A Versatile Influence Function for Data Attribution with Non-Decomposable Loss

Anonymous Author(s)
Affiliation
Address
email

Abstract

Influence function, a technique rooted in robust statistics, has been adapted in modern machine learning for a novel application: data attribution—quantifying how individual training data points affect a model's predictions. However, the common derivation of influence functions in the data attribution literature is limited to loss functions that decompose into a sum of individual data point losses, with the most prominent examples known as M-estimators. This restricts the application of influence functions to more complex learning objectives, which we refer to as *non-decomposable losses*, such as contrastive or ranking losses, where a unit loss term depends on multiple data points and cannot be decomposed further. In this work, we bridge this gap by revisiting the general formulation of influence function from robust statistics, which extends beyond M-estimators. Based on this formulation, we propose a novel method, the *Versatile Influence Function* (VIF), that can be straightforwardly applied to machine learning models trained with any non-decomposable loss. In comparison to the classical approach in statistics, the proposed VIF is designed to fully leverage the power of auto-differentiation, hereby eliminating the need for case-specific derivations of each loss function. We demonstrate the effectiveness of VIF across three examples: Cox regression for survival analysis, node embedding for network analysis, and listwise learning-to-rank for information retrieval. In all cases, the influence estimated by VIF closely resembles the results obtained by brute-force leave-one-out retraining, while being up to 1000 times faster to compute. We believe VIF represents a significant advancement in data attribution, enabling efficient influence-function-based attribution across a wide range of machine learning paradigms, with broad potential for practical use cases.

1 Introduction

Influence function (IF) is a well-established technique originating from robust statistics and has been adapted to the novel application of *data attribution* in modern machine learning (Koh & Liang, 2017). Data attribution aims to quantify the impact of individual training data points on model outputs, which enables a wide range of data-centric applications such as mislabeled data detection (Koh & Liang, 2017), data selection (Xia et al., 2008), and copyright compensation (Deng & Ma, 2023).

Despite its broad potential, the application of IFs for data attribution has been largely limited to loss functions that decompose into a sum of individual data point losses—such as those commonly used in supervised learning objectives or maximum likelihood estimation, which are also known as M-estimators. This limitation arises from the specific way IFs are typically derived in the data attribution literature (Koh & Liang, 2017; Grosse et al., 2023), where the derivation involves perturbing the weights of individual data point losses. As a result, this restricts the application of IF-based data

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

attribution methods to more complex machine learning objectives, such as contrastive or ranking losses, where a unit loss term depends on multiple data points and cannot be decomposed into individual data point losses. We refer to these loss functions as *non-decomposable losses*.

To address this limitation, we revisit the general formulation of IF in the literature of statistics (Huber & Ronchetti, 2009), where statistical estimators are viewed as functionals of probability measures, and the IF is derived as a functional derivative in a specific perturbation direction. This formulation extends beyond M-estimators and, in principle, applies to any estimator (which corresponds to the learned parameters in the context of machine learning) defined as the minimizer of a loss function that depends on an (empirical) probability measure. However, directly applying this general formulation to modern machine learning models poses significant challenges. Firstly, deriving the precise IF for a particular loss function often requires complex, case-by-case mathetical derivations, which can be challenging for intricate loss functions and models. Secondly, for non-convex models, the mapping from the probability measure to the model parameters is not well-defined, making the IF derivation unclear.

To overcome these challenges, we propose the Versatile Influence Function (VIF), a novel method that extends IF-based data attribution to models trained with non-decomposable losses. The proposed VIF serves as an approximation of the general formulation of IF but can be efficiently computed using auto-differentiation tools available in modern machine learning libraries. This approach eliminates the need for case-specific derivations of each loss function. Furthermore, like existing IF-based data attribution methods, VIF does not require model retraining and can be generalized to non-convex models using similar heuristic tricks.

We validate the effectiveness of VIF both theoretically and empirically. In special cases like M-estimators, VIF recovers the classical IF exactly. For Cox regression, we show that VIF closely approximates the classical IF. Empirically, we demonstrate the practicality of VIF across several tasks involving non-decomposable losses: Cox regression for survival analysis, node embedding for network analysis, and listwise learning-to-rank for information retrieval. In all cases, VIF closely approximates the influence obtained from the brute-force leave-one-out retraining while significantly reducing computational time—achieving speed-ups of up to 1000 times. We also provide case studies demonstrating VIF can help interpret the behavior of the models.

By extending IF to non-decomposable losses, VIF opens new opportunities for data attribution in modern machine learning models, enabling data-centric applications across a wider range of domains.

2 The Versatile Influence Function

2.1 Non-Decomposable Loss

In practice, there are many common loss functions that are *not* decomposable. Below we list a few examples. Please refer to Appendix B for detailed information.

Example 1: Cox's Partial Likelihood. The Cox regression model (Cox, 1972) is one of the most widely used models in survival analysis, designed to predict the time until specific events occur (e.g., patient death or a customer's next purchase).

Example 2: Contrastive Loss. Contrastive losses are commonly seen in unsupervised representation learning across various modalities, such as word embeddings (Mikolov et al., 2013), image representations (Chen et al., 2020), or node embeddings (Perozzi et al., 2014).

Example 3: Listwise Learning-to-Rank. Learning-to-rank is a core technology underlying information retrieval applications such as search and recommendation. In this context, listwise learning-to-rank methods aim to optimize the ordering of a set of documents or items based on their relevance to a given query. One prominent example of such methods is ListMLE (Xia et al., 2008).

A General Loss Formulation. The examples above can be viewed as special cases of the following formal definition of *non-decomposable loss*.

Definition 2.1 (Non-Decomposable Loss)**.** *Given n objects of interest within the training data, let a binary vector $b \in \{0,1\}^n$ indicate the presence of the individual objects for training, i.e., for $i = 1, \ldots, n$,*

$$b_i = \begin{cases} 1 & \textit{if the } i\textit{-th object presents,} \\ 0 & \textit{otherwise.} \end{cases}$$

Suppose the machine learning model parameters are denoted as $\theta \in \mathbb{R}^d$, a non-decomposable *is any function $\mathcal{L} : \mathbb{R}^d \times \{0,1\}^n \to \mathbb{R}$, that maps given model parameters θ and the object presence vector b to a loss value $\mathcal{L}(\theta, b)$.*

Generalizing the notation $\hat{\theta}(b) = \arg\min_\theta \mathcal{L}(\theta, b)$ on any non-decomposable loss $\mathcal{L}(\theta, b)$, the LOO effect of data point i on the learned parameters is still well-defined by $\hat{\theta}(\mathbf{1}_{-i}) - \hat{\theta}(\mathbf{1})$.

However, in this case, we can no longer use the partial derivative with respect to b_i to approximate the LOO effect, as $\hat{\theta}(b)$ is only well-defined for binary vectors b.

Remark 2.2 ("Non-Decomposable" v.s. "Not Decomposable")**.** *The class of non-decomposable loss in Definition 2.1 includes the decomposable loss in Eq.* (6) *as a special case when $\mathcal{L}(\theta, b) := \sum_{i:b_i=1} l_i(\theta)$. In fact, the method we will develop is applicable to all the loss in Definition 2.1 (with some nice properties such as convexity), including the decomposable ones (in which case our method reduces to the conventional IF-based method as shown in Appendix D). Throughout this paper, we will call loss functions that cannot be written in the form of Eq.* (6) *as "not decomposable". We name the general class of loss functions in Definition 2.1 as* non-decomposable loss *to highlight that they are generally not decomposable.*

Remark 2.3 (Randomness in Losses)**.** *Strictly speaking, many contrastive losses are not deterministic functions of training data points as there is randomness in the construction of the triplet set D, due to procedures such as negative sampling or random walk. However, our method derived for the deterministic non-decomposable loss still gives meaningful results in practice for losses with randomness.*

2.2 The Statistical Perspective of Influence Function

The Statistical Formulation of IF. To derive IF-based data attribution for non-decomposable losses, we revisit a general formulation of IF in robust statistics (Huber & Ronchetti, 2009). Let Ω be a sample space, and T is a vector-valued statistics that maps from a subset of the probability measures on Ω to a vector in $\mathbb{R}^d$. Let P and Q be two probability measures on Ω. The IF of a statistics $T(P)$ measures the infinitesimal change of the statistics towards a specific perturbation direction Q, which is defined as

$$\text{IF}(T(P); Q) := \lim_{\varepsilon \to 0} \frac{T((1-\varepsilon)P + \varepsilon Q) - T(P)}{\varepsilon}.$$

In the context of machine learning, the learned model parameters, denoted as $\hat{\theta}(P)$, can be viewed as statistics derived from the data distribution P. Specifically, the learned model parameters are typically obtained by minimizing a loss function, i.e., $\hat{\theta}(P) = \arg\min_\theta \mathcal{L}(\theta, P)$, where the loss depends on both the parameters and P.

Assuming the loss is strictly convex and twice-differentiable with respect to the parameters, the learned parameters $\hat{\theta}(P)$ are then implicitly determined by the following equation

$$\nabla_\theta \mathcal{L}(\hat{\theta}(P), P) = \mathbf{0}.$$

Moreover, the IF of $\hat{\theta}(P)$ for a perturbation towards Q is[1]

$$\text{IF}(\hat{\theta}(P); Q) = -\left[\nabla_\theta^2 \mathcal{L}(\hat{\theta}(P), P)\right]^{-1} \lim_{\varepsilon \to 0} \frac{\nabla_\theta \mathcal{L}(\hat{\theta}(P), (1-\varepsilon)P + \varepsilon Q) - \nabla_\theta \mathcal{L}(\hat{\theta}(P), P)}{\varepsilon}. \quad (1)$$

The advantage of the IF formulation in Eq. (1) is that it can be applied to more general loss functions by properly specifying P, Q, and $\mathcal{L}$.

Example: Application of Eq. (1) to M-Estimators. As an example, the following Lemma 2.4 states that the IF used by Koh & Liang (2017) in Eq. (8) can be viewed as a special case of the formulation in Eq. (1). This is a well-known result in robust statistics (Huber & Ronchetti, 2009), and the proof of which can be found in Appendix C.2. Intuitively, with the choice of P, Q, and $\mathcal{L}$ in Lemma 2.4, $(1-\varepsilon)P + \varepsilon Q = (1-\varepsilon)P + \varepsilon\delta_{z_i}$ exactly leads to the effect of upweighting the loss weight of z_i with a small perturbation, which is how the IF in Eq. (8) is derived.

Lemma 2.4 (IF for M-Estimators)**.** *Eq.* (1) *reduces to Eq.* (8) *when we specify that 1) P is the empirical distribution over a dataset $\{z_i\}_{i=1}^n$, i.e., $\Pr(z_i) = \frac{1}{n}$, for all $i = 1, \ldots, n$; 2) Q is δ_{z_i}, i.e., $\Pr(z_i) = 1$ and $\Pr(z_j) = 0, j \neq i$; and 3) $\mathcal{L}(\theta, P) := \mathbb{E}_{z\sim P}\left[\ell(\theta, z)\right] = \frac{1}{n}\sum_{i=1}^n \ell(\theta, z_i)$.*

[1]See Appendix C.1 for the derivation.

Challenges of Applying Eq. (1) in Modern Machine Learning. While the IF in Eq. (1) is a principled and well-established notion in statistics, there are two unique challenges when applying it to modern machine learning models. Firstly, solving the limit in the right hand side of Eq. (1) requires case-by-case derivation for different loss functions and models, which can be mathematically challenging (see an example of IF for the Cox regression in Appendix C.3). Secondly, the mapping $\hat{\theta}(P)$, hence the limit, are not well-defined for non-convex loss functions. A similar problem exists in the IF for decomposable loss in Eq. (8) and Koh & Liang (2017) mitigate this problem through heuristic tricks specifically designed for Eq. (8). However, the IF in Eq. (1) is more complicated and how to generalize it to modern setups like neural networks remains unclear.

2.3 VIF as A Finite Difference Approximation

We now derive the proposed VIF method by applying Eq. (1) to the non-decomposable loss while addressing the aforementioned challenges through an approximation.

Properties of P, Q, and $\mathcal{L}$ in Eq. (1) for Non-Decomposable Loss. Recall that our goal is to derive an IF under non-decomposable loss that approximates the LOO effect $\hat{\theta}(\mathbf{1}_{-i}) - \hat{\theta}(\mathbf{1})$. P is the empirical distribution corresponding to all the data objects present, i.e., $b = \mathbf{1}$. While the exact form of P depends on the data, it should satisfy the following property that holds for any θ:

$$\mathcal{L}(\theta, P) = \mathcal{L}(\theta, \mathbf{1}), \tag{2}$$

where we have slightly abused the notations of $\mathcal{L}$ defined in Section 2.1 and Section 2.2. On the other hand, Q should be defined as the direction towards the empirical distribution corresponding to the data indexed by $b = \mathbf{1}_{-i}$. A plausible choice of Q is to directly define it as the empirical distribution of data for $b = \mathbf{1}_{-i}$, which leads to the following property that holds for any θ:

$$\mathcal{L}(\theta, Q) = \mathcal{L}(\theta, \mathbf{1}_{-i}). \tag{3}$$

As a result of Eqs. (2) and (3), we will also have $\hat{\theta}(P) = \hat{\theta}(\mathbf{1}), \hat{\theta}(Q) = \hat{\theta}(\mathbf{1}_{-i})$.

Sanity Check on M-Estimators. When instantiating P and Q for M-estimators with $\mathcal{L}(\theta, P) = \mathbb{E}_{z\sim P}\left[\ell(\theta, z)\right]$, it can be shown that defining P and Q as uniform distributions over $\{z_j\}_{j=1}^n$ and $\{z_j\}_{j=1}^n \setminus \{z_i\}$, respectively, satisfies Eqs. (2) and (3). In this case, P matches the specification in Lemma 2.4 while Q corresponds to a distribution where $\Pr(z_i) = 0$ and $\Pr(z_j) = \frac{1}{n-1}, j \neq i$. For this specification, we have the following result.

Lemma 2.5 (IF for M-Estimators with Downweighting.)**.** *With the specification of P, Q, and L above, we have*

$$\textit{IF}(\hat{\theta}(P); Q) = -\textit{IF}(\hat{\theta}(P); \delta_{z_i}).$$

The difference of a negative sign compared to the specification in Lemma 2.4 arises because $(1-\varepsilon)P + \varepsilon\delta_{z_i}$ upweights the loss of z_i, whereas $(1-\varepsilon)P + \varepsilon Q$ with the current specification downweights the loss of z_i. Aside from this minor difference, our specification of P, Q, and $\mathcal{L}$ leads to the same result as the standard derivation of IF for M-estimators.

Finite Difference Approximation. Next, we address the challenge of solving the limit in Eq. (1) in general cases. We propose to approximate the limit with a finite difference with $\varepsilon = 1$, which results in the following approximation[2]:

$$\lim_{\varepsilon\to 0} \frac{\nabla_\theta \mathcal{L}(\hat{\theta}(P), (1-\varepsilon)P + \varepsilon Q) - \nabla_\theta \mathcal{L}(\hat{\theta}(P), P)}{-\varepsilon} \approx \nabla_\theta \mathcal{L}(\hat{\theta}(P), P) - \nabla_\theta \mathcal{L}(\hat{\theta}(P), Q).$$

The Proposed VIF. Combining the properties of P and Q in Eqs. (2) and (3), and replacing the limit in Eq. (1) with the finite difference approximation, we propose the following method to approximate the LOO effect for any non-decomposable loss.

[2] We note that, due to the nuance in Lemma 2.5 caused by the choice of upweighting v.s. downweighting, we have added a negative sign to ε in the denominator to make the result consistent with the standard IF formulation when applied to M-estimators.

Definition 2.6 (Versatile Influence Function)**.** *The* Versatile Influence Function *(VIF) that measures the influence of a data object i on the parameters $\hat{\theta}(\mathbf{1})$ learned from a non-decomposable loss $\mathcal{L}$ is defined as following*

$$VIF(\hat{\theta}(\mathbf{1}); i) := -\left[\nabla_\theta^2 \mathcal{L}(\hat{\theta}(\mathbf{1}), \mathbf{1})\right]^{-1} \nabla_\theta \left(\mathcal{L}(\hat{\theta}(\mathbf{1}), \mathbf{1}) - \mathcal{L}(\hat{\theta}(\mathbf{1}), \mathbf{1}_{-i})\right). \quad (4)$$

Computational Advantages. The VIF defined in Eq. (4) enjoys a few computational advantages. Firstly, VIF depends on the parameters only at $\hat{\theta}(\mathbf{1})$ and does not require $\hat{\theta}(\mathbf{1}_{-i})$. Therefore, it does not require model retraining. Secondly, compared to Eq. (1), VIF only involves gradients and the Hessian of the loss, which can be easily obtained through auto-differentiation provided in modern machine learning libraries. Thirdly, VIF can be applied to more complicated models and accelerated with similar heuristic tricks employed by existing IF-based data attribution methods for decomposable losses (Koh & Liang, 2017; Grosse et al., 2023). Finally, note that VIF calculates the difference $\mathcal{L}(\hat{\theta}(\mathbf{1}), \mathbf{1}) - \mathcal{L}(\hat{\theta}(\mathbf{1}), \mathbf{1}_{-i})$ before taking the gradient with respect to the parameters. In some special cases (see, e.g., the M-estimator case in Appendix D), this difference significantly simplifies.

Attributing a Target Function. In practice, we are often interested in attributing certain model outputs or performance. Similar to Koh & Liang (2017), given a target function of interest, $f(z, \theta)$, that depends on both some data z and the model parameter θ, then the influence of a training data point i on this target function can be obtained through the chain rule:

$$\nabla_\theta f(z, \hat{\theta}(\mathbf{1}))^\top \mathrm{VIF}(\hat{\theta}(\mathbf{1}); i). \quad (5)$$

3 Experiments

3.1 Experimental setup

We conduct experiments on three examples listed in Section 2.1: Cox Regression, Node Embedding, and Listwise Learning-to-Rank. In this section, we present the performance and runtime of VIF compared to brute-force LOO retraining. We also provide two case studies to demonstrate how the influence estimated by VIF can help interpret the behavior of the trained model in Appendix F.

Datasets and Models. We evaluate our approach on multiple datasets across different scenarios. For Cox Regression, we use the METABRIC and SUPPORT datasets (Katzman et al., 2018). For both of the datasets, we train a Cox model using the negative log partial likelihood following Eq. (9). For Node Embedding, we use Zachary's Karate network (Zachary, 1977) and train a DeepWalk model (Perozzi et al., 2014). Specifically, we train a two-layer model with one embedding layer and one linear layer optimized via contrastive loss following Eq. (10), where the loss is defined as the negative log softmax. For Listwise Learning-to-Rank, we use the Delicious (Tsoumakas et al., 2008) and Mediamill (Snoek et al., 2006) datasets. We train a linear model using the loss defined in Eq. (11). Please refer to Appendix E for more detailed experiment settings.

Target Functions. We apply VIF to estimate the change of a target function, $f(z, \theta)$, before and after a specific data object is excluded from the model training process. Below are our choice of target functions for difference scenarios.

For Cox Regression, we study how the relative risk function, $f(x_{test}, \theta) = \exp(\theta^\top x_{test})$, of a test object, x_{test}, would change if one training object were removed. For Node Embedding, we study how the contrastive loss, $f((u, v, N), \theta) = l(\theta; (u, v, N))$, of an arbitrary pair of test nodes, (u, v), would change if a node $w \in N$ were removed from the graph. For Listwise Learning-to-Rank, we study how the ListMLE loss of a test query, $f((x_{test}, y_{test}^{[k]}), \theta) = -\sum_{j=1}^{k}\left(f(x_{test}; \theta)_j - \log \sum_{l \in [n] \setminus \{y_{test}^{(1)}, \ldots, y_{test}^{(j-1)}\}} \exp(f(x_{test}; \theta)_l)\right)$, would change if one item $l \in [n]$ were removed from the training process.

3.2 Performance

We utilize the Pearson correlation coefficient to quantitatively evaluate how closely the influence estimated by VIF aligns with the results obtained by brute-force LOO retraining. Furthermore, as a reference upper limit of performance, we evaluate the correlation between two brute-force LOO

Table 1: The Pearson correlation coefficients of VIF and brute-force LOO retraining under different experimental settings. Specifically, "Brute-Force" refers to the results of two times of brute-force LOO retraining using different random seeds, which serves as a reference upper limit of performance.

Scenario	Dataset	Method	Pearson Correlation
Cox Regression	METABRIC	VIF	0.997
		Brute-Force	0.997
	SUPPORT	VIF	0.943
		Brute-Force	0.955
Node Embedding	Karate	VIF	0.407
		Brute-Force	0.419
Listwise Learning-to-Rank	Mediamill	VIF	0.823
		Brute-Force	0.999
	Delicious	VIF	0.906
		Brute-Force	0.999

retraining with different random seeds. As noted in Remark 2.3, some examples like contrastive losses are not deterministic, which could impact the observed correlations.

Table 1 presents the Pearson correlation coefficients comparing VIF with brute-force LOO retraining using different random seeds. The performance of VIF matches the brute-force LOO in all experimental settings. Except for the Node Embedding scenario, the Pearson correlation coefficients are close to 1, indicating a strong resemblance between the VIF estimates and the retraining results. In the Node Embedding scenario, the correlations are moderately high for both methods due to the inherent randomness in the random walk procedure for constructing the triplet set in the DeepWalk algorithm. Nevertheless, VIF achieves a correlation that is close to the upper limit by brute-force LOO retraining.

3.3 Runtime

We report the runtime of VIF and brute-force LOO retraining in Tabel 2. The computational advantage of VIF is significant, reducing the runtime by factors up to $1097\times$. This advantage becomes more pronounced as the dataset size increases. The improvement ratio on the Karate dataset is moderate due to the overhead from the random walk process and potential optimizations in the implementation. All runtime measurements were recorded using an Intel(R) Xeon(R) Gold 6338 CPU.

Table 2: Runtime comparison of VIF and brute-force LOO retraining on different experimental settings.

Senario	Dataset	Brute-Force	VIF	Improvement Ratio
Cox Regression	METABRIC	24 min	2.43 sec	593×
	SUPPORT	225 min	12.3 sec	1097×
Network Embedding	Karate	204 min	109 min	1.87×
Listwise Learning-to-Rank	Mediamill	52 min	2.6 min	20×
	Delicious	660 min	2.8 min	236×

4 Conclusion

In this work, we introduced the Versatile Influence Function (VIF), a novel method that extends IF-based data attribution to models trained with non-decomposable losses. The key idea behind VIF is a finite difference approximation of the general IF formulation in the statistics literature, which eliminates the need for case-specific derivations and can be efficiently computed with the auto-differentiation tools provided in modern machine learning libraries. Our theoretical analysis demonstrates that VIF accurately recovers classical influence functions in the case of M-estimators and provides strong approximations for more complex settings such as Cox regression. Empirical evaluations across various tasks show that VIF closely approximates the influence obtained by brute-force leave-one-out retraining while being orders-of-magnitude faster. By broadening the scope of IF-based data attribution to non-decomposable losses, VIF opens new avenues for data-centric applications in machine learning, empowering practitioners to explore data attribution in more complex and diverse domains.

References

Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. Relatif: Identifying explanatory training samples via relative influence. In *International Conference on Artificial Intelligence and Statistics*, pp. 1899–1909. PMLR, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In Hal Daumé Iii and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607. PMLR, 2020.

D R Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B, Statistical methodology*, 34(2):187–202, January 1972. ISSN 1369-7412,1467-9868. doi: 10.1111/j.2517-6161.1972.tb00899.x.

Junwei Deng and Jiaqi Ma. Computational Copyright: Towards A Royalty Model for Music Generative AI. *arXiv [cs.AI]*, December 2023.

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2242–2251. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/ghorbani19c.html`.

Roger 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 Samuel R Bowman. Studying large language model generalization with influence functions. *arXiv [cs.LG]*, August 2023.

Han Guo, Nazneen Fatema Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. Fastif: Scalable influence functions for efficient model interpretation and debugging. *arXiv preprint arXiv:2012.15781*, 2020.

Zayd Hammoudeh and Daniel Lowd. Training data influence analysis and estimation: A survey. *Machine Learning*, 113(5):2351–2403, 2024.

Frank R Hampel. The influence curve and its role in robust estimation. *Journal of the American Statistical Association*, 69(346):383–393, June 1974. ISSN 0162-1459,1537-274X. doi: 10.1080/01621459.1974.10482962.

Peter J Huber and Elvezio M Ronchetti. *Robust Statistics*. Wiley Series in Probability and Statistics. Wiley-Blackwell, Hoboken, NJ, 2 edition, January 2009. ISBN 9780470129906,9780470434697. doi: 10.1002/9780470434697.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. Datamodels: Predicting predictions from training data. *arXiv preprint arXiv:2202.00622*, 2022.

Luca Invernizzi, Paolo Milani Comparetti, Stefano Benvenuti, Christopher Kruegel, Marco Cova, and Giovanni Vigna. Evilseed: A guided approach to finding malicious web pages. In *2012 IEEE symposium on Security and Privacy*, pp. 428–442. IEEE, 2012.

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 (eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 1167–1176. PMLR, 16–18 Apr 2019. URL `https://proceedings.mlr.press/v89/jia19a.html`.

Jared L Katzman, Uri Shaham, Alexander Cloninger, Jonathan Bates, Tingting Jiang, and Yuval Kluger. Deepsurv: personalized treatment recommender system using a cox proportional hazards deep neural network. *BMC medical research methodology*, 18:1–12, 2018.

Pang Wei Koh and Percy Liang. Understanding Black-box Predictions via Influence Functions. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1885–1894. PMLR, 2017.

Yongchan Kwon and James Zou. Beta shapley: a unified and noise-reduced data valuation framework for machine learning. *arXiv preprint arXiv:2110.14049*, 2021.

Yongchan Kwon, Eric Wu, Kevin Wu, and James Zou. Datainf: Efficiently estimating data influence in lora-tuned llms and diffusion models. *arXiv preprint arXiv:2310.00902*, 2023.

T Mikolov, I Sutskever, K Chen, G S Corrado, and J Dean. Distributed representations of words and phrases and their compositionality. *Neural information processing systems*, 2013.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. Trak: Attributing model behavior at scale. *arXiv preprint arXiv:2303.14186*, 2023.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. DeepWalk: online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710, New York, NY, USA, August 2014. ACM. ISBN 9781450329569. doi: 10.1145/2623330.2623732.

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

N Reid and H Crepeau. Influence functions for proportional hazards regression. *Biometrika*, 72(1):1, April 1985. ISSN 0006-3444,1464-3510. doi: 10.2307/2336329.

Xin Rong. word2vec Parameter Learning Explained. *arXiv [cs.CL]*, November 2014.

Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. Scaling up influence functions. In *Proc. Conf. AAAI Artif. Intell.*, volume 36, pp. 8179–8186. Association for the Advancement of Artificial Intelligence (AAAI), June 2022. doi: 10.1609/aaai.v36i8.20791.

Cees GM Snoek, Marcel Worring, Jan C Van Gemert, Jan-Mark Geusebroek, and Arnold WM Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of the 14th ACM international conference on Multimedia*, pp. 421–430, 2006.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *Proc. ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, volume 21, pp. 53–59, 2008.

A W van der Vaart. *Asymptotic Statistics*. Cambridge University Press, Cambridge, England, June 2012. ISBN 9780511802256,9780521496032. doi: 10.1017/cbo9780511802256.

Jiachen T Wang and Ruoxi Jia. Data banzhaf: A robust data valuation framework for machine learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 6388–6421. PMLR, 2023.

Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning - ICML '08*, pp. 1192–1199, New York, New York, USA, 2008. ACM Press. ISBN 9781605582054. doi: 10.1145/1390156.1390306.

Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. Representer point selection for explaining deep neural networks. *Advances in neural information processing systems*, 31, 2018.

Wayne W Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 33(4):452–473, 1977.

Tong Zhao, Julian McAuley, Mengya Li, and Irwin King. Improving recommendation accuracy using networks of substitutable and complementary products. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 3649–3655. IEEE, 2017.

A Preliminaries: IF-Based Data Attribution for Decomposable Loss

We begin by reviewing the formulation of IF-based data attribution in prior literature (Koh & Liang, 2017; Schioppa et al., 2022; Grosse et al., 2023). IF-based data attribution aims to approximate the effect of leave-one-out (LOO) retraining—the change of model parameters after removing one training data point and retraining the model—which could be used to quantify the influence of this training data point.

Formally, suppose we have the following loss function,

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} \ell_i(\theta), \tag{6}$$

where θ is the model parameters; the total number of training data points is n; and each $\ell_i(\cdot), i = 1, \ldots, n$, corresponds to the loss function of one training data point. The IF-based data attribution is derived by first inserting a binary weight w_i in front of each $\ell_i(\cdot)$ to represent the inclusion or removal of the individual data points, transforming $\mathcal{L}(\theta)$ to a weighted loss[3]

$$\mathcal{L}(\theta, w) = \sum_{i=1}^{n} w_i \ell_i(\theta). \tag{7}$$

Note that $w = \mathbf{1}$ corresponds to the original loss in Eq. (6); while removing the i-th data point is to set $w_i = 0$ or, equivalently, $w = \mathbf{1}_{-i}$, where $\mathbf{1}_{-i}$ is a vector of all one except for the i-th element being zero. Denote the learned parameters as $\hat{\theta}(w) := \arg\min_\theta \mathcal{L}(\theta, w)$[4]. The LOO effect for data point i is then characterized by $\hat{\theta}(\mathbf{1}_{-i}) - \hat{\theta}(\mathbf{1})$.

However, evaluating $\hat{\theta}(\mathbf{1}_{-i})$ is computationally expensive as it requires model retraining. Koh & Liang (2017) proposed to approximate the LOO effect by relaxing the binary weights in w to the continuous interval $[0, 1]$ and measuring the influence of the training data point i on the learned parameters as

$$\left.\frac{\partial \hat{\theta}(w)}{\partial w_i}\right|_{w=\mathbf{1}} = -\left[\nabla_\theta^2 \mathcal{L}(\hat{\theta}(\mathbf{1}), \mathbf{1})\right]^{-1} \nabla_\theta \ell_i(\hat{\theta}(\mathbf{1})), \tag{8}$$

which can be evaluated using only $\hat{\theta}(\mathbf{1})$, hence eliminating the need for expensive model retraining.

However, by construction, this approach critically relies on the introduction of the loss weights w_i's, and is thus limited to loss functions that are *decomposable* with respect to the individual training data points, taking the form of Eq. (6).

B Non-Decomposable Loss

In practice, there are many common loss functions that are *not* decomposable. Below we list a few examples.

Example 1: Cox's Partial Likelihood. The Cox regression model (Cox, 1972) is one of the most widely used models in survival analysis, designed to predict the time until specific events occur (e.g., patient death or a customer's next purchase). A unique challenge in survival analysis is handling censored observations, where the exact event time is unknown because the event has either not occurred by the end of the study or the individual is lost to follow-up. These censored data points contain partial information about the event timing and must be properly accounted for to avoid biased estimates and inaccurate conclusions in the analysis. Given a set of data points $\{(X_i, Y_i, \Delta_i)\}_{i=1}^n$, where X_i represents the features for the i-th data point, Y_i denotes the observed time (either the event time or the censoring time), and Δ_i is the binary event indicator ($\Delta_i = 1$ if the event has occurred

[3] In the rest of the paper, we will overload the notations of $\mathcal{L}$ and $\hat{\theta}$ several times for writing convenience. But their definitions should be clear from context.

[4] While this definition is technically valid only under specific assumptions about the loss function (e.g., strict convexity), in practice, methods developed based on these assumptions (together with some heuristics tricks) are often applicable to more complicated models such as neural networks (Koh & Liang, 2017).

and $\Delta_i = 0$ if the observation is censored), the Cox regression model is defined through specifying the hazard function

$$h(t \mid x) = h_0(t) \exp(\theta^\top x),$$

where θ is the model parameters to be estimated while $h_0(t)$ is called the baseline hazard function without parametric parameters and $\exp(\theta^\top x)$ is called the relative risk function. The parameters θ can be learned through minimizing the following *negative log partial likelihood*

$$\mathcal{L}(\theta) = -\sum_{i:\Delta_i=1} \left(\theta^\top X_i - \log \sum_{j \in R_i} \exp(\theta^\top X_j) \right), \tag{9}$$

where $R_i := \{j : Y_j > Y_i\}$ is called the *at-risk set*.

In Eq. (9), each data point may appear in multiple loss terms if it belongs to the at-risk sets of other data points. Consequently, we can no longer characterize the effect of removing a training data point by simply introducing the loss weight.

Example 2: Contrastive Loss. Contrastive losses are commonly seen in unsupervised representation learning across various modalities, such as word embeddings (Mikolov et al., 2013), image representations (Chen et al., 2020), or node embeddings (Perozzi et al., 2014). Generally, contrastive losses rely on a set of triplets, $D = \{(u_i, v_i, N_i)\}_{i=1}^m$, where u_i is an anchor data point, v_i is a positive data point that is relevant to u_i, while N_i is a set of negative data points that are irrelevant to u_i. The contrastive loss is then the summation over such triplets:

$$\mathcal{L}(\theta) = \sum_{i=1}^{m} \ell(\theta; (u_i, v_i, N_i)), \tag{10}$$

where the loss $l(\cdot)$ could take many forms. In word2vec (Mikolov et al., 2013) for word embeddings or DeepWalk (Perozzi et al., 2014) for node embeddings, θ corresponds to the embedding parameters for each word or node, while the loss $l(\cdot)$ could be defined by heirarchical softmax or negative sampling (see Rong (2014) for more details).

Similar to Eq. (9), each single term of the contrastive loss in Eq. (10) involves multiple data points. Moreover, taking node embeddings as an example, the set of triplets D is constructed by running random walks on the network. Removing one data point, which is a node in this context, could also affect the proximity of other pairs of nodes and hence the construction of D.

Example 3: Listwise Learning-to-Rank. Learning-to-rank is a core technology underlying information retrieval applications such as search and recommendation. In this context, listwise learning-to-rank methods aim to optimize the ordering of a set of documents or items based on their relevance to a given query. One prominent example of such methods is ListMLE (Xia et al., 2008). Suppose we have annotated results for m queries over n items as a dataset $\{(x_i, (y_i^{(1)}, y_i^{(2)}, \ldots, y_i^{(k)})\}_{i=1}^m$, where x_i is the query feature, $y_i^{(1)}, y_i^{(2)}, \ldots, y_i^{(k)} \in [n] := \{1, \ldots, n\}$ indicate the top k items for query i. Then the ListMLE loss function is defined as following

$$\mathcal{L}(\theta) = -\sum_{i=1}^{m} \sum_{j=1}^{k} \left(f(x_i; \theta)_j - \log \sum_{l \in [n] \setminus \{y_i^{(1)}, \ldots, y_i^{(j-1)}\}} \exp(f(x_i; \theta)_l) \right), \tag{11}$$

where $f(\cdot; \theta)$ is a model parameterized by θ that takes the query feature as input and outputs n logits for predicting the relevance of the n items.

In this example, Eq. (11) is decomposable with respect to the queries while not decomposable with respect to the items. The influence of items could also be of interest in information retrieval applications. For example, in a search engine, we may want to detect webpages with malicious search engine optimization (Invernizzi et al., 2012); in product co-purchasing recommendation (Zhao et al., 2017), both the queries and items are products.

C Omitted Derivations

C.1 Derivation of Eq. (1)

Consider an ε perturbation towards another distribution Q, i.e., $(1-\varepsilon)P+\varepsilon Q$. Note that $\hat{\theta}((1-\varepsilon)P+\varepsilon Q)$ solves $\nabla_\theta \mathcal{L}(\theta,(1-\varepsilon)P+\varepsilon Q)=0$. We take derivative with respect to ε and evaluate at $\varepsilon=0$ on both side, which leads to

$$\nabla^2_\theta \mathcal{L}(\hat{\theta}(P),P)\lim_{\varepsilon\to 0}\frac{\hat{\theta}((1-\varepsilon)P+\varepsilon Q)-\hat{\theta}(P)}{\varepsilon}+\lim_{\varepsilon\to 0}\frac{\nabla_\theta \mathcal{L}(\hat{\theta}(P),(1-\varepsilon)P+\varepsilon Q)-\nabla_\theta \mathcal{L}(\hat{\theta}(P),P)}{\varepsilon}=0.$$

Given the strict convexity, the Hessian is invertible at the global optimal. By plugging the definition of IF, we have

$$IF(\hat{\theta}(P);Q)=-\left[\nabla^2_\theta \mathcal{L}(\hat{\theta}(P),P)\right]^{-1}\lim_{\varepsilon\to 0}\frac{\nabla_\theta \mathcal{L}(\hat{\theta}(P),(1-\varepsilon)P+\varepsilon Q)-\nabla_\theta \mathcal{L}(\hat{\theta}(P),P)}{\varepsilon}.$$

C.2 Proof of Lemma 2.4

Proof. Under M-estimation, the objective function becomes the empirical loss, i.e., $\mathcal{L}(\theta,P)=\mathbb{E}_{z\sim P}[\ell(\theta;z)]$, where $P=\sum_{i=1}^n \delta_{z_i}/n$ is the empirical distribution over the dataset. Similarly, the gradient and Hessian become

$$\nabla_\theta \mathcal{L}(\hat{\theta}(P),P)=\mathbb{E}_{z\sim P}[\nabla_\theta \ell(\hat{\theta}(P);z)]=0$$

and

$$\nabla^2_\theta \mathcal{L}(\hat{\theta}(P),P)=\mathbb{E}_{z\sim P}[\nabla^2_\theta \ell(\hat{\theta}(P);z)]=\sum_{i=1}^n \nabla^2_\theta \ell(\hat{\theta}(P);z_i)/n,$$

respectively. The infinitesimal change on the gradient towards the distribution $Q=\delta_{z_i}$ equals to

$$\begin{aligned}
&\lim_{\varepsilon\to 0}\frac{\nabla_\theta \mathcal{L}(\hat{\theta}(P),(1-\varepsilon)P+\varepsilon Q)-\nabla_\theta \mathcal{L}(\hat{\theta}(P),P)}{\varepsilon}\\
=&\lim_{\varepsilon\to 0}\frac{\mathbb{E}_{z\sim(1-\varepsilon)P+\varepsilon Q}[\nabla_\theta \mathcal{L}(\hat{\theta}(P),z)]-0}{\varepsilon}\\
=&\lim_{\varepsilon\to 0}\frac{(1-\varepsilon)\mathbb{E}_{z\sim P}[\nabla_\theta \mathcal{L}(\hat{\theta}(P),z)]+\varepsilon\mathbb{E}_{z\sim Q}[\nabla_\theta \mathcal{L}(\hat{\theta}(P),z)]}{\varepsilon}\\
=&\lim_{\varepsilon\to 0}\frac{(1-\varepsilon)\cdot 0+\varepsilon\mathbb{E}_{z\sim Q}[\nabla_\theta \mathcal{L}(\hat{\theta}(P),z)]}{\varepsilon}\\
=&\mathbb{E}_{z\sim Q}[\nabla_\theta \mathcal{L}(\hat{\theta}(P),z)]=\nabla_\theta \mathcal{L}(\hat{\theta}(P),z_i).
\end{aligned}$$

Plugging the above equations into Eq. (1), it becomes the IF defined in Eq. (8). □

C.3 Analytic Expression of IF and VIF for Cox Regression

Recall that the objective function for Cox regression is negative log-partial likelihood:

$$\begin{aligned}
\mathcal{L}_n(\theta)&=-\sum_{i=1}^n \Delta_i\left(\theta^\top X_i-\log\left(\sum_{j\in R_i}\exp\left(\theta^\top X_j\right)\right)\right)\\
&=-\sum_{i=1}^n \Delta_i\left(\theta^\top X_i-\log\left(\sum_{j=1}^n I(Y_j\geq Y_i)\exp\left(\theta^\top X_j\right)\right)\right).
\end{aligned}$$

Define

$$\begin{aligned}
S_n^{(0)}(t;\theta)&=\frac{1}{n}\sum_{i=1}^n I\left(Y_i\geq t\right)\exp\left(\theta^\top X_i\right),\\
S_n^{(1)}(t;\theta)&=\frac{1}{n}\sum_{i=1}^n I\left(Y_i\geq t\right)\exp\left(\theta^\top X_i\right)X_i,
\end{aligned}$$

and

$$S_n^{(2)}(t;\theta) = \frac{1}{n}\sum_{i=1}^{n} I\left(Y_i \geq t\right)\exp\left(\theta^\top X_i\right) X_i X_i^\top.$$

Note that the maximum partial likelihood estimator $\hat{\theta}$ solves the following score equation:

$$\nabla_\theta \mathcal{L}_n(\hat{\theta}) = -\sum_{i=1}^{n} \Delta_i \left(X_i - \frac{S_n^{(1)}(Y_i;\hat{\theta})}{S_n^{(0)}(Y_i;\hat{\theta})} \right) = 0.$$

We define $\nabla_\theta \ell_n(\hat{\theta}; Z_i)$ as shorthand for $-\Delta_i \left(X_i - \frac{S_n^{(1)}(Y_i;\hat{\theta})}{S_n^{(0)}(Y_i;\hat{\theta})} \right)$. It is worth noting that $\ell_n(\hat{\theta}; Z_i)$ does not only depend on Z_i but also other data points in its at-risk set. The Hessian of the objective function at $\hat{\theta}$ is given by

$$\nabla^2_\theta \mathcal{L}(\hat{\theta}) = \sum_{i=1}^{n} \Delta_i \left(\frac{S_n^{(2)}(Y_i;\hat{\theta})}{S_n^{(0)}(Y_i;\hat{\theta})} - \frac{S_n^{(1)}(Y_i;\hat{\theta})}{S_n^{(0)}(Y_i;\hat{\theta})} \cdot \frac{S_n^{(1)}(Y_i;\hat{\theta})}{S_n^{(0)}(Y_i;\hat{\theta})}^\top \right).$$

For simplicity, assume there is no tied event. Reid & Crepeau (1985) derived the influence function for the observation $Z_i = (X_i, Y_i, \Delta_i)$ by evaluating the limit in (1) with $Q = \delta_{Z_i}$:

$$\mathrm{IF}(i) = -\,[\nabla^2_\theta \mathcal{L}(\hat{\theta})]^{-1} \nabla_\theta \ell_n(\hat{\theta}; Z_i) - [\nabla^2_\theta \mathcal{L}(\hat{\theta})]^{-1} C_i(\hat{\theta}),$$

where

$$C_i(\hat{\theta}) = \frac{1}{n}\sum_{j=1}^{n} I(Y_j \leq Y_i)\Delta_j \exp(\hat{\theta}^\top X_i) \cdot \frac{X_i \cdot S_n^{(0)}(Y_j;\hat{\theta}) - S_n^{(1)}(Y_j;\hat{\theta})}{\left(S_n^{(0)}(Y_j;\hat{\theta})\right)^2}.$$

The first term is analogous to the standard influence function for M-estimators and the second term captures the influence of the i-th observation in the at-risk set. Denote $\epsilon_{ij} = \frac{\exp(\hat{\theta}^\top X_i)/n}{S_n^{(0)}(Y_j;\hat{\theta})}$ for j such that $Y_j \leq Y_i$. The IF can be rewritten as

$$\mathrm{IF}(i) = -\,[\nabla^2_\theta \mathcal{L}(\hat{\theta})]^{-1} \left(\nabla_\theta \ell_n(\hat{\theta}; Z_i) + \sum_{j: Y_j \leq Y_i} \Delta_j \left(X_i - \frac{S_n^{(1)}(Y_j;\hat{\theta})}{S_n^{(0)}(Y_j;\hat{\theta})} \right) \cdot \boldsymbol{\epsilon_{ij}} \right).$$

On the other hand, under the Cox regression, the proposed VIF becomes

$$\mathrm{VIF}(i) := -\left[\nabla^2_\theta \mathcal{L}(\hat{\theta})\right]^{-1} \left(\nabla_\theta \mathcal{L}_n(\hat{\theta}) - \nabla_\theta \mathcal{L}_{n-1}^{(-i)}(\hat{\theta}) \right),$$

where $\nabla_\theta \mathcal{L}_{n-1}^{(-i)}(\hat{\theta})$ is the gradient of the negative log-partial likelihood after excluding the i-th data point at $\hat{\theta}$. Given no tied events, we can rewrite it as

$$\nabla_\theta \mathcal{L}_{n-1}^{(-i)}(\hat{\theta}) = -\sum_{j: Y_j < Y_i} \Delta_j \left(X_j - \frac{S_n^{(1)}(Y_j;\hat{\theta}) - \exp(\hat{\theta}^\top X_i) X_i / n}{S_n^{(0)}(Y_j;\hat{\theta}) - \exp(\hat{\theta}^\top X_i)/n} \right) - \sum_{j: Y_j > Y_i} \Delta_j \left(X_j - \frac{S_n^{(1)}(Y_j;\hat{\theta})}{S_n^{(0)}(Y_j;\hat{\theta})} \right).$$

Then it follows that

$$\begin{aligned}
\mathrm{VIF}(i) &= -\,[\nabla^2_\theta \mathcal{L}(\hat{\theta})]^{-1} \left(\nabla_\theta \mathcal{L}_n(\hat{\theta}) - \nabla_\theta \mathcal{L}_{n-1}^{(-i)}(\hat{\theta}) \right) \\
&= -\,[\nabla^2_\theta \mathcal{L}(\hat{\theta})]^{-1} \left(\nabla_\theta \ell_n(\hat{\theta}; Z_i) + \sum_{j: Y_j < Y_i} \Delta_j \left(\frac{S_n^{(1)}(Y_j;\hat{\theta})}{S_n^{(0)}(Y_j;\hat{\theta})} - \frac{S_n^{(1)}(Y_j;\hat{\theta}) - \exp(\hat{\theta}^\top X_i) X_i / n}{S_n^{(0)}(Y_j;\hat{\theta}) - \exp(\hat{\theta}^\top X_i)/n} \right) \right) \\
&= -\,[\nabla^2_\theta \mathcal{L}(\hat{\theta})]^{-1} \left(\nabla_\theta \ell_n(\hat{\theta}; Z_i) + \sum_{j: Y_j < Y_i} \Delta_j \left(X_i - \frac{S_n^{(1)}(Y_j;\hat{\theta})}{S_n^{(0)}(Y_j;\hat{\theta})} \right) \cdot \frac{\boldsymbol{\epsilon_{ij}}}{\mathbf{1} - \boldsymbol{\epsilon_{ij}}} \right).
\end{aligned}$$

D Approximation Quality in Special Cases

To provide insights into how accurately the proposed VIF approximates Eq. (1), we examine the following special cases. Although there is no formal guarantee of approximation quality in general, our analysis in these cases suggests that VIF may perform well in practice in many situations.

M-Estimators. Under M-estimation, we have $\nabla_\theta \mathcal{L}(\hat{\theta}(\mathbf{1}), \mathbf{1}) = \sum_{i=1}^n \nabla_\theta \ell(\hat{\theta}(P), z_i)$ and $\nabla_\theta \mathcal{L}(\hat{\theta}(\mathbf{1}), \mathbf{1}_{-i}) = \sum_{j=1, j\neq i}^n \nabla_\theta \ell(\hat{\theta}(P), z_j)$. Then the VIF in Eq. (4) becomes

$$\mathrm{VIF}(\hat{\theta}(\mathbf{1}); i) := -\left[\nabla_\theta^2 \mathcal{L}(\hat{\theta}(\mathbf{1}), \mathbf{1})\right]^{-1} \nabla_\theta \ell(\hat{\theta}(P), z_i),$$

which indicates that the VIF is identical to the IF obtained by Eq. (1) without approximation under M-estimation.

Cox Regression. The IF obtained by applying Eq. (1) to the Cox regression model exists in the statistics literature (Reid & Crepeau, 1985). We also derive and compare analytic expressions of IF and VIF for the Cox regression model below. The exact derivations and notation definitions can be found in Appendix C.3.

$$\mathrm{IF}(i) = -\,[\nabla_\theta^2 \mathcal{L}(\hat{\theta})]^{-1} \left(\nabla_\theta \ell_n(\hat{\theta}; Z_i) + \sum_{j: Y_j \leq Y_i} \Delta_j \left(X_i - \frac{S_n^{(1)}(Y_j; \hat{\theta})}{S_n^{(0)}(Y_j; \hat{\theta})} \right) \cdot \boldsymbol{\epsilon_{ij}} \right).$$

$$\mathrm{VIF}(i) = -\,[\nabla_\theta^2 \mathcal{L}(\hat{\theta})]^{-1} \left(\nabla_\theta \ell_n(\hat{\theta}; Z_i) + \sum_{j: Y_j < Y_i} \Delta_j \left(X_i - \frac{S_n^{(1)}(Y_j; \hat{\theta})}{S_n^{(0)}(Y_j; \hat{\theta})} \right) \cdot \frac{\boldsymbol{\epsilon_{ij}}}{\mathbf{1} - \boldsymbol{\epsilon_{ij}}} \right).$$

As can be seen from the comparison, the analytic expressions of IF and VIF differ only in minor terms that may be empirically negligible.

E Detailed experiment setup

Datasets. For Cox regression, both METABRIC and SUPPORT datasets are split into training, validation, and test sets with a 6:2:2 ratio. The training objects and test objects are defined as the full training and test sets. For node embedding, the test objects are all valid pairs of nodes, i.e., $34 \times 34 = 1156$ objects, while the training objects are the 34 individual nodes. In the case of listwise learning-to-rank, we sample 500 test samples from the pre-defined test set as the test objects. For the Mediamill dataset, we use the full label set as the training objects, while for the Delicious dataset, we sample 100 labels from the full label set (which contains 983 labels in total). The brute-force leave-one-out retraining follows the same training hyperparameters as the full model, with one training object removed at a time.

Scenario	Dataset	Training obj	Test obj
Cox regression	METABRIC	1217 samples	381 samples
	SUPPORT	5677 samples	1775 samples
Node embedding	Karate	34 nodes	1156 pairs of nodes
Listwise learning-to-rank	Mediamill	101 labels	500 samples
	Delicious	100 labels	500 samples

Table 3: Training objects and test objects in different experiment settings.

Models. For Cox regression, we train a CoxPH model with a linear function on the features for both the METABRIC and SUPPORT datasets. The model is optimized using the Adam optimizer with a learning rate of 0.01. We train the model for 200 epochs on the METABRIC dataset and 100 epochs on the SUPPORT dataset. For node embedding, we sample 1,000 walks per node, each with a length of 6, and set the window size to 3. The dimension of the node embedding is set to 2. For listwise learning-to-rank, the model is optimized using the Adam optimizer with a learning rate of 0.001, weight decay of 5e-4, and a batch size of 128 for 100 epochs on both the Mediamill and Delicious datasets. We also use TruncatedSVD to reduce the feature dimension to 8.

F Case Studies

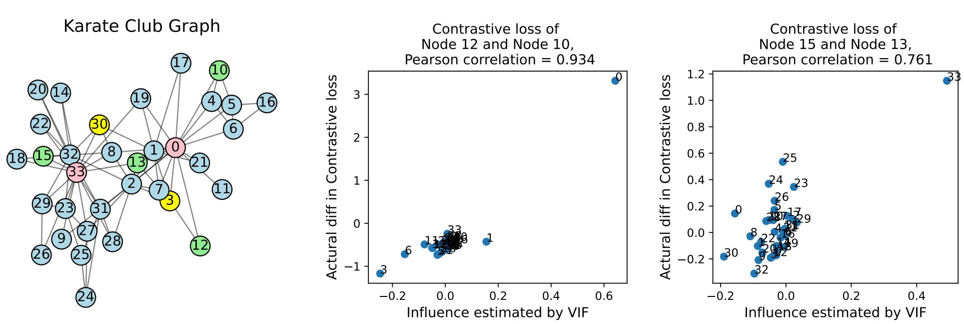
We present two case studies to show how the influence estimated by VIF can help interpret the behavior of the trained model.

Case study 1: Cox Regression. In Table 4, we show the top-5 most influential training samples, as estimated by VIF, for the relative risk function of two randomly selected test samples. We observe that removing two types of data samples in training will significantly increase the relative risk function of a test sample: (1) training samples that share similar features with the test sample and have long survival times (e.g., training sample ranks 1, 3, 4, 5 for test sample 0 and ranks 5 for test sample 1) and (2) training samples that differ in features from the test sample and have short survival times (e.g., training sample ranks 2 for test sample 0 and ranks 1, 2, 3, 4 for test sample 1). These findings align with domain knowledge.

Table 4: The top-5 influential training samples to 2 test samples in the METABRIC dataset. "Features Similarity" is the cosine similarity between the feature of the influential training sample and the test sample. "Observed Time" and "Event Occurred" are the Y and Δ of the influential training sample as defined in Eq. (9).

Influence Rank	Test Sample 0			Test Sample 1		
	Feature Similarity	Observed Time	Event Occurred	Feature similarity	Observed time	Event occurred
1	0.84	322.83	False	-0.49	16.57	True
2	-0.34	9.13	True	-0.22	30.97	True
3	0.77	258.17	True	-0.39	15.07	True
4	0.23	131.27	False	-0.65	4.43	True
5	0.81	183.43	False	0.72	307.63	False

Case study 2: Node Embedding. in Figure 1b and 1c, we show the influence of all nodes to the contrastive loss of 2 pairs of test nodes. The spring layout of the Karate dataset is provided in Figure 1a. We observe that the most influential nodes (on the top right in Figure 1b and 1c) are the hub nodes that lie on the shortest path of the pair of test nodes. For example, the shortest path from node 12 to node 10 passes through node 0, while the shortest path from node 15 to node 13 passes through node 33. Conversely, the nodes with the most negative influence (on the bottom left in Figure 1b and 1c) are those that likely "distract" the random walk away from the test node pairs. For instance, node 3 distracts the walk from node 12 to node 10, and node 30 distracts the walk from node 15 to node 13.



(a) Karate Club Graph (b) Node 12 and Node 10 (c) Node 15 and Node 13

Figure 1: VIF is applied to Zachary's Karate network to estimate the influence of each node on the contrastive loss of a pair of test nodes. Figure 1a is a spring layout of the Karate network. Figure 1b and Figure 1c illustrate the alignment between the influence estimated by VIF (x-axis) and the brute-force LOO retrained loss difference (y-axis).

G Related Work

Data Attribution. Data attribution methods can be roughly categorized into two groups: retraining-based and gradient-based methods (Hammoudeh & Lowd, 2024). Retraining-based methods (Ghorbani & Zou, 2019; Jia et al., 2019; Kwon & Zou, 2021; Wang & Jia, 2023; Ilyas et al., 2022) typically estimate the influence of individual training data points by repeatedly retraining models on subsets of the training dataset. While these methods have been shown effective, they are not scalable for large-scale models and applications. In contrast, gradient-based methods (Koh & Liang, 2017; Guo et al., 2020; Barshan et al., 2020; Schioppa et al., 2022; Kwon et al., 2023; Yeh et al., 2018; Pruthi et al., 2020; Park et al., 2023) estimate the training data influence based on the gradient and higher-order gradient information of the original model, avoiding expensive model retraining. In particular, many gradient-based methods (Koh & Liang, 2017; Guo et al., 2020; Barshan et al., 2020; Schioppa et al., 2022; Kwon et al., 2023; Pruthi et al., 2020; Park et al., 2023) can be viewed as variants of IF-based data attribution methods. Therefore, extending IF-based data attribution methods to a wider domains could lead to a significant impact on data attribution.

Influence Function in Statistics. The IF is a well-established concept in statistics, dating back at least to Hampel (1974), though it is typically applied for purposes other than data attribution. Originally introduced in the context of robust statistics, it was used to assess the robustness of statistical estimators (Huber & Ronchetti, 2009) and later adapted as a tool for developing asymptotic theories (van der Vaart, 2012). Notably, IFs have been derived for a wide range of estimators beyond M-estimators, including L-estimators, R-estimators, and others (Huber & Ronchetti, 2009; van der Vaart, 2012). Closely related to an example of this study, Reid & Crepeau (1985) developed the IF for the Cox regression model. However, the literature in statistics often approaches the derivation of IFs through precise definitions specific to particular estimators, requiring case-specific derivations. In contrast, this work proposes an approximation for the general IF formulation in statistics, which can be straightforwardly applied to a broad family of modern machine learning loss functions for the purpose of data attribution. While this approach involves some degree of approximation, it benefits from being more versatile and computationally efficient, leveraging auto-differentiation capabilities provided by modern machine learning libraries.