Scalability vs. Utility: Do We Have to Sacrifice One for the Other in Data Importance Quantification?

Ruoxi Jia[1] Fan Wu[2*] Xuehui Sun[3*] Jiacen Xu[4*] David Dao[5]
Bhavya Kailkhura[6] Ce Zhang[5] Bo Li[2] Dawn Song[7]
[1]Virginia Tech [2]UIUC [3]Shanhai Jiaotong University [4]UC Irvine [5]ETH Zurich
[6]Lawrence Livermore National Laboratory [7]UC Berkeley
ruoxijia@vt.edu {fanw6,lbo}@illinois.edu zidaneandmessi@sjtu.edu.cn jiacenx@uci.edu
{david.dao,ce.zhang}@inf.ethz.ch kailkhura1@llnl.gov dawnsong@gmail.com

Abstract

Quantifying the importance of each training point to a learning task is a fundamental problem in machine learning and the estimated importance scores have been leveraged to guide a range of data workflows such as data summarization and domain adaption. One simple idea is to use the leave-one-out error of each training point to indicate its importance. Recent work has also proposed to use the Shapley value, as it defines a unique value distribution scheme that satisfies a set of appealing properties. However, calculating Shapley values is often expensive, which limits its applicability in real-world applications at scale. Multiple heuristics to improve the scalability of calculating Shapley values have been proposed recently, with the potential risk of compromising their utility in real-world applications.

How well do existing data quantification methods perform on existing workflows? How do these methods compare with each other, empirically and theoretically? Must we sacrifice scalability for the utility in these workflows when using these methods? In this paper, we conduct a novel theoretical analysis comparing the utility of different importance quantification methods, and report extensive experimental studies on existing and proposed workflows such as noisy label detection, watermark removal, data summarization, data acquisition, and domain adaptation. We show that Shapley value approximation based on a KNN surrogate over pre-trained feature embeddings obtains comparable utility with existing algorithms while achieving significant scalability improvement, often by orders of magnitude. Our theoretical analysis also justifies its advantage over the leave-one-out error.

The code is available at https://github.com/AI-secure/Shapley-Study.

*Equal contribution.

1. Introduction

Understanding the *importance* of a single training example, relative to other training examples, to a learning task is a fundamental problem in machine learning (ML) which could have profound impact on a range of applications including interpretability, robustness, data acquisition, data valuation, among others [12, 7, 14].

In this paper, we are driven by *two* questions around this fundamental problem. Our contribution is a novel theoretical analysis and thorough experimental studies towards understanding both questions.

Q1: Leave-one-out vs. Shapley? Given a training set D, a validation set D_{val} and a learning algorithm $\mathcal{A}$, let the utility $U_{\mathcal{A},D_{val}}(D)$ be the validation accuracy of the model trained on D using $\mathcal{A}$, recently there have been two lines of work in assigning relative importance to a data point $z \in D$.

A. Leave-one-out (LOO)-based Method & Influence Function. One natural way to assign importance to z is by calculating its contribution to the rest of training data:

$$v_{loo}(z) \propto U_{\mathcal{A},D_{val}}(D) - U_{\mathcal{A},D_{val}}(D\backslash\{z\})$$

When we need to assign such an importance score to all data points in the training set, we need to train a large number of models. Thus researchers have proposed efficient techniques to approximate this score, e.g., via influence function [14].

B. Shapley-based Method. Another natural way to assign importance to z is inspired by cooperative game theory and to use the Shapley value [12, 7]:

$$\nu_{\text{shap}}(z) \propto \frac{1}{N}\sum_{S\subseteq D\backslash\{z\}} \frac{1}{\binom{N-1}{|S|}}\left[U_{\mathcal{A},D_{val}}(S\cup\{z\}) - U_{\mathcal{A},D_{val}}(S)\right]$$

Both approaches have recently been explored by researchers and have been applied to a range of ML tasks including noisy label detection, watermark removal, data summarization, active data acquisition, and domain adaptation [12, 7, 14].

However, one question remains: *What's the relationships and differences, both theoretically and empirically, between these two lines of approaches?*

Q2: Exact Shapley vs. Shapley over Surrogates? As we will show in this paper, Shapley-based methods often outperforms leave-one-out-based methods, both theoretically and empirically. However, Shapley-based approaches can be expensive as one needs to train, for general classifiers, exponentially many models. Thus, many state-of-the-art approaches resort to a sampling-based approach [12, 7] to approximate this score. On the other hand, a recent work by Jia *et al.* [11] has shown that for a certain family of classifiers, i.e., K-Nearest Neighbor (KNN), calculating this score can be done efficiently, in $\mathcal{O}(|D| \log |D|)$ time for *all* data points in D.

Despite this, there still remains a question: *Can we use a K-Nearest Neighbor classifier as a surrogate model to calculate the Shapley value, and how does it perform on real-world applications compared with the vanilla exact Shapley value?*

Technical Contributions. In this paper, we take the first step towards understanding the above questions. We make contributions on both theoretical and empirical fronts.

- We conduct a novel theoretical analysis aiming at rigorously analyzing the differences between the leave-one-out-based and the Shapley-based methods. Specifically, we formalize two performance metrics specific to data importance: one focuses on the predictive power of data importance, studying whether it is indicative of a training point's contribution to a random set; the other focuses on the ability of a data to discriminate "good" training points from "bad" ones. We show that for both performance metrics, under certain technical conditions, the Shapley-based method can outperform leave-one-out–based approaches. To our best knowledge, this is the first theoretical analysis reasoning the relative performances of different data importance quantification techniques.

- We conduct a thorough empirical study on a range of ML tasks, including noisy label detection, watermark removal, data summarization, active data acquisition, and domain adaptation on different benchmark datasets. Some have been used by previous work as a use case of data valuation methods, and some are proposed by us. On these tasks, we empirically investigate the relative performance between (1) leave-one-out–based methods and Shapley-based methods, and (2) exact Shapley-based methods and Shapley over KNN Surrogates.

- Our empirical study suggests that the Shapley-over-KNN-Surrogates method performs well and achieves comparable results with, and often outperforms, all other methods in quality while being orders of magnitude faster. This gives us the first practical algorithm over large-scale datasets that returns useful data importance scores for a range of important ML tasks.

2. Background: General Frameworks for Data Importance Quantification

There are two lines of work in estimating the importance of a single training point for supervised learning [12, 7, 14]. In this section, we describe these methods and the KNN-surrogate-based method to set the context for our analysis in Section 3-4.

We first set up the notations to characterize the main ingredients of a supervised learning problem, including the training and validation data, the learning algorithm, and the performance measure. Let $D = \{z_i\}_{i=1}^N$ be the training set, where z_i is a feature-label pair (x_i, y_i), and D_{val} be the validation data. Let $\mathcal{A}$ be the learning algorithm which maps a training dataset to a model. Let U be a performance measure which takes as input training data, any learning algorithm, and validation data and returns a score. We write $U(S, \mathcal{A}, D_{\text{val}})$ to denote the performance score of the model trained on a subset S of training data using the learning algorithm $\mathcal{A}$ when testing on D_{val}. When the learning algorithm and validation data are self-evident, we will suppress the dependence of U on them and just use $U(S)$ for short. Our goal is to assign a score to each training point z_i, denoted by $\nu(z_i, D, \mathcal{A}, D_{\text{val}}, U)$, indicating its *importance* to the supervised learning problem specified by $D, \mathcal{A}, D_{\text{val}}, U$. We will often write it as $\nu(z_i)$ or $\nu(z_i, U)$ to simplify notation.

2.1. Leave-One-Out Method

One simple way to quantify data importance is to measure one data point's contribution to the rest of the training data:

$$\nu_{\text{loo}}(z_i) = U(D) - U(D \setminus \{z_i\}) \tag{1}$$

This data importance measure is referred to as the Leave-One-Out (LOO) value. The exact evaluation of the LOO values for N training points requires re-training the model for N times and the associated computational cost is prohibitive for large training datasets and large models. For deep neural networks, Koh *et al.* [14] proposed to estimate the model performance change due to the removal of each training point via influence functions. However, in order to obtain the influence functions, one will need to evaluate the inverse of the Hessian for the loss function. With N training points and p model parameters, it requires $\mathcal{O}(Np^2 + p^3)$ operations. Koh *et al.* [14] approximate the influence function with $\mathcal{O}(Np)$ complexity, which is still expensive for large networks.

2.2. Shapley Value-based Method

The Shapley value is a classic concept in cooperative game theory to distribute the total gains generated by the

coalition of all players. One can think of a supervised learning problem as a cooperative game among training data instances and apply the Shapley value to value the contribution of each training point. Given a performance measure U, the Shapley value for training data z_i is defined as the average marginal contribution of z_i to all possible subsets of D formed by other training points:

$$\nu_{\text{shap}}(z_i) = \frac{1}{N} \sum_{S \subseteq D \setminus \{z_i\}} \frac{1}{\binom{N-1}{|S|}} \big[U(S \cup \{z_i\}) - U(S)\big] \quad (2)$$

However, calculating the Shapley value can be expensive: evaluating the exact Shapley value involves computing the marginal contribution of each training point to all possible subsets, whose complexity is $\mathcal{O}(2^N)$. Such complexity is clearly impractical for valuating a large number of training points. Even worse, for ML tasks, evaluating the utility function *per se* (e.g., validation accuracy) is computationally expensive as it requires re-training an ML model.

MCMC-based Approximation. Ghorbani *et al.* [7] introduced two approaches to approximating the Shapley value based on Monte Carlo approximation. The central idea behind these approaches is to treat the Shapley value of a training point as its expected contribution to a random subset and use sample average to approximate the expectation. By the definition of the Shapley value, the random set has size 0 to $N-1$ with equal probability and is also equally likely to be any subset of a given size (corresponding to the $1/\binom{N-1}{|S|}$ factor). In practice, one can implement an equivalent sampler by drawing a random permutation of the training set. Then, the Shapley value can be estimated by computing the marginal contribution of a point to the points preceding it and averaging the marginal contributions across different permutations. However, these Monte Carlo-based approaches cannot circumvent the need to re-train models and therefore are not viable for large models. In our experiments, we found that the approaches in Ghorbani *et al.* [7] can manage data size up to one thousand for simple models such as logistic regression and shallow neural networks, while failing to estimate the Shapley value for larger data sizes and deep nets in a reasonable amount of time. We evaluate runtime in more details in Section 4.

KNN Surrogate-based Approach. In one recent paper, Jia *et al.* [11] developed an exact, efficient algorithm to compute the Shapley value for KNN classifiers. In principle, we can use a KNN classifier to act as a surrogate model and use it instead of the target learning algorithm. Given a single validation point x_{val} with the label y_{val}, the simplest, unweighted version of a KNN classifier first finds the top-K training points $(x_{\alpha_1}, \cdots, x_{\alpha_K})$ that are most similar to x_{val} and outputs the probability of x_{val} taking the label y_{val} as $P[x_{\text{val}} \to y_{\text{val}}] = \frac{1}{K}\sum_{i=1}^{K} \mathbb{1}[y_{\alpha_i} = y_{\text{val}}]$. We assume that the confidence of predicting the right label is used as the performance measure, i.e.,

$$U(S) = \frac{1}{K} \sum_{k=1}^{\min\{K,|S|\}} \mathbb{1}[y_{\alpha_k(S)} = y_{\text{val}}] \quad (3)$$

where $\alpha_k(S)$ represents the index of the training feature that is the kth closest to x_{val} among the training examples in S. Particularly, $\alpha_k(D)$ is abbreviated to α_k. Under this performance measure, the Shapley value can be calculated exactly using the following theorem.

Theorem 1 (Jia *et al.* [11])**.** *Consider the model performance measure in (3). Then, the Shapley value of each training point can be calculated recursively as follows:*

$$\nu(z_{\alpha_N}) = \frac{\mathbb{1}[y_{\alpha_N} = y_{val}]}{N} \quad (4)$$

$$\nu(z_{\alpha_i}) = \nu(z_{\alpha_{i+1}}) + \frac{\mathbb{1}[y_{\alpha_i} = y_{val}] - \mathbb{1}[y_{\alpha_{i+1}} = y_{val}]}{K} \frac{\min\{K,i\}}{i} \quad (5)$$

Theorem 1 can be readily extended to the case of multiple validation points by summing up the Shaplley value with respect to each validation point. We will call the scores obtained from (4) and (5) *the KNN-Shapley value* hereinafter. For each validation point, computing *the KNN-Shapley value* requires only $\mathcal{O}(N \log N)$ time, which circumvents the exponentially many utility evaluations entailed by the Shapley value definition.

Using Pre-trained Embeddings. One problem of using KNN as a surrogate model is that KNN often does not perform well on high-dimensional data. As many works have illustrated the power of pre-trained embeddings on a different, new task [17, 16, 24, 30], we address this problem by using *pre-trained embeddings* as a feature extractor and then apply KNN. Note that this feature needs to be trained on a *different dataset* for KNN surrogate to respect Shapley value semantics.

3. Theoretical Comparison Between LOO and Shapley Value

We now focus on the first question: *What's the relationships and differences, both theoretically and empirically, between these two lines of approaches?* Specifically, we define two performance metrics and conduct theoretical analysis *under different technical assumptions*. To our best knowledge, this is the first theoretical analysis reasoning about the relative performances of different techniques that measure data importance.

3.1. Performance Metric 1: Order-Preservation

Both the LOO-based method and the Shapley-based method only measure the importance of a data point relative to other points in the given dataset. Since it is still

uncertain what data will be used in tandem with the point being valued after its importance is measured, *in the first performance metric*, we hope that the data importance measures of a point are indicative of the expected performance boost when combining the point with a random set of data points.

In particular, we consider two points that have different scores under a given data importance measure and study whether the expected model performance improvements due to the addition of these two points will have the same order as the importance scores. With the same order, we can confidently select the higher-importance point in favor of another when performing ML tasks. We formalize this desirable property in the following definition.

Definition 1. *We say a data importance measure ν is order-preserving at a pair of training points z_i, z_j with different scores if $\big(\nu(z_i, U) - \nu(z_j, U)\big) \times \mathbb{E}\big[U(T \cup \{z_i\}) - U(T \cup \{z_j\})\big] > 0$ where T is an arbitrary random set drawn from some distribution.*

For general model performance measures U, it is difficult to analyze the order-preservation of the corresponding data importance measures. However, for KNN, we can precisely characterize this property for both the LOO and the Shapley value. The formula for *the KNN-Shapley value* is given in Theorem 1 and we present the expression for *the KNN-LOO value* in the following lemma.

Lemma 1 (KNN-LOO Value)**.** *Consider the model performance measure in (3). Then, the KNN-LOO value of each training point can be calculated by $\nu_{loo}(z_{\alpha_i}) = \frac{1}{K}\big(\mathbb{1}[y_{\alpha_i} = y_{val}] - \mathbb{1}[y_{\alpha_{K+1}} = y_{val}]\big)$ if $i \leq K$ and 0 otherwise.*

Now, we are ready to state the theorem that exhibits the order-preservation of *the KNN-LOO value* and *the KNN-Shapley value*.

Theorem 2. *For any given $D = \{z_1, \ldots, z_N\}$, where $z_i = (x_i, y_i)$, and any given validation point $z_{val} = (x_{val}, y_{val})$, assume that $z_1, \ldots, z_N$ are sorted according to their similarity to x_{val}. Let $d(\cdot, \cdot)$ be the feature distance metric according to which D is sorted. Suppose that $P_{(X,Y)\in\mathcal{D}}(d(X, x_{val}) \geq d(x_i, x_{val})) > \delta$ for all $i = 1, \ldots, N$ and some $\delta > 0$. Then, $\nu_{shap\text{-}knn}$ is order-preserving for all pairs of points in I; $\nu_{LOO\text{-}knn}$ is order-preserving only for (z_i, z_j) such that $\max i, j \leq K$.*

Due to the space limit, we will omit all proofs to the appendix. The assumption that $P_{(X,Y)\in\mathcal{D}}(d(X, x_{\text{val}}) \geq d(x_i, x_{\text{val}})) > \delta$ in Theorem 2 intuitively means that it is possible to sample points that are further away from x_{val} than the points in D. This assumption can easily hold for reasonable data distributions in continuous space.

Theorem 2 indicates that *the KNN-Shapley value* has more predictive power than *the KNN-LOO value—the KNN-Shapley value* can predict the relative utility of any two points in D, while *the KNN-LOO value* is only able to correctly predict the relative utility of the K-nearest neighbors of x_{val}. In Theorem 2, the relative data utility of two points is measured in terms of the model performance difference when using them in combination with a random dataset.

Theorem 2 can also be generalized to the setting of multiple validation points using the additivity property. Specifically, for any two training points, *the KNN-Shapley value* with respect to multiple validation points is order-preserving when the order remains the same on each validation point, while *the KNN-LOO value* with respect to multiple validation points is order-preserving when the two points are within the K-nearest neighbors of all validation points and the order remains the same on each validation point. We can see that similar to the single-validation-point setting, the condition for *the KNN-LOO value* with respect to multiple validation points to be order-preserving is more stringent than that for *the KNN-Shapley value*.

3.2. Performance Metric 2: Value Distinguishness

In the second performance metric, we are interested in conditions under which LOO-based method and Shapley-based method cannot distinguish between different data points, independently of their importance. The technical tool that we use as a demonstration is to consider the setting in which the classifier is trained in a differentially private (DP) manner.

Definition 2 (Differential privacy)**.** *$\mathcal{A} : \mathcal{D}^N \to \mathcal{H}$ is (ϵ, δ)-differentially private if for all $R \subseteq \mathcal{H}$ and for all $D, D' \in \mathcal{D}^N$ such that D and D' differ only in one data instance: $P[\mathcal{A}(D) \in R] \leq e^{\epsilon} P[\mathcal{A}(D') \in R] + \delta$.*

By definition, differentially private learning algorithms hide the influence of one training point on the model performance. Thus, it may be more difficult to differentiate "good" data from "bad" ones for differentially private models. We will show that the Shapley value could have more discriminative power than the LOO value when the learning algorithms satisfy DP.

The following theorem states that for differentially private algorithms, the values of training data are gradually indistinguishable from each other as the training size grows larger using both the LOO and the Shapley value measures; nonetheless, the value differences vanish faster for the LOO value than the Shapley value.

Theorem 3. *For a learning algorithm $\mathcal{A}(\cdot)$ that achieves $(\epsilon(N), \delta(N))$-DP when training on N data points, let the performance measure be $U(S) = -\frac{1}{M}\sum_{i=1}^{M} \mathbb{E}_{h\sim\mathcal{A}(S)} l(h, z_{val,i})$ for $S \subseteq D$. Let $\epsilon'(N) = e^{c\epsilon(N)} - 1 + ce^{c\epsilon(N)}\delta(N)$. It holds that*

$$\max_{z_i \in D} \nu_{loo}(z_i) \leq \epsilon'(N-1) \quad \max_{z_i \in D} \nu_{shap}(z_i) \leq \frac{1}{N-1} \sum_{i=1}^{N-1} \epsilon'(i).$$

For typical differentially private learning algorithms, such as adding random noise to stochastic gradient descent, the privacy guarantees will be weaker if we reduce the size of training set (e.g., see Theorem 1 in Abadi *et al.* [1]). In other words, $\epsilon(n)$ and $\delta(n)$ are monotonically decreasing functions of n, and so is $\epsilon'(n)$. Therefore, it holds that $\epsilon'(N) < \frac{1}{N}\sum_{i=1}^{N}\epsilon'(i)$. The implications of Theorem 3 are three-fold. Firstly, the fact that the maximum score of all training points is directly upper bounded by ϵ' signifies that stronger privacy guarantees will naturally increase the difficulty to distinguish the importance of different points. Secondly, the monotonic dependency of ϵ' on N indicates that both the LOO and the Shapley value converge to zero when the training size is very large. Thirdly, by comparing the upper bound for the LOO and the Shapley value, we see that the convergence rate of the Shapley value is slower and thus it has a better chance to differentiate "good" data from the "bad" ones compared with the LOO value.

Our results are extendable to general *stable* learning algorithms, which are insensitive to the removal of an arbitrary point in the training dataset [4]. Stable learning algorithms are appealing as they enjoy provable generalization error bounds. Indeed, differentially private algorithms are subsumed by the class of stable algorithms [28]. We leave the details to the appendix.

4. Empirical Studies

Here we conduct a thorough empirical study on a range of real-world ML applications with different datasets to investigate the performance comparison between (a) leave-one-out–based method and Shapley-based method, and (b) exact Shapley-based method and Shapley over KNN surrogates. We first compare the runtime for different data importance quantification methods, followed by the data importance predictive power comparison, which is demonstrated on applications including mislabeled data detection, watermark removal, data summarization, active data acquisition, and domain adaptation. Due to the space limit, we leave the detailed experimental settings to appendix.

4.1. Data Importance Quantification Approaches

Here we we mainly compare the up-to-date data importance quantification approaches, including the exact Shapley-based method, leave-one-out method, as well as the ones using KNN as surrogates for both.

Truncated Monte Carlo Shapley (TMC-Shapley). This is a Monte Carlo-based approximation algorithm proposed in Ghorbani *et al.* [7]. Monte Carlo-based methods regard the Shapley value as the expectation of a training instance's marginal contribution to a random set and then use the sample mean to approximate it.

Gradient Shapley (G-Shapley). This is another Monte Carlo-based method proposed in Ghorbani *et al.* [7] with a different heuristic to accelerate the algorithm. G-Shapley approximates the model performance change due to the addition of one training point by taking a gradient descent step at that point and calculating the performance difference. This method is applicable only to the models trained with gradient methods; hence, the method will be included as a baseline in our experimental results when the underlying models are trained using gradient methods.

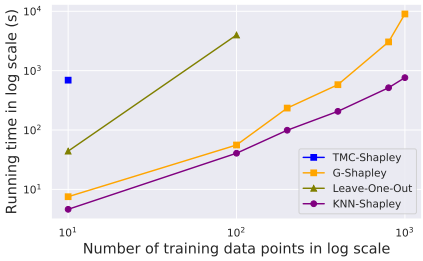


Figure 1: runtime comparison.

Leave-One-Out (LOO). We use LOO to refer to the algorithm that calculates the exact model performance due to the removal of a training point. Evaluating the LOO error requires to re-train the model on the reduced dataset for every training point, thus also impractical for large models.

KNN-LOO. Leave-one-out is efficient for KNN according to Theorem 1. To use the KNN-LOO for valuing data, we first use the pre-trained models offered in PyTorch [23] to extract features for KNN and compute the KNN-LOO value over the extracted features.

KNN-Shapley. We use KNN-Shapley to refer to the following algorithm: similar to in KNN-LOO, we first use the pretrained models in PyTorch to extract features. We then directly apply Theorem 1 to compute the Shapley value over pre-trained feature transformations. When pre-trained feature transformations are not available, we directly compute *the KNN-Shapley value* on the raw data as a surrogate for the true Shapley value. The complexity of the above algorithm is $\mathcal{O}(Nd + N\log N)$ where d is the dimension of feature representation. As opposed to Monte Carlo-based methods (e.g., [7, 12]), the proposed algorithm does not require retraining models. It is well suited for approximating scores for large models.

Random. The random baseline does not differentiate importance between different data points and selects data randomly from training set to perform a given task.

4.2. Runtime Comparison

First, we compare the runtime between the KNN-Shapley approach and other baselines. Fig. 1 corresponds to ResNet-18 [9] on CIFAR-10 [18], implemented on a machine with 1.80 GHz and 32 GB memory. We can see that KNN-Shapley (using pre-trained MobileNet [10] embedding) outperforms other approaches by several orders of magnitude

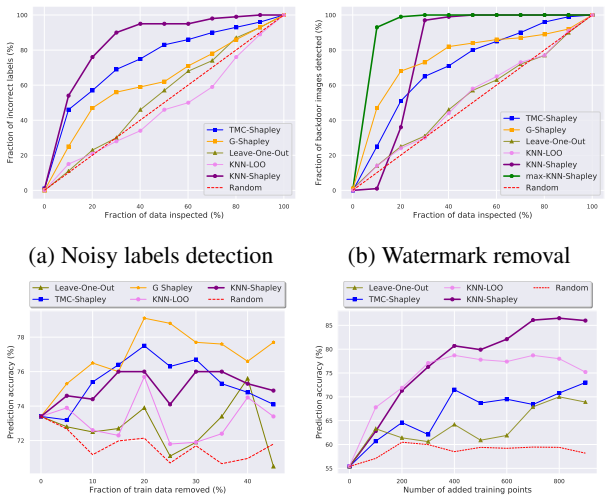
Table 1: a summary of experiments in Appendix E.

Task (Section)	Datasets
1. Noisy labels Detection (E.1)	Spam [3], Flower[1]
2. Pattern-based watermark removal (E.2)	Fashion-MNIST [29], MNIST [21], PubFig-83 [19]
3. Instance-based watermark removal (E.1)	CIFAR-10 [18], SVHN [22]
4. Data summarization (E.3)	Tiny ImageNet [20]
5. Data acquisition (E.4)	Tiny ImageNet [20]
6. Domain adaptation (E.5)	MNIST [21] → SVHN [22]

for large training data size and large model size.

4.3. Comparisons on Applications

We study the efficacy of data importance estimated by different approaches on a range tasks, including noisy label detection, watermark removal, data summarization, active data acquisition, and domain adaptation. While most of the applications are used in a recent work [7], the watermark removal including both pattern-based and instance-based watermark removal evaluations are proposed by us here. We consistently use the MobileNet embedding pretrained on ImageNet on image inputs. For the spam dataset [3] and the flower dataset[1], we do not apply embedding as the provided features are not of image form. We compare different embeddings in Appendix F.



(a) Noisy labels detection (b) Watermark removal

(c) Data summarization (d) Data acquisition

Figure 2: The experiment result of (a) noisy label detection on fashion-MNIST dataset; (b) instance-based watermark removal on MNIST dataset; (c) data summarization on UCI Adult Census dataset [15]; (d) data acquisition on MNIST dataset with injected noise. In (a)-(b) the "random" line shows the results of random guess; while in (c)-(d), the "random" line corresponds to the empirical results of the random baseline introduced in Section 4.1.

[1] https://www.tensorflow.org/tutorials/load_data/images

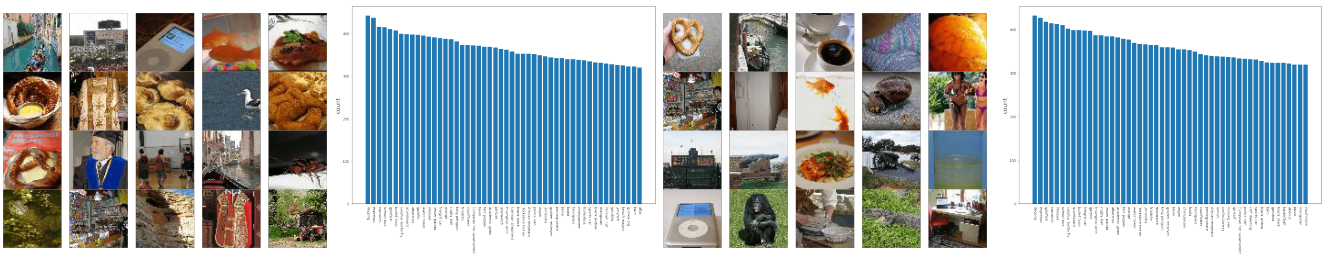
Summary of Experiments in Appendix. In this paper, we evaluate different methods on 6 applications, each of which on 2 to 3 datasets following the practice of previous work. Due to the space limitation, we describe only *one* representative dataset in the main body and leave the experiment details and results on other datasets to Appendix E. Table 1 is a summary of the experiments in the appendix.

Noisy Labels Detection. Labels in the real world are often noisy due to automatic labeling, non-expert labeling, or label corruption by data poisoning adversaries. We show that the notion of data importance can help prioritize the verification process, allowing experts to review only the examples that are most likely to be contaminated. *The key idea is to rank the data points according to their data importance and prioritize the points with the lowest importance scores.* Following Ghorbani *et al.* [7], we perform experiments in three settings and present the result of a three-layer convolutional network trained on the fashion-MNIST dataset here in the main body. The noise flipping ratio is 10% for this dataset. The performance of different data importance measures is illustrated in Fig. 2a. We examine the label of the training instances that have the lowest scores, and plot the change of the fraction of detected mislabeled data (in percentage) with the fraction of the checked training data (in percentage). We can see that *the KNN-Shapley value* outperforms all other methods. Also, the Shapley value–based measures, including TMC-Shapley, G-Shapley, and our KNN-Shapley, are more effective than the LOO-based measures.

Watermark Removal. One prevalent way to claim the ownership of a trained deep net is to embed watermarks into the model. There are two classes of watermarking techniques, namely, pattern-based techniques and instance-based techniques. The watermark examples are displayed in Fig. 4. Here, we present the experiment results for instance-based techniques. The details on how they the watermarks are generated and how they work, as well as the experiment results for the pattern-based techniques are left to Appendix E.2.

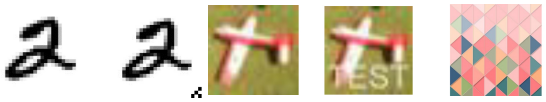
In this application, we demonstrate that it is always possible for the model trainer to remove the watermarks based on data importance. The idea is that the watermarks should have low data importance by nature, since they contribute little to predict the normal validation data. Note that this experiment constitutes a new type of attack, which might be of independent interest itself.

We consider the setting of one logistic regression model trained on 10000 images from MNIST for instance-based watermark removal. The watermark ratio is 10%. For this experiment, we found that both watermarks and benign instances tend to have low scores on some validation instances; therefore, they are not quite separable in terms of the score averaged over the whole validation set. We instead propose



(a) (b) (c) (d)

Figure 3: (a) (c) Top 20 selected images with the highest Shapley values for Tiny ImageNet, using MobileNet and VGG11 embeddings. (b) (d) Histogram of images in top 50 classes after summarization (sorted by decreasing order). It is clear that within the top 50 classes, there are many overlapped classes between different embeddings.



(a) (b) (c)

Figure 4: Examples of watermarks generated by (a)-(b) pattern-based and (c) instance-based techniques.

to compute the max score across the validation set for each training point, which we name as *max-KNN-Shapley*, and remove the instances with lowest *max-KNN-Shapley* values. The intuition is that out-of-distribution samples are inessential to the prediction of normal validation instances and thus the maximum of their Shapley values w.r.t. different validation instances should be low. In plotting Fig. 2b, we examine the label of the training instances that have the lowest scores and plot the change of the fraction of the detected watermarks (in percentage) with the fraction of the checked training data (in percentage). The figure reveals that our *max-KNN-Shapley* is more effective in detecting instance-based watermarks than all other baselines.

Data Summarization. Data summarization aims to select a small representative subset from a massive dataset, which can retain a comparable utility to that of the whole dataset. This is a natural application of data importance, since we can directly reduce the dataset size by eliminating data of low importance.

We use a single hidden layer neural network trained on UCI Adult Census dataset. In Fig. 2c, we plot the change of prediction accuracy (in percentage) with the change of the fraction of data removed (in percentage). The figure reveals that the instances selected by the Shapley value–based data importance measures are more representative than the LOO-based measures. Though TMC-Shapley and G-Shapley can achieve slightly better performance than KNN-Shapley, our method still retains a high performance even after reducing 50% of the whole training set, which is notable.

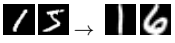
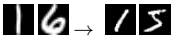
Apart from the quantitative results above, we provide the qualitative visualization of images drawn from Tiny ImageNet in Fig. 3, where we show the images of the highest Shapley value (*i.e.*, representative images), as well as the top 50 classes that their summarization belongs to. It is intriguing that a similar set of images (*e.g.*, dugong, espresso, monarch, goldfish) stand out as the most representative samples even when they are pre-processed using different feature extractors. Why these classes are more representative is an interesting open question that deserves further investigation. Another observation is the high diversity of the top 20 images displayed, which further corroborates the capability of our Shapley enriched method in producing a high-quality miniature for the original massive dataset.

Active Data Acquisition. Annotated data is often hard and expensive to obtain, particularly for specialized domains where only experts can provide reliable labels. Active data acquisition aims to facilitate the data collection process by automatically deciding which instances an annotator should label to train a model. To simulate this scenario, we start with a small training set, and then train a random forest to predict the score for new data based on their features. We repeat the process and iteratively add new data with highest data importance to the training set.

Here, we choose MNIST as our dataset and inject noise to part of it. We start with a small training set with 100 images and add Gaussian white noise into half of them. We use another 100 images to calculate the scores of training data and a held-out validation dataset of size 1000 to evaluate the performance. In Fig. 2d we plot the change of prediction accuracy with the number of added training points. Evidently, new data selected based on KNN-Shapley value improves model accuracy faster than all other methods.

Domain Adaptation. Domain adaptation aims to leverage the dataset from one domain for the prediction tasks in another domain. We will show that the data importance

Table 2: Domain adaptation between MNIST and USPS.

Method	MNIST → USPS 	USPS → MNIST 
KNN-Shapley	**31.70%** → **47.00%**	**23.35%** → **29.80%**
KNN-LOO	31.70% → 37.40%	23.35% → 24.50%
TMC-Shapley	31.70% → 44.90%	23.35% → 29.55%
LOO	31.70% → 29.40%	23.35% → 23.53%

measures will be useful for domain adaptation. Specifically, we first compute the importance of data in the source domain with respect to a held-out set from the target domain. We then train the model using only positive-valued points in source domain and evaluate the model in target domain.

We perform experiments on MNIST and USPS following the setups in Ghorbani *et al.* [7] and present the transfer results between the two. We first train a multinomial logistic regression classifier. We randomly sample 1000 images from the source domain as the training set, calculate the scores for the training data based on 1000 instances from the target domain, and evaluate the performance of the model on another 1000 target domain instances. The results are summarized in Table 2. As it shows, KNN-Shapley performs the best.

Summary of Results. Based on extensive empirical observations, we conclude that: (1) the KNN-Shapley-based method requires the minimal runtime compared with the rest approaches on large scale training data and models (some methods such as TMC-Shapley cannot even finish running within reasonable time); (2) for different ML applications (e.g. mislabeled data detection, watermark removal, data summarization active data acquisition, and domain adaptation), different variants of Shapley-based methods consistently outperform the leave-one-out–based methods; (3) the KNN-Shapley based methods including both *KNN-* and *max-KNN-Shapley*, always achieve the best or at least comparable performance compared to other Shapley approximation methods.

4.4. Comparisons of Different Embeddings

In Section 4.3 we provide results corresponding to the embeddings extracted by MobileNett [10] classifier pre-trained on ImageNet. In this section, we leverage different embeddings extracted by 4 other pre-trained classifiers for evaluation: ResNet18 [9], VGG11 [25], Inception-V3 [26], and EfficientNet B7 [27].

We provide part of results in Fig. 5 for selected applications and datasets. In each figure, there are clearly two groups of curves: one group for KNN-Shapley and the other for KNN-LOO. We can see that the utility of KNN-LOO is roughly the same as random, while the KNN-Shapley presents high utility for different applications. In conclusion, the difference induced by using different *embeddings* is marginal compared to using different *measures*. Furthermore, our KNN-Shapley based importance measure is insensitive to the selection of embeddings and can achieve superb

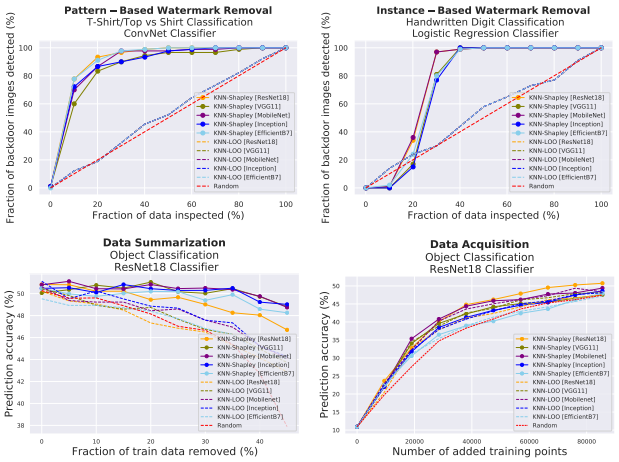


Figure 5: Comparisons of different embeddings on different applications and datasets.

performance without the need of carefully selecting feature extractors. A more comprehensive set of results are left to Appendix F.

5. Conclusion

This paper provides the first theoretical and large-scale empirical studies towards answering the fundamental questions about what method should be used for evaluating data importance and how to efficiently do so. Particularly, we prove that the Shapley-based method provides higher utility than a leave-one-out–based approach, in terms of evaluating the predictive power of the data importance as well as the data discrimination ability. Extensive experiments are conducted on five applications, showing that the Shapley-based methods outperform the leave-one-out–based ones in terms of both runtime and experimental performance. Specifically, the KNN-Shapley approach provides the most efficient solution and usually achieves the best or comparable performance among all. In addition, we are the first to leverage data importance approaches to perform watermark removal, which is a challenging task currently, and achieve promising results. This particular application would shed light on future research on watermark analysis and other related tasks.

Acknowledgement This work was performed under the auspices of the NSF grant #1910100 and U.S. Department of Energy by the Lawrence Livermore National Laboratory under Contract No. DE-AC52-07NA27344, Lawrence Livermore National Security, LLC. The views and opinions of the authors do not necessarily reflect those of the U.S. government or Lawrence Livermore National Security, LLC neither of whom nor any of their employees make any endorsements, express or implied warranties or representations or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of the information contained herein. LLNL-CONF-820505.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318. ACM, 2016.

[2] Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 1615–1631, 2018.

[3] Tiago Almeida, Jose Gomez Hidalgo, and Tiago Silva. Towards sms spam filtering: Results under a new dataset. *International Journal of Information Security Science*, 2:1–18, 01 2013.

[4] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526, 2002.

[5] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728*, 2018.

[6] Qingrong Chen, Chong Xiang, Minhui Xue, Bo Li, Nikita Borisov, Dali Kaarfar, and Haojin Zhu. Differentially private data generative models. *arXiv preprint arXiv:1812.02274*, 2018.

[7] Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. *arXiv preprint arXiv:1904.02868*, 2019.

[8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

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

[10] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017. cite arxiv:1704.04861.

[11] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gurel, Bo Li, Ce Zhang, Costas Spanos, and Dawn Song. Efficient task-specific data valuation for nearest neighbor algorithms. *Proceedings of the VLDB Endowment*, 12(11):1610–1623, 2019.

[12] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gurel, Bo Li, Ce Zhang, Dawn Song, and Costas Spanos. Towards efficient data valuation based on the shapley value. *arXiv preprint arXiv:1902.10275*, 2019.

[13] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *Ann. Math. Statist.*, 23(3):462–466, 09 1952.

[14] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1885–1894. JMLR. org, 2017.

[15] Ron Kohavi. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 202–207. AAAI Press, 1996.

[16] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1920–1929, 2019.

[17] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

[18] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[19] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *2009 IEEE 12th International Conference on Computer Vision*, pages 365–372, 2009.

[20] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. 2015.

[21] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[22] Yuval Netzer, Tiejie Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[23] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[24] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained models for natural language processing: A survey. *arXiv preprint arXiv:2003.08271*, 2020.

[25] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[26] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[27] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2019. cite arxiv:1905.11946Comment: Published in ICML 2019.

[28] Yu-Xiang Wang, Jing Lei, and Stephen E Fienberg. Learning with differential privacy: Stability, learnability and the sufficiency and necessity of ERM principle. Feb. 2015.

[29] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[30] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.