FedServing: A Federated Prediction Serving Framework Based on Incentive Mechanism

Jiasi Weng[1 2†], Jian Weng[1*], Hongwei Huang[1], Chengjun Cai[2], and Cong Wang[2]
[1]Jinan University; [2]City University of Hong Kong

IEEE INFOCOM 2021 - IEEE Conference on Computer Communications | 978-1-6654-0325-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/INFOCOM42981.2021.9488807

***Abstract*—Data holders, such as mobile apps, hospitals and banks, are capable of training machine learning (ML) models and enjoy many intelligence services. To benefit more individuals lacking data and models, a convenient approach is needed which enables the trained models from various sources for prediction serving, but it has yet to truly take off considering three issues: (*i*) incentivizing prediction truthfulness; (*ii*) boosting prediction accuracy; (*iii*) protecting model privacy.**

We design FedServing, a federated prediction serving framework, achieving the three issues. First, we customize an incentive mechanism based on *Bayesian game theory* which ensures that joining providers at a *Bayesian Nash Equilibrium* will provide truthful (not meaningless) predictions. Second, working jointly with the incentive mechanism, we employ *truth discovery algorithms* to aggregate truthful but possibly inaccurate predictions for boosting prediction accuracy. Third, providers can locally deploy their models and their predictions are securely aggregated inside TEEs. Attractively, *our design supports popular prediction formats, including top-1 label, ranked labels and posterior probability.* Besides, blockchain is employed as a complementary component to enforce exchange fairness. By conducting extensive experiments, we validate the expected properties of our design. We also empirically demonstrate that FedServing reduces the risk of certain *membership inference attack*.

***Index Terms*—Prediction serving, Incentive mechanism, Privacy, Aggregation**

I. Introduction

Machine learning (ML) is revolutionizing our world and the global market for ML driven services is expected to reach $\$5,330$ million by 2024 [1]. Many data holders, such as mobile apps, hospitals and banks, are able to train models based on the available data they hold, and use the trained models to achieve functionality and business innovation [2]. From another perspective, most individuals lacking data and power are incapable of training models, so that they hardly benefit from ML. Even if an individual is in possession of a model, it still has the real-world demand to collaborate with other parties' models, demonstrated by existing real-world cases, *e.g.*, two banks in North America collaborate to detect money laundering. Obviously, due to privacy concerns, intellectual property issues or business competition, model owners are unwilling to share their trained models. Thus, it is necessary to build a bridge which connects model owners who have no incentives of sharing models with individuals who need models.

Building such a bridge inevitably needs to support three essential requirements as following: (*i*) providing sufficient *incentives* to the model owners so that they are willing to contribute their models; (*ii*) enabling individual users of interest to enjoy as *high-performance* as possible models; (*iii*) guaranteeing *model privacy*, since models imply private information about their training data [3]. However, there exists no work to realize such a bridge, so that it has yet to truly take off.

While Machine-Learning-as-a-Service (MLaaS) platforms enable monetizing models for prediction serving on a pay-per-query basis, trained models have to reside on the untrusted servers, causing model privacy concerns. Although earlier works present effective approaches [4–6] for protecting models against the untrusted servers, they still are not really satisfactory. Specifically, cryptographic methods are computation-consuming and inefficient when handling large-sized models [4], but high-performance models usually are large. Differential privacy based defenses would sacrifice prediction accuracy [5]. Trusted hardware-enabled approaches are relatively practical but still have efficiency limitation. It is due to that trusted hardwares are majorly restricted to CPUs, but running large models usually needs GPUs [6].

Motivated by our observations, our objective is to make model owners freely deploy their models without limits, collectively contribute their models to make profits and securely use models without privacy leakage concerns. Towards the goal, we present a *federated prediction serving framework, FedServing,* towards model owners from various sources in an open setting. Our starting point is allowing model owners to deploy models at local devices and provide aggregated predictions for exchanging with monetary rewards. Standing on top of it, we especially make efforts to design our solution for enforcing prediction accuracy, due to the following two-fold challenges:

Challenge (*i*): Strategic behaviors of model owners. Model owners (hereafter called as providers) are likely to be rational and selfish, so that they may be strategic to report meaningless predictions without effort. In addition, ground truths with respect to given prediction queries are usually unknown, which makes the truthfulness of predictions hard to be verified.

Challenge (*ii*): Varying quality of models. While aggregating predictions (*e.g.*, via majority voting or averaging) is a classic strategy for improving accuracy, they may be less effective in our case. The issue of majority voting and averaging is the assumption of equally reliable prediction sources (*i.e.*, models). Yet, the assumption cannot hold in our open setting. It is due to that (1) the qualities of models from various sources are varying, and due to local deployment, there is no available au-

†Work was done when the author was a visiting student with City University of Hong Kong. *E-mail: cryptjweng@gmail.com.

thority enforcing the model quality upon answering prediction queries; (2) even well-trained models are not always generalized well over the whole feature space of all prediction queries, so producing predictions are probably not always accurate.

In light of the two challenging issues, available solutions [7–12] usually resort to incentive mechanisms in conjunction with quality-aware aggregation algorithms. Unfortunately, previous work cannot be used to mitigate our challenging issues because of their inability to simultaneously handle categorical and continuous data covered by popular prediction outputs [13]. As a concrete instance of solving a sentiment analysis task, the prediction outputs can be ***top-1 label***, *e.g.*, [upset], ***ranked labels***, *e.g.*, [upset, scared, distressed, guilty], and ***posterior probability*** for each label, *e.g.*, $\langle 95.0\%, 2.0\%, 1.0\%, 2.0\% \rangle$, which are supported by Google Photos and Google Cloud Vision API, for example.

Our key design. We customize a complementary mechanism by integrating an incentive design with "truth-finding" algorithms. Concretely, our mechanism (1) uses *Bayesian game theory* to model the honest and strategic behaviors of providers and ensures the existence of a *Bayesian Nash Equilibrium*, where all providers will offer truthful not meaningless predictions for given prediction queries; (2) employs *truth discovery (TD) algorithms* to learn highly accurate predictions from the truthful but possibly inaccurate predictions to eliminate the effect of inaccurate predictions; (3) allocates the providers with fair rewards in proportion to the truthfulness of their predictions; (4) handles prediction output formats including both labels and posterior probabilities.

Despite that models are locally deployed, privacy concerns still exist due to disclosing predictions of a single model. Concretely, a model's predictions can be exploited to infer if a data record was used to train the model, *e.g.*, identifying if an individual was a patient at the hospital, known as membership inference attacks [14]. To address the privacy concern, we leverage trusted execution environments (TEEs) to aggregate predictions from multiple providers, and only aggregated predictions are revealed to users [15]. Owing to the confidentiality and integrity provided by TEEs, a model's predictions are not revealed and aggregated predictions are correctly generated. It is noteworthy that our proposed incentive mechanism also can benefit from the TEEs' integrity, since the procedure of evaluating the truthfulness of predictions from each model can be correctly executed, which further enforces fair rewards guided by the truthfulness. Notably, we do not use privacy-preserving verifiable cryptography, considering that TEEs are relatively more performant.

Besides, we need to facilitate an open setting for model owners from various sources freely joining in FedServing. But meanwhile, we also need a regulation complementary to our incentive mechanism for fulfilling the transparent process of money settlement and deterring providers' selfish behaviors, *e.g.*, abortion, thereby achieving the fairness of money-prediction exchange among users and providers. In light of the issues, we choose the blockchain to facilitate the open setting and enforce the regulation.

We note that FedServing can be extended to support the existing prediction serving systems and now we shed light on the service manner of our FedServing framework. A prediction serving server can deploy a smart contract as a uniform query interface for charging users and as an entrance for participating providers. When receiving the queries and deposits from a user, the server resorts to its off-chain TEEs-empowered component to collect predictions from participating providers who undertake the prediction task. The TEE strategically aggregates predictions and submits aggregated predictions to the blockchain. Finally, the user obtains the predictions and meanwhile the smart contract allocates the user's fees to the participating providers according to the truthfulness of their predictions.

In conclusion, the main contributions are as following:

- We propose a blockchain-empowered federated prediction serving framework in an open setting, providing as accurate as possible prediction services with truthful contributions from multi-source models.
- We customize an incentive mechanism for eliciting truthful contributions, by carefully applying a technique of *peer prediction* [16] while respecting popular prediction formats.
- We extend a widely-adopted truthful discovery algorithm to support our setting, jointly working with our incentive mechanism, finally enhancing prediction accuracy.
- We implement our design and conduct extensive experiments in terms of the performance, validity and ability against certain privacy attack. For reproducibility, our code is publicly available at https://github.com/H-W-Huang/FedServing.

II. Related Work

Prediction Serving System. Existing excellent systems [17, 18] centralizedly manage models as well as deploy models for low-latency and high-throughput prediction serving, where models are off-the-shelf. To enhance prediction accuracy, they generally support ensemble models which aggregate predictions from multiple models.

Different from them, our work focuses on the models from various sources for prediction serving in an open setting. More precisely, we consider *how to incentivize model owners to provide truthful prediction services while respecting model privacy and ensuring prediction accuracy.* To the end, we present a distributed framework achieving the following threefold components which are less considered by the existing systems [17, 18].

(*i*) **Pricing mechanism.** We customize a pricing mechanism for compensating participating providers and incentivizing prediction truthfulness, instead of using an one-price-fits-all pricing structure, which still respects the pay-per-query business pattern of the current MLaaS platforms.

(*ii*) **Quality-aware aggregation.** Considering that model quality and ground truths of prediction queries are unknown in our open setting, we use TD algorithms to aggregate predictions

rather than simply averaging, thereby eliminating the effect of low-accuracy predictions.

(*iii*) **Model and prediction protection.** We make models never leave local devices and multiple predictions are securely aggregated inside TEEs, so that users only obtain aggregated predictions. Due to local deployment, providers retain control over when and how their in-house models are used to make predictions, *e.g.*, joining in an ensemble to produce predictions for reducing privacy leakage [14].

Incentive Mechanism. Prior incentive mechanisms [7–12, 19–34] are designed for stimulating participation by compensating workers' costs with monetary rewards, and implementing economic properties, such as platform profit maximization, individual rationality and budget feasibility, which greatly promote the development of crowdsourcing. These incentive mechanisms generally resort to game-theoretic methods, such as reverse auction [21, 22, 25, 26, 31], double auction [23, 34] and all-pay auction [20], or other game theory [9, 11, 33]. With the game-theoretic analysis, they consider workers' strategic behaviors and investigate how to encourage workers to behave truthfully.

In this paper, we aim to stimulate the truthfulness of collective predictions, considering providers' strategic behaviors of providing meaningless predictions. Existing mechanisms do not solve our problem, since the following four *requirements* cannot be simultaneously satisfied:

(*i*) **Incentivizing truthfulness.** Most mechanisms [21, 22, 25, 26] focus on motivating workers to reveal their costs truthfully. A few excellent mechanisms like [9, 11] incentivize the truthfulness of crowd data as this paper, but they are unsatisfactory to us due to that (*ii*) below cannot be supported.

(*ii*) **Simultaneously handling categorical and continuous data.** Theseus [9] proposes a truthful mechanism for quality and efforts elicitation while focusing on continuous sensing data. [11] creatively studies the joint elicitation of quality, efforts and data while focusing on binary data. Their techniques do not solve our problem, since we simultaneously consider categorical and continuous data.

(*iii*) **No reliance on prior knowledge.** Prior arts [8, 23] assume workers' reliability or reputation as prior knowledge for allocating rewards. We do not assume such prior knowledge, since a provider's predictions for historical tasks are considered irrelevant to the current task[1].

(*iv*) **Jointly addressing incentive and quality concerns.** Most mechanisms for incentivizing truthfulness do not jointly work with TD, except to [7–10, 12]. These works can be deployed with TD, but they still are unsatisfactory to us: Theseus [9], [8] and [12] mainly consider the continuous data stream; [7] does not focus on workers' strategic behaviors; [10] cares about binary answers and assumes that most workers are reliable.

III. System Overview

In this section, we present our FedServing framework. It begins with the system model, and then figures out the threat assumptions and design goals.

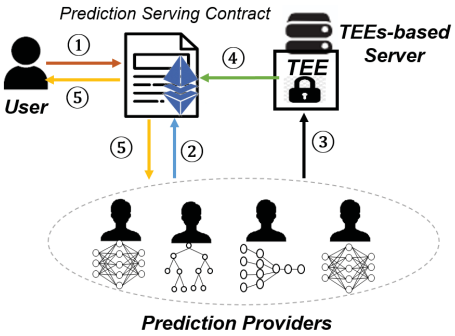

Fig. 1: Overview of the FedServing framework.

A. System Model

At the high level, our FedServing consists of four entities as shown in Fig. 1: *prediction providers, user, smart contract* and *TEEs-based server*. Specifically, the prediction providers who own various ML models monetize their prediction query services on the blockchain (*e.g.*, Ethereum). They could publicize non-private model profiles like service APIs for user accessing their models. User is able to browse model profiles on the blockchain, and query prediction services via a smart contract, named as *prediction serving contract* (PS contract). PS contract aims at receiving the user's query request, relaying the request, receiving aggregated predictions and achieving the fair payment finalization. As the intermediator between the prediction providers and the PS contract, the TEEs-based server is responsible for strategically aggregating predictions sent by multiple providers, and calculating accuracy-aware scores used to guide allocating rewards. The basic workflow in Fig. 1 is described as following:

① User sends a transaction which contains a description about her requested task, *e.g.*, a sentiment analysis task, and makes a deposit for payment to the PS contract. Note that the task's input data, *e.g.*, text files, can be securely stored in an accessible system like IPFS, and then be securely authorized to participating providers.

② Providers participate in the task by submitting a deposit to the PS contract for potential penalty, *e.g.*, punishing abortion. Here, we omit the phase that they can authentically obtain the task's input data from IPFS.

③ Participating providers evaluate local models on the input data, and lastly submit predictions to the TEE via an authenticated communication channel.

④ The TEE strategically aggregates predictions from multiple participating providers and compute accuracy-aware scores for each provider. After that, the aggregated predictions are correctly encrypted using the user's public key and submitted to the blockchain.

⑤ User retrieves and decrypts the aggregated predictions using her private key, and meanwhile, her deposit is allocated to the participating providers according to the respective accuracy-aware scores.

[1]A prediction task usually is requested with batch queries, where the queries belonging to the same task are relevant.

B. Threat Model and Assumptions

Prediction Provider. A prediction provider answering certain query is named as a participating provider and assumed not to collude with other participating providers. We consider that participating providers are rational and self-interested, and thus they act to maximize their profits by submitting arbitrary predictions. We also assume that the input data received by participating providers are benign; perturbed input data known as *adversarial examples* [35] are out of our consideration.

TEEs. We trust TEEs, *e.g.*, Intel Software Guard Extensions (SGX), to securely execute specific programs against external observation and manipulation, *i.e.*, ensuring confidentiality and integrity. Like prior TEEs-empowered work [36], side-channel attacks and rollback attacks on TEEs are out of the scope of this paper, owing to many off-the-shelf defence mechanisms [37, 38]. We rely on the authenticated communication channels built between a TEE and a remote party, *e.g.*, Intel SGX's Enhanced Privacy ID (EPID) remote attestation protocol.

Blockchain. We trust the blockchain for integrity and availability. Smart contract autonomously and faithfully executes defined functions, *e.g.*, correctly locking deposits and settling rewards, which is assumed not vulnerable to software bugs.

Remarks. We aware that FedSeving can suffer from Sybil attacks [39], where a prediction provider may maliciously use multiple fake accounts to join in certain task. For demoralizing Sybil attacks, a widely adopted solution is to increase the attack cost like solving proof-of-work puzzles and making deposits. In this paper, we require each participating provider to make a deposit before undertaking a task.

C. Design Goals

Truthfulness and accuracy. It means that user can obtain aggregated predictions with truthfulness and accuracy guarantees. Specifically, each participating provider provides truthful (but possibly inaccurate) predictions, and meanwhile, the truth discovery algorithm is correctly conducted on the provided truthful predictions to produce truths, *i.e.*, aggregated predictions, which are regarded accurate enough.

Fairness. It includes the fairness of reward allocation and the fairness of money-prediction exchange. First, each participating provider in a task gets a fair reward guided by a *strictly proper score* which is computed based on the truthfulness of their predictions. A comparatively truthful prediction leads to a higher score, and the prediction's provider obtains comparatively more rewards. Second, all participating providers receive rewards *iff* the user obtains the final predictions.

IV. Design of Prediction Aggregation

Considering that our FedServing is built in an open setting, participating models might produce inaccurate predictions. The reasons include that (*i*) varying quality models can freely participate in FedServing, and meanwhile, there is no available authority enforcing the quality of participating models; (*ii*) trained models are not always generalized well over the whole feature space of every prediction task [40].

In light of this issue, we study the lessons from the earlier works [8–10, 41, 42] and leverage TD algorithms [41] to aggregate predictions, so as to learn as accurate predictions as possible from varying quality models in absence of ground truths. We support three common prediction formats in practice. To the best of our knowledge, there is no existing scheme dealing with the issue as this paper. The previous work [42] is similar to our design of prediction aggregation. Yet, it focuses on one single format, *i.e.*, probability vector, while we especially consider other popular prediction outputs including ranked label list, used in Google Photos.

For ease of presentation, we begin with an instance of a sentiment analysis task. Then, we elaborate three prediction formats and demonstrate how to aggregate them.

Instance Description. Suppose that a social psychologist has a sentiment analysis task with a set of consulting letters from anonymous citizens. She needs to label the set of consulting letters using proper emotion states for studying social projection. With the task, she can query the PS contract in our FedServing: what is the emotion state for each consulting letter, distressed, upset, guilty or scared?

Prediction Formats. In the above instance, we introduce three types of prediction formats [13]: (1) **Abstract**: a top-1 class label, *e.g.*, 'upset', (2) **Rank**: a ranked list of labels, *e.g.*, [upset, scared, distressed, guilty], and (3) **Measurement**: a probability vector for possible class labels, *e.g.*, $\langle 2.0\%, 95.0\%, 1.0\%, 2.0\% \rangle$ for [distressed, upset, guilty, scared] (their sum is 100%).

Apparently, the last type contains the most detailed prediction information while the first type contains less information. Note that here we mainly discuss classification tasks, but our method can be easily extended to regression tasks which are associated with real-valued predictions.

Prediction Aggregation. We now introduce the algorithm of aggregating predictions adapted to three formats mentioned above. Specifically, we carefully transform the later two formats into continuous data vectors, so as to apply Algorithm 1 into all predictions, regardless of their formats before the transformation. For ease of explanation, we suppose that there are three models predicting a given consulting letter with the corresponding label list [distressed, upset, guilty, scared]. Their predictions with respect to the three formats are demonstrated in TABLE I.

We now explain how we uniformly represent the three-format predictions by using continuous data vectors. For the abstract format, the three models separately produce labels 'upset', 'distressed' and 'distressed'. We transform them into the corresponding $0/1$ value vectors, where the index with value 1 is the most possible label, as shown in the abstract row of TABLE I. For the rank format, the three models provide the ranked lists of possible labels as presented in TABLE II. For example, a ranked list [upset, scared, distressed, guilty] is given by the first model. We set ranked integer values to each ranking level. A largest integer represents the highest ranking level while a smallest integer represents the lowest one. With this representation rule, the ranked lists in TABLE II

TABLE I: Examples of prediction formats

Format	Model1	Model2	Model3
Abstract	$\langle 0, 1, 0, 0\rangle$	$\langle 1, 0, 0, 0\rangle$	$\langle 1, 0, 0, 0\rangle$
Rank	$\langle 2, 4, 1, 3\rangle$	$\langle 4, 2, 1, 3\rangle$	$\langle 4, 2, 3, 1\rangle$
Measurement	$\langle 2.0\%, 49.0\%, 1.0\%, 48.0\%\rangle$	$\langle 92.0\%, 2.0\%, 1.0\%, 5.0\%\rangle$	$\langle 93.0\%, 2.0\%, 3.0\%, 2.0\%\rangle$

are transformed into the vectors with integer values in the rank row of TABLE I. Last, the probability vectors in the measurement format are presented without change. Hereafter, we call the vector values as *confidence values*.

TABLE II: Examples of ranked lists.

Value	Model1	Model2	Model3
4	upset	distressed	distressed
3	scared	scared	guilty
2	distressed	upset	upset
1	guilty	guilty	scared

With the uniform representation, multiple predictions for the set of consulting letters in each format will be aggregated via Algorithm 1 including two steps [41]. Specifically, we suppose that there are multiple predictions from m $(m \geq 3)$ providers for n consulting letters. Each prediction is a c-length vector containing the confidence values for each class label, where c is the number of given possible class labels. They are represented as $\{I_i^j\}_{i=1,j=1}^{m,n}$, where I_i^j is a continuous data vector $\mathbf{v}_i^j = (v_{i1}^j, ..., v_{ic}^j)$. Now, by using Algorithm 1 whose correctness has been analysed in [41], we iteratively estimate the truths on $\{I_i^j\}_{i=1,j=1}^{m,n}$ and update m providers' weights until convergence. The algorithm finally outputs the truths as the aggregated predictions $\{O^{j(\varepsilon)}\}_{j=1}^{n}$ with respect to each consulting letter.

Algorithm 1 Truth discovery

Input: provider predictions $\{I_i^j\}_{i=1,j=1}^{m,n}$
Output: truth predictions $\{O^{j(\varepsilon)}\}_{j=1}^{n}$
1: Initialize $r = 1$ and weights $\{w_i^{(r)} = 1\}_{i=1,...,m}$.
2: **repeat**
3: **for** each $j \in [1, n]$ **do**
4: $O^{j(r+1)} \leftarrow \frac{\sum_{i=1}^{m} w_i^{(r)} I_i^j}{\sum_{i=1}^{m} w_i^{(r)}}$ (1)
5: **end for**
6: **for** each $i \in [1, m]$ **do**
7: $w_i^{(r+1)} \leftarrow -\log(\frac{\sum_{j=1}^{n} f_{loss}(O^{j(r+1)}, I_i^j)}{\sum_{k=1}^{m}\sum_{j=1}^{n} f_{loss}(O^{j(r+1)}, I_k^j)})$ (2)
8: **end for**
9: $r = r + 1$
10: **until** $r \leq \varepsilon$
11: **return** $\{O^{j(\varepsilon)}\}_{j=1}^{n}$

Initially, we set each provider's weight with 1 and denote an iteration threshold ε. Then, with fixed weights, m providers' predictions are aggregated via the weighted mean method (Step (1)). During the iterative computation, the aggregated predictions are closer to that offered by the providers having higher weights. With the aggregated predictions, each provider's weight is updated based on the distances between his predictions and the aggregated predictions with respect to n consulting letters (Step (2)). The provider whose predictions are closer to the aggregated predictions will be assigned with a higher weight. Here, the loss function $f_{loss}(\cdot)$ is used to characterize the distance and specifically, we use the normalized squared loss function. Step (1) and (2) are iteratively computed until r reaches the pre-defined threshold ε.

V. Design of Pricing Mechanism

The previous section introduces the process of aggregating predictions with the aim to filter out less accurate predictions. Yet, the accuracy of aggregated predictions still cannot be guaranteed if a majority of self-interested providers offer meaningless predictions. In order to motivate the self-interested providers to provide truthful predictions, we jointly design our pricing mechanism by employing the Bayesian game theory. Notably, predictions contain categorical and continuous data which will be simultaneously handled.

This section begins with the setting definitions and design objectives, and then presents the formulation of our pricing mechanism as well as an approximate solution. To the end, we demonstrate the existence of a Bayesian Nash Equilibrium for participating providers by analysing the pricing mechanism.

A. Mechanism Setting

We use the game theory method to model the strategic behaviors of participating providers inspired by the works [9, 11]. Concretely, we model participating providers $P = \{i, ..., m\}$ playing a *non-cooperative game*, where each of them independently gives a private prediction for each query requested by certain user. Note that a requested task can include multiple queries, *e.g.*, labeling multiple consulting letters.

In the game, participating providers behave as utility maximizers. They behave strategically by evaluating their expected utility. Specifically, they will not participate if the expected utility is negative, and otherwise, they offer predictions via a specific strategy for maximizing the expected utility. In general, the evaluation needs some technical assumptions [16]. We assume that participating providers undertaking the same task have a common prior belief, and meanwhile, they use the same belief updating procedure, *i.e.*, Bayes' rule.

A provider's behavior is described by *strategy*. A strategy is denoted by $s = (\mathbf{l}, \mathbf{v})$ representing a prediction for a query, or $\perp$ meaning abort. Herein, $\mathbf{l}$ is a list of claimed possible class labels and each label in $\mathbf{l}$ is from discrete set Ω; $\mathbf{v}$ is the corresponding posterior probability values which are drawn from probability density distributions Ψ. Thus, the strategy space is $\{(\Omega, \Psi)\} \cup \{\perp\}$. Then, the strategy profile of participating providers is $\mathbf{S} = (s_1, ..., s_m)$, if we suppose that there are m participating providers.

Next, we continue to formulate the provider model, the user model and a Bayesian Nash Equilibrium for providers.

Provider Model. Within the defined game, a provider's payoff depends on his own strategy with regard to other providers' strategies. Specifically, given a payment function $p(\cdot)$, a cost function $c(\cdot)$ and deposit d_0, we define any provider's utility $u_i(\mathbf{S})$, $i \in P$ in a game with a strategy profile $\mathbf{S}$ as following:

$$u_i(\mathbf{S}) = p_i(\mathbf{S}) - c(s_i) - d_0.$$

Next, any provider can evaluate the expected utility:

$$\mathbb{E}_{\mathbf{S}_{-s_i}}[u_i(s_i, \mathbf{S}_{-s_i})] = \mathbb{E}_{\mathbf{S}_{-s_i}}[p_i(s_i, \mathbf{S}_{-s_i})] - c(s_i) - d_0,$$

where $\mathbf{S}_{-s_i}$ is the strategy profile excluding s_i. Note that a participating provider's deposit for n queries is $d = n \times d_0$.

User Model. A user's objective is to obtain the aggregated predictions whose accuracy is as close as possible to the accuracy of truths. To exchange the aggregated predictions $\{\mathbf{v}^j\}_{j=1,...,n}$ of n queries from m participating providers, she makes amount of deposits, namely budget B, on the blockchain. Assume that the market publicizes budget curves relative to the number of employed providers via market survey. With the budget curves, the user deposits a budget level that enables soliciting certain number of prediction providers.

Bayesian Nash Equilibrium. A strategy profile $\mathbf{S}^*$ is denoted as a Bayesian Nash Equilibrium (BNE) in the defined game, if no provider $i \in P$ can increase her expected utility by changing the current strategy s_i^* with regard to other providers' strategies $\mathbf{S}^*_{-s_i}$:

$$\mathbb{E}_{\mathbf{S}^*_{-s_i}}[u_i(s_i^*, \mathbf{S}^*_{-s_i})] \geq \mathbb{E}_{\mathbf{S}^*_{-s_i}}[u_i(s_i, \mathbf{S}^*_{-s_i})].$$

At the BNE, our mechanism aims to achieve several design objectives in Section V-B.

B. Design Objectives

With the strategy $\mathbf{S}^*$ at the BNE, we state three design objectives below.

Definition 1. *(**Truthfulness**) An aggregated prediction for a query is truthful if and only if (i) the aggregation computation is correctly executed, and meanwhile, (ii) every participating provider $i \in P$ at BNE $\mathbf{S}^*$ provides a prediction $s_i^* = (\mathbf{l}_i, \mathbf{v}_i)$ satisfying the following condition:*

$$\mathbf{l}_i = \mathbf{l}^p \wedge \mathbf{D}_{KL}(\mathbf{v}^T || \mathbf{v}_i) \leq \theta.$$

Here, vector $\mathbf{l}^p$ contains the public possible class labels, *e.g.*, [distressed, upset, guilty, scared] in Section IV. $\mathbf{v}^T$ is the true posterior probability for $\mathbf{l}^p$. $\mathbf{D}_{KL}(||)$ is the Kullback-Leibler (KL) divergence function. $\mathbf{D}_{KL}(\mathbf{v}^T||\mathbf{v}_i)$ measures the information lost using $\mathbf{v}_i$ to approximate $\mathbf{v}^T$. Clearly, condition (*i*) can be guaranteed by leveraging TEEs. Next, we design a pricing mechanism to meet condition (*ii*), so that every provider has no motivation to provide a prediction which deviates from the truthful labels and the corresponding truthful posterior probability. However, $\mathbf{v}^T$ is unknown in our setting. Our designed pricing mechanism will take it into consideration.

Definition 2. *(**Individual Rationality**) A pricing mechanism satisfies individual rationality (IR) iff every participating provider $i \in P$ at the BNE has non-negative expected utility:*

$$\mathbb{E}_{\mathbf{S}^*_{-s_i}}[u_i(s_i^*, \mathbf{S}^*_{-s_i})] \geq 0.$$

Definition 3. *(**Budget Feasibility**) A pricing mechanism satisfies budget feasibility (BF) iff the total payment allocated to the participating providers $i \in P$ at the BNE is not more than a user's given budget for every query:*

$$\mathbb{E}_{\mathbf{S}^*}[\sum_{i=1}^{m} p_i(\mathbf{S}^*)] \leq \frac{B}{n},$$

where m is the number of providers while n is the number of queries.

C. Pricing Mechanism Formulation

We are now ready to formulate the optimization problem of designing our pricing mechanism for participants' predictions (called as PPP), *i.e.*,

$$\begin{aligned} \max_{p(\cdot)} \quad & \sum_{i=1}^{m} \mathbf{Pr}(\mathbf{D}_{KL}(\mathbf{v}^T||\mathbf{v}_i) \leq \theta) \\ \text{s.t.} \quad & \mathbb{E}_{\mathbf{S}^*_{-s_i}}[u_i(s_i^*, \mathbf{S}^*_{-s_i})] \geq 0 \\ & \mathbb{E}_{\mathbf{S}^*}[\sum_{i=1}^{m} p_i(\mathbf{S}^*)] \leq \frac{B}{n}. \end{aligned}$$

As elaborated, given a set of participating providers $P = \{1, ..., m\}$, n queries and budget B, we aim to customize a payment function $p(\cdot)$ which satisfies both constraints of IR and BF, as well as maximizes the objective function, that is, the overall probability of the KL divergence between every provider's prediction at BNE $\mathbf{S}^*$ and the true prediction which is less than given threshold θ.

Solving PPP optimization problem will effectively minimize the loss between the accuracy of the aggregated predictions via truth discovery and the accuracy of truths, which is the user's objective. First, given n queries, $\sum_{j=1}^{n}\sum_{i=1}^{m} \mathbf{D}_{KL}(\mathbf{v}^{Tj}||\mathbf{v}_i^j)$ is apparently minimized, if PPP optimization problem is solved for every query. Next, we can achieve that the result accuracy via truth discovery is as close as possible to the accuracy of truths due to $\sum_{j=1}^{n}\sum_{i=1}^{m} \mathbf{Pr}(\mathbf{D}_{KL}(\mathbf{v}^{Tj}||\mathbf{v}_i^j) \leq \theta) \geq \sum_{j=1}^{n} \mathbf{Pr}(\mathbf{D}_{KL}(\mathbf{v}^{Tj}||\mathbf{v}^j) \leq \theta)$. The conclusion is according to the following derivation:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} \mathbf{D}_{KL}(\mathbf{v}^{Tj}||\mathbf{v}_i^j) \geq \frac{\sum_{i=1}^{m} w_i(\sum_{j=1}^{n} \mathbf{D}_{KL}(\mathbf{v}^{Tj}||\mathbf{v}_i^j))}{\sum_{i=1}^{m} w_i}$$

$$= \sum_{j=1}^{n} \frac{\sum_{i=1}^{m} w_i \mathbf{D}_{KL}(\mathbf{v}^{Tj}||\mathbf{v}_i^j)}{\sum_{i=1}^{m} w_i} \geq \sum_{j=1}^{n} \mathbf{D}_{KL}(\mathbf{v}^{Tj}||\frac{\sum_{i=1}^{m} w_i \mathbf{v}_i^j}{\sum_{i=1}^{m} w_i})$$

$$= \sum_{j=1}^{n} \mathbf{D}_{KL}(\mathbf{v}^{Tj}||\mathbf{v}^j).$$

However, solving PPP optimization problem is hard and ground truths are unavailable, namely $\mathbf{v}^T$. Hence, we approximately solve it by applying the idea of divergence-based

Bayesian Truth Serum (BTS) method [16]. Its main idea is rewarding a player based on the divergence between her reports and a randomly selected counterpart' reports, when there is no ground truths for verification. It is an effective approach to incentivize report truthfulness and control report quality [16]. We inherit such desirable properties from the divergence-based BTS method, and in the meantime, we handle both categorical data and continuous data, which is different from prior works [9, 11] considering either continuous data or categorical data.

Derived from the divergence-based BTS method, we denote our payment function. It rewards a participating provider i based on its strategy s_i and a randomly selected provider r's s_r by calculating two scores. The payment function is $p_i(s_i) = \alpha_i \times (score_{\mathrm{I}i} + score_{\mathrm{P}i} + 1)^2$, where $\alpha_i > 0$. The two scores are denoted accordingly as following:
(1) $score_{\mathrm{I}i} = score_{\mathrm{I}}(s_i, s_r)$ measures a *penalty* value if s_i reports the same labels with s_r, but the corresponding posterior probability disagrees with each others.

$$score_{\mathrm{I}}(s_i, s_r) = -\mathbb{I}_{\mathbf{l}_i=\mathbf{l}_r \wedge \mathbf{D}_{KL}(\mathbf{v}_i||\mathbf{v}_r)>\theta}.$$

Herein, $\mathbb{I}_a$ is an indicator. Its value is 1, if condition a is valid; otherwise, its value is 0.
(2) $score_{\mathrm{P}i} = score_{\mathrm{P}}(s_i, s_r)$ measures a *reward* value if s_i's posterior probability fits close to the distribution of the class labels provided by s_r.

$$score_{\mathrm{P}}(s_i, s_r) = \frac{1}{c}\sum_{k=1}^{c}[2 - (1 - \mathbf{v}_i(l_{rk}))^2 - \sum_{l_r \in \Omega/\{l_{rk}\}} \mathbf{v}_i(l_r)^2].$$

Herein, c is the number of possible class labels; $\mathbf{v}_i(l)$ means the posterior probability for label l and $(\mathbf{v}_i(l_1), ..., \mathbf{v}_i(l_c))$ constitute $\mathbf{v}_i$ with constraint $\sum_{k=1}^{c} \mathbf{v}_i(l_k) = 1$. Concretely, $\mathbf{v}_i(l_{rk}) = \mathbf{Pr}(l_{rk}|l_{ik})$ represents i's the posterior probability for label $l_{rk} \in \Omega$. If $\mathbf{v}_i(l_{rk}) = \mathbf{Pr}(l_{rk}|l_{ik} = l_{rk})$, the score value is maximized, being equal to 2. If $\mathbf{v}_i(l_{rk}) = \mathbf{Pr}(l_{rk}|l_{ik} \neq l_{rk})$, the score value is minimized, being equal to 0.

With the definitions above, the value of $(score_{\mathrm{I}i} + score_{\mathrm{P}i} + 1)$ falls in the range $[0, 3]$. Also, it is worth noting that the scoring rule consisting of the two scores has been proved *strictly Bayes-Nash incentive-compatible* relying on *stochastic relevance* in [16]. It means that a truthful prediction is always configured with a higher score compared to an untruthful prediction so as to achieve the goal of fairness (refer it to Section III-C).

Considering the potential aborting of a participating provider, we revise our payment function. If provider i does not abort, her deposit d_0 should be refunded, that is, $p_i(s_i) = p_i(s_i) + d_0$. Otherwise, her deposit d_0 will be forfeited.

D. Analysis

In this section, we proceed to analyze how to achieve the design objectives in Section V-B by using the presented pricing function as an approximate solution.

To begin with, we quantify the cost function with respect to different participating providers, which is useful to estimate the providers' expected utility. For simplicity, we assume that the truthful costs spent by participating providers are known, which refers to the *complete information* scenario. The assumption can be reasonable due to a bunch of existing arts for motivating cost truthfulness [21, 22, 25, 26]. Their costs derive from two identical cost parameters $c_1 > 0$ and $c_2 > 0$ which are far smaller than a user's budget B. We assume that the cost of generating a product linearly increases with the product's quality. Recall that we measure the truthfulness of a prediction via two scores mentioned in Section V-C. Thus, we next naturally regard the two scores as the quality metric to calculate the corresponding cost of every strategy s_i. That is, $c(s_i) = c_1 \cdot (score_{\mathrm{I}i} + score_{\mathrm{P}i} + 1) + c_2$. It is noteworthy that the cost monotonically increases with higher $(score_{\mathrm{I}i} + score_{\mathrm{P}i} + 1)$.

We are now ready to analyze that with our pricing mechanism, there exists a BNE achieving our design objectives via parameter constraints. Specifically, we set constraint conditions on parameter α_i considering the design objectives of individual rationality and budget feasibility, based on which we find a BNE, where all participants adopt the strategy of offering truthful predictions. Below, we demonstrate and prove this finding by Theorem 1.

Theorem 1. *In the non-cooperative game, there exists a BNE* $\mathbf{S}^* = (s_1^*, ..., s_m^*)$*, where every participating provider* $i \in \{1, ..., m\}$ *provides* s_i^* *containing* $\mathbf{l}_i = \mathbf{l}_r$ *and* $\mathbf{D}_{KL}(\mathbf{v}_i||\mathbf{v}_r) \leq \theta$ *compared with* s_r^* $(r \neq i)$ *when parameter* α_i *satisfies (1)* $\alpha_i \geq \frac{c_1}{2 \cdot score_i}$*, (2)* $\alpha_i \geq \frac{c_1 \cdot score_i + c_2}{score_i^2}$ *and (3)* $\alpha_i \leq \frac{B}{nm} \cdot \frac{1}{score_i^2}$*, where* $score_i = (score_{Ii} + score_{Pi} + 1)$.

Proof. Given other participating providers' strategies $\mathbf{S}_{-s_i}^*$ and a randomly selected provider's strategy s_r^*, every provider i can estimate her expected utility by

$$\begin{aligned}\mathbb{E}_{\mathbf{S}_{-s_i}}[u_i(s_i, \mathbf{S}_{-s_i})|s_r^*] &= \mathbb{E}_{\mathbf{S}_{-s_i}}[p_i(s_i, \mathbf{S}_{-s_i})|s_r^*] - c(s_i) - d_0 \\ &= \alpha_i \times (score_{\mathrm{I}i} + score_{\mathrm{P}i} + 1)^2 + d_0 \\ &\quad - c_1 \cdot (score_{\mathrm{I}i} + score_{\mathrm{P}i} + 1) \\ &\quad - c_2 - d_0.\end{aligned}$$

Here, we suppose that provider i does not abort. If she aborts, apparently her expected utility is equal to $-d_0$ which is negative. For every provider i not aborting, she can maximize her expected utility when her strategy $s_i^* = (\mathbf{l}_i, \mathbf{v}_i)$ leads to $score_i$ reaching the maximum among $[\frac{c_1}{2\alpha_i}, 3]$. Therefore, every rational provider i is doomed to chose the strategy which enables $score_i$ being equal to 3. To be more clear, we summarize the possible cases for every provider i's strategy and her expected utility as following:
(a). If $s_i^* = (\mathbf{l}_i, \mathbf{v}_i)$, where $\mathbf{l}_i \neq \mathbf{l}_r$, her expected utility is negative due to $(score_{\mathrm{I}i} + score_{\mathrm{P}i} + 1) = 0$ leading to $\mathbb{E}_{\mathbf{S}_{-s_i}}[u_i(s_i, \mathbf{S}_{-s_i})|s_r^*] = -c_2$.
(b). If $s_i^* = (\mathbf{l}_i, \mathbf{v}_i)$, where $\mathbf{l}_i = \mathbf{l}_r$ and $\mathbf{D}_{KL}(\mathbf{v}_i||\mathbf{v}_r) \leq \theta$, her expected utility is equal to $9\alpha_i - 3c_1 - c_2$ which is positive due to parameter constraint (2), and maximized due to $(score_{\mathrm{I}i} + score_{\mathrm{P}i} + 1) = 3$.
(c). If abort, her expected utility is negative due to $\mathbb{E}_{\mathbf{S}_{-s_i}}[u_i(s_i, \mathbf{S}_{-s_i})|s_r^*] = -d_0$.

Hence, strategy profile $\mathbf{S}^* = (s_1^*, ..., s_m^*)$ in Theorem 1, where s_i^* satisfies $\mathbf{l}_i = \mathbf{l}_r$ and $\mathbf{D}_{KL}(\mathbf{v}_i||\mathbf{v}_r) \leq \theta$ is a BNE. □

VI. Experiment

A. Implementation and Setup

Prediction Aggregation with TEEs. We initialize TEEs by utilizing SGX SDK of version 2.5. In the SGX environment, we implement the prediction aggregation program (*i.e.*, Algorithm 1) by using C/C++ programming language.

Smart Contract. We also implement the PS contract with the Solidity programming language of Ethereum and deploy it on the Ropsten Test Network via MetaMask[2].

Dataset. We totally use three datasets to simulate three prediction tasks. Specifically, we use two well-studied image datasets, including MNIST[3] and ImageNet[4] for image prediction, as well as a public text dataset, namely 20 Newsgroups[5] for text prediction. With respect to three datasets, we will correspondingly sample a number of test data for evaluation. Note that MNIST, ImageNet and 20 Newsgroups contain 10K, 100K and near 8K test data, respectively. More concrete information of the three datasets are shown in TABLE III.

TABLE III: Real-world datasets used in the experiment.

Dataset	Type	Size	Features	Labels
MNIST	Image	70K	20x20	10
ImageNet	Image	1.26M	224x224x3	1000
20 Newsgroups	Text	18846	–	20

Provider Simulation. We collect three groups of various trained models which are used to simulate providers for serving prediction. We separately collect 6, 10 and 15 models under various frameworks which are evaluated on MNIST, 20 Newsgroups and ImageNet. Specifically, we implement and train the models for the MNIST and 20 Newsgroups by ourselves, and download off-the-shelf models for ImageNet from two public model sources[6,7].

We simulate distrustful predictions by perturbing normal predictions, where perturbations are sampled from the uniform distribution on interval $(0, 1)$. Besides, we simulate a distrusting provider by perturbing a model's all predictions.

We will consider three cases in perturbing predictions of models, including (a) no perturbation, (b) perturbing no more than $\frac{M}{2}$ models' predictions, and (c) perturbing more than $\frac{M}{2}$ models' predictions, where M is the total number of models. Note that case (a) is used to simulate the BNE setting induced by Theorem 1, where each provider is incentivized to provide truthful predictions; case (c) creates the setting, where providers lack sufficient motivation for prediction truthfulness; case (b) refers to the setting between case (a) and (c).

In addition, our experiments are conducted in a Ubuntu 16.04 server equipped with a CPU of 3.40GHz, 32 GB RAM and a GPU of Nvidia GTX-1080.

[2]https://metamask.io/
[3]http://yann.lecun.com/exdb/mnist/
[4]http://www.image-net.org/challenges/LSVRC/2012/
[5]http://qwone.com/ jason/20Newsgroups/
[6]https://keras.io/api/applications/
[7]https://pytorch.org/docs/stable/torchvision/models.html

B. Evaluation

Our evaluation is four-fold: (*i*) To highlight the advantage of Algorithm 1, we compare the accuracy of predictions generated by Algorithm 1 and that by averaging (a traditional ensemble strategy); (*ii*) To demonstrate the effectiveness of the incentive mechanism, we plot and compare simulation results of prediction aggregation regarding case (a), (b) and (c) in terms of accuracy; (*iii*) To show service cost, we estimate the computation complexity of prediction aggregation with a TEE and evaluate gas costs caused by the interaction between the PS contract and the TEE; (*iv*) To answer whether or not prediction aggregation via Algorithm 1 is effective to resist membership inference attacks, we conduct state-of-the-art attacks [43] and present empirical evidences.

TABLE IV: Accuracy comparison.

Dataset	Avg.	Label	Rank	Probability
MNIST	0.907	0.978	0.973	0.981
ImageNet	0.724	0.790	0.764	0.789
20 Newsgroups	0.721	0.862	0.836	0.862

First of all, as shown in TABLE IV, for each dataset, the accuracy of the predictions generated by Algorithm 1 regrading three output formats (*i.e.*, 3_{th} to 5_{th} column) is always better than the averaging accuracy (*i.e.*, 2_{th} column) of all participating models. We can see that on ImageNet dataset, the accuracy improvement is relatively small, but as pointed out by [44], spending a lot of time and energy to achieve minor accuracy improvement on difficult object recognition task is deserved.

Second, Fig. 2 and Fig. 3 show the accuracy of aggregated predictions regarding three perturbation cases on MNIST, 20 Newsgroups and ImageNet. For each dataset, it can be clearly seen that the accuracy in case (a) is always higher than that in case (b) and (c), which is because that participating providers offer truthful predictions with sufficient incentives. We also can see that in case (c), where a vast majority of participating providers report meaningless predictions, the accuracy is never better than 0.5. The reason is that Algorithm 1 fails to learn the truth when a majority of predictions are not enough accurate, and thus our incentive mechanism is necessary to handle case (c). In addition, from Fig. 2, the evaluated accuracy slightly grows up with the increasing queries. According to Fig. 3, we also notice that the accuracy of the rank-level predictions on ImageNet drops more obviously than the other two datasets in more serious perturbation cases. It might be caused by the large number of labels, *i.e.*, 1000, on ImageNet dataset.

Third, Fig. 4 presents the estimated time costs of prediction aggregation inside the TEE over three datasets. Note that we omit the one-time cost of setting up a TEE. Clearly, more queries spend more times. By comparing the three sub-figures, we also can know that the time complexity becomes higher as the number of labels of the query task increases. Recall that the number of class labels of MNIST, 20 Newsgroups and ImageNet is 10, 20 and 1000, respectively. Besides, gas costs are mainly derived from two parts: (1) execution costs of the PS contract when its three entry points, Deposit, Request and Response, are correspondingly invoked, and (2) execution

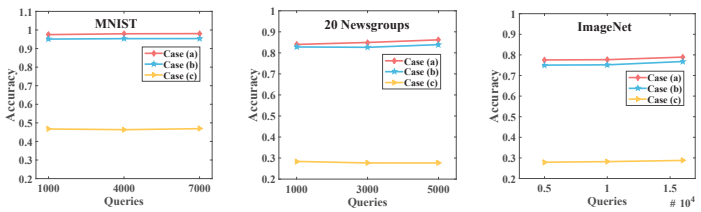

Fig. 2: Accuracy with increasing queries.

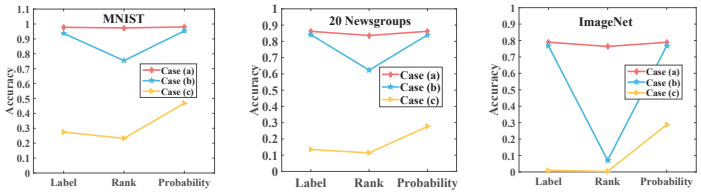

Fig. 3: Accuracy on different-format predictions.

costs of the TEE's transaction (on entry point Response) which contains outputs $outp$, $outp_{attr}$ and signatures σ, σ_{attr} (entirely 2×70 bytes). Also, the gas costs grow up with increasing participating providers. Note that encrypted input data and predictions are transmitted off-chain, and thus the magnitude of query makes negligible effect on the gas costs. We only test the gas costs by simulating 6 providers (on MNIST). Specifically, part (1) totally spends $510,815$ units gas, including $389,373$ units for Deposit, $102,200$ units for Request and $19,242$ units for Response. The gas costs for sending the response transaction in part (2) are about $74,370$ units.

TABLE V: Comparison of adversaries' attack performance.

Type	Target model	Precision	Recall
Adversary 1	Single	0.996	0.503
	Ensemble	0.056	0.054
Adversary 2	Single	0.997	0.504
	Ensemble	0.987	0.499

Last, we launch membership inference attacks using two types of adversaries with increasingly strong attack capabilities in prior work (*i.e.*, adversary 1 and 2, detailed in [43]'s TABLE I) and show the attack results. Similar to the work [43], we adopt three models as an ensemble, but the difference is that our ensemble strategy is Algorithm 1 rather than stacking. Besides, the used three models are CNN, RNN and MLP trained on the MNIST dataset. For comparison, we also conduct the same attacks on the single CNN model. As shown in TABLE V, the attack results demonstrate that ensemble model under Algorithm 1 is able to reduce the attack performance of adversary 1, but not adversary 2. Concretely, for adversary 1, the precision drops from 0.996 to 0.056 and the recall drops from 0.503 to 0.054. But for adversary 2, there has no effect. It is hard to suggest an explanation with confidence for the attack results like the previous work [43].

VII. Limitation and Future Work

Other prediction formats. Our work focuses on the prediction formats, including top-1 label, ranked labels and posterior probability, but fails to support other formats, such as text data, in Natural language processing (NLP) tasks. Taking language translation as an example, Sequence-to-Sequence models are usually used, which take as input a sequence of words in certain language and output another sequence of words in a target language, where output format belongs to text data.

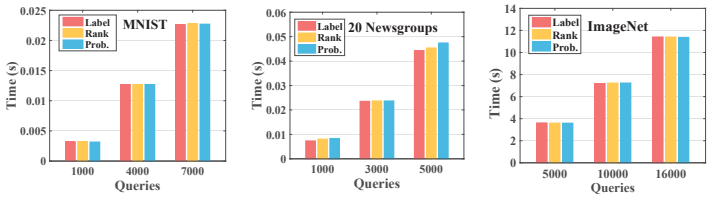

Fig. 4: Time complexity of prediction aggregation inside a TEE.

Adversarial examples. We assume benign input data and do not consider adversarial examples (AEs), *i.e.*, input data injected with imperceptible perturbations [35]. AEs can mislead a deep neural network to incorrectly classify an originally correctly classified input. Recently, a promising approach against AEs is to create a robust ensemble model by carefully considering the diversity of individual models [45, 46]. In our future work, we will follow this direction and take into account the factors regarding model diversity to refine our incentive mechanism for FedServing.

VIII. Conclusion

In this paper, we present a prediction serving framework, named as FedServing, towards trained models from various sources. FedServing enables locally deploying models and provides collective prediction services for charging users. For motivating truthful predictions, we customize an incentive mechanism based on Bayesian game theory. For boosting prediction accuracy, we use truth discovery algorithms working jointly with the incentive mechanism to eliminate the effect of low-accuracy predictions. Our proposed design supports popular prediction formats, including top-1 label, ranked labels and posterior probability. Besides, we build FedServing on the blockchain to ensure exchange fairness and leverage TEEs to securely aggregate predictions as well. With extensive experiments, we effectively validate the expected properties of our mechanism and empirically demonstrate its capability of reducing the risk of certain membership inference attack.

Acknowledgment

Jian Weng was supported by the National Key Research and Development Plan of China under Grant Nos. 2018YFB1003701 and 2020YFB1005600, the National Natural Science Foundation of China under Grant Nos. 61825203, U1736203 and 61732021, the Major Program of Guangdong Basic and Applied Research Project under Grant No. 2019B030302008, the Guangdong Provincial Science and Technology Project under Grant No. 2017B010111005. Dr. Cong Wang was supported by the Research Grants Council of Hong Kong under Grant CityU 11217819 and Grant CityU 11217620, the Innovation and Technology Commission of Hong Kong under ITF Project ITS/145/19.

REFERENCES

[1] "Global machine learning market research report," https://www.marketresearchfuture.com/reports/machine-learning-market-2494, 2019.

[2] G. Bello-Orgaz, J. J. Jung, and D. Camacho, "Social big data: Recent achievements and new challenges," *Information Fusion*, vol. 28, pp. 45–59, 2016.

[3] C. Song, T. Ristenpart, and V. Shmatikov, "Machine learning models that remember too much," in *Proc. of ACM CCS*, 2017.

[4] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "{GAZELLE}: A low latency framework for secure neural network inference," in *Proc. of USENIX Security*, 2018.

[5] F. Mireshghallah, M. Taram, P. Ramrakhyani *et al.*, "Shredder: Learning noise distributions to protect inference privacy," in *Proc. of ASPLOS*, 2020.

[6] F. Tramer and D. Boneh, "Slalom: Fast, verifiable and private execution of neural networks in trusted hardware," in *ICML*, 2018.

[7] D. Peng, F. Wu, and G. Chen, "Pay as how well you do: A quality based incentive mechanism for crowdsensing," in *Proc. of ACM MobiHoc*, 2015.

[8] S. Yang, F. Wu, S. Tang *et al.*, "On designing data quality-aware truth estimation and surplus sharing method for mobile crowdsensing," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 4, pp. 832–847, 2017.

[9] H. Jin, L. Su, and K. Nahrstedt, "Theseus: Incentivizing truth discovery in mobile crowd sensing systems," in *Proc. of ACM MobiHoc*, 2017.

[10] P. Sun, Z. Wang, Y. Feng *et al.*, "Towards personalized privacy-preserving incentive for truth discovery in crowdsourced binary-choice question answering," in *INFOCOM*, 2020.

[11] X. Gong and N. Shroff, "Incentivizing truthful data quality for quality-aware mobile data crowdsourcing," in *Proc. of ACM MobiHoc*, 2018, pp. 161–170.

[12] B. Zhao, S. Tang, X. Liu, and X. Zhang, "Pace: privacy-preserving and quality-aware incentive mechanism for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, 2020.

[13] U. G. Mangai, S. Samanta, S. Das, and P. R. Chowdhury, "A survey of decision fusion and feature fusion strategies for pattern classification," *IETE Technical review*, vol. 27, no. 4, pp. 293–307, 2010.

[14] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE S&P*, 2017.

[15] N. Papernot, "A marauder's map of security and privacy in machine learning: An overview of current and future research directions for making machine learning secure and private," in *Proc. of ACM AISec*, 2018, pp. 1–1.

[16] G. Radanovic and B. Faltings, "Incentives for truthful information elicitation of continuous signals," in *AAAI*, 2014.

[17] D. Crankshaw, X. Wang, G. Zhou, M. J. Franklin, J. E. Gonzalez, and I. Stoica, "Clipper: A low-latency online prediction serving system," in *Proc. of USENIX NSDI*, 2017.

[18] Y. Lee, A. Scolari, B.-G. Chun, M. D. Santambrogio, M. Weimer, and M. Interlandi, "Pretzel: Opening the black box of machine learning prediction serving systems," in *Proc. of USENIX OSDI*, 2018.

[19] Y. Zhang and M. Van der Schaar, "Reputation-based incentive protocols in crowdsourcing applications," in *INFOCOM*, 2012.

[20] T. Luo, H.-P. Tan, and L. Xia, "Profit-maximizing incentive for participatory sensing," in *INFOCOM*, 2014.

[21] H. Jin, L. Su, B. Ding, K. Nahrstedt, and N. Borisov, "Enabling privacy-preserving incentives for mobile crowd sensing systems," in *IEEE ICDCS*, 2016.

[22] H. Jin, L. Su, H. Xiao, and K. Nahrstedt, "Inception: Incentivizing privacy-preserving data aggregation for mobile crowd sensing systems," in *Proc. of ACM MobiHoc*, 2016.

[23] H. Jin, L. Su, and K. Nahrstedt, "Centurion: Incentivizing multi-requester mobile crowd sensing," in *INFOCOM*, 2017.

[24] H. Jin, H. Guo, L. Su, K. Nahrstedt, and X. Wang, "Dynamic task pricing in multi-requester mobile crowd sensing with markov correlated equilibrium," in *INFOCOM*, 2019.

[25] Q. Zhang, Y. Wen, X. Tian, X. Gan, and X. Wang, "Incentivize crowd labeling under budget constraint," in *INFOCOM*, 2015.

[26] H. Wang, S. Guo, J. Cao, and M. Guo, "Melody: A long-term dynamic quality-aware incentive mechanism for crowdsourcing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 4, pp. 901–914, 2017.

[27] X. Zhang, G. Xue, R. Yu, D. Yang, and J. Tang, "Truthful incentive mechanisms for crowdsourcing," in *INFOCOM*. IEEE, 2015, pp. 2830–2838.

[28] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing," in *Proc. of MobiCom*, 2012.

[29] K. Han, H. Huang, and J. Luo, "Posted pricing for robust crowdsensing," in *Proc. of ACM MobiHoc*, 2016.

[30] D. Zhao, X.-Y. Li, and H. Ma, "How to crowdsource tasks truthfully without sacrificing utility: Online incentive mechanisms with budget constraint," in *INFOCOM*, 2014.

[31] X. Zhang, Z. Yang, Z. Zhou, H. Cai, L. Chen, and X. Li, "Free market of crowdsourcing: Incentive mechanism design for mobile sensing," *IEEE transactions on parallel and distributed systems*, vol. 25, no. 12, pp. 3190–3200, 2014.

[32] Y. Chen, B. Li, and Q. Zhang, "Incentivizing crowdsourcing systems with network effects," in *INFOCOM*, 2016.

[33] C. Huang, H. Yu, J. Huang, and R. A. Berry, "Crowdsourcing with heterogeneous workers in social networks," in *IEEE GLOBECOM*, 2019.

[34] H. Zhang, B. Liu, H. Susanto, G. Xue, and T. Sun, "Incentive mechanism for proximity-based mobile crowd service systems," in *INFOCOM*, 2016.

[35] C. Szegedy, W. Zaremba *et al.*, "Intriguing properties of neural networks," https://arxiv.org/abs/1312.6199, 2013.

[36] T. Hunt, Z. Zhu, Y. Xu, S. Peter, and E. Witchel, "Ryoan: A distributed sandbox for untrusted computation on secret data," *Proc. of TOCS*, 2018.

[37] A. Ahmad, K. Kim, M. I. Sarfaraz, and B. Lee, "Obliviate: A data oblivious filesystem for intel sgx." in *Proc. of NDSS*, 2018.

[38] G. Kaptchuk, M. Green, and I. Miers, "Giving state to the stateless: Augmenting trustworthy computation with ledgers," in *Proc. of NDSS*, 2019.

[39] J. R. Douceur, "The sybil attack," in *Proc. of IPTPS*, 2002.

[40] L. I. Kuncheva, "Switching between selection and fusion in combining classifiers: An experiment," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 32, no. 2, pp. 146–156, 2002.

[41] Q. Li, Y. Li, J. Gao, B. Zhao, W. Fan, and J. Han, "Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation," in *Proc. of ACM SIGMOD*, 2014.

[42] L. Su, Q. Li, S. Hu, S. Wang, J. Gao, H. Liu, T. F. Abdelzaher, J. Han, X. Liu, Y. Gao *et al.*, "Generalized decision aggregation in distributed sensing systems," in *IEEE RTSS*, 2014.

[43] A. Salem, Y. Zhang, M. Humbert *et al.*, "Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models," in *Proc. of NDSS*, 2019.

[44] O. Russakovsky, J. Deng, H. Su, J. Krause *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[45] T. Pang, K. Xu, C. Du, N. Chen, and J. Zhu, "Improving adversarial robustness via promoting ensemble diversity," in *ICML*, 2019.

[46] L. Liu, W. Wei, K.-H. Chow, M. Loper *et al.*, "Deep neural network ensembles against deception: Ensemble diversity, accuracy and robustness," in *IEEE MASS*, 2019.