|_F^2 = \left\|\widetilde{Y} \Pi_{\mathcal{U}}\right\|_F^2. \tag{2}$$

Thus, we see that optimizing for the LDS likewise maximizes the captured energy of $\widetilde{Y}$.

3.4 SELECTING THE COMBINATIONS TO MEASURE

The top-B right-singular subspace of Y is optimal for reconstruction, and that of $\widetilde{Y}$ is optimal for the attribution surrogate in Equation 2 (Fan, 1949). While the truncated SVD is ordinarily computed from the full matrix, it can also be assembled one direction at a time. To do so, we can probe (i.e., send through replay) the leading eigenvector of the residual Gram matrix, $R = Y(I_n - \Pi_{\mathcal{U}})Y^\top$, where we suppress the iteration index so that $\mathcal{U}$ and R denote the current span and residual. After B iterations, the span of the measurements recovers the top-B right-singular subspace (Hotelling, 1933; Golub & Van Loan, 2013).

However, forming the residual Gram matrix would require the rows of Y (or $\widetilde{Y}$ analogously). Instead, MAGE and SPELL run the iterations using principled stand-ins built from the query gradients and the measurements taken so far, with MAGE tracking the truncated SVD of Y and SPELL that of its row-normalized counterpart.

MAGE substitutes the query Gram matrix $G_V = V^\top V$ for R and probes its leading eigenvectors in order. Here, $z^\top G_V z = \|Vz\|^2$ measures the energy before and $z^\top R z$ the unmeasured energy after replay, so the substitution is reasonable when J preserves the inner products between query gradients.

To derive the measurements for **SPELL**, we first note that for any probe r on $\widetilde{Y}$, $r^\top \widetilde{Y} = (D^{-1}r)^\top Y$, which is equivalent to probing $z \propto D^{-1}r$ on Y. Next, since the residual Gram matrix is given by

$$\widetilde{Y}(I_n - \Pi_{\mathcal{U}})\widetilde{Y}^\top = D^{-1} R D^{-1},$$

we can probe $v_{\max}(D^{-1}RD^{-1})$ on $\widetilde{Y}$, which amounts to probing $z \propto D^{-1}v_{\max}(D^{-1}RD^{-1})$ on Y. Here, both D and R are unknown. We can first approximate the entries of D with the row norms of the estimate in the current iteration. However, since the row norm, and thus its estimate, enters the probe quadratically, any query whose row norm is badly underestimated could commandeer the next probe. To prevent this, we impose a percentile floor

$$\widehat{D} = \mathrm{diag}(\max\{\widehat{d}_q, \tau\}), \qquad \text{and} \qquad \tau = Q_{0.1}(\{\widehat{d}_q : \widehat{d}_q > 0\}), \qquad \text{where} \qquad \widehat{d}_q = \|\widehat{Y}_q\|$$

with $Q_{0.1}$ the empirical tenth percentile. In practice, we observe that using nearby percentiles ($Q_{0.05}$, $Q_{0.15}$, or $Q_{0.2}$) does not noticeably change the estimate.

To approximate R, SPELL follows MAGE's substitution of G_V. However, since $\widehat{D}^{-1}$ is updated after every round, the probes no longer come from one fixed eigenbasis. Consequently, the deflation that MAGE performs implicitly by probing in order, which removes directions already measured before each round, must now be carried out explicitly.

Since the final estimate depends on any probe z only through the combined query gradient Vz, it follows that deflating z amounts to removing the part of Vz measured by the previous replays, which we can achieve by solving the least squares problem $\min_\beta \|Vz - VZ\beta\|$ where $Z = [z_1, \ldots, z_{a-1}]$, with the minimizer given by $\beta = (Z^\top G_V Z)^+ Z^\top G_V z$. Writing the map $z \mapsto z - Z\beta$ in matrix form, we obtain the RESIDUAL-GRAM matrix in Algorithm 1, $P = I_k - Z(Z^\top G_V Z)^+ Z^\top G_V$ and $G_{\mathrm{res}} = P^\top G_V P$. Assembling the approximation and residual projection, we arrive at

$$z \;\propto\; \widehat{D}^{-1} v_{\max}\Big(\widehat{D}^{-1} G_{\mathrm{res}} \widehat{D}^{-1}\Big).$$

Finally, since the estimates may be unreliable in the initial iterations, we warm up SPELL by following MAGE for the first a_0 rounds.

4 EXPERIMENTS

In this section, we evaluate MAGE and SPELL across two settings. In both settings, we first compute the exact influence matrix Y by running a separate replay for each of the k queries. For SPELL, we set the first $a_0 = \max(\lceil 0.1B \rceil, 10)$ calls for the warm-up phase in Algorithm 1.

We compare our methods against three baselines at the same budget of B replays. Random probing replaces MAGE's eigendirection with a random Gaussian combination $z \sim \mathcal{N}(0, I_k)$ of the queries. The first-B-queries baseline replaces it with the coordinate vector $z = e_a$ for $a = 1, \ldots, B$, where these B queries are chosen by class-balanced round robin on CIFAR-10 and by random permutation on TinyStories.

Principal component analysis (PCA) reconstructs $\widehat{Y} = V_B^\top J$, where V_B is the rank-B SVD of V, by passing the top-B principal directions of the query gradients V directly through replay. We also include two oracle baselines computable only with access to Y, which are the rank-B singular value decomposition (SVD) of Y, which achieves the smallest reconstruction error at rank B, and the rank-B SVD of the row-normalized counterpart of Y.

We evaluate all methods on the two aforementioned objectives in Section 2.3. For reconstruction, we report the relative Frobenius error

$$\Big\|Y - \widehat{Y}\Big\|_F \Big/ \Big\|Y\Big\|_F.$$

For data attribution, we report the linear datamodeling score (LDS). To approximate this quantity for a given test query q and removal fraction p, we take the following steps:

1. Draw M subsets $S_1, \ldots, S_M \subseteq [n]$ of size np uniformly at random to be removed from the training set.
2. Retrain the model on each subset $[n] \setminus S_j$ by setting the weight of every removed example to zero, holding all other training randomness fixed.
3. Record for every test query q the predicted change $\widehat{\Delta}_q(S_j) = -\langle \widehat{Y}_q, \mathbf{1}_{S_j} \rangle$ and true change $\Delta_q(S_j) = f_q(\mathbf{1}_n - \mathbf{1}_{S_j}) - f_q(\mathbf{1}_n)$ in the model output function.
4. Compute the Spearman correlation between the predicted and true change over all M subsets.

Finally, we average the LDS over all k test queries, which gives

$$\overline{\mathrm{LDS}} = \frac{1}{k}\sum_{q=1}^{k} \rho\left(\left[\widehat{\Delta}_q(S_j)\right]_{j=1}^{M}, \left[\Delta_q(S_j)\right]_{j=1}^{M}\right).$$

We use $M = 300$ retrained models at two removal fractions of 1% and 5%, and hold the subsets fixed across test queries and methods. We also plot the estimate given by the exact influence matrix $-\langle Y_q, \mathbf{1}_{S_j}\rangle$ as the ceiling (MAGIC).

4.1 ResNet-9 on CIFAR-10

We first consider a ResNet-9 architecture trained on CIFAR-10 (Krizhevsky, 2009), where our goal is to attribute the cross-entropy loss on up to $k = 1000$ held-out test images back to the training data. The model contains 10M parameters and is trained on the full set of 50K training examples (see Appendix B.1 for further details). We then average the results over ten random seeds, where the resulting models attain a mean 89.88% test accuracy on the full CIFAR-10 test set. For the figures below, we plot the relative Frobenius error and linear datamodeling score against the number of replays B over the total number of test queries k.

We see in Figure 1 that MAGE attains the lowest reconstruction error of any implementable method at every budget and query set size. For data attribution, the top panel of Figure 2 shows that SPELL consistently outperforms computable alternatives at every size k once the budget reaches a fifth of the query set size, when 1% of the training data is dropped. Furthermore, as shown in the bottom panel of Figure 2, SPELL retains the same advantage when 5% of the training data is dropped. Table 1 details both metrics at $k = 500$ under the budget of $B = 50$. At $B = 50$, MAGE attains a relative Frobenius error of 0.49 against 0.53 for PCA, 0.62 for random probes, and 0.92 for the first-B queries, while SPELL obtains an LDS of 0.34 against 0.25 for random probes, 0.22 for the first-B queries, and 0.15 for PCA when dropping 1% of the data.

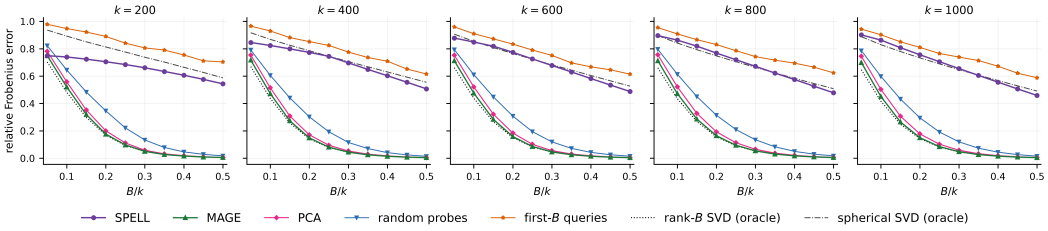


Figure 1: Reconstruction error across 10 seeded ResNet-9 models trained on CIFAR-10 and query set size k. MAGE is nearly indistinguishable from the oracle rank-B SVD.

4.2 GPT-2 on TinyStories

For our second domain, we fine-tune a 12-layer GPT-2-style decoder (Radford et al., 2019) on TinyStories (Eldan & Li, 2023). We consider a chunk of 512 consecutive tokens as a single training example, and the query value is the negative log-likelihood the model assigns to the single token following a held-out chunk of the same length. We fine-tune the model on 2048 training examples, where our goal is to attribute up to 600 test queries back to the training data. The model has 92M parameters, and only the fine-tuning stage is attributed starting from a fixed pretrained checkpoint shared across all runs (see Appendix B.2 for further details). We then average the results over four runs, where the fine-tuned models attain a mean 65.04% top-1 next-token accuracy on the held-out query corpus.

In Figure 3, we see that MAGE attains the lowest reconstruction error of any implementable method at every budget and query set size. For data attribution, the top panel of Figure 4 shows that SPELL also outperforms every computable alternative at every k once the budget reaches a fifth of the query set size, when 1% of the fine-tuning data is dropped. We also see in the bottom panel of Figure 4 that SPELL remains performant when 5% of the data is dropped. In greater detail, Table 2 shows that at $k = 500$ with budget $B = 50$, MAGE attains a relative Frobenius error of 0.70 against 0.71 for

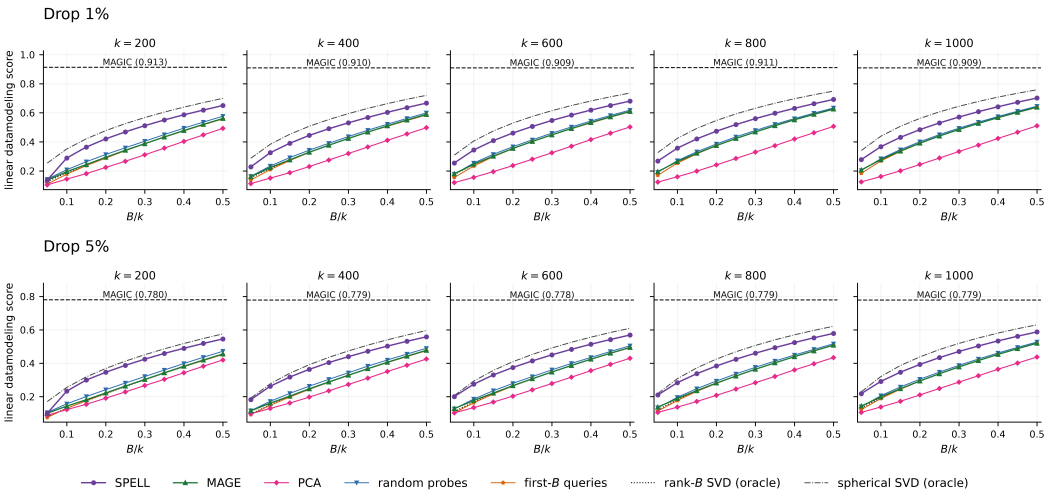


Figure 2: Linear datamodeling score across 10 seeded ResNet-9 models trained on CIFAR-10 and query set size k after removing a random 1% (top) and 5% (bottom) of the training set. SPELL outperforms all computable methods at every k once $B \geq 0.2k$.

Table 1: Performance of methods on ResNet-9 at $k = 500$ with $B = 50$ on CIFAR-10, evaluated against models retrained after removing a random 1% and 5% of the training set. We **bold** the best computable method in each column, where MAGIC is the ceiling given by the exact influence matrix at $B = k$. The values are averaged over ten runs and reported along with their standard errors.

			Linear datamodeling score ↑	
Budget	Method	Relative Frobenius error ↓	1% removed	5% removed
$B = 50$	▲ MAGE	**0.4872** ±0.0064	0.2358 ±0.0017	0.1698 ±0.0032
	● SPELL	0.8419 ±0.0061	**0.3353** ±0.0023	**0.2696** ±0.0045
	▼ Random probes	0.6231 ±0.0049	0.2461 ±0.0013	0.1803 ±0.0032
	◆ PCA	0.5316 ±0.0061	0.1541 ±0.0010	0.1327 ±0.0017
	★ First B queries	0.9177 ±0.0040	0.2226 ±0.0024	0.1568 ±0.0032
	···· Rank-B SVD	0.4483 ±0.0058	0.2328 ±0.0022	0.1599 ±0.0030
	-·- Spherical SVD	0.8611 ±0.0049	0.3948 ±0.0023	0.2852 ±0.0056
$B = k$	-- MAGIC	–	0.9087 ±0.0038	0.7778 ±0.0058

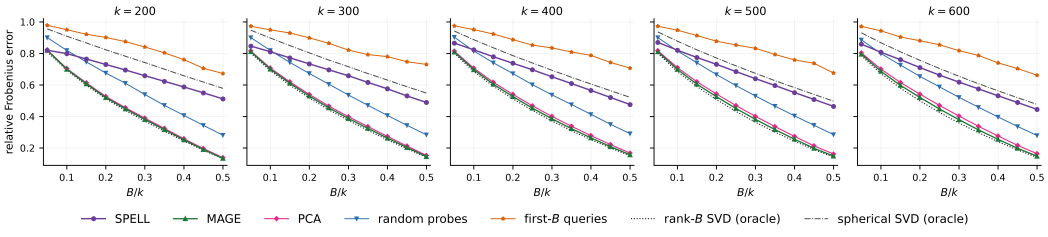


Figure 3: Reconstruction error across 4 seeded GPT-2-style decoders fine-tuned on TinyStories and query set size k. MAGE is nearly indistinguishable from the oracle rank-B SVD.

PCA, 0.82 for random probes, and 0.95 for the first-B queries, while SPELL obtains an LDS of 0.39 against 0.32 for random probes, 0.27 for the first-B queries, and 0.23 for PCA when dropping 1% of the data.

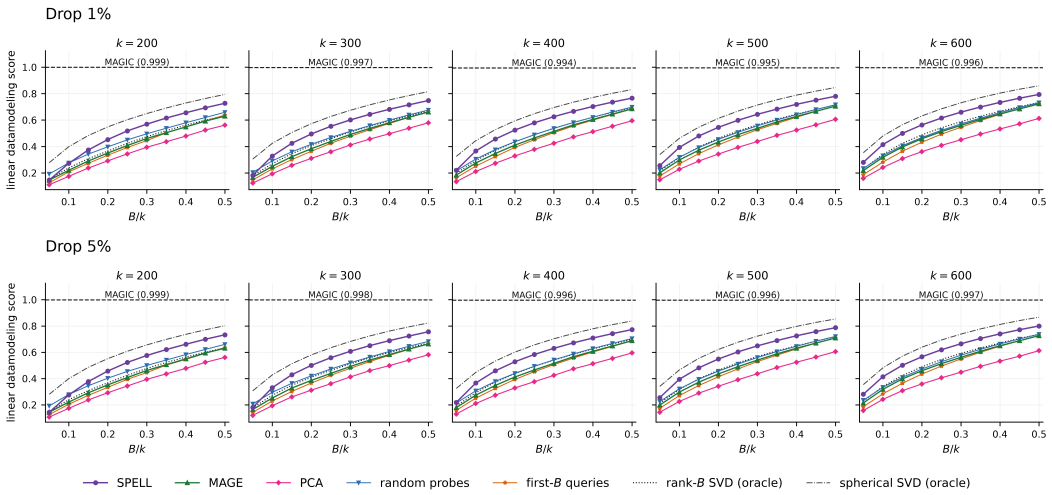


Figure 4: Linear datamodeling score across 4 seeded GPT-2-style decoders fine-tuned on TinyStories and query set size k after removing a random 1% (top) and 5% (bottom) of the fine-tuning set. SPELL outperforms all computable methods at every k once $B \geq 0.2k$.

Table 2: Performance of methods on GPT-2 at $k = 500$ with $B = 50$ on TinyStories, evaluated against models retrained after removing a random 1% and 5% of the fine-tuning set. We **bold** the best computable method in each column, and MAGIC is the ceiling given by the exact influence matrix at $B = k$. The values are averaged over four runs and reported along with their standard errors.

			Linear datamodeling score ↑	
Budget	Method	Relative Frobenius error ↓	1% removed	5% removed
$B = 50$	▲ MAGE	**0.6986** ±0.0012	0.2980 ±0.0018	0.2970 ±0.0017
	● SPELL	0.8206 ±0.0013	**0.3931** ±0.0004	**0.3947** ±0.0017
	▼ Random probes	0.8165 ±0.0010	0.3191 ±0.0014	0.3221 ±0.0018
	◆ PCA	0.7104 ±0.0012	0.2275 ±0.0022	0.2258 ±0.0012
	★ First B queries	0.9478 ±0.0003	0.2683 ±0.0015	0.2704 ±0.0014
	···· Rank-B SVD	0.6847 ±0.0009	0.3164 ±0.0027	0.3168 ±0.0023
	-·- Spherical SVD	0.8777 ±0.0008	0.4618 ±0.0016	0.4654 ±0.0018
$B = k$	-- MAGIC	–	0.9952 ±0.0004	0.9964 ±0.0002

5 CONCLUSION

In this paper, we cast data attribution at scale as a problem of estimating the influence matrix from a small number of measurements, using the observation that a single replay measures any linear combination of its rows. Next, we introduce MAGE and SPELL, algorithms tailored to recover the matrix itself and to preserve data attribution fidelity respectively. Across a ResNet-9 on CIFAR-10 and a GPT-2-style decoder on TinyStories, both methods consistently outperform every computable alternative on their targeted objectives. Given that the cost of both procedures is set by the measurement budget rather than by the number of queries, they enable efficient attribution for many queries simultaneously where recovering the exact influence matrix may be computationally prohibitive.

One promising avenue for future work is to investigate whether better selection rules exist, and in particular whether the attribution objective admits a measurement that is optimal in the same way that truncated SVD is optimal for reconstruction, which would help guide the design of practical selection rules at a fixed measurement budget.

AI USE STATEMENT

In this work, we used generative AI tools for suggesting edits for the manuscript, implementing methods, creating and modifying the figures, and identifying gaps in related works. We take responsibility for the final content of this submission.

REFERENCES

Juhan Bae, Nathan Ng, Alston Lo, Marzyeh Ghassemi, and Roger Grosse. If influence functions are the answer, then what is the question? In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2022.

Juhan Bae, Wu Lin, Jonathan Lorraine, and Roger Grosse. Training data attribution via approximate unrolling. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, 2024.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine Unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, May 2021.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting Training Data from Large Language Models. In *Proceedings of the 30th USENIX Security Symposium*, 2021.

Ronen Eldan and Yuanzhi Li. TinyStories: How Small Can Language Models Be and Still Speak Coherent English?, 2023. arXiv:2305.07759 [cs.CL].

Logan Engstrom, Andrew Ilyas, Benjamin Chen, Axel Feldmann, William Moses, and Aleksander Madry. Optimizing ML Training with Metagradient Descent, 2025. arXiv:2503.13751 [stat.ML].

Ky Fan. On a Theorem of Weyl Concerning Eigenvalues of Linear Transformations I. *Proceedings of the National Academy of Sciences*, 35(11), 1949.

Amirata Ghorbani and James Zou. Data Shapley: Equitable Valuation of Data for Machine Learning. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, 4 edition, 2013.

Sam Gunn. How to sketch a learning algorithm, 2026. arXiv:2604.07328 [cs.LG].

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified Data Removal from Machine Learning Models. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Harold Hotelling. Analysis of a Complex of Statistical Variables into Principal Components. *Journal of Educational Psychology*, 24(6), 1933.

Andrew Ilyas and Logan Engstrom. MAGIC: Near-Optimal Data Attribution for Deep Learning, 2025. arXiv:2504.16430 [cs].

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial Examples Are Not Bugs, They Are Features. In *Proceedings of the 32nd International Conference in Neural Information Processing Systems*, 2019.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. Datamodels: Understanding Predictions with Data and Data with Predictions. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

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 *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, 2019.

Pang Wei Koh and Percy Liang. Understanding Black-box Predictions via Influence Functions. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2017.

Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009.

William H. Kruskal. Ordinal Measures of Association. *Journal of the American Statistical Association*, 53(284), 1958.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Josh Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alexandros G. Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. DataComp-LM: in search of the next generation of training sets for language models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, 2024.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.

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*, 2023.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners, 2019.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond Neural Scaling Laws: Beating Power Law Scaling via Data Pruning. In *Advances in Neural Information Processing Systems*, 2022.

Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. Poisoning Language Models During Instruction Tuning. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting Influential Data for Targeted Instruction Tuning. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: less is more for alignment. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, 2023.

A Proof of the Pearson Correlation Identity

First, we define

$$H = I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top, \qquad \alpha = \frac{p(1-p)n}{n-1}.$$

For a uniformly random subset S of size np, $\mathrm{Cov}_S(\mathbf{1}_S) = \alpha H$, and we can write

$$\begin{aligned}
\text{Pearson-}\rho_S\left(-\langle \widehat{Y}_q, \mathbf{1}_S\rangle,\ -\langle Y_q, \mathbf{1}_S\rangle\right) &= \frac{\widehat{Y}_q \mathrm{Cov}_S(\mathbf{1}_S) Y_q^\top}{\sqrt{\widehat{Y}_q \mathrm{Cov}_S(\mathbf{1}_S)\widehat{Y}_q^\top}\sqrt{Y_q \mathrm{Cov}_S(\mathbf{1}_S) Y_q^\top}} \\
&= \frac{\alpha \widehat{Y}_q H Y_q^\top}{\sqrt{\alpha \widehat{Y}_q H \widehat{Y}_q^\top}\sqrt{\alpha Y_q H Y_q^\top}} \\
&= \frac{\widehat{Y}_q H Y_q^\top}{\sqrt{\widehat{Y}_q H \widehat{Y}_q^\top}\sqrt{Y_q H Y_q^\top}} \\
&= \frac{\big(\widehat{Y}_q H\big)\big(Y_q H\big)^\top}{\big\|\widehat{Y}_q H\big\|\big\|Y_q H\big\|} \\
&= \text{Pearson-}\rho(\widehat{Y}_q, Y_q),
\end{aligned}$$

where the penultimate equality holds because H is idempotent and symmetric.

B Architecture Details

B.1 ResNet-9 on CIFAR-10

Our ResNet-9 consists of eight 3×3 convolutions, each followed by batch normalization and GELU nonlinearity, and a final linear head. The first layer widens the input image to 80 channels at full resolution, and the second layer widens it to 160 and applies a 2×2 average pooling. Two more layers at 160 channels form the first residual block. The next two layers widen and downsample twice more, to 320 channels at 8×8 and then to 640 channels at 4×4. The last two convolutional layers form a second residual block at 640 channels. A final 4×4 average pooling reduces each channel to a single value, and a linear head maps the resulting 640-dimensional vector to the ten class logits.

Table 3: Training hyperparameters for ResNet-9 on CIFAR-10. The learning rate and the weight decay are both divided by the momentum correction $1 + 1/(1 - 0.85)$.

Hyperparameter	Value
Learning rate	0.085
Weight decay	5×10^{-4}
Batch size	500
Epochs	20
Training steps T	2,000
Training examples n	50,000
Optimizer	SGD
Momentum	0.85
LR schedule	One-cycle linear
LR start multiplier	0.1
LR end multiplier	0.0
LR peak time	0.3

The $k = 1000$ queries are randomly chosen to ensure balance at 100 images per class and prefix-nested in k, so that the $k = 100$ set is the first 100 rows of the $k = 1000$ set.

B.2 GPT-2 ON TINYSTORIES

Our model is a 12-layer pre-norm transformer decoder of width 768 with 12 attention heads of 64 dimensions. The input is a chunk of 512 tokens and is mapped through an embedding table and added to a learned position embedding, where we tokenize with a vocabulary of 8192 entries that contains the 7936 most frequent words of the pretraining text together with 256 single-byte fallbacks. Each block applies causal self-attention and a feed-forward network that widens from 768 to 3072 through a GELU nonlinearity, preceded by layer normalization and wrapped in a residual connection.

We partition the TinyStories training corpus into chunks of length 512 and divide them into three splits with no overlap, reserving 896,000 chunks for pretraining, 4473 for fine-tuning, and 1024 for testing. We attribute on the test split, where each of our four runs fine-tunes on a different draw of $n = 2048$ chunks from the fine-tuning split.

Table 4: Training hyperparameters for the language model on TinyStories.

Hyperparameter	Pretraining	Fine-tuning
Learning rate	6×10^{-4}	5×10^{-6}
Weight decay	0.1	10^{-5}
Batch size	128	16
Epochs	2	2
Training steps T	14,000	256
Training examples n	896,000	2,048
Optimizer	AdamW	AdamW
(β_1, β_2)	$(0.95, 0.999)$	$(0.95, 0.999)$
LR schedule	One-cycle linear	One-cycle linear
LR start multiplier	10^{-6}	10^{-6}
LR end multiplier	0.1	0.1
LR peak time	0.25	0.25
Matmul precision	TF32	fp32

C GEOMETRY OF THE INFLUENCE MATRIX

In Section 3.3, we have shown that reconstruction and attribution weight queries differently, and have seen in Section 4 that MAGE and SPELL indeed excel at their respective tasks. Here, we provide some further intuition behind this phenomenon. In particular, we show that for the two scenarios considered in Section 4, the rows of the influence matrix differ by many orders of magnitude in norm.

In Figure 5, we sort the queries using their row norms $\|Y_q\|$ in decreasing order and plot them along with the share of total energy accumulated thus far. For the ResNet-9 setting considered in Section 4.1 with $k = 1000$, the norms span 8.6 orders of magnitude on CIFAR-10, and the largest quartile of queries holds 99.20% of the total energy. For the GPT-2 setting shown in Section 4.2 with $k = 600$, the row norms span 7.4 orders of magnitude, where the largest quartile of queries captures 77.14% of the total energy on TinyStories.

Since the relative Frobenius error weights each query by the square of its row norm $\|Y_q\|^2$, it follows that on CIFAR-10 the error is determined almost entirely by the top quartile. The linear datamodeling score instead averages over every query equally regardless of its norm. Consequently, the same estimate can be scored very differently under the two objectives. To measure this behavior, we consider the mean per-query relative error

$$\frac{1}{k} \sum_{q=1}^{k} \left\| Y_q - \widehat{Y}_q \right\| \Big/ \left\| Y_q \right\|,$$

which normalizes each error by its ground-truth influence. Table 5 shows that MAGE attains a relative Frobenius error of 0.0045 on CIFAR-10 at $k = 1000$ and $B = 500$ against a mean per-query relative error of 0.447. In contrast, SPELL attains a balanced performance of 0.458 against 0.584.

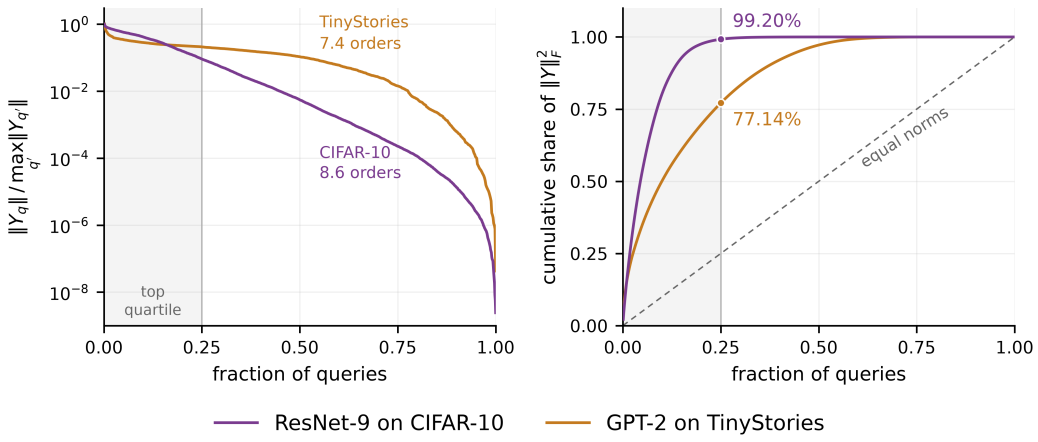


Figure 5: Row norms of the exact influence matrix Y at $k = 1000$ on CIFAR-10 and $k = 600$ on TinyStories, sorted in decreasing order. The left panel plots the row norm of each query divided by the largest, and the right panel plots the fraction of the total energy the queries accumulate, where the shaded region represents the top quartile of queries. The values are averaged over ten runs for CIFAR-10 and four on TinyStories.

Table 5: Performance of methods across ResNet-9 at $k = 1000$ with $B = 500$ on CIFAR-10, and GPT-2 at $k = 600$ with $B = 300$ on TinyStories. The values are averaged over ten runs for CIFAR-10 and four on TinyStories.

Setting	Method	Relative Frobenius error	Mean per-query error	Ratio
CIFAR-10	MAGE	0.0045	0.447	100
	SPELL	0.458	0.584	1.3
	Random probes	0.0152	0.491	32
	PCA	0.0057	0.576	101
TinyStories	MAGE	0.148	0.459	3.1
	SPELL	0.445	0.545	1.2
	Random probes	0.279	0.568	2.0
	PCA	0.164	0.507	3.1

To capture a clearer picture of the underlying geometry, we sort the queries by $\|Y_q\|$ and partition them into quartiles. Table 6 below reports the per-query relative error and the linear datamodeling score within every quartile. Across both scenarios, MAGE reconstructs the largest quartile almost exactly but learns little about the smallest quartile, and its attribution quality follows the same pattern. In contrast, SPELL remains nearly flat on both measures.

While the smallest quartile may appear unpredictable, the exact influence matrix suggests otherwise, scoring 0.88 on the smallest quartile and 0.92 on the largest, whereas MAGE attains only 0.35 on the smallest against 0.59 for SPELL on CIFAR-10. This suggests that it could be helpful for practitioners to understand the imbalance in the row norms before deciding on the optimal attribution strategy.

Surprisingly, we also find that the row norms of Y in the scenarios above are nearly perfectly ranked by the norms of the query gradients themselves, with a mean Spearman correlation of 0.999 on CIFAR-10 and 0.996 on TinyStories across the same runs. Given that the query gradients are already accessible after training, a practitioner could read off how concentrated a query set is in advance, and decide whether to invoke MAGE or SPELL before spending any measurement budget. We leave a theoretical justification of why this phenomenon occurs for future investigation.

Table 6: Performance of methods across ResNet-9 at $k = 1000$ with $B = 500$ on CIFAR-10, and GPT-2 at $k = 600$ with $B = 300$ on TinyStories. Queries are partitioned into quartiles by $\|Y_q\|$ in increasing order, with Q1 being the smallest and Q4 being the largest. The linear datamodeling score is measured against models retrained after removing a random 1% of the training set. The values are averaged over ten runs for ResNet-9 and four on TinyStories.

		Per-query relative error				Linear datamodeling score				
Setting	Quartile	MAGE	SPELL	Random	PCA	MAGE	SPELL	Random	PCA	MAGIC
CIFAR-10	Q1	0.88	0.69	0.88	1.14	0.35	0.59	0.35	0.09	0.88
	Q2	0.87	0.57	0.82	1.12	0.37	0.73	0.44	0.12	0.92
	Q3	0.04	0.56	0.25	0.05	0.91	0.73	0.87	0.91	0.92
	Q4	0.00	0.52	0.02	0.00	0.92	0.76	0.92	0.92	0.92
TinyStories	Q1	0.89	0.64	0.89	0.98	0.43	0.69	0.43	0.20	0.98
	Q2	0.85	0.54	0.73	0.94	0.48	0.81	0.63	0.28	1.00
	Q3	0.09	0.54	0.41	0.10	0.98	0.82	0.90	0.97	1.00
	Q4	0.01	0.45	0.24	0.01	1.00	0.86	0.96	1.00	1.00