Private Data Valuation and Fair Payment in Data Marketplaces

Zhihua Tian[‡], Jian Liu[*], Jingyu Li[*], Xinle Cao[*], Ruoxi Jia[†], Jun Kong[§], Mengdi Liu[§], Kui Ren[*]

[‡]Zhejiang University and ZJU-Hangzhou Global Scientific and Technological Innovation Center,
[*]Zhejiang University, [†]Virginia Tech, [§] Zhejing Big Data Exchange Center
{zhihuat, liujian2411, jingyuli, xinle, kuiren}@zju.edu.cn, ruoxijia@vt.edu, {kongj, liumd}@zjdex.com

***Abstract*—Data valuation is an essential task in a data marketplace. It aims at fairly compensating data owners for their contribution. There is increasing recognition in the machine learning community that the Shapley value—a foundational profit-sharing scheme in cooperative game theory—has major potential to value data, because it uniquely satisfies basic properties for fair credit allocation and has been shown to be able to identify data sources that are useful or harmful to model performance. However, calculating the Shapley value requires accessing original data sources. It still remains an open question how to design a real-world data marketplace that takes advantage of the Shapley value-based data pricing while protecting privacy and allowing fair payments.**

In this paper, we propose the *first* prototype of a data marketplace that values data sources based on the Shapley value in a privacy-preserving manner and at the same time ensures fair payments. Our approach is enabled by a suite of innovations on both algorithm and system design. We firstly propose a Shapley value calculation algorithm that can be efficiently implemented via multiparty computation (MPC) circuits. The key idea is to learn a performance predictor that can directly predict model performance corresponding to an input dataset without performing actual training. We further optimize the MPC circuit design based on the structure of the performance predictor. We further incorporate fair payment into the MPC circuit to guarantee that the data that the buyer pays for is exactly the same as the one that has been valuated. Our experimental results show that the proposed new data valuation algorithm is as effective as the original expensive one. Furthermore, the customized MPC protocol is efficient and scalable.

***Index Terms*—Data marketplace, Fair payment, Data valuation**

I. Introduction

Data, a new factor of production, is now treated equally as land, labour, capital and technology. As a result, data commoditization has become an emerging trend: data buyers seek to purchase high-quality data, and data owners attempt to maximize data monetization. A *data marketplace* is a platform where users can buy and sell data. It facilitates data owners marketing, managing, and selling their data; it also enables data buyers to browse, compare, and purchase data.

Data marketplaces can be roughly classified into three categories based on what they sell and their *pricing mechanisms*: data-based, query-based, and model-based. Data marketplaces with data-based pricing sell data to buyers directly, for example Dawex [1], Datarade [2], Quandl [3] and Bloomberg [4]. Such marketplaces give data owners limited control over their data usages, and data buyers have to purchase the whole dataset even if they are only interested in particular information. Data marketplaces with query-based pricing charge buyers and compensate data owners on a per-query basis, e.g., Google Bigquery [5]. This type of marketplace can usually only allow simple queries; in particular, they are unable to support widely used machine learning (ML) based data analytics.

To address these issues, model-based pricing [6] has recently been proposed as an alternative to the two types of pricing mechanisms mentioned above. A model-based pricing mechanism valuates each data source according to its contribution to training a given ML model. For instance, a collection of multiple data sources is used jointly to train an ML model, which achieves certain performance, say classification accuracy 0.9. The model-based pricing mechanism assigns price among all data sources, so that each source receives a fair share for its contribution towards achieving the 0.9 accuracy. Most works have focused on leveraging the Shapley value as the metric to quantify the contribution of individual data sources [7], [6], [8]. Particularly, it *uniquely* satisfies several natural properties of an equitable data pricing mechanism: 1) it requires that the the contribution of a data source is sensitive only to how the model performance responds to the presence of the source; 2) it ensures that the contribution of data to predicting multiple test inputs is equal to the sum of contributions to predicting each individual test input; and 3) it distributes the full yield of an ML pipeline. Aside from the theoretical soundness, it has been demonstrated in [9], [8] that the Shapley value can reliably differentiate between high-quality and low-quality data.

Despite the appealing properties of the Shapley value, none of the existing data marketplace platforms supports its adaption. Some platforms only allow buyers to request samples to *qualitatively* understand the quality of the data that they intend to buy [2], [3]. Indeed, deploying SV-based data pricing mechanisms into the real-world data marketplace requires systematic innovations to tackle the following dilemmas.

- *Who performs the data valuation?* Calculation of the Shapley value must access all individual data sources. If the buyer performs the data valuation, the buyer has to access the data and as a result, the buyer no longer needs to buy

the data. If the data owners perform the valuation, they have to access the valuation algorithm and therefore, they can potentially manipulate their data to maximize the profit.
- *Who delivers first?* If the buyer delivers the payment first, a data owner might refuse to deliver the data or deliver inferior data. If the data owners deliver data first, the buyer might refuse to pay.

Firstly, introducing a broker between the buyer and the data owners does not resolve these two dilemmas. The broker could potentially collude with either the buyer or the data owners. Secondly, it may seem natural to resort to cryptographic techniques to resolve the first dilemma. For example, to address the first dilemma, data owners and the buyer can jointly run data valuation via *multiparty computation* (MPC) [10], such that the buyer can get SV without seeing the data. However, even computing SV in plaintext could be expensive because it requires retraining models on different subsets of data; calculating them via MPC remains an open problem. As for the second dilemma, one natural idea is to leverage *zero-knowledge contingent payment* (ZKCP) [11], [12], [13], [14], which allows fair payment of digital goods and payments over a blockchain. However, simply applying ZKCP cannot ensure that data owners use the same data for both MPC-based data valuation and ZKCP. As a result, a malicious data owner could use high-quality data to get a high SV in MPC-based data valuation, and then deliver inferior data in ZKCP. *Overall, while a combination of MPC and ZKCP provides a natural pathway to approach the dilemmas, it suffers significant computational issues and also exposes new vulnerabilities for a malicious user to game the system.*

Our contributions. In this paper, we propose the *first* framework to solve the aforementioned dilemmas, thus enabling private data valuation and fair payment in a data marketplace. The proposed framework includes a suite of sophisticated designs.

First of all, we **design an MPC-friendly algorithm for SV calculation**. Given n data sources, the exact SV calculation requires repeatedly training an ML model over every possible combination of data sources, leading to $\mathcal{O}(2^n)$ complexity. Although existing work leverages Monte Carlo-based algorithms to approximate the SV by training over polynomially many randomly chosen subsets, training ML models is still too heavy for MPC. Inspired by the recent advance in active learning, we propose to learn an ML *performance predictor* that directly predicts the performance of the model corresponding to a training set without performing actual training. Note that in existing data marketplaces [2], [3], sellers often pre-share a small portion of representative samples of their data to facilitate buyers' assessment of the quality and relevance of the data. Our performance predictor is learned in a supervised manner with pre-shared samples. The learned performance predictor can be implemented in MPC to support calculation of the SV of each dataset in a way that does not reveal the original data. Compared with the existing data marketplaces in which the buyers can only acquire a qualitative understanding of a dataset from pre-shared samples, our system makes a better use of pre-shared data by offering a quantitative evaluation of the dataset's contribution.

Second, we **optimize the MPC circuit design for the performance predictor** to further improve efficiency. It is well known that the complexity of MPC increases *quadratically* with the number of parties. Therefore, directly running MPC for data valuation in data marketplaces cannot support a large number of data owners. Recall that we directly predict the performance of each combination of data with a parameter model. The prediction process consists of an encoding phase that encodes each data sample into an embedding and a mapping phase that aggregates the embeddings to a vector and maps the vector to the performance. To that end, we optimize the MPC circuit such that it only requires all data owners to get involved in the mapping phase. The encoding phase can be achieved by data owners running the two-party computation (2PC) separately with the buyer. Since the mapping phase consists of only a few fully connected layers, which is efficient to implement via MPC, the cost of data valuation is nearly unaffected by the number of data owners.

Finally, instead of using ZKCP, we **innovatively incorporate fair payment into our MPC circuit**. In more detail, each data owner inputs an encryption key k together with her data D to MPC. Other than SV, the MPC protocol will also outputs $E(k, D)$ and $H(k)$ to the buyer, where $E()$ is a symmetric-key encryption scheme and $H()$ is a cryptographic hash function. If the buyer decides to buy the data, he issues a *hash-locked transaction* [15] on a blockchain, requiring the data owner to post the preimage of $H(k)$ to the blockchain to redeem the payment. Most cryptocurrencies like Bitcoin and Ethereum allow a payer to make a payment by specifying a condition that needs to be met in order for the money to be redeemed by the payee. A hash-locked transaction is such a condition, where a payment can be redeemed by presenting a preimage of a hash value. Then, the data owner can simply post k to the blockchain to redeem the payment; meanwhile, the buyer can decrypt $E(k, D)$ and obtain the purchased data. The MPC protocol ensures that the data being encrypted is exactly the data being valuated. We are the first to adopt the idea of incorporating fair payment into the MPC circuit to address the fair payment problem in data marketplaces.

We provide a full implementation of our data valuation algorithms as well as the MPC circuits. Our experimental results show that the proposed data valuation algorithms are as effective as the original expensive one. Furthermore, the customized MPC protocol is efficient and scalable.

Paper Organization. The rest of the paper is organized as follows. Section II presents the overview of our method. We provide our data valuation algorithms in Section III. The customized MPC protocol is presented in Section IV-B and we show how to incorporate fair payment into it in Section IV-C. Section V presents the experiments for evaluating the effectiveness of the data valuation algorithms as well as the efficiency of the MPC circuits. Section VI presents related work and Section VII concludes the paper.

Notations. We summarize the frequently used notations in Table I for the convenience of readers.

Table I: Summary of notations

Notation	Description
$\mathcal{P}$	data owner
$\mathcal{B}$	the buyer
u	data utility function
C	coalition of datasets
SV	Shapley values of all data owners
v_{sv}	Shapley value of one data owner
v_{loo}	Leave-one-out value of one data owner
G_f	feature extractor
f_{DS}	DeepSets model
$\mathcal{A}$	learning algorithm
$\mathcal{L}_{tr}$	training set
$\mathcal{L}_{val}$	validation set
$\mathcal{L}_{pub}$	public set
M	size of the train dataset
$\mathcal{S}_n$	a n secret sharing scheme
$\langle x \rangle$	shared variable
$\langle x \rangle_n$	shares calculated via n secret sharing
$\langle x \rangle_{i/n}$	individual share of $\langle x \rangle_n$ held by player i

II. Overview

We consider the common setting of a data marketplace: N data owners $\mathcal{P}_1, ..., \mathcal{P}_N$, holding datasets $D_1, ..., D_N$, respectively; a data buyer $\mathcal{B}$ wanting to know the Shapley value (SV) for each dataset, so that $\mathcal{B}$ can decide how much she should pay for data owners. In prior work [16], a broker is typically introduced to calculate the SV on behalf of $\mathcal{B}$. However, such a design relies on the honesty of the broker, which could be a strong assumption and challenging to meet realistically. In our framework, we do not assume the existence of a broker. On the other hand, we assume data owners and the buyer cannot be malicious simultaneously. Specifically, at least all data owners or the buyer are (is) honest. Malicious behaviors could occur in both data valuation and payment processes.

Following a setting proposed in [6], wherein the buyer has test data to verify the utility of a dataset, we have the same assumption and treat test data as the validation set denoted as $\mathcal{L}_{val}$. Meanwhile, we mainly focus on evaluating the value of unlabeled data as data sellers may prefer to sell features only, considering that label information is usually more sensitive than features [17], and labeling data is notoriously costly. Data owners and the buyer have different goals: each data owner shares her data with the buyer for compensation, and the buyer purchases datasets that are useful for her learning task from data owners.

Given the above setting and assumptions, we aim to solve the dilemmas of considering both data valuation and fair payment in data marketplaces:

The buyer can valuate the data without seeing the data; once the buyer decides to pay, the payment happens atomically.

Intuition. Inspired by the work of [18], we propose to approximate the mapping from a training set to the performance of the ML model trained on the set suing a parametric model, which is referred to as a *data utility model* (cf. Section III). With a well-trained data utility model, the process of calculating utility scores is in fact a prediction process, which can be efficiently executed via a customized MPC protocol (cf. Section IV). The SV can be calculated with the utility scores of different coalitions of data using Equation 2.

Figure 1 depicts the overview of our scheme considering both data valuation and fair payment. It roughly works as follows:

1) Each $\mathcal{P}_i$ pre-shares a small subset of data $d_i \subseteq D_i$ to the buyer $\mathcal{B}$. $\mathcal{B}$ then trains a data utility model U with the data she received from data owners.
2) Each $\mathcal{P}_i$ inputs her data D_i together with a encryption key k_i to MPC. $\mathcal{B}$ inputs (the parameters of) the data utility model U to MPC.
3) The MPC protocol outputs (SV_i, $E(k_i, D_i)$, $H(k_i)$)s to $\mathcal{B}$, where $E()$ is a symmetric-key encryption scheme and $H()$ is a cryptographic hash function.
4) $\mathcal{B}$ initiates a hash-locked transaction for each $\mathcal{P}_i$, wherein the amount of funds is based on SV_i. Each $\mathcal{P}_i$ claims the funds by putting k_i on-chain.
5) Each $\mathcal{P}_i$ gets the funds, and $\mathcal{B}$ gets D_i by decrypting $E(k_i, D_i)$ with k_i.

III. MPC-Friendly Data Valuation

In this section, we first introduce the notion of data value and explain in detail how to learn an ML performance predictor, which could output model performance corresponding to a training set in an MPC-friendly manner.

A. What is the Notion of Data Value

In the design of a data marketplace, multiple data owners contribute their data according to the format that buyers require. As different data contribute variously, the main challenge is how to compensate data owners fairly. A natural way is to allocate the revenue based on the usefulness of data from different sources for the buyer's task. Such usefulness is treated as the value of data. There has been a surge of research efforts to formalize the notion of data value [9], [8], [19], with the two most commonly used being Leave-one-out (LOO) [20] and the Shapley value [21].

Leave-one-out Method. For N data owners, each of them holding a dataset $D_i, i \in I = \{i, ..., N\}$, LOO quantifies the importance of D_i by measure its contribution to the rest of datasets:

$$v_{loo}(D_i) = u(\bigcup_{j \in I} D_j) - u(\bigcup_{j \in I \setminus \{i\}} D_j), \tag{1}$$

where u is a function that evaluates the utility of the data. We will formally define u later.

Shapley Value-based Method. The SV characterize the importance of each dataset inspired by cooperative game theory. A cooperative game is defined by a pair (I, u), where $I = \{1, ..., N\}$ denotes the set of all players and $u : 2^N \to \mathbb{R}$ is the utility function, which maps each possible coalition to a real

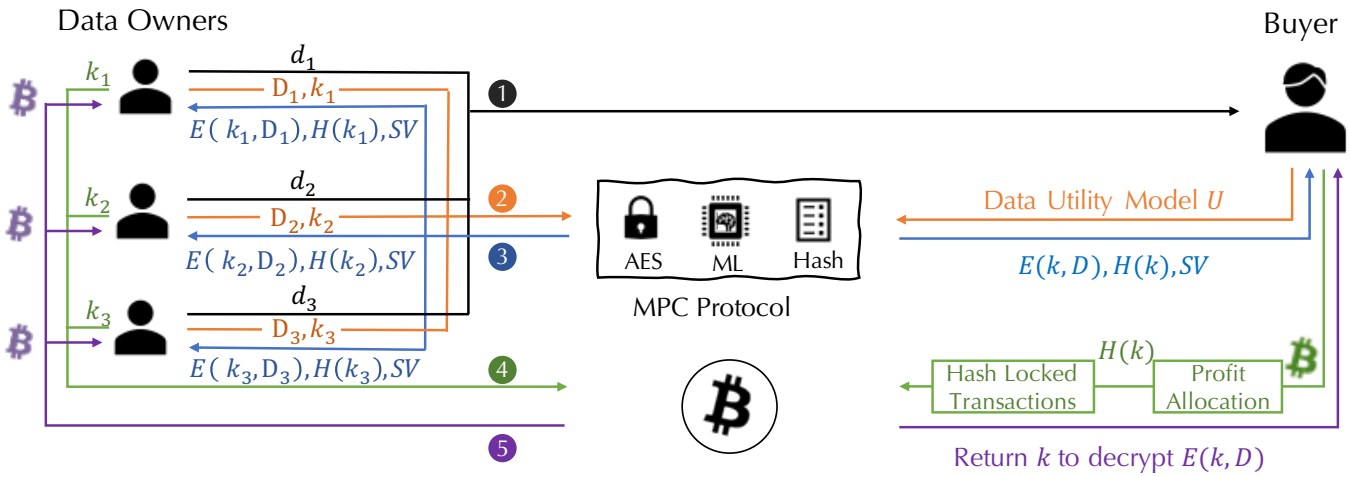


Figure 1: Overview of our proposed framework addressing both private data valuation and fair payment.

value indicating the payment of the coalition, i.e., collective payoff a set of players can gain by forming a coalition.

Given the utility function u, the SV of player i is calculated by averaging the marginal contribution of player i to all possible coalitions formed by other players $C \subseteq I \backslash \{i\}$:

$$v_{sv}(u, i) = \frac{1}{N} \sum_{C \subseteq I \backslash \{i\}} \frac{1}{\binom{N-1}{|C|}} [u(C \cup \{i\}) - u(C)] \quad (2)$$

Suppressing the dependency on u, we use $v_{sv}(i)$ to represent the SV of the player i.

Transforming the game theory concepts to data valuation, one can think of players as training data sources and the utility function $u(C)$ as a performance metric function, which measure the performance of the model trained on the set of training data C. Therefore, the SV of each source measures its contribution to training an ML model.

We present the formal definition of the data utility function u as below, which is also suitable for the LOO method. Given a learning algorithm $\mathcal{A}$, which takes a set of instances C as input and outputs a classifier $\hat{f} \leftarrow \mathcal{A}(C)$. The metric function u takes $\hat{f}$ as input and outputs the utility of C. In the context of machine learning, we often use test accuracy as the metric $u(\hat{f}, \mathcal{V}) = \frac{1}{|\mathcal{V}|} \sum_{(x,y) \in \mathcal{V}} \mathbb{1}[\hat{f}(x) = y]$ for a test set $\mathcal{V}$. With a potential stochastic learning algorithm $\mathcal{A}$ and a corresponding metric function u, the data utility function can be defined as $U_{\mathcal{A},u}(C) = \mathbb{E}_{\mathcal{A}}[u(\mathcal{A}(C), \mathcal{V})]$. We omit the subscript when the context is clear and simply write $U(C)$.

Although LOO has the advantage of low complexity, only $\mathcal{O}(N)$ where SV has $\mathcal{O}(2^N)$, prior works [7], [8], [19], [9] empirically show that SV can better reflect the utility of the data and identify bad and good quality data. Meanwhile, Jia et al. [22] theoretically prove that SV is more robust against inherent stochasticity of ML models induced by stochastic gradient descent, which produce inconsistent data value rankings across different runs. Furthermore, SV is theoretically proven to satisfy property of rigorous fairness: (1) if sellers i and j are equivalent in the sense of $v_{sv}(C \cup \{i\}) = v_{sv}(C \cup \{j\})$, $\forall C \subseteq I \backslash \{i, j\}$, then $s_{sv}(i) = s_{sv}(j)$, where $s_{sv}(i)(s_{sv}(j))$ is the SV of the player $i(j)$. That means two sellers who are identical in terms of the contribution to ML performance should have the same value. (2) $s_i = 0$ if $v_{sv}(C \cup \{i\}) = v_{sv}(C)$ for all $C \subseteq I \backslash \{i, j\}$. This means that sellers should receive zero pay-off if their data have no contributions to improving ML performance when combined with all the other data sources. Those reasons motivate us to adapt SV for data valuation.

B. Efficient Computation of Shapley Value

Exactly calculating the SV of a set of data requires retraining a model for every coalition of individuals. To solve such a problem, we propose to approximate the data utility function with a parametric model, referred to as the data utility model. Next, we explain in detail how we can build a data utility model. It consists of two phases: *training dataset construction* and *model training*.

1) Training dataset construction: As we described in Section II, each data owner $\mathcal{P}$ pre-shares a small portion of its dataset with the buyer $\mathcal{B}$; we denote such data collectively as $\mathcal{L}_{tr}$. The training dataset consists of pairs of data subsets and the corresponding utility scores, wherein each subset is sampled from $\mathcal{L}_{tr}$.

We describe the training dataset construction phase in Algorithm 1. Specifically, given $\mathcal{L}_{tr}$, $\mathcal{L}_{val}$ and a learning algorithm $\mathcal{A}$, $\mathcal{B}$ first randomly samples a subset $S_i \subseteq \mathcal{L}_{tr}$ (line 3) and uses it to train a model f_i with algorithm $\mathcal{A}$ (line 4). Next, $\mathcal{B}$ evaluates the utility of f_i on $\mathcal{L}_{val}$, denoted as $u(f_i, \mathcal{L}_{val})$. Then, the pair of S_i and $u(f_i, \mathcal{L}_{val})$ forms a training instance to learn the data utility model, where $u(f_i, \mathcal{L}_{val})$ is considered the label of S_i. Repeating the above procedure M times, we can get a set S_{DS} to serve as the training dataset for the data utility model.

Algorithm 1 requires the instances in $\mathcal{L}_{tr}$ to be labeled, which is not the case in our setting. To resolve this problem, a

Algorithm 1: Training Dataset Construction

Input: training set $\mathcal{L}_{tr}$; validation set $\mathcal{L}_{val}$; learning algorithm $\mathcal{A}$;
Output: utility learning dataset S_{DS} for training data utility model

1 Initialize value dataset $S_{DS} = \emptyset$
2 **for** $i = 1 \rightarrow M$ **do** `for each subset`
3 Sample a subset $S_i \subseteq \mathcal{L}_{tr}$
4 Train classifier $f_i \leftarrow \mathcal{A}(S_i)$
5 $u(f_i, \mathcal{L}_{val}) = \frac{1}{|\mathcal{L}_{val}|} \sum_{(x,y)\in\mathcal{L}_{val}} \mathbb{1}[f_i(x) = y]$
6 $S_{DS} = S_{DS} \bigcup (S_i, u(\mathcal{L}_{val}))$
7 **end**
8 **return** $S_{DS} = (X, U)$, *where* $X = (S_1, ..., S_M), U = (u_1, ..., u_M)$

straightforward method is to label the data before being shared to $\mathcal{B}$.

2) Data Utility Training: Next, we show how $\mathcal{B}$ trains the data utility model U. As each instance in the training dataset is a set, we adopt a canonical *set function model* - DeepSets [23] - to train U. Furthermore, we also need a *feature extractor* to extract the feature embedding from each instance in S_{DS}, so that the DeepSets model could map the set of feature embeddings to its corresponding utility score. To this end, we treat the data utility model U as a composition of a DeepSets model f_{DS} and a feature extractor G_f.

Algorithm 2 details the data utility training phase. We name it **Labeled Pre-sharing Method**. Note that different sets S_i and S_j, $i \neq j$ may contain the same data points, i.e., $S_i \cap S_j \neq \emptyset$. In practice, we first fix G_f to optimize f_{DS} (lines 4-5) and then fix f_{DS} to optimize G_f with a single batch/subset (line 6).

Algorithm 2: Data Utility Model Training

Input: training set $\mathcal{L}_{tr}$, valuation set $\mathcal{L}_{val}$;
Output: feature extractor G_f and DeepSets model f_{DS}

1 Initialize models: feature extractor G_f and DeepSets model f_{DS}
2 $S_{DS} = (X, U) \leftarrow \mathcal{B}$ run Algorithm 1 with inputs $\mathcal{L}_{tr}, \mathcal{L}_{val}$
3 **for** $i = 1 \rightarrow T$ **do** `for each epoch`
4 Fix G_f, extract the feature embedding E_S of dataset, $E_S \leftarrow G_f(X)$
5 Train the DeepSets value model f_{DS} on (E_S, U)
6 Fix f_{DS}, optimize G_f with (X, U)
7 **end**
8 **return** G_f, f_{DS}

*a) **Data Utility Training with Domain Adaption**:* Recall that we need each $\mathcal{P}_i$ to manually label its data before they are shared to $\mathcal{B}$, which requires extra human work. To that end, we propose to train the data utility model with *domain adaption* (DA) to eliminate the requirement of labels. The core idea is to train the data utility model on public data that are different from but related to data sources to be purchased and mitigate the impact of domain shift by domain adaption. Adopting domain adaptation is not rare, and such public data exists in many real-world applications. For example, when a service company aims to expand its business into new regions, it is natural to treat data from the original area as public data.

A domain adaption framework usually consists of three components: a feature extractor Ext which maps the inputs to a lower dimension feature embedding, a discriminator Dis which aims to distinguish between source domain and target domain data, and a regressor Reg that takes the output embedding of Ext as input and output predictions. The DA method has two goals: 1) map examples from two different domains to a common feature space; and 2) keep useful information for original tasks. They can be achieved by optimizing the GAN loss L_{GAN} and the prediction loss L_{DS} written as follows:

$$\begin{aligned}\min_{G_f}\max_{G_d} L_{GAN} = &\sum_{x\in\mathcal{L}_{pub}} \log G_d\left(G_f(x)\right) \\ &+ \sum_{x\in\mathcal{U}_{tgt}} \log\left(1 - G_d\left(G_f(x)\right)\right),\end{aligned} \tag{3}$$

$$\min_{f_{DS}}\min_{G_f} L_{DS} = \sum_{i=1}^{M} \left\| U_s\left(G_f\left(S_i\right)\right) - u_i \right\|^2. \tag{4}$$

To leverage domain adaption to train a data utility model $U = f_{DS} \circ G_f$ useful for evaluating data utility, we treat G_f, f_{DS} as Ext, Reg separately and introduce a model G_d that serves as a discriminator Dis to help training G_f. The training process of the data utility model U is integrated in the DA training process.

In more detail, we assume the buyer $\mathcal{B}$ has a public dataset $\mathcal{L}_{pub}$ from the source domain, which can be seen as the training set $\mathcal{L}_{tr}$ to construct S_{DS}. Meanwhile, each data owner $\mathcal{P}_i$ sends a subset of unlabeled data $d_i \in D_i$ to the buyer. The data consolidation $\mathcal{U}_{tgt} = \{d_1, ..., d_N\}$ is treated as samples from the target domain.

The workflow to train the data utility model with domain adaptation is depicted in Algorithm 3. We name it **Unlabeled Pre-sharing Method**. In practice, we train k steps of general domain adaptation training (line 4-6) followed by one step of utility training (line 7-9). Note that our method can be combined with any state-of-the-art DA frameworks, and we use CyCADA [24] in this paper.

We discuss how to determine the number of pre-shared data. More pre-shared data leads to training a better utility model but at the same time introduces the potential risk that the buyer gets enough data for free. To that end, data owners first pre-share a small amount of data, say 1%, with the buyer. The buyer makes public the utility learning dataset S_{DS} as well as the training process of the utility model, such as the initial weight, hyper-parameters, gradients, etc. The data owner can verify the utility model with public information and re-share more data if she finds that the model generalizes badly on her local data (e.g., the predicted utility score on her local data is significantly different from that on pre-shared data.). The experiments in Section V-E show that 4% of pre-shared data is enough to train a good utility model, and it is almost impossible for the buyer to learn a good model for her original task using aggregated pre-shared data.

Algorithm 3: Data Utility Model Training with DA

Input: public dataset $\mathcal{L}_{pub}$ serves as the training set $\mathcal{L}_{tr}$, valuation set $\mathcal{L}_{val}$ and the unlabeled dataset $\mathcal{U}_{tgt}$ from target domain

Output: feature extractor G_f and DeepSets model f_{DS}

Initialize models: feature extractor G_f, discriminator G_d and DeepSets model f_{DS}
$S_{DS} = (X, U) \leftarrow \mathcal{B}$ run Algorithm 1 with inputs $\mathcal{L}_{pub}, \mathcal{L}_{val}$
for $i = 1 \rightarrow T$ **do** `for each epoch`
  for k *steps* **do**
    Train G_f and G_d with $(\mathcal{L}_{pub}, \mathcal{U}_{tgt})$ via optimizing Equation 3
  end
  Fix G_f, extract the feature embedding E_S of dataset, $E_s \leftarrow G_f(X)$
  Train the DeepSets model f_{DS} on (E_s, U)
  Fix f_{DS}, optimize G_f with (X, U) via optimizing Equation 4
end
return G_f, f_{DS}

IV. Customized MPC

In this section, we further optimize the MPC circuit design such that the cost of data valuation increases almost unaffected by the number of parties. We first introduce the background of MPC in Section IV-A. Then, we present in detail how the MPC circuit is being designed in Section IV-B. We finally show how to achieve fair payment while ensuring that the data being encrypted is exactly the data being valuated in Section IV-C.

A. Background: Multiparty Computation.

MPC is a type of protocol that allows parties to jointly compute a function $f(x_1, ..., x_n) \leftarrow \mathcal{F}(x_1, ..., x_n)$ over their input while protecting the privacy of those inputs. It offers the same security guarantee achieved by a trustworthy outside party who receives input from all parties, computes, and returns the corresponding outputs to them so that all parties learn no more except the information that can be inferred from the output and their input. Generally speaking, MPC protocols are designed to achieve the following two goals: input privacy and correctness. Input privacy implies that engaging in the protocol does not leak any information about their private data. And correctness comes when the honest parties are guaranteed to compute the correct output or abort if they find an error.

Most MPC protocols make use of the idea of secret sharing, which distribute a secret among a group of parties and each party is allocated a share of that secret. The protocol takes the shares as input and, after a series of calculations, outputs the shares of the result. The result can only be reconstructed when a sufficient number of shares are combined. Two types of secret sharing schemes are commonly used: Shamir secret sharing and additive secret sharing. Shamir secret sharing is also known as t-out-of-n secret sharing, which means only when getting the collection of t or more shares can reconstruct the secret, while additive secret sharing needs all shares to reconstruct the secret. MPC protocols have been implemented in several systems with secret sharing schemes, where the most popular one is SPDZ [25], which is based on additive secret sharing.

B. Private Data Valuation

Running MPC for data valuation has two challenges: (1) the data utility model contains some operations (e.g., comparison) that cannot be supported efficiently by MPC, and (2) the complexity of MPC increases quadratically with the number of parties. Next we show how we overcome these challenges.

MPC-friendly operations. Recall that the data utility model is a composition of a DeepSets model and a feature extractor. The feature extractor contains ReLU and sigmoid, which are slow when running inside the MPC circuit. Firstly, we use the square function as the activation layer, a common practice to substitute ReLU, which has demonstrated only a slight decrease in accuracy for Neural Network models in [26], [27]. Notice that that we also use the square function during training to prevent the performance drop caused by the change of activation function. Secondly, as the sigmoid function is used in the last layer mapping the output to the values in a range $[0, 1]$, we detach it from MPC and run it in plaintext after getting the output of the MPC circuit, which saves much time while having no accuracy loss.

Optimizations with 2PC. We decompose the DeepSets model f_{DS} into three parts $f_{DS}^{trans}, Readout, f_{DS}^{network}$, i.e.,

$$f_{DS} = f_{DS}^{network}(Readout(f_{DS}^{trans})),$$

where f_{DS}^{trans} and $f_{DS}^{network}$ are two fully connected layers with different parameters, and Readout is a function to aggregate the representations. U's forward propagation process can be described as follows:

- Encode each instance $x_m \in C$ into a low-dimension feature embedding $G_f(x_m)$;
- Transform each embedding $G_f(x_m)$ into some representations $\varphi(x_m)$ by f_{DS}^{trans}, i.e., $\varphi(x_m) = f_{DS}^{trans}(G_f(x_m))$;
- Aggregate the representations of all embeddings and input the result into $f_{DS}^{network}$ to get the utility score.

Notice that the feature extractor G_f and f_{DS}^{trans} only involve one data owner. Thus they can be done via 2PC circuits. Although the remaining process ($f_{DS}^{network}$) still requires an MPC circuit, it consists of only a few fully connected layers, which is more efficient to implement compared with convolution layers in G_f and f_{DS}^{trans} for image data. Meanwhile, note that the $Readout$ function is a simple permutation-invariant function such as summation or average. We use average here.

To scale to more data owners, we run most of the operations in 2PC instead of MPC. To this end, we transfer a big portion of the MPC circuit to 2PC circuits while keeping the same level of security. The core idea is to share the secret shares of the output of the 2PC protocol with m ($m > 2$) data owners and treat them as input to the MPC protocol. We denote a shared variable x as $\langle x \rangle$ and all shares calculated via n secret sharing scheme are referred as $\langle x \rangle_n$. The individual share of $\langle x \rangle_n$ held by $\mathcal{P}_i$ as $\langle x \rangle_{i/n}$. The details of the conversion from $\langle x \rangle_2$ to $\langle x \rangle_n$ are shown in Protocol 4. For simplicity, we

Protocol 4: Conversion of 2 shares calculated via 2 secret sharing scheme to n shares

Input: $\mathcal{P}_i$ holds $\langle x\rangle_{1/2}$, $\mathcal{P}_j$ holds $\langle x\rangle_{2/2}$, a n secret share scheme $\mathcal{S}_n$
Output: n shares $\langle x\rangle_n$ of x held by $\mathcal{P}_i, ..., \mathcal{P}_n$, separately, where $1 \leq i, j \leq n$

1 $\mathcal{P}_i$ runs $\{s_1^0, ..., s_n^0\} \leftarrow \mathcal{S}_n(\langle x\rangle_{1/2})$, and send $\{s_1^0, ..., s_{i-1}^0, s_{i+1}^0, ..., s_n^0\}$ to other players
2 $\mathcal{P}_j$ runs $\{s_1^1, ..., s_n^1\} \leftarrow \mathcal{S}_n(\langle x\rangle_{2/2})$, and send $\{s_1^1, ..., s_{j-1}^1, s_{j+1}^1, ..., s_n^1\}$ to other players
3 **for** $k \in \{1, ..., n\}$ **do**
4 $\quad$ $\mathcal{P}_k$ run $\langle x\rangle_{k/n} = s_k^0 + s_k^1$ to get the individual share
5 **end**

present the conversion of $\langle x\rangle_2$ held by $\mathcal{P}_i$ and $\mathcal{P}_j$ to $\langle x\rangle_n$ held among data owners $1, ..., n$, where $i \neq j$ and $1 \leq i, j \leq n$. $\mathcal{P}_i$ and $\mathcal{P}_j$ first share the shares of $\langle x\rangle_{1/2}$ and $\langle x\rangle_{2/2}$ with other data owners (line 1-2), who can directly add the shares they received to obtain the individual $\langle x\rangle_{k/n}, k \in \{1, ..., n\}$.

In practice, we adopt a widely-used system SPD$\mathbb{Z}_{2^k}$ [28] to implement our protocol, which build MPC protocol with additive secret sharing. Thus, only collecting all shares can reconstruct the secret x.

C. Fair Payment

The fair payment is achieved using the technique of *hash-locked* transactions.

The hash-locked transaction [15] is build upon the Bitcoin system. A hash-locked transaction can be redeemed only after the redeemer provides the preimage of a given hash. Specifically, to create such a transaction, the payer attaches the value of the hash h, such that $h = H(m)$ (where H is a cryptographic hash function) for some m, and a script that specifically asks for the preimage of h to the transaction. And the transaction can only be finalized by providing a preimage m, which hash is exactly h. Meanwhile, combined with the technique of time locks, it guarantees that the funds return to the payer if not redeemed after the deadline, creating bidirectional micro-payment channels between the payer and the receiver [29]. The use of hash time-locked contracts allows to securely transfer Bitcoins between parities while minimizing the transactions stored on the blockchain, gaining a lot of popularity in the Bitcoin system.

Adopting the hash-locked transactions in our setting, each data owner $\mathcal{P}_i$ are required to calculate $E(k_i, D_i)$ and $H(k_i)$, where $E()$ is a symmetric-key encryption scheme, k_i is an encryption key and $H()$ is a cryptographic hash function. If the buyer is willing to purchase data from $\mathcal{P}_i$, she issues a *hash-locked* transaction [15] on the blockchain, which can only be redeemed by providing the preimage of $H(k_i)$. $\mathcal{P}_i$ then post k_i to the blockchain to redeem the payment. After getting k_i, the buyer can decrypt the encryption to obtain the purchased data.

ZKCP uses *zero-knowledge proof* (ZKP) to guarantee the correctness of the data and the encryption key. Specifically, each $\mathcal{P}_i$ is required to provide the following ZKP:

- $E(k_i, D_i)$ is the encryption of the data that the buyer intents to buy;
- $H(k_i)$ is the hash of encryption key that can be used to decrypt $E(k_i, D_i)$.

However, it does not guarantee that the encrypted data is exactly the evaluated data.

In our work, the "proof" is inherent inside the MPC circuit. Specifically, treat the MPC circuit as a black box, which takes data D, the encryption key k, as well as the data utility model U as inputs and outputs $E(k, D)$, $H(k)$ together with SVs. In other words, $E(k, D)$ and $H(k)$ are obtained collaboratively by each data owner and the buyer via MPC. As the MPC protocol is secure against malicious players, it guarantees that no one can modify the output [28]. Thus, it is guaranteed that the data being encrypted is exactly the data being evaluated, and the encryption key is exactly the preimage of the hash value.

V. Experiments

In this section, we conduct experiments to evaluate the effectiveness, scalability, and robustness of our approach to data valuation. We also evaluate the efficiency for the MPC circuit by simulating the scenario of domestic transactions, where data owners and the buyer in the same country, and cross-border transactions (e.g., China and America), where data owners and the buyer in the different countries.

A. Experimental Setup

We conduct experiments on 5 datasets covering census records: *a9a* [30] and images: *MNIST*[31], *USPS* [32], *CIFAR-10* [33], *STL-10* [34] as a benchmark.

Experiments for Data valuation All experiments are implemented at least 3 times on a single server equipped with a 2.5 GHz CPU and Tesla V100 GPU. For the sake of efficiency, we implement the prediction process of the data utility model in plaintext. Meanwhile, when experiments are simulated on more than 8 data owners, we approximate SV using a sampling-based approach that requires $O(NlogN)$ samples to achieve the desired approximation error [35]. We adapt the Logistic Regression algorithm as the proxy algorithm to construct the training dataset (cf. Algorithm 1) for efficiency.

Experiments for MPC circuit Each participant is simulated as a single server equipped with a 2.5 GHz Intel CPU and 32 GB RAM. The bandwidth is set to 100 Mbps. We implement the MPC protocol with the MP-SPDZ library [36], which includes more than 30 variants of the MPC protocol with the same Python-based high-level programming interface. Specifically, we implement with the SPD$\mathbb{Z}_{2^k}$ protocol [28]. The data encryption and hash are implemented with the Bristol Fashion circuits [37], which have also been implemented well in the MP-SPDZ library.

B. Effectiveness of data valuation algorithms

We start by exploring the effectiveness of our data valuation algorithms. We bootstrap dataset to synthesize data owned by data owners and the buyer. For simplicity, we call the data

Dataset	Number of Data Owners	Size of Group
a9a	40	400
MNIST	60	300
USPS	50	100
CIFAR-10	40	400

Source Domain	Target Domain	$\lvert\mathcal{L}_{pub}\rvert$	Number of Data Owners	Size of Group
MNIST	USPS	7 000	50	100
STL-10	CIFAR-10	4 000	40	400

Table II: Data division for the labeled pre-sharing method (Up) and the unlabeled pre-sharing method (Low).

owned by each data owner as a **group**. Each group can be seen as a data owner who wants to sell to the broker. Unless otherwise indicated, each data owner pre-shares 10% data with the buyer to train the data utility model. We also randomly selected 300 samples from the remaining training dataset for the target domain as a validation set $\mathcal{L}_{val}$. The experimental setting for data division is shown in Table II.

During the experiments, we manually add different noises to datasets owned by each data owner. Specifically, for the *a9a* dataset, which is a tabular dataset and all features are binary variables, we assign to each group a probability $p \in [0, 1]$ such that it has the probability $1 - p$ to flip the feature values. For the data owner $\mathcal{P}_i, i \in \{1, ..., N\}$, $p_i = \frac{i-1}{N}$. We add Gaussian noise with scale $\sigma_i = 1+9\cdot\frac{i}{N}$ for each $\mathcal{P}_i$ for the other digital datasets.

We show the effectiveness by calculating the correlation coefficient [38] of the rank of noise magnitude and the rank of SV. The results are presented in Figure 2. It shows that SV calculated in our method has a coefficient bigger than 0.5 in all settings. Although the data quality is not strictly related to added noise, the data with more noise tend to be low quality. Hence, a high coefficient implies that our method effectively evaluates data quality.

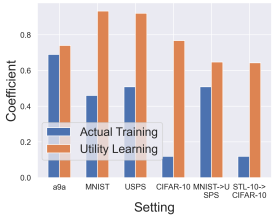


Figure 2: Correlation coefficient of the rank of noise magnitude and the rank of SV.

Next, we conduct experiments of removing data in the same group until only one group remains according to the SV calculated in advance and observe the test accuracy in $\mathcal{L}_{val}$ of the model trained with the remaining data. Intuitively, a better SV estimate can better identify the importance of each group. Therefore, when the data with the highest (lowest) SV estimates are removed, a better data value estimation method would lead to a faster (slower) performance drop. The results are depicted in Figure 3 and Figure 4.

It shows that our methods perform well in identifying the quality of data groups and have similar performance to calculating SV via actual training. Notice that the accuracy of the model trained on the a9a dataset increases with the removal of low-SV data, probably because we add too much noise to the data, making it difficult to train an effective model.

C. Scalability of data valuation algorithms

Next, we vary the number of data owners and the size of each group to verify the scalability of our methods in data removal tasks. For simplicity of presentation, we propose an intuitive metric named *effectiveness score* to represent the effectiveness of our methods. The metric is calculated as follows:

$$a = \begin{cases} \sum_{t=1}^{T} (acu_l^t - acu_r^t) \cdot \frac{1}{T} \ \textit{if Removing low-value data}, \\ \sum_{t=1}^{T} (acu_r^t - acu_h^t) \cdot \frac{1}{T} \ \textit{if Removing high-value data}, \end{cases} \tag{5}$$

where T is the number of removal times, acu represents accuracy, the subscripts l, r, h represent the removal of low SV data, the random removal and the removal of high SV data. Intuitively, a can be treated as the area between two accuracy lines caused by data removing according to SV and randomly removing. A positive a represents that our data valuation method is effective. The results are presented in Table III. All of them show that our methods are effective even with different numbers of data owners and different sizes of groups, explaining their good scalability.

D. Robustness of data valuation algorithms

We conduct a series of experiments to verify the robustness of our methods. Specifically, given some data owners, who own imbalance datasets or adversarial datasets, we experimentally prove that our data valuation methods still have a good performance. We further consider two types of malicious data owners: 1) data owners who intentionally select good data to pre-share and 2) data owners who share incorrect labels to train the data utility model for the labeled pre-sharing method. A malicious data owner may pre-share "poisoned" data during the sample step, specifically designed to allow the data owner to lower the value of other data owners' data. However, developing such an attack is beyond the scope of this paper, and we leave it for future work.

*a) **SV for Imbalanced Dataset**:* We artificially simulate data owners who own class-imbalance datasets for *CIFAR-10* and experiment with both the labeled pre-sharing method and the unlabeled pre-sharing method. We distribute data to data owners according to the Dirichlet distribution [39]. Specifically, for each class, we randomly generate a set of real values $\alpha \in [20, 100]$ with probability 0.2 and generate a set of real values $\alpha \in [80, 100]$ with probability 0.8. We then draw a distribution sample $p^i = (p_1^i, .., p_N^i)$ from Dirichlet distribution Dirichlet$(\alpha_1^i, ..., \alpha_N^i)$ where i represents the i-th class for *CIFAR-10* and N represents the number of data owners. We distribute p_j^i to data owner $\mathcal{P}_j$ for data of class

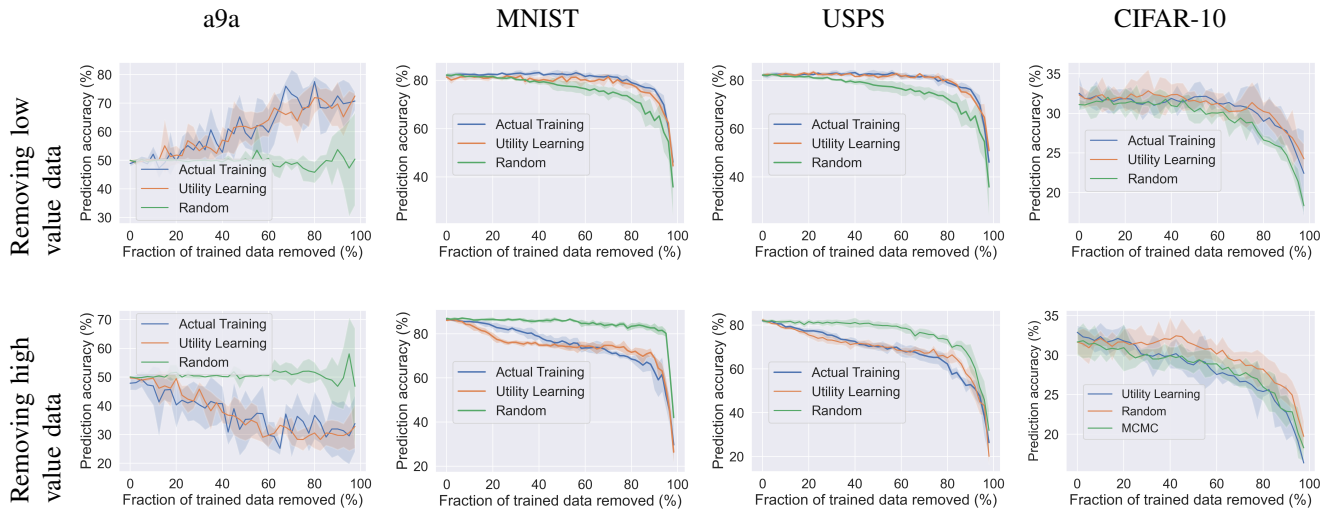


Figure 3: Experiments for labeled pre-sharing method. Removing data in ascending order (Up) or descending (Low) order of SV.

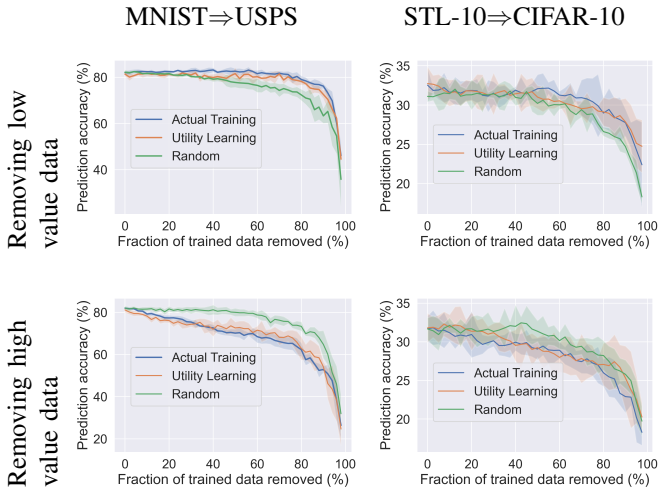


Figure 4: Experiments for the unlabeled pre-sharing method. Data removal in ascending order (Up) or descending order (Low) of SV.

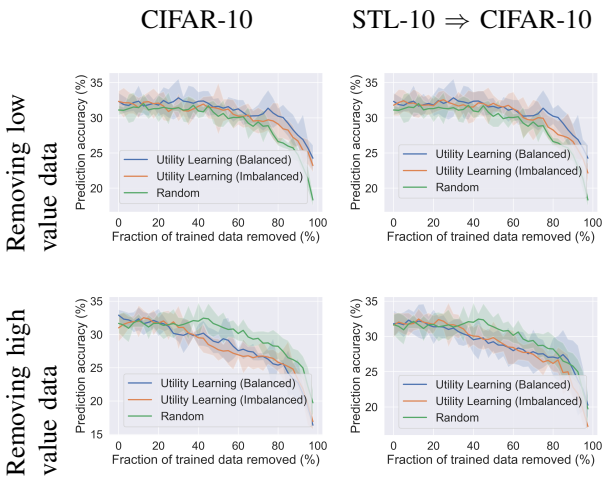


Figure 5: Models' validation accuracy vs. the proportion of removed instances for different settings. The accuracy line for the randomly removing strategy is generated with the balanced dataset.

i. Finally, the proportion of class j in D_j is $p_j^i / \sum_{k=1}^{10} p_j^k$. In our experiments, we simulate 40 data owners, and each data owner holds a dataset with different sizes $|D_i| \in [300, 500]$. We also simulate 40 data owners who hold balanced datasets with the same size for experiments as comparison. The results for labeled pre-sharing method are shown in Figure 5(a) and the results for unlabeled pre-sharing method are shown in Figure 5(b). The accuracy line for imbalanced datasets is similar to that for balanced datasets, demonstrating that our methods are robust against imbalanced datasets.

*b) **SV for Adversarial Dataset**:* Given some small intentional feature perturbations, an adversarial example can cause a machine learning model to make a false prediction. However, training with adversarial examples could improve the robustness of an ML model. With more adversarial examples are added to the validation dataset, the adversarial dataset is expected to become more valuable. We artificially constructed a data owner who owns adversarial examples (we call such a data owner the *adversarial data owner*), while other data owners own benign examples. The public dataset contains only benign examples.

All experiments are conducted under the setting of 8 data owners each hold a dataset with 1 000 instances. We vary the adversarial-benign mixing ratios to synthesize different validation datasets and observe the utility change of the adversarial data owner. We adapt the popular adversarial attack algorithm, namely Projected Gradient Descent (PGD) [40] to generate adversarial samples. In more detail, we use the PGD-20 attack in our experiments. We set the $\epsilon = 0.3$ for *MNIST*

setting	Removing low value data					Removing high value data					setting	Removing low value data					Removing high value data				
	dataset size	Number of data owners				dataset size	Number of data owners					dataset size	Number of data owners				dataset size	Number of data owners			
		10	20	30	40		10	20	30	40			30	40	50	60		30	40	50	60
a9a	100	0.235	0.151	0.254	0.357	100	0.023	0.038	0.060	0.022	MNIST	100	0.043	0.016	0.033	0.024	100	0.089	0.120	0.181	0.283
	200	0.276	0.161	0.358	0.262	200	0.017	0.038	0.043	0.008		200	0.052	0.014	0.010	0.027	200	0.190	0.158	0.255	0.156
	300	0.368	0.129	0.248	0.178	300	0.020	0.086	0.032	0.050		300	0.017	0.029	0.014	0.020	300	0.260	0.056	0.240	0.114
	400	0.274	0.378	0.255	0.160	400	0.048	0.051	0.028	0.034		400	0.017	0.004	0.033	0.016	400	0.109	0.260	0.079	0.128
		10	20	30	40		10	20	30	40			30	40	50	60		30	40	50	60
CIFAR-10	400	0.024	0.016	0.007	0.014	100	0.019	0.018	0.004	0.005	MNIST⇒ USPS	100	0.063	0.059	0.043	0.015	100	0.024	0.009	0.013	0.044
	500	0.020	0.007	0.038	0.042	200	0.014	0.020	0.052	0.013		200	0.017	0.008	0.028	0.025	200	0.019	0.028	0.014	0.074
	600	0.015	0.034	0.023	0.047	300	0.022	0.021	0.025	0.016		300	0.014	0.024	0.046	0.008	300	0.041	0.019	0.059	0.006
	700	0.004	0.046	0.027	0.013	400	0.039	0.032	0.029	0.026		400	0.044	0.002	0.009	0.009	400	0.023	0.002	0.001	0.020

Table III: Experiments for the setting of different data owners and each data owner hold different number of data points.

and *USPS* dataset and $\epsilon = 0.031$ for *CIFAR-10* dataset [41].

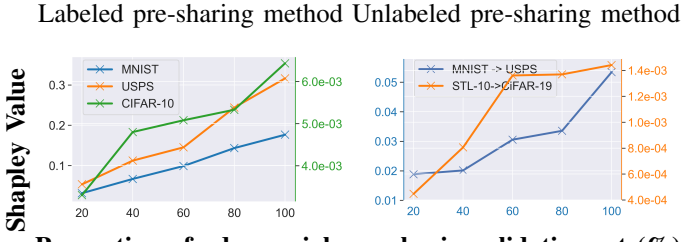


Figure 6: Shapley value for adversarial data owner VS. The proportion of adversarial samples in the validation set.

The results are presented in Figure 6. It shows that the SV of the adversarial data owner increases as the validation data become more adversarial, consistent with our expectations. Meanwhile, the SV for *CIFAR-10* is smaller than *MNIST* and *USPS*, because the learning algorithm $\mathcal{A}$ for *CIFAR-10* is unstable during the training dataset construction phase. In other words, the validation accuracy of datasets with different utilities may be close, resulting in the data utility model cannot distinguish the utility of datasets well. However, such problem can be resolved by running the learning algorithms multiple times or adopting a more robust learning algorithm when constructing the training set $\mathcal{L}_{tr}$.

*c) **SV for Intentionally Selected Dataset**:* Recall that each data owner pre-shares a proportion of data with the buyer, who exploits these data to train the data utility model. However, what if data owners who own mostly low-SV data intentionally select high-SV data for pre-sharing to bias the distribution of the training set? To verify the robustness of our method against such activity, we conduct experiments to artificially simulate such data owners and verify the robustness of our methods. For simplicity, we call data owners who intentionally select good data to pre-share as *dishonest data owner* and those who randomly select data to pre-share as *honest data owner*. We simulate 8 data owners each holding a dataset with 1 000 instances. Dishonest data owners are simulated by adding Gaussian noise with $\sigma = 10$ to its data, but still pre-share clean data without noise. We also vary the number of dishonest data owners to verify the performance of our methods under majority dishonest, i.e., most of the data owners are dishonest.

Figure 7 (a) presents the SV for minority dishonest data owners, and Figure 7 (b) presents the SV for majority dishonest data owners. It shows that all dishonest data owners with low-SV data have negative SV, and honest data owners with high-SV data have positive SV. We conjecture it is probably because the data utility model can still learn the useful feature from the natural variance of data in the distribution. Furthermore, the results also show that the unlabeled pre-sharing method is more sensitive for dishonest data owners than the labeled pre-sharing method, probably because the data in the source domain enrich the variety of training data for the data utility model.

*d) **SV for Noisy Labeled Dataset**:* We then conduct experiments to explore the influence of noisy labels of the labeled pre-sharing method with the *MNIST* dataset. For a label that is originally assigned as class i, we add noise using the following strategy:

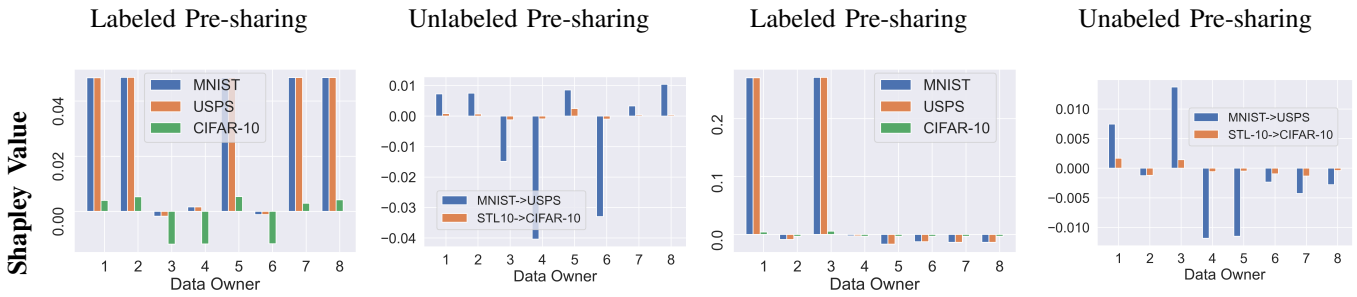
- with probability p, it keeps unchanged;
- with probability $1 - p$, it changed to the jth class that is picked uniformly at random.

We simulate 60 data owners and each data owner has a class-balanced data set of 300 instances. Figure 8 shows the changes in the effectiveness score 5 when p ranges from 0.6 to 1.0. It shows that mislabeled data do degrade the effectiveness of the utility model. However, our method can still evaluate the utility of data even if 40% data points are mislabeled. Meanwhile, the effectiveness score of removing high-SV data bigger than that of removing low-SV data, probably because high-SV data contributed more to train a better model, and removing more high-value has a greater impact on accuracy.

E. Proportion of Pre-shared Data

The number of pre-shared data needs to be appropriately large to train a data utility model that can generalize well to unseen data points and subsets. However, pre-sharing too much data to the buyer increases privacy leakage risk while having no compensation. To avoid such threats for data owners, training a practical data utility model while pre-sharing as little data as possible is needed.

Figure 9(a) depicts the results of pre-sharing different proportion of data to the buyer. We experiment with the *MNIST* dataset for the labeled pre-sharing method and *USPS⇒MNIST* for the unlabeled pre-sharing method. We simulate 10 data owners, each of them holds 100 instances. The proportion of pre-shared data ranged from 4% to 20%. We find that pre-sharing more data do perform better in training a



(a) Specify data owners $\mathcal{P}_3$, $\mathcal{P}_4$, $\mathcal{P}_6$ as dishonest **(b) Specify data owners $\mathcal{P}_2$, $\mathcal{P}_4$, $\mathcal{P}_5$, $\mathcal{P}_6$, $\mathcal{P}_7$, $\mathcal{P}_8$ dishonest**

Figure 7: Shapley value for data owners. Dishonest data owners who intentionally select good data to pre-share.

practical utility function. However, only pre-sharing 5% data is still enough, which means each data owner is only required to pre-share five samples with the buyer. Meanwhile, we also conduct experiments to compare the utility of data that the buyer gets without extra costs (i.e., the public dataset and the received pre-shared data) with all those data that she intends to buy. We train models with four different datasets:

1) public dataset,
2) labeled pre-shared dataset, which is consistent with our labeled setting,
3) public dataset plus unlabeled pre-shared dataset, which is consistent with our unlabeled setting,
4) public dataset plus full unlabeled data.

The results are depicted in Figure 9 (b). In the labeled setting where the buyer only has the validation set, training with 5% of pre-shared data can only train a model with about 60% accuracy. In the unlabeled setting where the buyer has access to a public dataset, with an extra 5% of pre-shared data, the buyer trains a model of 75% accuracy and about 10% of marginal contribution beyond only training with the public dataset. However, the marginal contribution of complete data can achieve above 35% accuracy. Combined with the results in Figure 9(a), it shows that we can train a utility function with data of low marginal contribution over what the buyer owns and the public dataset.

F. Experiments for MPC circuit

We evaluate the performance of the MPC circuit for SV calculation, data encryption, and hashing. Recall that we optimize our MPC circuit by replacing partial MPC circuit with 2PC circuit (cf. Section IV-B). The data encryption and key hashing process is also implemented using a 2PC circuit. For simplicity, we name the process of running 2PC circuit as **2PC part** and running MPC circuit as **MPC part**. We start by analyzing the 2PC part which takes most of the time when the number of data owners is relatively small.

1) Performance of the 2PC Part: We use AES-256 as the encryption algorithm and SHA-256 as the hash algorithm, which is widely used on the blockchain for fair payment. We only evaluate the performance of the online phase of the SPD$\mathbb{Z}_{2^k}$ protocol in our experiments.

*a) **Running time** :* Figure 10 shows the running time of 2PC circuits with different number of data points for *MNIST* and *CIFAR-10*. As the prediction processes of utility models trained by labeled pre-sharing method and the unlabeled pre-sharing method are the same, we do not distinguish them here. The running time almost linearly grows with the number of instances. The 2PC part in a cross-border transaction, which is more time-consuming than a domestic transaction, costs no more than 10 hours for 2 000 *MNIST* instances and 16 hours for 2 000 *CIFAR-10* instances. In a domestic transaction, running the 2PC part for 2 000 *MNIST* instances takes no more than 4 hours. For 2 000 *CIFAR-10* it costs no more than 5 hours.

*b) **Communication cost**:* We also evaluate the communication cost of the 2PC part. The results is shown in Figure 11(a). It show that communication cost grows linearly with the number of instances, which corresponds to the change of running time in Figure 10.

2) Performance of the MPC Part: The MPC part consists of the remaining prediction process of the data utility model (i.e. Average and $f_{DS}^{network}$), which mainly consists of a few fully-connected layers. Figure 11 (b) depicts the results of the running time for 2PC and MPC for 100 instances. We find that when the number of participants is small (e.g. less than 8), the running time of MPC is negligible. Moreover, the running time of the MPC part will be longer than running the 2PC part only when achieving 13 participants. It is worth noting that the running time of the MPC part would not change with the number of data growing since the Average operation map all inputs to a vector of the same shape. When the number of instances grows, the running time of the MPC part will take a smaller percentage. This means that the running time of our circuits almost unaffected by the number of data owners.

VI. Related Work

Data commoditization has recently become an emerging trend. The problem of data pricing, especially, gains a lot of attention. The pricing schemes currently deployed in the data marketplace are simplistic: a buyer either buy the whole or parts of the dataset for a fixed price. Existing data marketplace platforms like Datarade [2] sells data based on the data type, size, query frequency, etc. Another kind of platforms like

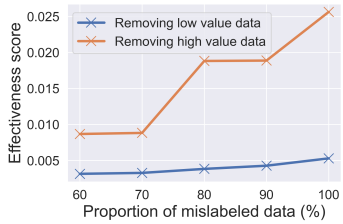
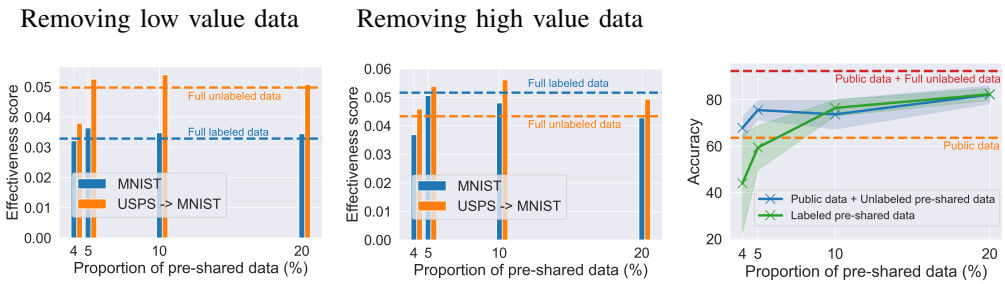


Figure 8: Effectiveness score VS. The proportion of correctly labeled data



(a) (b)

Figure 9: (a) Effectiveness score VS. The proportion of pre-shared data, (b) Marginal contribution of pre-share data.

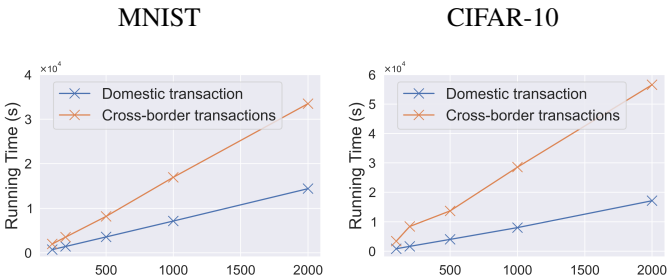
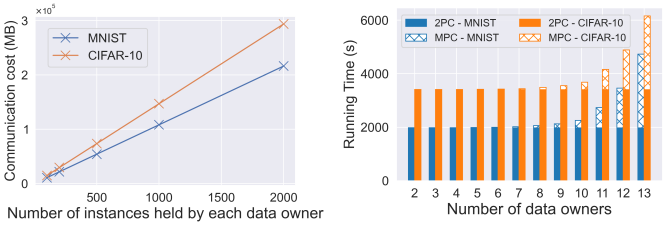


Figure 10: Running time of 2-PC protocol VS the number of instances



(a) (b)

Figure 11: (a) Communication cost of the 2PC circuits; (b) Running time of the 2PC and MPC parts.

Google BigQuery [5] also charging for running (relational) queries over the dataset.

A bunch of previous works have studied how to price data over different kinds of data marketplaces. A recent line of works [42], [43] have formally studied pricing schemes for fine-gained queries over a dataset. With the query-based pricing schemes, given a dataset D and a query Q, the seller assigns a price $p(D, Q)$ based on the information disclosed by the query answers. Two important properties that the pricing function must satisfy namely arbitrage-freeness and discount-freeness was proposed by Koutris el al [43]. The arbitrage-freeness indicated that when query Q_1 discloses more information than Q_2, we want to ensure that $P(Q_1) > p(Q_2)$; otherwise, buyers have an arbitrage opportunity to take advantage of the marketplace by combining datasets for lower prices into a high-price dataset to escape the designated price for that dataset. The discount-freeness requires that the prices offers no additional discounts than the ones specified by the data seller. In fact, discount-freeness is the discrete version of arbitrage-freeness. Menwhile, with the increasing pervasiveness of machine learning-based analysis, model-based pricing has become a new interest in studying the cost of acquiring data for machine learning. Chen et al. [44] proposed the first model-based pricing framework that directly prices instances of the ML model with different noise, in which an optimization problem was formulated to find the arbitrage-free price that maximizes the revenue of a seller. Liu et al. [16] proposed an end-to-end model marketplace aiming to maximize the revenue for data owners while supplying the demands of model buyers, wherein a broker is introduced to collect data from data owners and sell ML models to buyers.

The problem of fairness in the data marketplace also attracts researchers deeply, wherein Shapley value has become the widely used notion of fairness due to its rigorous fairness guarantees. The use of the SV for pricing personal data can be traced to [45], which studied the use of SV in the context of marketing survey, collaborative filtering, and recommendation systems. Recently, Jia et al. [6] formally studies the notion of fairness based on the SV and designed an algorithm that can calculate the SV more efficient based on Nearest Neighbor Algorithms (KNN). Although the interaction between data analytics and economics has been extensively studied, data security is often neglected, especially in the data evaluation process, which is fundamental in the data marketplace. Xu et al. [46] developed three methods for data appraisal: Norm of Parameter Gradients, Model Fine-tuning and Influence Functions, and proposed to perform data appraisal using MPC. However, their method cannot be applied to calculate SV. Azcoitia et al. [47] proposed a algorithm namely *Try Before You Buy* (TBYB) to evaluate data value before paying for them. However, it requires a sandbox to train the target model multiple times using different datasets.

VII. Conclusion and Future Work

In response to the increasing demand for protecting security during data transactions in the data marketplace, we propose a fair payment scheme suitable for the data marketplace

while satisfying the demand for data valuation. We develop algorithms that are useful for data valuation while friendly be implemented via MPC circuit. To guarantee that the data to be paid is exactly the data evaluated before, we innovatively incorporate fair payment into the MPC circuit instead of using ZKCP. Since it is general, efficient, and also conceptually easy to understand, we believe it to have the potential to serve as the de facto method for real-world data marketplace platforms. In the future, it would be promising to see further studies on reducing the number of pre-shared data with the buyer and improving the efficiency of the framework.

REFERENCES

[1] Dawex, "Dawex," https://www.dawex.com/en/, 2022.

[2] Datarade, "Datarade," https://datarade.ai/, 2022.

[3] Quandl, "Quandl," https://data.nasdaq.com/, 2022.

[4] Bloomberg, "Bloomberg," https://www.bloomberg.com/markets/, 2022.

[5] BigQuery, "Google bigquery," https://cloud.google.com/bigquery/, 2022.

[6] R. Jia, D. Dao, B. Wang, F. A. Hubis, N. M. Gürel, B. Li, C. Zhang, C. J. Spanos, and D. Song, "Efficient task-specific data valuation for nearest neighbor algorithms," *Proc. VLDB Endow.*, vol. 12, no. 11, pp. 1610–1623, 2019. [Online]. Available: http://www.vldb.org/pvldb/vol12/p1610-jia.pdf

[7] R. Jia, D. Dao, B. Wang, F. A. Hubis, N. Hynes, N. M. Gürel, B. Li, C. Zhang, D. Song, and C. J. Spanos, "Towards efficient data valuation based on the shapley value," in *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and M. Sugiyama, Eds., vol. 89. PMLR, 2019, pp. 1167–1176. [Online]. Available: http://proceedings.mlr.press/v89/jia19a.html

[8] A. Ghorbani and J. Y. Zou, "Data shapley: Equitable valuation of data for machine learning," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 2019, pp. 2242–2251. [Online]. Available: http://proceedings.mlr.press/v97/ghorbani19c.html

[9] R. Jia, F. Wu, X. Sun, J. Xu, D. Dao, B. Kailkhura, C. Zhang, B. Li, and D. Song, "Scalability vs. utility: Do we have to sacrifice one for the other in data importance quantification?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8239–8247.

[10] O. Goldreich, "Secure multi-party computation," *Manuscript. Preliminary version*, vol. 78, 1998.

[11] Z. Zhou, X. Cao, J. Liu, B. Zhang, and K. Ren, "Zero knowledge contingent payments for trained neural networks," in *Computer Security – ESORICS 2021*, E. Bertino, H. Shulman, and M. Waidner, Eds. Cham: Springer International Publishing, 2021, pp. 628–648.

[12] M. Campanelli, R. Gennaro, S. Goldfeder, and L. Nizzardo, "Zero-knowledge contingent payments revisited: Attacks and payments for services," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, Eds. ACM, 2017, pp. 229–243. [Online]. Available: https://doi.org/10.1145/3133956.3134060

[13] K. Nguyen, M. Ambrona, and M. Abe, "WI is almost enough: Contingent payment all over again," in *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, J. Ligatti, X. Ou, J. Katz, and G. Vigna, Eds. ACM, 2020, pp. 641–656. [Online]. Available: https://doi.org/10.1145/3372297.3417888

[14] Y. Li, C. Ye, Y. Hu, I. Morpheus, Y. Guo, C. Zhang, Y. Zhang, Z. Sun, Y. Lu, and H. Wang, "Zkcplus: Optimized fair-exchange protocol supporting practical and flexible data exchange," in *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, Y. Kim, J. Kim, G. Vigna, and E. Shi, Eds. ACM, 2021, pp. 3002–3021. [Online]. Available: https://doi.org/10.1145/3460120.3484558

[15] Hashlock, "Hashlock," 2022. [Online]. Available: https://en.bitcoin.it/wiki/Hashlock

[16] J. Liu, J. Lou, J. Liu, L. Xiong, J. Pei, and J. Sun, "Dealer: An end-to-end model marketplace with differential privacy," *Proc. VLDB Endow.*, vol. 14, no. 6, pp. 957–969, 2021. [Online]. Available: http://www.vldb.org/pvldb/vol14/p957-liu.pdf

[17] B. Ghazi, N. Golowich, R. Kumar, P. Manurangsi, and C. Zhang, "Deep learning with label differential privacy," in *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021, pp. 27 131–27 145. [Online]. Available: https://proceedings.neurips.cc/paper/2021/hash/e3a54649aeec04cf1c13907bc6c5c8aa-Abstract.html

[18] T. Wang, Y. Yang, and R. Jia, "Learnability of learning performance and its application to data valuation," *arXiv preprint arXiv:2107.06336*, 2021.

[19] Y. Kwon and J. Zou, "Beta shapley: a unified and noise-reduced data valuation framework for machine learning," in *International Conference on Artificial Intelligence and Statistics, AISTATS 2022, 28-30 March 2022, Virtual Event*, ser. Proceedings of Machine Learning Research, G. Camps-Valls, F. J. R. Ruiz, and I. Valera, Eds., vol. 151. PMLR, 2022, pp. 8780–8802. [Online]. Available: https://proceedings.mlr.press/v151/kwon22a.html

[20] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 2017, pp. 1885–1894. [Online]. Available: http://proceedings.mlr.press/v70/koh17a.html

[21] L. Shapely, "A value for n-person games. contributions to the theory of games," pp. 307–318, 1953.

[22] T. Wang and R. Jia, "Data banzhaf: A data valuation framework with maximal robustness to learning stochasticity," *CoRR*, vol. abs/2205.15466, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2205.15466

[23] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Póczos, R. Salakhutdinov, and A. J. Smola, "Deep sets," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 3391–3401. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/f22e4747da1aa27e363d86d40ff442fe-Abstract.html

[24] J. Hoffman, E. Tzeng, T. Park, J. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, "Cycada: Cycle-consistent adversarial domain adaptation," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, ser. Proceedings of Machine Learning Research, J. G. Dy and A. Krause, Eds., vol. 80. PMLR, 2018, pp. 1994–2003. [Online]. Available: http://proceedings.mlr.press/v80/hoffman18a.html

[25] I. Damgård, V. Pastro, N. P. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, ser. Lecture Notes in Computer Science, R. Safavi-Naini and R. Canetti, Eds., vol. 7417. Springer, 2012, pp. 643–662. [Online]. Available: https://doi.org/10.1007/978-3-642-32009-5_38

[26] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*. IEEE Computer Society, 2017, pp. 19–38. [Online]. Available: https://doi.org/10.1109/SP.2017.12

[27] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via minionn transformations," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 619–631. [Online]. Available: https://doi.org/10.1145/3133956.3134056

[28] R. Cramer, I. Damgård, D. Escudero, P. Scholl, and C. Xing, "Spd$\mathbb{Z}2^k$: Efficient mpc mod 2^k for dishonest majority," in *Advances in Cryptology – CRYPTO 2018*, H. Shacham and A. Boldyreva, Eds. Cham: Springer International Publishing, 2018, pp. 769–798.

[29] S. Delgado-Segura, C. Pérez-Solà, J. Herrera-Joancomartí, and G. Navarro-Arribas, "Bitcoin private key locked transactions," *Inf.*

Process. Lett., vol. 140, pp. 37–41, 2018. [Online]. Available: https://doi.org/10.1016/j.ipl.2018.08.004

[30] Z.-Q. Zeng, H.-B. Yu, H.-R. Xu, Y.-Q. Xie, and J. Gao, "Fast training support vector machines using parallel sequential minimal optimization," in *2008 3rd international conference on intelligent system and knowledge engineering*, vol. 1. IEEE, 2008, pp. 997–1001.

[31] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[32] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[33] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[34] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 215–223.

[35] S. Maleki, L. Tran-Thanh, G. Hines, T. Rahwan, and A. Rogers, "Bounding the estimation error of sampling-based shapley value approximation with/without stratifying," *CoRR*, vol. abs/1306.4265, 2013. [Online]. Available: http://arxiv.org/abs/1306.4265

[36] M. Keller, "MP-SPDZ: A versatile framework for multi-party computation," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020. [Online]. Available: https://doi.org/10.1145/3372297.3417872

[37] B. Fashion, "Bristol fashion," https://bit.ly/3aDWBFp, 2022.

[38] Wikipedia, "Spearman's rank correlation coefficient," https://bit.ly/3EmUOB3, 2022.

[39] ——, "Dirichlet distribution," https://bit.ly/3cgG9vs, 2022.

[40] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=rJzIBfZAb

[41] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. [Online]. Available: https://openreview.net/forum?id=BJx040EFvH

[42] B. Lin and D. Kifer, "On arbitrage-free pricing for general data queries," *Proc. VLDB Endow.*, vol. 7, no. 9, pp. 757–768, 2014. [Online]. Available: http://www.vldb.org/pvldb/vol7/p757-lin.pdf

[43] P. Koutris, P. Upadhyaya, M. Balazinska, B. Howe, and D. Suciu, "Query-based data pricing," *J. ACM*, vol. 62, no. 5, pp. 43:1–43:44, 2015. [Online]. Available: https://doi.org/10.1145/2770870

[44] L. Chen, P. Koutris, and A. Kumar, "Towards model-based pricing for machine learning in a data marketplace," in *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, P. A. Boncz, S. Manegold, A. Ailamaki, A. Deshpande, and T. Kraska, Eds. ACM, 2019, pp. 1535–1552. [Online]. Available: https://doi.org/10.1145/3299869.3300078

[45] J. Kleinberg, C. H. Papadimitriou, and P. Raghavan, "On the value of private information," in *Proceedings of the 8th Conference on Theoretical Aspects of Rationality and Knowledge*, ser. TARK '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, p. 249–257.

[46] M. Xu, L. van der Maaten, and A. Y. Hannun, "Data appraisal without data sharing," *CoRR*, vol. abs/2012.06430, 2020. [Online]. Available: https://arxiv.org/abs/2012.06430

[47] S. A. Azcoitia and N. Laoutaris, "Try before you buy: A practical data purchasing algorithm for real-world data marketplaces," *CoRR*, vol. abs/2012.08874, 2020. [Online]. Available: https://arxiv.org/abs/2012.08874