On the Support Vector Effect in DNNs: Rethinking Data Selection and Attribution

Syed Hasan Amin Mahmood
hasanamin@purdue.edu
Purdue University
West Lafayette, IN, USA

Ming Yin
mingyin@purdue.edu
Purdue University
West Lafayette, IN, USA

Rajiv Khanna
rajivak@purdue.edu
Purdue University
West Lafayette, IN, USA

Abstract

In Deep Neural Networks (DNNs), manipulating gradients is central to various algorithms, including data subset selection and instance attribution. For better tractability, practitioners often resort to using only the gradients of the last layer as a heuristic, instead of the full gradient across all model parameters, which we show is detrimental due to the *Support Vector Effect (SVE)*. We introduce SVE, a max-margin-like behavior in the last layer(s) of DNNs and employ it to thoroughly scrutinize prevalent data selection and attribution methods relying on last layer gradients. Our investigation exposes limitations in these techniques and not only provides explanations for previously observed pitfalls, like lack of diversity and temporal performance degradation, but also offers fresh insights, including the vulnerability of existing methods to basic poisoning attacks and the potential for competitive performance using much simpler alternatives. Based on insights from SVE, we craft new methods `RandE` and `PAE` for data subset selection and instance attribution, respectively, which often outperform the purported state-of-the-art at a fraction of the cost, emphasizing the practical advantages of more efficient and less complex approaches.

CCS Concepts

• **Computing methodologies** → **Regularization**; **Neural networks**; *Instance-based learning*; *Support vector machines.*

Keywords

Support Vector Effect, Deep Neural Network, Data Subset Selection, Instance Attribution, Interpretability, Last Layer Approximation

ACM Reference Format:
Syed Hasan Amin Mahmood, Ming Yin, and Rajiv Khanna. 2025. On the Support Vector Effect in DNNs: Rethinking Data Selection and Attribution. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3690624.3709295

1 Introduction

Deep Neural Networks (DNNs) have revolutionized predictive modeling across diverse domains, driven largely by increasing model scale and dataset sizes. Yet, their remarkable success contrasts with a limited understanding of their inner workings. As models and datasets continue to grow, a critical question emerges: *how do individual data points shape these vast models?* Answering this is key to understanding model behavior and developing more efficient, sustainable training methods.

The exploration of the importance of individual data points for high-performing DNNs has given rise to at least two prominent use cases: *instance attribution* and *data subset selection*. Instance attribution utilizes the notion of importance to select training instances that the model capitalized on to make a given prediction for insights into model behavior, while data subset selection uses it to curate a smaller, impactful data subset for accelerated model training without substantial performance degradation.

Newer methods continue to emerge for both instance attribution [12, 18, 32, 38, 42] and data subset selection [11, 13, 14, 23, 30, 31, 41], often to mitigate perplexing issues linked with prior approaches. In instance attribution literature, some methods are observed to lead to only a few training instances repeatedly being considered most important for multiple test instances [10, 38, 43]. Attempts to overcome such limitations have generally been method-specific. For example, Sui et al. [38] reformulates work on Representer Point Selection [42] to get rid of a regularization term, which helps ameliorate the lack of diversity. Additionally, Basu et al. [2] find that Influence Functions [18] for instance attribution are fragile when non-convex models like DNNs are involved, and a follow-up investigation by Schioppa et al. [34] reveals that influence estimates may be reasonable initially but steadily degrade as DNN training progresses. Similar issues have also intriguingly emerged in data subset selection recently. Specifically, a few instances are observed to dominate the selection process, and performance gains using meticulously crafted coreset methods—instead of simple random baseline—are found to diminish when large subset sizes are involved [41]. These pervasive concerns, shared across multiple methods and diverse applications, signal a larger underlying problem.

In the present work, we find a surprising commonality across these methods that explicate the concerns. Although not intrinsic to the design of all methods, the computational costs involved with these vast networks encourage a predominant *focus on the last layer, which we find is a bad idea.* Our investigations reveal an intriguing phenomenon, the *Support Vector Effect (SVE)*: the last layer of DNNs behaves like a Support Vector Machine, forming a maximum-margin linear decision boundary dependent on a sparse set of support vectors. This exposes a critical flaw in using last-layer approximations to assess data point importance, as they are disproportionately influenced by a small set of support vectors, which in turn results in data selection and attribution methods yielding counterproductive outcomes.

KDD '25, Toronto, ON, Canada

ACM ISBN 979-8-4007-1245-6/25/08
https://doi.org/10.1145/3690624.3709295

We rigorously establish the foundations of the SVE, leveraging the implicit regularization effect of gradient descent adjacent algorithms. We further illustrate the SVE in practice, identify key limitations that are direct consequences of the SVE, and offer amelioration strategies. Specifically, we make the following contributions:

(1) We introduce the Support Vector Effect, illuminating how the last layer(s) of DNNs exhibit SVM-like behavior. We not only provide a comprehensive framework for understanding the phenomenon but also offer both theoretical insights and empirical evidence into the substantial impact of the SVE.
(2) We utilize SVE as a unifying framework to explain perplexing shortcomings in varied methods across two prominent use cases of instance attribution and data subset selection. We illustrate that concentrating on the last layer(s) in DNNs to overcome computational limitations is often suboptimal.
(3) We utilize SVE to uncover fresh limitations and insights, including group-level influence and vulnerability to basic adversarial attacks, as well as how embarrassingly simple and efficient alternatives will often perform competitively.
(4) We utilize SVE to design new methods for data selection and attribution, which achieve state-of-the-art performance with significantly improved efficiency. We also provide guidelines for further optimization and refinement of these methods, aiming to stimulate further method development through a different, and much more efficient, lens.

2 Background and Related Work

Implicit biases. Popular optimization algorithms like Gradient Descent and Adam [16] have been found to implicitly regularize the training process, steering parameters to certain kinds of global minima without any explicit regularization [27, 40]. Some of these works [7, 26, 37] support observation of max-margin behavior in DNNs, providing a foundation for our work on the Support Vector Effect. These "implicit biases" have helped elucidate why highly non-convex DNNs may generalize well. In this work, we empirically verify a similar implicit behavior in the last layer of DNNs, and discuss its impact on relative importance of individual data points for instance attribution and data subset selection.

Instance attribution. A growing body of research [12, 15, 18, 32, 35, 42] has explored methods for understanding and interpreting DNNs, with work on gradient-based approaches to understanding the influence of individual training instances on the prediction process being of particular relevance to us. Traditional approaches like leave-one-out methodologies are computationally expensive, leading to the emergence of relatively efficient attribution score methods. These include Influence Functions [18] that employ first- and second-order derivatives to estimate influence, Representer Point Selection [42] that utilizes a representer theorem in the last layer of DNNs, and TracIn [32] that tracks gradients during optimization. Some recent works [10, 29] also introduce methods to select important instances that help explain entire learning algorithms or model classes. However, these are less relevant to our study, as we focus on understanding specific model instantiations and methods employing last layer gradients.

For illustration through a concrete instantiation, Influence Functions [18] estimate $A_{IF}(\mathbf{x}, \mathbf{x}')$, the influence on a test point $\mathbf{x}'$ attributed to a training point $\mathbf{x}$, through the sensitivity of the model's loss ℓ with respect to its parameters Θ, expressed as:

$$A_{IF}(\mathbf{x}, \mathbf{x}') = -\nabla_\Theta \ell(\mathbf{x}, \Theta)^\top H_\Theta^{-1} \nabla_\Theta \ell(\mathbf{x}', \Theta), \tag{1}$$

where H_Θ is the Hessian of the loss over the training data. For large DNNs, calculating (1) is intractable, which is why last layer gradients are often used as a proxy to the full model gradients. We use SVE to uncover, explain and improve limitations in such sensitivity-based methods.

Data subset selection. To address efficiency and sustainability concerns in large-scale machine learning, data-efficient learning methods have emerged with the objective of judiciously selecting a small subset of training data that yields performance comparable to the full dataset. In the context of efficient model training, data subset selection methods [13, 14, 23, 30, 41] delve into the iterative creation of weighted data subsets, known as coresets, by aligning subset and full gradients.

For illustration through a concrete instantiation, CRAIG [23] employs a greedy algorithm to identify a coreset S^* that minimizes the subset size while ensuring that its gradients approximate the gradients of the entire dataset $\mathcal{D}$ within an ϵ tolerance:

$$\begin{aligned} S^* = &\arg\min_{S \subseteq \mathcal{D}, \gamma_j \geq 0 \, \forall \mathbf{x}_j \in S} |S|, \quad \text{subject to:} \\ &\max_\Theta \| \sum_{\mathbf{x}_i \in \mathcal{D}} \nabla_\Theta \ell(\mathbf{x}_i, \Theta) - \sum_{\mathbf{x}_j \in S} \gamma_j \nabla_\Theta \ell(\mathbf{x}_j, \Theta) \| \leq \epsilon, \end{aligned} \tag{2}$$

where γ_j are weights assigned to the selected subset data points. The model is then updated using the weighted gradient sum of just the identified coreset. Again, because of the size of DNNs, solving (2) is hard and last layer gradients are often used as a proxy to the full model gradients. We use SVE to uncover, explain and improve limitations in such methods.

Pervasive use of last layer. Given the large number of parameters in DNNs and resulting high dimensional gradients, it is common practice to resort to using only (the gradients of) the last layer as a heuristic approximation. This is quite popular for instance attribution and data subset selection, which is the focus of our work. However, its adoption also extends to multiple other use cases, including DNN compression [6], analyzing DNN limitations [36], active learning [1] and robust training [17]. Therefore, the implications of our work elucidating how and when last layer approximations could be detrimental are expected to extend beyond discussion on the two aforementioned use cases, and opens route for significant further research.

3 Support Vector Effect

One can view neural network training as a combination of two serial steps: *representation learning*, where the network learns to extract meaningful features represented as the output of the penultimate layer, and *learning linear classifiers* using these last layer representations. In a 'good' DNN with (near) zero training error, the learned representations are expected to be linearly separable. Crucially, we observe that the last layer classifier in such good DNNs does not learn just any linear boundary but a very specific

one: the maximum margin solution. In other words, this last layer classifier exhibits behavior akin to learning the maximum margin linear decision boundary—a characteristic reminiscent of Support Vector Machines (SVMs). We coin this SVM-like training phenomenon in the last layer(s) of DNNs as the Support Vector Effect. The core of the SVE lies in the sparsity of gradient contributions during training: as training progresses, only a few data points significantly influence the updates of the last layer's parameters.

We now formalize the SVE and provide theoretical insights into its manifestation in DNNs. Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$, be the training data. We consider a DNN parameterized by $\Theta = \{\Theta_L, \Theta_{\bar{L}}\}$, where $\Theta_L \in \mathbb{R}^p$ denotes the parameters of the last layer, and $\Theta_{\bar{L}}$ denotes the parameters of all preceding layers. We conceive the output of the penultimate layer as the feature mapping $\phi(\mathbf{x}; \Theta_{\bar{L}}) \in \mathbb{R}^p$, and the output of the DNN is given by $\Phi(\mathbf{x}; \Theta) = \Theta_L^\top \phi(\mathbf{x}; \Theta_{\bar{L}})$. For simplicity, we focus on binary classification using logistic loss, defined as $\ell(\mathbf{x}_i, y_i; \Theta_L) = \log\left(1 + \exp\left(-y_i \Theta_L^\top \phi(\mathbf{x}_i)\right)\right)$, but the extension to multiclass classification and the softmax loss is straightforward.

Our main result shows that the gradient updates with respect to the last layer parameters Θ_L become dominated by the support vectors as training progresses, with the contributions from non-support vectors becoming increasingly minuscule. For easier exposition, we make a couple assumptions. Firstly, we assume that the *last layer feature representations $\phi(\cdot)$ are fixed for all training data points*. This can also be seen as freezing the weights $\Theta_{\bar{L}}$ and training only the last layer parameters Θ_L. Secondly, we assume that the *data is separable in the last layer*, i.e., there exists a classifier that achieves perfect training accuracy, which is often true for real-world DNNs. Both of these assumptions can be removed at the expense of clarity. We also empirically verify that the central conclusion holds in practical DNNs.

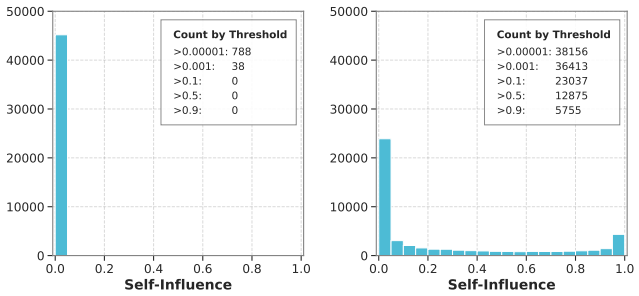
Theorem 3.1 (Support Vector Effect). *Under the above assumptions, the difference between the full gradient sum and the sum over support vectors decreases as:*

$$\left\| \sum_{i=1}^{n} \nabla_{\Theta_L} \ell(\mathbf{x}_i, y_i; \Theta_L(t)) - \sum_{i \in \mathcal{S}} \nabla_{\Theta_L} \ell(\mathbf{x}_i, y_i; \Theta_L(t)) \right\| = O\left(\frac{1}{t^{\gamma+\delta}}\right),$$

where $\gamma = \min_{i \in \mathcal{S}} \gamma_i$ is the maximum margin, $\delta = \min_{i \notin \mathcal{S}} (\gamma_i - \gamma) > 0$, t is the iteration number of gradient descent, $\mathcal{S}$ is the set of support vectors, and the size of the set $|\mathcal{S}| \leq \dim(\phi(\mathbf{x}))$ is independent of the number of data points n.

The proof of the Support Vector Effect (Theorem 3.1) is deferred to Appendix A.2. Intuitively, it implies that the gradient updates for the last layer parameters Θ_L become increasingly dominated by the contributions of a small set of support vectors, as contributions from non-support vectors decay rapidly. For interpretation purpose, these *support vectors can be considered the most difficult (to learn) data points*, which we clarify further in Section 4.1.

Explaining existing limitations. The SVE implies sparsity of gradient updates in the last layer and resulting dominance of a small set of support vectors. This can explain observed issue of *lack of diversity* in instance attribution methods [10, 38, 43]. Moreover, since the SVE intensifies over time, leading to a growing imbalance in the importance assigned to individual data points, it can explain the *temporal performance degradation* highlighted by Schioppa et al. [34]. Together, these findings underscore the *risks of last layer approximations* for assessing data importance, as such methods tend to overemphasize a narrow subset of data points while failing to capture the broader dataset's influence.



(a) Last Layer Estimate (b) Ground Truth

Figure 1: Comparison of self-influence estimates for CIFAR-100 using (a) last layer and (b) input layer representations, highlighting sparsity in last layer estimates, in strong contrast to a more diverse spread in ground truth estimates.

3.1 Illustrating the SVE

To demonstrate the SVE's relevance beyond the theoretical scope of Theorem 3.1, we conduct empirical evaluation under multiclass setting and common configurations of training time, optimizer, weight initialization etc. While this could lead to falling short of near-zero training error or even perfect training accuracy, the exploration highlights the SVE's significance across diverse, real-world scenarios. Our experiments confirm the existence of SVE under these settings: *(1) with penultimate layer embeddings as features, few data points have non-zero influence* (Figure 1), and *(2) number of data points with non-zero gradients decrease rapidly* (Figure 2). Additionally, they help attain a more nuanced understanding of the manifestation of SVE under popular DNN setups.

3.1.1 Validation of SVE in Practice. For direct illustration purposes, we explicitly compute importance of individual data points using last layer representations. Following Feldman and Zhang [5], we use a sample-level notion of influence, defining a sample's importance by its impact on a specific prediction. We can define the influence [3] of a data point $\mathbf{x}$ on a data point $\mathbf{x}'$, $\mathcal{I}_{\text{true}}(\mathbf{x}, \mathbf{x}')$, as the change in the loss ℓ at $\mathbf{x}'$ between having learned the model parameters Θ without and with $\mathbf{x}$ in the training data $\mathcal{D}$, i.e.,

$$\mathcal{I}_{\text{true}}(\mathbf{x}, \mathbf{x}') = \ell(\mathbf{x}' \mid \Theta : \mathbf{x} \notin \mathcal{D}) - \ell(\mathbf{x}' \mid \Theta : \mathbf{x} \in \mathcal{D}) \tag{3}$$

For the sake of illustration, we concentrate on self-influence, where $\mathbf{x} = \mathbf{x}'$. While naive computation of Equation (3) for all $\mathbf{x}$ is prohibitively expensive, we employ sub-sampling estimators [5] that make the calculation relatively efficient. This involves training multiple models on randomly chosen subsets of the data. We keep track of which subsets each instance belongs to, allowing us to estimate its influence without retraining the full model for every instance.

Similar to Feldman and Zhang [5], we train ResNet-50 [9] on CIFAR-100 images. However, we compute influence scores using last layer embeddings[1], and use the available influence scores computed using input layer embeddings (i.e., raw images) with full model training as ground truth [5]. Under the SVE, most data points are expected to show zero influence when focusing on the last layer, which is indeed the case in Figure 1a where fewer than 0.08% data points are found to have non-zero self-influence ($\hat{I}(\mathbf{x}', \mathbf{x}') > 0.001$). We further compare these values with ground truth estimates in Figure 1b, to verify that the behavior actually arises from SVE rather than some intrinsic data or model properties. Here, we find that over 70% of data points exhibit non-zero self-influence, a nearly 1000× increase, with more than 10% having substantial self-influence exceeding ($\hat{I}(\mathbf{x}', \mathbf{x}') > 0.9$).

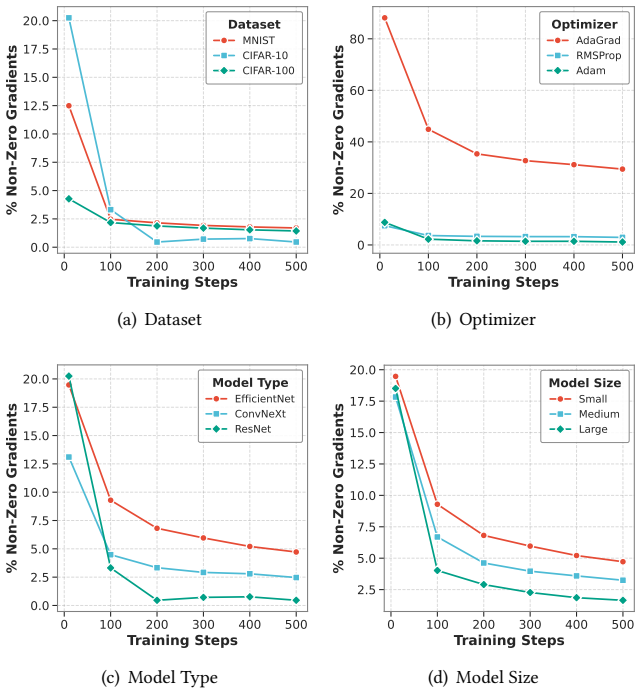
Remark. While the focus of this work is on the last layer, the behavior may not intuitively be very different in second last layer where similar linear separability may often be attained. A preliminary study using second and third last layer embeddings under aforementioned setup suggests that while moving from the last to earlier layers may alleviate the dominance of support vectors, the SVE still influences gradient sparsity, albeit less pronouncedly.

3.1.2 Extent of Impact of SVE. We also evaluate how significantly SVE manifests across a range of practical settings, going beyond the specific configuration under which its theoretical foundations are laid out. While we rigorously uncover the SVE and present Theorem 3.1 under a particular choice of loss and optimizer, recent literature on implicit bias of multiple optimization algorithms towards maximum margin solutions [21, 22, 25, 40] help extend our insights to more relaxed settings. Nonetheless, different optimizers like AdaGrad [4] may demonstrate different implicit biases [33], so the SVE may not always naturally follow.

We train DNNs for 500 steps under varied choices of dataset, optimizer, model type and model size, while keeping track of percentage of non-zero gradients in the last layer—which the SVE suggests should drop rapidly over time. We use CIFAR-10, ResNet-50 and Adam as default choices for dataset, model and optimizer, respectively. While the precise numbers end up being different, the key expected trend of *rapid drop* to a *small percentage of non-zero (i.e., greater than* 0.001*) gradients* persists almost throughout, as seen in Figure 2. Quite paradoxically, *better models are generally impacted worse*, i.e., models that attain lower error quickly experience faster decay in non-zero gradients, which could be explained by the criteria of SVE manifestation, like near-zero training loss, being attained faster with better models.

Impact of dataset (Figure 2a). We experiment with three image datasets of increasing complexity: MNIST, CIFAR-10 and CIFAR-100. While we stick with ResNet-50 for CIFAR datasets, we use a multi-layer perceptron (MLP) for training on MNIST since it is a relatively simple dataset. Moreover, the ResNet-50 models are pretrained and initialized with ImageNet weights, while the MLP is trained from scratch. Percentage of non-zero gradients drops to under 2.5 after just 200 training steps across all datasets, illustrating significant impact of SVE throughout.

[1]Models $h(\cdot)$ are trained on last layer representations $\phi(\mathbf{x})$, and performance is evaluated by checking if $h(\phi(\mathbf{x})) = y$.



(a) Dataset (b) Optimizer

(c) Model Type (d) Model Size

Figure 2: Percentage of non-zero gradients in the last layer decreases rapidly over time across multiple datasets, optimizers, model types and model sizes, suggesting the Support Vector Effect is observed in varied, real-world settings.

Impact of optimizer (Figure 2b). We experiment with three popular optimizers: AdaGrad [4], RMSProp [24] and Adam [16]. In line with SVE, most last layer gradients drop to zero for RMSProp and Adam, with just 3% and 1% non-zero gradients remaining after 500 training steps for RMSProp and Adam, respectively. However, AdaGrad demonstrates a significantly different trend, with over 25% gradients being non-zero even at the end of training. This illustrates that optimizer choice could influence SVE manifestation, as well as consequent negative impact on downstream applications like data selection and attribution. The trend may be explained by AdaGrad having different implicit biases [33] than others. Still, we speculate that such behavior is not unique to AdaGrad and is worthy of deeper investigation in the future.

Impact of model type (Figure 2c). We experiment with three prominent model architectures, from the popular ResNets [9] to the scaling-focused EfficientNets [39] to the ConvNeXts [20] that incorporate learnings from vision transformers [19]. Specifically, we use ResNet-50, EfficientNet-B0 and ConvNeXt-Base in our experiments. All models are evidently influenced by the SVE.

Impact of model size (Figure 2d). We experiment with three EfficientNet [39] models of increasing sizes: EfficientNet-B0 (small), EfficientNet-B2 (medium), and EfficientNet-B4 (large). Although the SVE is evident across all model sizes, larger models experience a more pronounced impact, with a sharper decline in the percentage of non-zero gradients.

4 Uncovering New Limitations Through SVE

Having established the foundations of the SVE, which helped explain some previously observed limitations of popular data selection and attribution methods, we now use SVE to unveil new limitations and insights. Our findings further highlight the risks of last layer approximations, emphasizing the need for data selection and attribution methods that account for broader dataset dynamics and diverse contributions from training points.

4.1 Group-level Influence and Exemplar Effect

A direct consequence of the SVE is that so-called 'instance' attribution methods exhibit neither local, instance-level behavior nor global, data-level behavior, but instead they identify class or *group-level exemplars*. We refer to this phenomenon as the *exemplar effect*: with last layer approximations, a few training instances (the support vectors) repeatedly receive the highest attribution scores for most test instances belonging to the same class. Since attribution score calculations rely on gradients [18], it is easy to see that these *group-level exemplars or support vectors are the most difficult (to learn) instances from each class*, as indicated by their high loss or large gradient norm. The exemplar effect has critical implications for the utility of instance attribution methods, revealing their tendency to act as class-level explainers rather than providing meaningful, instance-level insights.

To quantify this phenomenon, we define the set of exemplars as:

$$\mathcal{S}_{ex}^{r,k} = \{\mathbf{x}_i \mid \forall \mathbf{x}_i \in \mathcal{D}_{\text{train}},\ \text{count}\,(\mathbf{x}_i, \mathcal{D}_{\text{test}}, \mathcal{I}, k) \geq r\} \qquad (4)$$

where $\text{count}\,(\mathbf{x}_i, \mathcal{D}_{\text{test}}, \mathcal{I}, k)$ represents the number of times a training point $\mathbf{x}_i$ appears among the top-k highest attribution scores with any test instance $\mathbf{x}_j \in \mathcal{D}_{\text{test}}$ computed using $\mathcal{I}$. We are particularly interested in data points which repeat very frequently, say more than r times, which we can refer to as exemplars. In the absence of SVE, one would expect very few, if any, training instances to repeatedly influence large numbers of test points, particularly for small values of k. However, we find extensive repetition on experimenting with multiple datasets and DNNs, as shown in Table 1. Note that the first row depicts that just 3 out of 60,000 training points in MNIST are considered the most influential for all 1000 test instances per class. Additionally, the number of exemplars (i.e., $|\mathcal{S}_{\text{ex}}^{r,k}|$) are close to the number of classes when $k = 3$, serving as empirical evidence of class or group-level behavior.

While previous work has noted lack of diversity in attribution methods through anecdotal examples [38], our results demonstrate that this issue is systematic and pervasive *across entire test sets*. Moreover, we attribute this behavior to the SVE's emphasis on support vectors, even in settings where the data is not fully separable and training loss is not exactly zero.

Table 1: Illustration of the exemplar effect.

Dataset	k	r	# Exemplars
MNIST	1	1000	3
	1	900	7
	3	1000	11
	3	900	24
CIFAR-10	1	1000	0
	1	900	4
	3	1000	4
	3	900	10
CIFAR-100	1	100	78
	1	90	88
	3	100	220
	3	90	249

4.2 Vulnerability to Poisoning Attacks

Since the exemplars are often hard-to-learn data points, the exemplar effect makes the discussed attribution methods particularly bad choices for data with mislabeling. Yet, even more concerning is that this renders these methods vulnerable to data poisoning attacks involving intentional mislabeling or label flipping; a small proportion of flipped labels are enough to result in these malicious data points being considered the most influential. This can have unintended consequences such as a biased DNN being considered fair due to the most influential instances with flipped labels appearing unbiased, although the DNN does not actually rely on them.

As shown in Table 2, we verify this behavior through multiple experiments where we randomly flip a meager 1% of labels. With a small amount of poisoning, the model training should not be impacted much—and we indeed see little change in accuracy of models trained on poisoned instead of the original data. While the poisoned model continues to make very similar predictions, the most influential data points change dramatically. Since the poisoned instances do not naturally belong to their stated class, they are difficult to learn and end up becoming support vectors with disproportionately high influence. Moreover, due to the exemplar effect, the poisoned instances repeatedly show up as most influential training points for most test instances, which results in nonsensical outcomes. For example, a cat image with label poisoned to frog could end up being considered most influential for *correctly* predicting frogs. Table 2 confirms that a handful of training instances show up as most influential points for all test instances per class, and that an overwhelming majority of these data points is in fact poisoned. r is set to 1000 for MNIST and CIFAR-10, and 100 for CIFAR-100, aligning with the typical number of test instances per class for each dataset.

Table 2: Illustration of vulnerability of last layer gradient-based instance attribution methods to data poisoning.

Dataset	k	# Exemplars	# Poisoned	% Poisoned
MNIST	1	3	3	100.0
	3	12	12	100.0
	10	41	36	87.8
CIFAR-10	1	2	2	100.0
	3	11	11	100.0
	10	55	45	81.8
CIFAR-100	1	27	26	96.3
	3	188	178	94.7
	10	844	497	58.9

4.3 Coresets Are No Better than Random

To speed up model training for large data and model size, multiple coreset-based data subset selection methods have been proposed. These methods curate the selection of a fraction of the training data for faster training with minimal loss in predictive performance. They require representative selection of influential data points, which would naturally be impacted by the SVE when such influence is computed using last layer embeddings—which is the de facto choice for application of coreset methods to DNNs. Note that while the motivations of instance attribution and data subset selection may be related, the impact of SVE differs significantly because the influential instances are used differently. In instance attribution, just five or ten most influential instances may be under consideration, which happen to repeatedly be the same for different test instances due to SVE. On the other hand, in data subset selection, even though a small proportion of entire data is selected, this still typically translates to thousands of instances. The intriguing insight is that most of these instances may be as good as random because only the few support vectors may adequately be identified as being useful towards the model training. We formally discuss two critical implications: *(1) random selection performs at par with coreset-based methods for a sufficiently large subset selection size*, and *(2) random selection outperforms coreset-based methods when subset size is small.* Empirical validation is present in Figure 3 and Table 3.

Remember that coreset-based methods often greedily build the important data subset S by adding to it instance $\mathbf{x}_j$ with gradient contribution $\mathbf{g}_j$ if it helps approximate the (last layer) gradient $\mathbf{g}$ on entire training data. Specifically, we could have $\mathbf{g} = \sum_{\mathbf{x}_i \in D} \nabla_\Theta \ell(\mathbf{x}_i, \Theta)$ and $\mathbf{g}_j = w_j \nabla_\Theta \ell(\mathbf{x}_j, \Theta)$ [23]. A greedy selection may be practically seen as adding $\mathbf{x}_j^*$ to subset S from training data $\mathcal{D}$ such that:

$$\mathbf{x}_j^* = \arg \min_{j \in \mathcal{D} \setminus S} \|\mathbf{g} - \sum_{i \in S} \mathbf{g}_i - \mathbf{g}_j\| \tag{5}$$

Random performs at par with coreset-based methods for a sufficiently large subset selection size. We leverage SVE to explicitly link random and coreset-based selection. Techniques like CRAIG [23] rely on the last layer gradients and $\mathbf{g}_j = \nabla_{\Theta_L} \ell(\mathbf{x}_j)$. Through Theorem 3.1, we know that the gradients remain non-zero for largely the support vectors, i.e., $\mathbf{g}_j \to 0$ and $\mathbf{g}_{j'} \gg \mathbf{g}_j$ for $j' \in \mathcal{S}$ and $j \notin \mathcal{S}$.

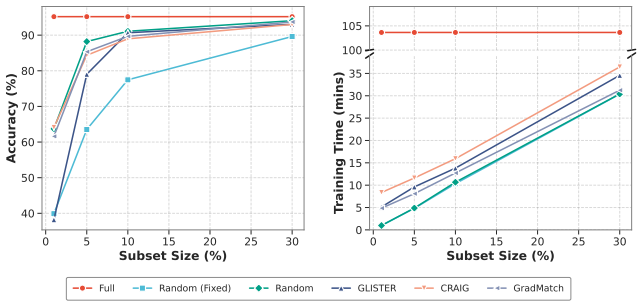


Figure 3: Accuracy (higher is better) and time (lower is better) comparison of various data subset selection strategies for training ResNet-18 on CIFAR-10.

Table 3: Accuracy comparison of simple random for data subset selection with the state-of-the-art CREST on CIFAR-100. Accuracy with full training (i.e., size=100) is 74.5%.

Size (%)	CREST	Random	% Change
1	23.2	33.3	+43.5
4	55.5	58.4	+5.2
7	64.1	65.3	+1.9
10	68.4	66.8	-2.3
15	71.6	71.0	-0.8
20	72.3	72.2	-0.1

This implies that, beyond the early training phase, significant gradients are limited to a small set of support vectors, making most other points indistinguishable for selection. Therefore, data subset selection via Equation (5) with $|S| \gg |\mathcal{S}|$ effectively reduces to random selection. This is in line with observations by Yang et al. [41], who noted that a few key points drive group performance and random selection rivals top methods when subset size exceeds 20%.

Random outperforms coreset-based methods when subset size is small. For smaller subset sizes, we hypothesize that the coreset-based methods tend to repeatedly select the same hard-to-learn data points, while random selection provides greater diversity. Over multiple steps, a random method samples a broader portion of the training set, leading to potentially outperforming even the best coreset-based methods, as seen in Table 3.

These findings collectively highlight that random selection can be highly competitive with popular coreset methods at a fraction of the computation cost. This is made even more explicit in Figure 3, where we compare uniformly random selection with three popular coreset-based methods: CRAIG [23], GLISTER [14] and GradMatch [13]. To ensure fairness, we use a random baseline that selects new subsets every e epochs, mirroring coreset methods. We also include comparison with fixed random subsets ("Random (Fixed)"), sometimes used unfairly as the baseline. Our findings hold across varied datasets (CIFAR-10, CIFAR-100 and Tiny ImageNet) and models (ResNet-18 and ResNet-101), confirming the generalizability of these insights (see Figure A2 in Appendix).

5 Countering SVE

Despite the limitations of popular attribution and subset selection methods, their continued use and associated positive impact might seem puzzling. However, it is important to be clear that the key takeaway from SVE is not that these methods are inherently flawed, but that their practical instantiation is suboptimal. Their apparent success can be attributed to incorporating various beneficial elements, albeit within complex frameworks that necessitate last layer approximations. We argue that by focusing on the simpler underlying principles driving the gains, we can achieve comparable or even superior results with significantly improved efficiency.

In this section, we introduce two novel methods informed by SVE: **P**rediction **A**s **E**xplanation (PAE) for instance attribution and **Rand**om sampling with **E**ntropy weighting (RandE) for data subset selection. We demonstrate that these methods can achieve state-of-the-art performance while being substantially more efficient

than existing approaches. Furthermore, we highlight the potential for further optimization and refinement of these methods, emphasizing the importance of continued exploration in this direction. We did not indulge in extensive exploration in current study since we wanted to test efficacy and efficiency of these SVE-informed methods rather than coming up with the best performing method.

5.1 Prediction As Explanation

Various previously discussed instance attribution approaches can be decomposed into a uniform framework characterized by a product of similarity between instances and the sensitivity of model outputs to the model loss [43]. The SVE unveils shortcomings associated with the sensitivity part, resulting from the sparsity of last layer gradients. However, the similarity part may still be used for effective attribution, either as part of some complex method or more efficiently to build independent alternatives.

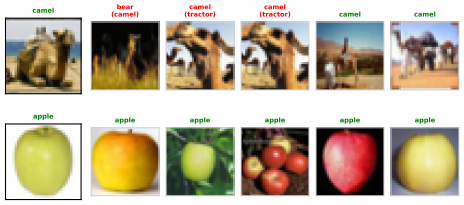
In fact, distance comparison or nearest neighbor search on last layer representations, often termed "penultimate layer embeddings," has been previously found to capture perceptual similarity well and be effective for model explanation [8, 28, 44]. However, these similarity comparisons for instance attribution have been neglected recently in support of the more complex methods considered to be superior [32]. To counter the SVE and allow for useful attribution, we revisit this idea and conduct experiments to validate the efficacy of instance attribution using nearest neighbor search with penultimate layer embeddings. It is much more efficient and versatile since it involves no gradient tracking or manipulation. While past work [32] may argue that similarity measures do not take into account the DNN model—which does hold when using input layer embeddings—the model is in fact central to penultimate embedding preparation and consequent similarity analyses. *We advocate for broader utilization of model-informed similarity measures at large*, finding a lot of past criticism to be too harsh.

In particular, we propose using prediction layer outputs themselves as embeddings in similarity computation for instance attribution. This method, which we term **P**rediction **A**s **E**xplanation, or simply PAE (Algorithm 1), offers several advantages. Firstly, since predictions directly reflect the model's decision-making process, PAE provides a faithful representation of how the model perceives the data, thus offering *high fidelity*. Secondly, since predictions are readily available and do not need access to internal model representations or gradients, PAE offers *high accessibility*. Thirdly, unlike sensitivity-based methods that can be unduly influenced by incorrect labels, PAE relies on the model's own understanding of the data, making it *robust to mislabeling*. Finally, the method eliminates the need for expensive gradient operations while also having low dimensionality—often much lower than even the penultimate one—resulting in significantly *improved computational efficiency*.

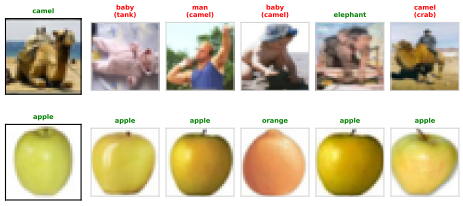
5.1.1 PAE vs Penultimate Layer Embeddings. As advocates for model-informed similarity, we consider penultimate embeddings to be a solid choice for instance attribution. We do find PAE slightly superior due to being more faithful, accessible and efficient by design, although these gains may not always be very significant or noticeable. Beyond these obvious advantages, however, there is a crucial use case where penultimate embeddings can be detrimental while PAE performs effectively: when data points are close to the decision boundary. Since penultimate embeddings disregard the last layer parameters and the linear boundaries they represent, if the test data point's embedding lies close to the decision boundary, its closest training data point could belong to another class, which is counterintuitive from an influence perspective. Using prediction vector entropy as a proxy for such closeness, we are able to uncover such failure cases of penultimate layer similarity comparison, as seen in Figure 4. Penultimate embeddings lead to problematic attribution on the camel image expected to be close to decision boundary with 0.40 entropy: a baby predicted as tank or a correctly predicted elephant may not explain why the image was correctly predicted as camel. On the other hand, PAE appears to provide reasonable explanations. Still, note that penultimate layer embeddings will often be effective, especially when it comes to providing perceptually similar attributions and when a data point is not close to the decision boundary, as seen in apple image in with 0.08 entropy.

Algorithm 1 Prediction As Explanation (PAE)

Input: Training data $\mathcal{D}$, test instance $\mathbf{x}'$, trained model $f(\cdot)$
Output: k most important training instances
for $\mathbf{x}_i$ in $\mathcal{D} \cup \{\mathbf{x}'\}$ **do**
  $\mathbf{p}_i \leftarrow f(\mathbf{x}_i)$
end for
for $\mathbf{x}_i \in \mathcal{D}$ **do**
  $s_i \leftarrow \text{Similarity}(\mathbf{p}_i, \mathbf{p}')$
end for
$\mathcal{R} \leftarrow$ Sort training instances by s_i in descending order
Return: Top-k instances $\mathcal{R}[:k]$

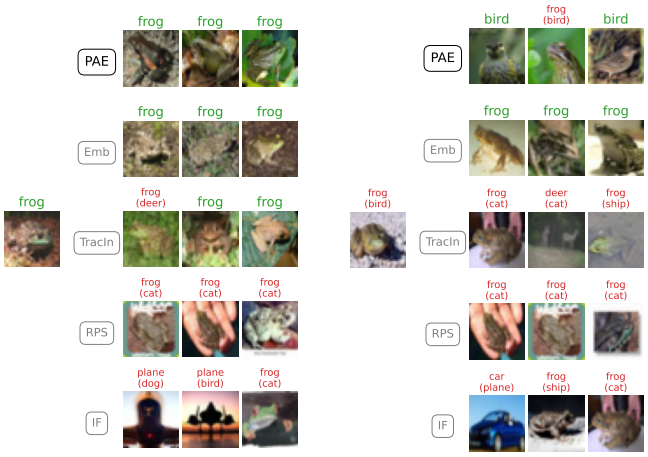


(a) PAE



(b) Penultimate Layer Embedding

Figure 4: Qualitative comparison of PAE and penultimate layer embeddings on a camel image close to decision boundary and an apple image far from decision boundary. The test image is shown on the left, followed by 5 most important training images selected by respective methods. Correctly predicted labels are shown in green, while incorrect labels are displayed in red along with the prediction (in brackets).



(a) Correctly Classified (b) Incorrectly classified

Figure 5: Qualitative comparison of various instance attribution methods on correctly and incorrectly classified data points from CIFAR-10. The test image is shown on the left, followed by the 3 most important training images selected by respective methods.

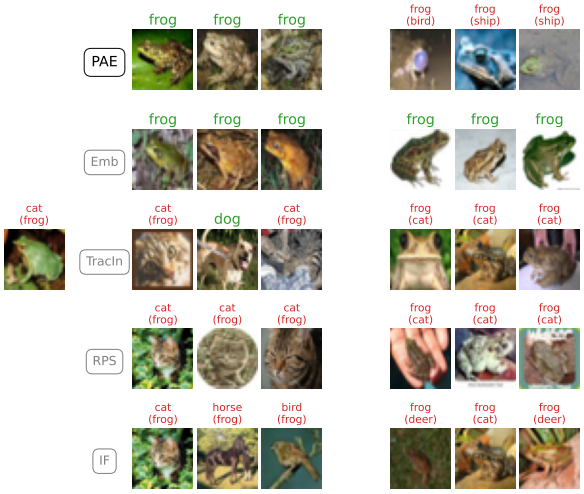


Figure 6: Qualitative comparison of various instance attribution methods on a mislabeled data point from CIFAR-10. The test image, of a frog labeled as cat yet correctly predicted to be frog, is shown on the left, followed by 3 most and least important training images selected by respective methods.

5.1.2 Qualitative Evaluation. Due to the inherent subjectivity in evaluating effectiveness of instance attribution, we primarily rely on visual inspection of top training instances for the purpose, closely resembling prior literature [32]. We find PAE, and simple embedding-based methods in general, to consistently outperform complex methods across multiple datasets. Figures 5 and 6 provide such qualitative comparison on select CIFAR-10 images, where PAE is found to perform quite well in comparison to four baseline methods: Penultimate Layer Embeddings (Emb), TracIn [32], Representer Point Selection (RPS) [42] and Influence Functions (IF) [18]. As seen in Figure 5b, PAE appears particularly insightful in explaining possible reasons behind incorrect prediction. Moreover, we also uncover a case of naturally mislabeled instance in CIFAR-10, and use it to evaluate robustness of the methods to mislabeling. As seen in Figure 6, and in line with our expectations, similarity-based methods significantly excel here. We also compare the *least* important images here, and find PAE to arguably be more informative than penultimate embeddings by providing a more diverse selection instead of just perceptually similar images with correct predictions. The examples highlight PAE's ability to provide meaningful and interpretable explanations, especially when naturally or maliciously mislabeled data is present.

5.1.3 Quantitative Evaluation. For more objective assessment, we applied the Identical Class and Identical Subclass tests from [9] on the various attribution methods, as shown in Figure 7. The Identical Class Test ensures that the most similar instances for a test instance belong to the same class, avoiding explanations that could undermine user trust in model predictions. Additionally, the more involved Identical Subclass Test adds the requirement that similar instances belong to the same 'latent subclass', which is unknown during model training. For this test, we create CIFAR-10B, a variant of CIFAR-10 with binary classes (vehicle or not), where known subclass labels aid evaluation. While there are understandable concerns about how well these tests capture true effectiveness of instance attribution methods, they at least provide some meaningful assessment of interpretability and reliability. Our findings reveal that similarity-based methods, including our proposed PAE, significantly outperform Influence Functions while being competitive with other complex, sensitivity-based methods. These experiments underscore the effectiveness of PAE as an efficient and insightful instance attribution method, particularly in scenarios where computational constraints or limited access to model internals pose challenges.

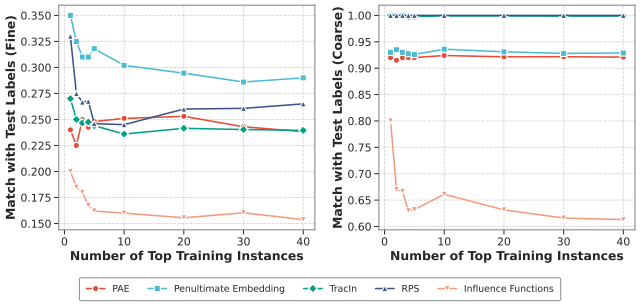


Figure 7: Comparative performance (higher is better) on Identical Subclass Test (left) and Identical Class Test (right) for different instance attribution methods on CIFAR-10B. Similarity-based methods demonstrate competitive, if not superior, performance. It is important to highlight that RPS involves explicit class-level filtering when selecting top training instances, trivially achieving a perfect score in the Identical Class Test.

Table 4: Relative error (lower is better) compared to training full model (Size=100). We also share wall-clock time improvement for our proposed RandE over state-of-the-art CREST. SGD shows accuracy of a standard mini-batch SGD pipeline at 10% training.

Dataset	Model	Size (%)	Random	SGD	CRAIG	GradMatch	GLISTER	CREST	RandE [Time]
CIFAR-10	ResNet-20	10	8.9±0.6	21.3±8.0	13.0±5.1	6.0±0.1	7.0±0.1	5.7±0.2	**5.2**±0.3 [↓ 6×]
CIFAR-100	ResNet-18	10	8.2±0.4	36.5±2.9	17.2±4.5	12.7±0.9	27.6±4.0	10.3±0.4	**5.6**±0.4 [↓ 3×]
TinyImageNet	ResNet-50	10	15.4±0.6	32.8±2.1	28.5±0.6	27.7±0.2	32.8±2.1	16.2±0.4	**10.9**±0.4 [↓ 10×]

5.2 Random Sampling with Entropy Weighting

While coreset-based methods excel at identifying influential training examples, they incur substantial computational overhead due to their iterative selection process. In contrast, random sampling offers exceptional efficiency but may miss out on crucial data points that contribute significantly to model training. We use SVE to bridge this gap, and introduce a new algorithm that is both efficient and effective. SVE highlights that, as DNNs converge, the influence of training instances concentrates on a small set of support vectors. Since these support vectors lie near the decision boundary, they exhibit high uncertainty or entropy in associated model predictions. By leveraging this insight, we hypothesize that prioritizing examples with high entropy during random sampling can lead to a more informative and effective subset selection, even without the complex computations of coreset methods.

We introduce **Rand**om sampling with **E**ntropy weighting, or simply RandE, a simple yet powerful data selection strategy that incorporates entropy as a guiding principle for random sampling. RandE can be conveniently incorporated into a data-efficient learning pipeline, as depicted in Algorithm 2. After every few training steps or epochs, we compute the entropy of the model's prediction vector for every training instance. We then randomly select a subset of the training data, where the probability of selecting each instance is proportional to its entropy. The model is trained on this selected subset for the next few epochs. This approach allows us to dynamically focus on the most uncertain or "difficult" examples during training, potentially leading to faster convergence and improved generalization. In fact, RandE can attain state-of-the-art performance on many datasets, models and subset sizes.

Algorithm 2 Model training with RandE

Input: Training data $\mathcal{D}$, model $f(\cdot)$, number of epochs T, subset ratio ρ, epoch interval e
Output: Trained model $f(\cdot)$
for $t = 1$ to T **step** e **do**
  for $\mathbf{x}_i$ in $\mathcal{D}$ **do**
    $w_i \leftarrow$ Entropy of $f(\mathbf{x}_i)$
  end for
  $\mathcal{S} \leftarrow$ Randomly select $\rho \cdot |\mathcal{D}|$ examples from $\mathcal{D}$ with probability proportional to w_i
  for $j = 0$ to $e - 1$ **do**
    Train f on $\mathcal{S}$ for one epoch
  end for
end for
Return: f

5.2.1 Quantitative Evaluation. Adopting an evaluation framework similar to the state-of-the-art, coreset-based CREST [41], we rigorously evaluate RandE against multiple prominent data subset selection methods. As seen in Table 4, RandE not only performs competitively in terms of error, but also requires significantly lower runtime. Additionally, as seen in Table 5, RandE has even more significant gains over CREST for smaller subset sizes, similar to simple random in Table 3. This demonstrates the effectiveness of incorporating entropy-based weighting into random sampling for data selection. Overall, RandE offers a compelling alternative to computationally expensive coreset methods, demonstrating that substantial gains can be achieved through simple yet informed modifications to random sampling.

Table 5: Performance comparison of RandE for data subset selection with the state-of-the-art CREST on CIFAR-100.

Size (%)	CREST	RandE	% Change
1	23.2	37.8	+62.9
4	55.5	61.2	+10.3
7	64.1	67.9	+5.9
10	68.4	70.3	+2.8
15	71.6	72.6	+1.4
20	72.3	73.2	+1.2

6 Conclusion

We unveil the Support Vector Effect (SVE)—a phenomenon arising in the last layer of DNNs due to implicit regularization effect of optimizers like gradient descent on interpolating DNNs. Using SVE, we are not only able to explain previous perplexing observations regarding prominent data selection and attribution methods that rely on last layer gradients, but also uncover various new limitations and insights. Furthermore, SVE, and the insights drawn from it, inform design of new methodologies and algorithms for both instance attribution and data subset selection that compete with the respective state of the art, despite being simple and order of magnitude faster. Our work motivates for better data selection and attribution methods, and advocates against the use of last layer approximations for downstream tasks. For future work, we aim to delve deeper into use of adaptive random methods by combining our proposed methods with varied sample importance criteria. Further, we plan to explore impact on SVE of different optimization algorithms that may lead to different forms of implicit regularization.

Acknowledgments

RK would like to acknowledge support from AnalytiXIN Indiana.

References

[1] Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2020. Deep Batch Active Learning by Diverse, Uncertain Gradient Lower Bounds. In *International Conference on Learning Representations*.

[2] Samyadeep Basu, Phil Pope, and Soheil Feizi. 2020. Influence Functions in Deep Learning Are Fragile. In *International Conference on Learning Representations*.

[3] R Dennis Cook and Sanford Weisberg. 1980. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics* 22, 4 (1980), 495–508.

[4] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12, 7 (2011).

[5] Vitaly Feldman and Chiyuan Zhang. 2020. What neural networks memorize and why: Discovering the long tail via influence estimation. *Advances in Neural Information Processing Systems* 33 (2020), 2881–2891.

[6] Fangcheng Fu, Yuzheng Hu, Yihan He, Jiawei Jiang, Yingxia Shao, Ce Zhang, and Bin Cui. 2020. Don't waste your bits! squeeze activations and gradients for deep neural networks via tinyscript. In *International Conference on Machine Learning*. PMLR, 3304–3314.

[7] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. 2018. Characterizing implicit bias in terms of optimization geometry. In *International Conference on Machine Learning*. PMLR, 1832–1841.

[8] Kazuaki Hanawa, Sho Yokoi, Satoshi Hara, and Kentaro Inui. 2021. Evaluation of Similarity-based Explanations. In *ICLR*.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[10] Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. 2022. Datamodels: Understanding Predictions with Data and Data with Predictions. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, 9525–9587.

[11] Athresh Karanam, Krishnateja Killamsetty, Harsha Kokel, and Rishabh Iyer. 2022. Orient: Submodular mutual information measures for data subset selection under distribution shift. *Advances in Neural Information Processing Systems* 35 (2022), 31796–31808.

[12] Rajiv Khanna, Been Kim, Joydeep Ghosh, and Sanmi Koyejo. 2019. Interpreting Black Box Predictions using Fisher Kernels. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 89)*, Kamalika Chaudhuri and Masashi Sugiyama (Eds.). PMLR, 3382–3390.

[13] Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. 2021. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*. PMLR, 5464–5474.

[14] Krishnateja Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, and Rishabh Iyer. 2021. Glister: Generalization based data subset selection for efficient and robust learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 8110–8118.

[15] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. 2016. Examples are not enough, learn to criticize! criticism for interpretability. *Advances in Neural Information Processing Systems* 29 (2016).

[16] Diederik P Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic gradient descent. In *ICLR*.

[17] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. 2023. Last Layer Re-Training is Sufficient for Robustness to Spurious Correlations. In *The Eleventh International Conference on Learning Representations*.

[18] Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*. PMLR, 1885–1894.

[19] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*. 10012–10022.

[20] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11976–11986.

[21] Kaifeng Lyu and Jian Li. 2019. Gradient Descent Maximizes the Margin of Homogeneous Neural Networks. In *International Conference on Learning Representations*.

[22] Kaifeng Lyu, Zhiyuan Li, Runzhe Wang, and Sanjeev Arora. 2021. Gradient descent on two-layer nets: Margin maximization and simplicity bias. *Advances in Neural Information Processing Systems* 34, 12978–12991.

[23] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. 2020. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*. PMLR, 6950–6960.

[24] Mahesh Chandra Mukkamala and Matthias Hein. 2017. Variants of rmsprop and adagrad with logarithmic regret bounds. In *International conference on machine learning*. PMLR, 2545–2553.

[25] Mor Shpigel Nacson, Nathan Srebro, and Daniel Soudry. 2019. Stochastic gradient descent on separable data: Exact convergence with a fixed learning rate. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 3051–3059.

[26] Behnam Neyshabur. 2017. Implicit Regularization in Deep Learning. arXiv:1709.01953 [cs.LG]

[27] Behnam Neyshabur, Russ R Salakhutdinov, and Nati Srebro. 2015. Path-sgd: Path-normalized optimization in deep neural networks. *Advances in Neural Information Processing Systems* 28 (2015).

[28] Giang Nguyen, Daeyoung Kim, and Anh Nguyen. 2021. The effectiveness of feature attribution methods and its correlation with automatic evaluation scores. *Advances in Neural Information Processing Systems* 34 (2021), 26422–26436.

[29] Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. 2023. TRAK: Attributing Model Behavior at Scale. In *Proceedings of the 40th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 202)*. PMLR, 27074–27113.

[30] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. 2021. Deep learning on a data diet: Finding important examples early in training. *Advances in Neural Information Processing Systems* 34 (2021), 20596–20607.

[31] Omead Pooladzandi, David Davini, and Baharan Mirzasoleiman. 2022. Adaptive second order coresets for data-efficient machine learning. In *International Conference on Machine Learning*. PMLR, 17848–17869.

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

[33] Qian Qian and Xiaoyuan Qian. 2019. The implicit bias of adagrad on separable data. *Advances in Neural Information Processing Systems* 32 (2019).

[34] Andrea Schioppa, Katja Filippova, Ivan Titov, and Polina Zablotskaia. 2023. Theoretical and Practical Perspectives on what Influence Functions Do. *Advances in Neural Information Processing Systems* (2023).

[35] Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. 2022. Scaling up influence functions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 8179–8186.

[36] Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. 2017. Failures of gradient-based deep learning. In *International Conference on Machine Learning*. PMLR, 3067–3075.

[37] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. 2018. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research* 19, 1 (2018), 2822–2878.

[38] Yi Sui, Ga Wu, and Scott Sanner. 2021. Representer point selection via local jacobian expansion for post-hoc classifier explanation of deep neural networks and ensemble models. *Advances in Neural Information Processing Systems* 34 (2021), 23347–23358.

[39] Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*. PMLR, 6105–6114.

[40] Bohan Wang, Qi Meng, Wei Chen, and Tie-Yan Liu. 2021. The implicit bias for adaptive optimization algorithms on homogeneous neural networks. In *International Conference on Machine Learning*. PMLR, 10849–10858.

[41] Yu Yang, Hao Kang, and Baharan Mirzasoleiman. 2023. Towards Sustainable Learning: Coresets for Data-efficient Deep Learning. *International Conference on Machine Learning (ICML)* (2023).

[42] Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. 2018. Representer point selection for explaining deep neural networks. *Advances in Neural Information Processing Systems* 31 (2018).

[43] Chih-Kuan Yeh, Ankur Taly, Mukund Sundararajan, Frederick Liu, and Pradeep Ravikumar. 2022. First is Better Than Last for Language Data Influence. *Advances in Neural Information Processing Systems* 35 (2022), 32285–32298.

[44] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 586–595.

A Theory

We reiterate our assumption that the feature representations $\phi(\mathbf{x}; \Theta_{\bar{L}})$ are fixed for all training data points. We also assume that the data is separable in the last layer, i.e., there exists a classifier that achieves perfect training accuracy. Note that both of these assumptions can be removed at the expense of clarity, and we also empirically verify that the central conclusion holds in practical DNNs.

A.1 Auxiliary Lemmas

We start by re-stating a result by Soudry et al. [37] that shows that for separable data, (stochastic) gradient descent converges to the max-margin solution for the logistic loss through *implicit* regularization, even though there could be several solutions that fit the training data with perfect accuracy.

Lemma A.1 (Soudry et al. [37]). *For a linearly separable dataset, logistic loss and for any step size $\eta < 2\beta^{-1}\lambda_{max}^{-2}(\mathbf{X})$, the gradient descent iterates will behave as:*

$$\Theta_L(t) = \Theta^* \log t + \boldsymbol{\rho}(t), \tag{6}$$

where Θ^* is the maximum ℓ_2-norm margin separator in the feature space defined by $\phi(\cdot)$ and the residual grows at most as $\|\boldsymbol{\rho}(t)\| = O(\log\log t)$, and so

$$\lim_{t\to\infty} \frac{\Theta_L(t)}{\|\Theta_L(t)\|} = \frac{\Theta^*}{\|\Theta^*\|} \tag{7}$$

Furthermore, the rate of convergence is $\left\| \frac{\Theta(t)}{\|\Theta(t)\|} - \frac{\Theta^*}{\|\Theta^*\|} \right\| \in O\left(\frac{1}{\log t}\right)$.

We now encapsulate the growth rate of the norm of the optimizer.

Lemma A.2 (Norm Growth of $\Theta_L(t)$). *Under the above assumptions, the norm of $\Theta_L(t)$ grows logarithmically with t:*

$$\|\Theta_L(t)\| = \Theta(\log t)$$

This follows from application of triangle inequalities to Equation (6), i.e., $\|\Theta^* \log t\| - \|\boldsymbol{\rho}(t)\| \le \|\Theta^* \log t + \boldsymbol{\rho}(t)\| \le \|\Theta^* \log t\| + \|\boldsymbol{\rho}(t)\|$. These lemmas indicate that gradient descent on logistic loss with linearly separable data leads to the weights growing in norm logarithmically and their direction converging to the maximum margin solution at a rate inversely proportional to $\log t$. Next, we establish a lemma regarding the decay of the gradient norms for individual data points.

Lemma A.3 (Decay of Gradient Norms). *The gradient norms satisfy:*

$$\|\nabla_{\Theta_L} \ell(\Theta_L(t); \mathbf{x}_i, y_i)\| = O\left(\frac{1}{t^{\gamma_i}}\right),$$

where $\gamma_i = y_i \Theta^{*\top} \phi(\mathbf{x}_i)$ is the margin of the data point $\mathbf{x}_i$ under Θ^*.

Proof. We begin by recalling the gradient of the logistic loss function with respect to Θ_L for a single data point $(\mathbf{x}_i, y_i)$:

$$\nabla_{\Theta_L} \ell(\Theta_L(t); \mathbf{x}_i, y_i) = -\sigma\left(-y_i \Theta_L(t)^\top \phi(\mathbf{x}_i)\right) y_i \phi(\mathbf{x}_i),$$

where $\sigma(u) = \frac{1}{1+e^{-u}}$ is the sigmoid function.

Estimating $y_i \Theta_L(t)^\top \phi(\mathbf{x}_i)$. Using Lemma A.1, we know that the direction of $\Theta_L(t)$ converges to that of Θ^*:

$$\lim_{t\to\infty} \frac{\Theta_L(t)}{\|\Theta_L(t)\|} = \frac{\Theta^*}{\|\Theta^*\|}$$

We define the following normalized vectors:

$$\hat{\Theta}_L(t) = \frac{\Theta_L(t)}{\|\Theta_L(t)\|}, \quad \hat{\Theta}^* = \frac{\Theta^*}{\|\Theta^*\|}$$

For $t \to \infty$,

$$y_i \Theta_L(t)^\top \phi(\mathbf{x}_i) \to \|\Theta_L(t)\| y_i \hat{\Theta}^{*\top} \phi(\mathbf{x}_i),$$

Next, we define the margin of data point $\mathbf{x}_i$ under the maximum margin separator Θ^*: $\gamma_i = y_i \Theta^{*\top} \phi(\mathbf{x}_i)$. Note that because of the separability assumption, $\gamma_i \ge 0$. Using this definition, we have:

$$y_i \hat{\Theta}^{*\top} \phi(\mathbf{x}_i) = \frac{\gamma_i}{\|\Theta^*\|}.$$

Substituting back, we get for $t \to \infty$:

$$y_i \Theta_L(t)^\top \phi(\mathbf{x}_i) \to \|\Theta_L(t)\| \frac{\gamma_i}{\|\Theta^*\|}.$$

Estimating $\|\Theta_L(t)\|$. From Lemma A.2, we know that $\|\Theta_L(t)\| = \|\Theta^*\| \log t + O(\log\log t)$. Thus for large t and ignoring the $\log\log t$ term,

$$\|\Theta_L(t)\| \to \|\Theta^*\| \log t.$$

Combining the Estimates. Substituting the estimate of $\|\Theta_L(t)\|$ into our previous approximation:

$$y_i \Theta_L(t)^\top \phi(\mathbf{x}_i) \approx (\|\Theta^*\| \log t) \left(\frac{\gamma_i}{\|\Theta^*\|}\right) = \gamma_i \log t.$$

Thus, we have:

$$y_i \Theta_L(t)^\top \phi(\mathbf{x}_i) \to \gamma_i \log t.$$

Estimating $\sigma\left(-y_i \Theta_L(t)^\top \phi(\mathbf{x}_i)\right)$. We now compute the sigmoid function:

$$\sigma\left(-y_i \Theta_L(t)^\top \phi(\mathbf{x}_i)\right) = \frac{1}{1 + e^{y_i \Theta_L(t)^\top \phi(\mathbf{x}_i)}}.$$

This gives:

$$\sigma\left(-y_i \Theta_L(t)^\top \phi(\mathbf{x}_i)\right) \to \frac{1}{1 + e^{\gamma_i \log t}} = \frac{1}{1 + t^{\gamma_i}}.$$

Estimating the Gradient Norm. The norm of the gradient is:

$$\|\nabla_{\Theta_L} \ell(\Theta_L(t); \mathbf{x}_i, y_i)\| = \sigma\left(-y_i \Theta_L(t)^\top \phi(\mathbf{x}_i)\right) \|\phi(\mathbf{x}_i)\|,$$

since $|y_i| = 1$.

Substituting the approximation for the sigmoid function we have, as $t \to \infty$:

$$\|\nabla_{\Theta_L} \ell(\Theta_L(t); \mathbf{x}_i, y_i)\| \to \frac{1}{t^{\gamma_i}} \|\phi(\mathbf{x}_i)\|.$$

Since $\|\phi(\mathbf{x}_i)\|$ is a constant that does not depend on t,

$$\|\nabla_{\Theta_L} \ell(\Theta_L(t); \mathbf{x}_i, y_i)\| = O\left(\frac{1}{t^{\gamma_i}}\right).$$

□

A.2 Proof of Theorem 3.1 (SVE)

Proof. We write $\psi_i = \sigma\left(-y_i \Theta_L(t)^\top \phi(\mathbf{x}_i)\right)$. From the gradient of the logistic loss, the total gradient at iteration t is:

$$G(t) = \sum_{i=1}^{n} \nabla_{\Theta_L} \ell(\mathbf{x}_i, y_i; \Theta_L(t)) = -\sum_{i=1}^{n} \psi_i y_i \phi(\mathbf{x}_i)$$

Similarly, the gradient sum over support vectors is:

$$G_{\mathcal{S}}(t) = \sum_{i\in\mathcal{S}} \nabla_{\Theta_L} \ell(\mathbf{x}_i, y_i; \Theta_L(t)) = -\sum_{i\in\mathcal{S}} \psi_i y_i \phi(\mathbf{x}_i)$$

The difference is:

$$D(t) = G(t) - G_{\mathcal{S}}(t) = \sum_{i\notin\mathcal{S}} \nabla_{\Theta_L} \ell(\mathbf{x}_i, y_i; \Theta_L(t))$$

We need to bound $\|D(t)\|$. Using Lemma A.3, for $i \notin \mathcal{S}$:

$$\|\nabla_{\Theta_L} \ell(\Theta_L(t); \mathbf{x}_i, y_i)\| = O\left(\frac{1}{t^{\gamma_i}}\right),$$

where $\gamma_i > \gamma$, since non-support vectors have larger margins.

Let $\delta_i = \gamma_i - \gamma$, and $\delta = \min_{i \notin \mathcal{S}} \delta_i > 0$. Therefore,

$$\|\nabla_{\Theta_L} \ell(\Theta_L(t); \mathbf{x}_i, y_i)\| = O\left(\frac{1}{t^{\gamma+\delta_i}}\right) \leq O\left(\frac{1}{t^{\gamma+\delta}}\right).$$

Summing over all non-support vectors:

$$\|D(t)\| \leq \sum_{i \notin \mathcal{S}} \|\nabla_{\Theta_L} \ell(\Theta_L(t); \mathbf{x}_i, y_i)\| \leq \left(\sum_{i \notin \mathcal{S}} \|\phi(\mathbf{x}_i)\|\right) \cdot \frac{1}{t^{\gamma+\delta}}.$$

□

To further elucidate the implications of the decay rates, note that for support vectors, $\gamma_i = \gamma$, so their gradient norms decrease as $O\left(\frac{1}{t^{\gamma}}\right)$. Additionally, for non-support vectors, $\gamma_i > \gamma$, so their gradient norms decrease faster, at rate $O\left(\frac{1}{t^{\gamma+\delta}}\right)$, where $\delta > 0$.

B Experiments

Sensitivity-Similarity Comparison through RPS. The instance attribution method RPS [42] has this neat formulation where the attribution score ("representer value") can be neatly decomposed into product of sensitivity and similarity terms. In Figure A1, we visualize the distribution of these as an additional means to illustrate SVE, or sparsity of gradients, and reflect how similarity terms may naturally be more meaningful.

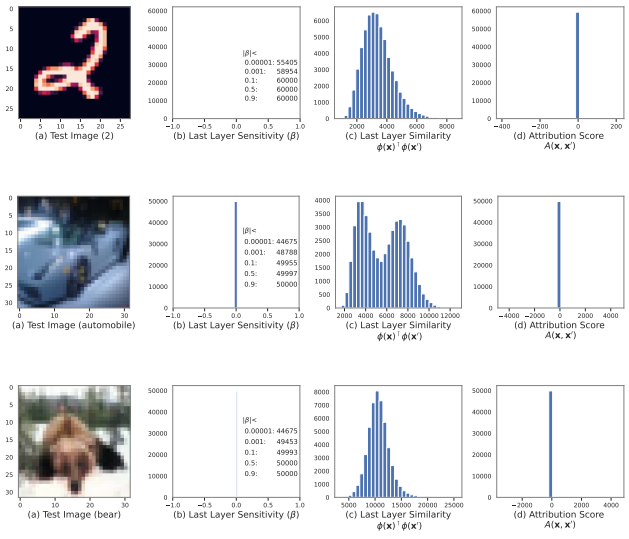
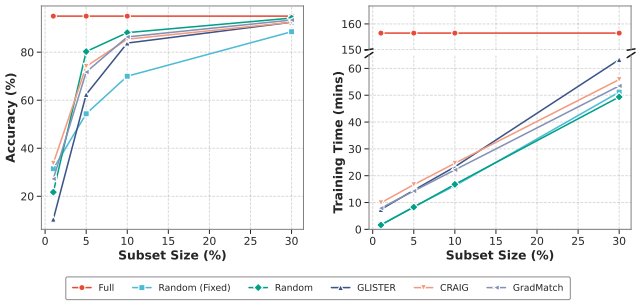
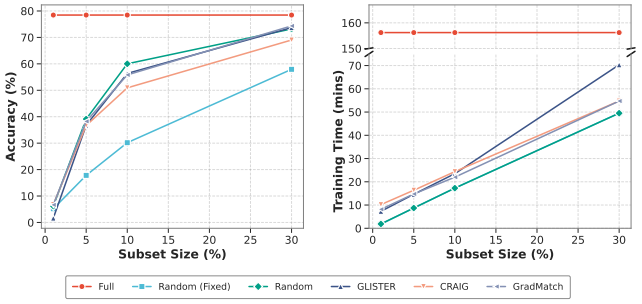


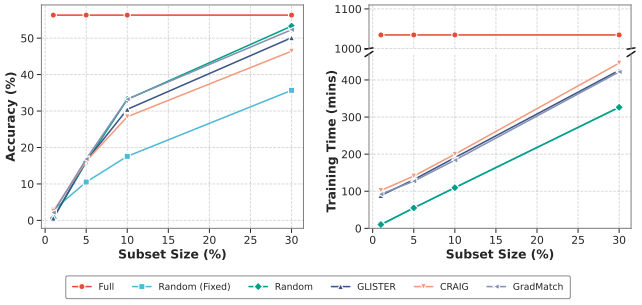
Figure A1: Illustration of SVE and its impact on instance attribution on instances from MNIST (top), CIFAR-10 (middle) and CIFAR-100 (bottom) datasets. We observe that almost all training instances have near-zero gradient (b), which dominates the representer values too (d), inadequately leading to most instances being considered to have no significant attribution. Similarity term (c) shows more diversity.



(a) CIFAR-10



(b) CIFAR-100



(c) Tiny Imagenet

Figure A2: Accuracy (higher is better) and time (lower is better) comparison of various data subset selection strategies for training ResNet-101 on different datasets.

Data Subset Selection. To further elucidate how coreset-based methods are no better than random for data subset selection, as discussed in Section 4.3 of main text, we conduct additional experiments on a larger model (ResNet-101) and with multiple datasets (CIFAR-10, CIFAR-100 and Tiny ImageNet). As seen in Figure A2, while the exact performance and gains vary across datasets, the general insights remain applicable throughout.

Experimental Details. Our experiments are implemented using TensorFlow, Keras, and PyTorch, depending on the available implementation for existing methods. We stick with default configurations wherever feasible. Training and evaluations are conducted on a mix of NVIDIA A10 and A100 GPUs. Code is made available at: https://github.com/shasanamin/sve.