IMPROVING UTILITY AND EFFICIENCY FOR PRIVACY PRESERVING DATA ANALYSIS

BY

BINGYU LIU

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science
in the Graduate College of the
Illinois Institute of Technology

Approved Yuan Hong
Adviser

Chicago, Illinois
December 2022

ACKNOWLEDGEMENT

First and foremost, I express my sincere gratitude to my advisor Professor Yuan Hong for all his advice and patience during the past 5 years and 2 months since I joined Illinois Tech as a Ph.D. student. His insightful ideas and suggestions have always helped me improve the quality of my research works. He has also advised me on technical and academic writing, research presentations, and technical reviews. Without his assistance at every stage of the research, I cannot finish this dissertation and continue my academic pursuit.

I would also like to thank my dissertation committee members, including Professor Rujia Wang, Professor Yue Duan and Professor Lulu Kang for their valuable comments and constructive suggestions.

Many thanks to my current lab colleagues Dr. Shangyu Xie, Han Wang and Dr. Meisam Mohammady for their assistance and valuable advice in my research. I would like to express my deepest gratitude to my husband Dr. Tianjiao Liu and my parents for their supports. They will always be right behind, whenever I needed.

I have also received a lot of inputs and supports from the Massachusetts General Hospital (MGH) and Harvard Medical School (HMS), where I have been working as a research assistant since 2022 summer. I am so grateful to all of my collaborators who have greatly helped me on medical imaging, especially Michelle Chua M.D. and Dr. Synho Do.

Last but not least, I would like to thank everyone not mentioned here who helped me along the way. Thanks for the guidance, friendship, and support.

AUTHORSHIP STATEMENT

I, Bingyu Liu, attest that the work in this thesis is substantially my own. In accordance with the disciplinary norm of computer science (see IIT Faculty Handbook, Appendix S), the following collaborations occurred in the thesis:

Professor Yuan Hong, as my Ph.D. supervisor, contributes to the general research ideas, helps the paper writing and guides the experimental evaluation. Some of the material presented here were the work of his collaborations. Professor Rujia Wang conducts the data analysis in the Chapter 4 and Chapter 5.

Shangyu Xie, Han Wang, Meisam Mohammady, Yuanzhou Yang, and Shanglin Zhou contribute to the design of experiments and interpretations in the Chapter 1, Chapter 4 and Chapter 5.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

In recent decades, the smart cities are incorporating with Internet-of-Things (IoT) infrastructures for improving the citizens' quality of life by leveraging information/data. The huge amount of data is extracted and generated from the devices (e.g., mobile applications, GPS navigation systems, urban traffic cameras, etc.), or city sectors such as Intelligent Transportation Systems (ITS), Resource Allocation, Utilities, Crime Detection, Hospitals, and other community services.

This dissertation aims to systematically research the Data Analysis in IoT System, which mainly consists of two aspects: *Utility* and *Efficiency*. First, ITS as a representative system in IoT in the smart city, I present the work on privacy preserving for the trajectories data, which is achieved by the differential privacy technique with a novel sanitation framework. Moreover, I have studied the resource allocation problem in two different approaches: *Cryptographic computation* and *Hardware enclaves* with the utility and efficiency accordingly. For the *Cryptographic computation* approach, I utilize Secure Multi-party Computation (SMC) technique for achieving the privacy-aware divisible double auction without a mediator. Besides, I also propose a hardware-based solution Trusted Execution Environment (TEE) for performance improvement. At the same time, integrity and confidentiality are also able to be guaranteed. The proposed hybridized Trusted Execution Environment (TEE)-Blockchain System is designed for securely executing smart contract. Finally, I have studied the Cryptographic Video DNN Inference for the smart city surveillance, which privately inferring videos (e.g., action recognition, and video, and classification) on 3D spatial-temporal features with the C3D and I3D pre-trained DNN models with high performance. This dissertation proposes the privacy preserving frameworks and mechanisms are able to be applied efficiently for IoT in the real-world.

CHAPTER 1

INTRODUCTION

The purpose of the the smart cities is to improve the people' quality of life via heterogeneous data sources, which are collected by IOT infrastructures and devices. With respect to the Intelligent Transportation System (ITS) area, the fine-grained vehicle trajectory data can be collected from the GPS navigation system, mobile application, urban traffic cameras. After that, it can be analyzed to significantly promote the development of ITS and urban traffic optimization (e.g., optimizing the mobility of urban traffic, and learning the signalized phases of traffic lights [1]) and the smart cities. Detecting the optimization allocation of divisible resource is one of purposes for the smart cities. Divisible resources (e.g., electricity, mobile data, and computation and storage resources in the cloud) have been frequently traded or allocated in a peer-to-peer mode for the smart cities. In order to seek the optimization for the resource allocation, the resource data are collected from different markets (e.g., electricity markets, cloud markets, financial markets and wireless network). All the agents can purchase or sell any amount of the resources in such markets. The strategies are made to improve the resource allocation performance, even make the reactive processes to become predictive as well. The crime prevention and community safety are also domains of the smart cities with IOT support. In fact, as number of the crimes and violence are increasing dramatically, it leads to the ubiquitous usage of surveillance cameras in the smart cities. Smart surveillance cameras are needed to monitor and distinguish the anomalies in real-time. Equipped with the neural network technique, the surveillance cameras are able to identify automatically the violent attack for deterring criminals.

Analyzing such fine-grained data (e.g., vehicle trajectories data, resource allocation data and surveillance cameras data) would significantly benefit the develop-

ment of smart cities, however, severe privacy and security problems are posed during the process. In ITS, trajectory data record the temporal pattern of locations, speeds and accelerations in multi-dimensional way. However, directly releasing or sharing such datasets for analyses would pose severe privacy concerns to vehicles and their drivers [2,3]. Specifically, sequences of locations can reveal a driver's frequently visited positions (e.g., residence, hospital) and preferred routes. Other attributes (i.e., speed and acceleration) can reveal his/her driving habits. Although vehicle/driver identities (i.e., VIN number and driver license numbers) have been replaced with pseudo-IDs in such datasets, privacy risks have not been addressed as re-identification attacks can still be applied to the dataset with certain background knowledge. For instance, if an adversary knows that a driver has visited some locations at specific times, even a small part of known traces can make the individual's entire data vulnerable to re-identification. After re-identification, it will be readily to track the driver/vehicle over any time period, learning all visited locations (e.g., hospital, gas station, office and residence) as well as his/her driving habits. For this reason, in the past decade, privacy concerns in some *similar datasets* have attracted significant interests [2, 4–7]. That most of the existing work either do not rely on a formal privacy notion (e.g., VTLs based techniques [2, 8]), or result in very limited utility (e.g., not fine-grained, without moving attributes and timestamps). To address all the above limitations, a novel privacy preserving technique is proposed to sanitize fine-grained vehicle trajectories (*all the attributes*) with differential privacy [9, 10], which provides rigorous privacy guarantee in datasets against arbitrary background knowledge. It ensures that adding or removing all trajectory of each vehicle (*all the attributes*) does not result in significant privacy risks. More details can be found in the chapter 2.

In the chapter 3, the generic divisible resource allocation in the smart cities is studied. There are a set of agents and divisible resources (i.e., electricity, computation and storage resources, stock shares, bandwidth). Since all the agents generally

compete with each other to maximize their payoffs, divisible double auction mechanisms [11] are designed to allow both buyers and sellers to dynamically submit their prices until convergence (e.g., achieving the Nash Equilibrium [12,13]) and then complete the transaction with resource allocation.

In such markets, each agent may sell resources with arbitrary amounts to any other buyers, and all the agents generally compete with each other by seeking for their maximum payoffs. Then, auction mechanisms have been extensively studied for exchanging such divisible resources to achieve the Nash Equilibrium [12, 13]. Since auctions request all the potential buyers to propose bid prices [13, 14] (in particular, double auction [15] requests both potential buyers and sellers to simultaneously submit their prices), a trusted-third party is established as the *market mediator* to coordinate the bidding and resource allocation in the auctions. The establishment of the mediator may result in high operational costs, extra charges to buyers/sellers, high computation burden, and high demand of trust on the mediator.

If directly eliminating the mediator in the auction, severe privacy concerns may occur since all the agents should disclose their local private data for completing the auction. In addition, some agents may try to win more payoffs in the auction by reporting untruthful bids, especially in sealed-bid auctions [16]. Even worse, agents (aka. potential buyers or sellers) may collect such information from their competitors [17], and misuse such private data, e.g., reselling the data (a mediator may also do so).

In the chapter 3, a novel auction framework (namely PANDA) is proposed by designing an efficient cryptographic protocol among all the buyers and sellers to privately execute double auction for divisible resources. Specifically, the cryptographic protocol is constructed with the fundamental cryptographic primitives: Homomorphic Encryption (HE) [18, 19] and Secure Function Evaluation (SFE) [20]. Then,

the cryptographic protocol enables all the agents to securely communicate with each other and complete the transactions with limited information disclosure. Per the secure multiparty computation (MPC) theory [21, 22], the cryptographic protocol can be proven to be equivalent to a mediator. Furthermore, a double auction [11] is designed based on the Vickrey-Clarke-Groves (VCG) [23, 24] mechanism in PANDA to ensure truthfulness.

Compared with other types of secure and private solutions (e.g., Secure Multiparty Computation (SMC) [18,25,26]), hardware-based solution TEE achieves stronger security and high efficiency for blockchain execution [27]. Thus, in the chapter 4, an efficient and privacy preserving divisible double auction with the TEE-Blockchain hybridized system (e.g., on the Intel SGX, which is a TEE supported by an architecture extension of Intel [28]) is proposed.

Moreover, the privacy concerns issue related to the smart surveillance video will be studied. To ensure public safety, the smart cities use the smart surveillance cameras to identify the violent attack for deterring criminals. The Surveillance video collects several realistic anomalies in real time. Due to the application of deep neural network, smart surveillance cameras are able to detect the real-time crime by human action recognition and classification. Yet, privacy leakage may be the paramount problem. These sensitive information, such as human face, users' home location, workspace and even the license plates of car, are recorded via surveillance video during the monitoring. In the chapter 5, the privacy of video inference is studied. Deep neural network (DNN) services have been widely deployed for efficient and accurate learning in many different domains. For instance, a client may send its private input data (e.g., images, text messages and videos) to the cloud, which provides the inferences (e.g., classification and prediction) with the pre-trained DNN models. However, significant privacy concerns would emerge in such use cases due to data or model sharing with

the cloud. *Secure inferences* with cryptographic techniques have been proposed to address such issues, and the system can perform *secure two-party inferences* between each client and cloud. However, most of the existing cryptographic systems only focus on DNNs for extracting 2D features for image inferences, which have major limitations on latency and scalability for extracting spatio-temporal (3D) features from videos for accurate inferences. To address such critical deficiencies on cryptographic DNN for video inferences, we design and implement the first cryptographic two-party inference system, `Crypto3D`, which privately infers videos on 3D features with rigorous privacy guarantees. More details can be found in chapter 5.

CHAPTER 2

DIFFERENTIAL PRIVACY ON TRAJECTORY DATA

With the rapidly growing deployment of intelligent transportation systems (ITS) and smart traffic applications, vehicle trajectory data are ubiquitously generated, e.g., from GPS navigation systems, mobile applications, and urban traffic cameras. Analyzing such fine-grained data would greatly benefit the development of ITS and smart cities, yet pose severe privacy risks due to the recorded drivers' visited locations, routes, and driving habits. Recently, some privacy enhancing techniques are proposed to sanitize such data. However, such schemes have some major limitations – they either lack formal privacy notions to quantify and bound the privacy risks, or result in very limited utility, e.g., only a sequence of locations or aggregated information can be released (without retaining the speeds, accelerations and the timestamps of vehicles). In this chapter, we propose a novel framework to sanitize the fine-grained *vehicle trajectories with differential privacy* (VTDP), which provides rigorous privacy protection against adversaries who possess arbitrary background knowledge. Our VTDP technique involves three phases of differentially private sampling, which sequentially generate all the three categories of data (besides a pseudo identity for each vehicle) – *position, moving, timestamps*. It also includes a *vehicle trajectory interpolation* procedure to further improve the output utility with the properties of fine-grained vehicle trajectory data.

The work presented in this chapter have been published at [29][1].

2.1 Background

With the rapidly growing deployment of intelligent transportation systems

[1]©2019, IEEE, Reprinted, with permission from Bingyu Liu, Shangyu Xie, Han Wang, Yuan Hong, Xuegang Ban, Meisam Mohammady, *VTDP: Privately sanitizing fine-grained vehicle trajectory data with boosted utility*

(ITS) and smart traffic applications, vehicle trajectory data are ubiquitously generated from GPS navigation systems, mobile applications (e.g., Uber), urban traffic cameras, roadside unit and connected vehicles to record temporal pattern of locations, speeds and accelerations for each vehicle in a fine-grained manner [30]. Such fine-grained time series data can be collected and analyzed to significantly promote the development of intelligent transportation systems, urban traffic optimization (e.g., optimizing the mobility of urban traffic, and learning the signalized phases of traffic lights [1]) and smart cities.

However, directly releasing or sharing such datasets for analyses would pose severe privacy concerns to vehicles and their drivers [2, 3]. Specifically, sequences of locations can reveal a driver's frequently visited positions (e.g., residence, hospital) and preferred routes. Other attributes (i.e., speed and acceleration) can reveal his/her driving habits. Although vehicle/driver identities (i.e., VIN number and driver license numbers) have been replaced with pseudo-IDs in such datasets, privacy risks have not been addressed as re-identification attacks can still be applied to the dataset with certain background knowledge [31]. For instance, if an adversary knows that a driver has visited some locations at specific times, even a small part of known traces can make the individual's entire data vulnerable to re-identification. After re-identification, it will be readily to track the driver/vehicle over any time period, learning all visited locations (e.g., hospital, gas station, office and residence) as well as his/her driving habits.

For this reason, in the past decade, privacy concerns in some *similar datasets* have attracted significant interests [2, 4–7, 32]. The existing techniques can be classified into two different categories: (1) data sanitization techniques [4–6], and (2) virtual trip lines (VTLs) [2, 8]. In the former category, each of the data sanitization techniques defines a privacy notion and proposes an algorithm to anonymize indi-

viduals or obfuscate the location traces while satisfying the defined privacy notion (e.g., generalization, suppression, or differential privacy [4–6]). However, most of such privacy preserving techniques can only generate either *spatially aggregated* data (e.g., traffic statistics [33–35]) or a sequence of locations *(by omitting the vehicle moving attributes, e.g., speed and acceleration, and even the timestamps*) [4–6]. Thus, the output utility would be constrained, and the privately released data (without indicators for traffic flow) may not function many urban traffic analyses for developing smart cities, which request the fine-grained data disclosure with moving/traffic information and timestamps [30, 36].

In the latter category, Hoh et al. [8] proposed the idea of virtual trip lines (VTLs) for protecting privacy, which are geographic markers that indicate where vehicles should provide location updates in their trajectories (*which are only a small subset of the trajectories around the signalized traffic intersections in general*). Ban and Gruteser [2] further showed that VTLs can be utilized to regulate location and speed reports, such that the data needs for intersection modeling (e.g., signal performance measurement) can be satisfied while simultaneously protecting privacy. However, the VTLs have the following limitations. First, the privacy risks in the output data cannot be formally quantified and bounded (e.g., via a privacy notion). Second, the output data are collected based on specific areas, and cannot span over the entire vehicle trajectories. Thus, the utility of the output data might be limited to only a few applications.

In summary, we argue that most of the existing work either do not rely on a formal privacy notion (e.g., VTLs based techniques [2, 8]), or result in very limited utility (e.g., not fine-grained, without moving attributes and timestamps). To address all the above limitations, we propose a novel privacy preserving technique to sanitize fine-grained vehicle trajectories (*all the attributes*) with differential privacy [9, 10],

which provides rigorous privacy guarantee in datasets against arbitrary background knowledge. It ensures that adding or removing all complete trajectory of each vehicle (*all the attributes*) does not result in significant privacy risks. Our differentially private scheme randomly samples the output data without aggregation while satisfying the defined rigorous privacy notion. Therefore, the major contributions of this paper are summarized as follows.

- To the best of our knowledge, this is the first work that sanitizes fine-grained vehicle trajectory data under differential privacy guarantee to generate *vehicle pseudo IDs, coordinates, speeds, accelerations*, and *timestamps*. We note that our technique can output one *record-per-0.1 second* fine-grained trajectories for vehicles, the existing work on trajectory sanitization [4–6] or privacy preserving traffic flows [37, 38] mainly consider incomplete or coarse-grained data., e.g., counts and occupancy times measured by the installed loop detectors on highways.

- We propose a novel sanitization framework (namely, VTDP) that includes three phases to sample all the attributes in sequence with differential privacy. Our framework also *interpolates* data to further improve the output utility by any untrusted data recipient.

- Our VTDP framework is proposed based on sampling mechanisms, which satisfy *non-interactive* differential privacy [39–42].

  Then, the non-aggregated output data (from non-interactive mechanisms) can be utilized for any utility-driven vehicle trajectory analysis such as traffic light signal phase learning and queue length estimation [43]. Furthermore, we propose a novel multi-phase sampling scheme which can efficiently compute the output trajectories from our fine-grained data, which ensuring both privacy and utility (see Example 1).

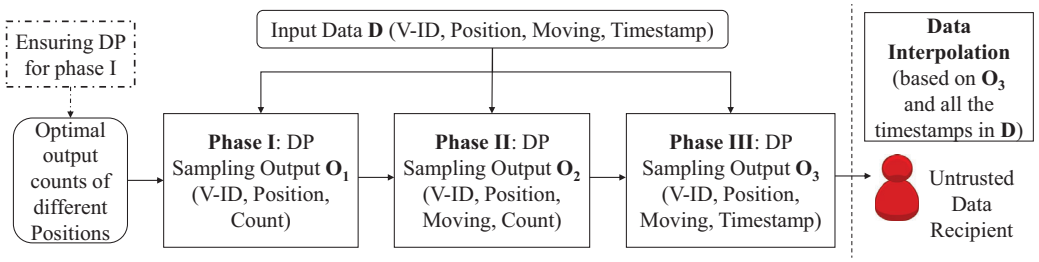


Figure 2.1. Data sanitization framework for VTDP (after preprocessing)

- We conduct experiments on real world vehicle trajectory datasets [44] (e.g., the NGSIM data [2]) and validate the effectiveness of our scheme.

2.2 Preliminaries

Let us use an example of the fine-grained vehicle trajectory data, e.g., the NGSIM data collected from traffic cameras. Such datasets include vehicle IDs (pseudo identities), lane ID ℓ, lateral/longitudinal coordinates of a position (x, y), speed v, acceleration a, day d, and time t, which belong to four different categories (*V-ID*, *position*, *moving* and *time*). Specifically, *position* consists of "lane" ℓ, "lateral/longitudinal coordinates" x and y where the coordinates x and y can uniquely determine its lane ℓ (thus we skip the lane in this paper).

2.2.1 Fine-grained Vehicle Trajectory Data. Moreover, we integrate "speed" (v) and "acceleration" (a) as the *moving* attributes, and the time includes "day" (d) and "time" (t). It is worth noting that all the position coordinates, speeds and accelerations are real numbers while the timestamps are discrete (e.g., with interval 0.1 second). To improve the output utility, all the position coordinates, speeds and accelerations can be approximated to discrete values (see Section 2.7.1).

Definition 1 (Vehicle Trajectory Data)**.** A collection of vehicles' fine-grained trajec-

[2]https://data.transportation.gov/api/views/8ect-6jqj

tories, each of which includes a pseudo-ID V_r denoting a vehicle, lane ID, coordinate (x, y), and moving speed v and acceleration a in day d at time t.

Vehicle trajectories can be formulated as above. In intelligent transportation systems (ITS), both vehicle speed and acceleration are considered as a part of vehicle trajectory data [2,8,30,45]. Thus, we define "vehicle trajectory data" to differentiate it from the definition of "trajectories" in the existing trajectory sanitization works (related to location-based services) [4–6].

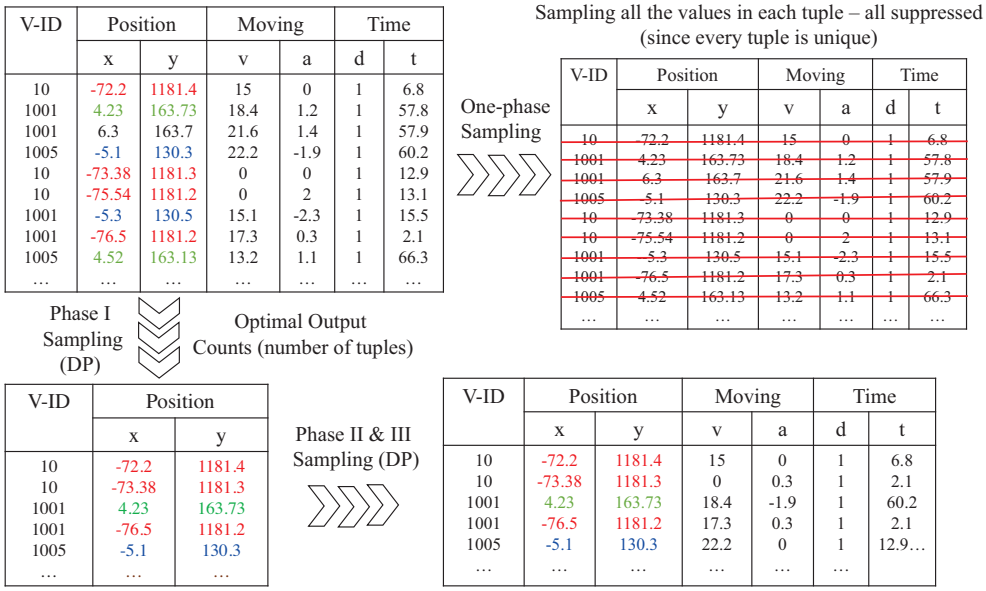
2.2.2 Privacy Notion. Before giving the definition of privacy notion, we first define vehicle trajectory in the dataset.

Definition 2. (VEHICLE TRAJECTORY) *Given a vehicle trajectory dataset D of n vehicles $V_1, \ldots, V_n$, vehicle trajectory $\Theta_r, r \in [1, n]$ is defined as all the tuples in D w.r.t. vehicle V_r.*

With the definition of vehicle trajectory, we consider two datasets D and D' as neighboring inputs if they differ in one vehicle trajectory Θ_r, which is the complete traveling data corresponds to any vehicle V_r. Thus, our differential privacy definition [4,39,44] would provide the guarantee that adding or removing *any vehicle trajectory* does not result in significant risk to the privacy of dataset. Although two neighboring inputs D and D' differ in only one vehicle trajectory, the possible sets of outputs for applying a randomization algorithm $\mathcal{A}$ to D and D' might be different since the extra vehicle trajectory Θ_r may generate items in the output that cannot be derived from D or D' with $\mathcal{A}$ (e.g., the pseudo-ID of vehicle V_r, an extreme speed, or a unique timestamp in Θ_r). In this case, a relaxed differential privacy [42,46] notion can be defined:

Definition 3 ((ϵ, δ)-differential privacy)**.** A randomization algorithm [47] $\mathcal{A}$ satisfies (ϵ, δ)-differential privacy if for all neighboring inputs D and D' and any set of possible

outputs S, we have $Pr[\mathcal{A}(D) \in S] \leq e^{\epsilon} Pr[\mathcal{A}(D') \in S] + \delta$ and $Pr[\mathcal{A}(D') \in S] \leq e^{\epsilon} Pr[\mathcal{A}(D) \in S] + \delta$.



V-ID	Position		Moving		Time	
	x	y	v	a	d	t
10	-72.2	1181.4	15	0	1	6.8
1001	4.23	163.73	18.4	1.2	1	57.8
1001	6.3	163.7	21.6	1.4	1	57.9
1005	-5.1	130.3	22.2	-1.9	1	60.2
10	-73.38	1181.3	0	0	1	12.9
10	-75.54	1181.2	0	2	1	13.1
1001	-5.3	130.5	15.1	-2.3	1	15.5
1001	-76.5	1181.2	17.3	0.3	1	2.1
1005	4.52	163.13	13.2	1.1	1	66.3
…	…	…	…	…	…	…

V-ID	Position	
	x	y
10	-72.2	1181.4
10	-73.38	1181.3
1001	4.23	163.73
1001	-76.5	1181.2
1005	-5.1	130.3
…	…	…

V-ID	Position		Moving		Time	
	x	y	v	a	d	t
10	-72.2	1181.4	15	0	1	6.8
10	-73.38	1181.3	0	0.3	1	2.1
1001	4.23	163.73	18.4	-1.9	1	60.2
1001	-76.5	1181.2	17.3	0.3	1	2.1
1005	-5.1	130.3	22.2	0	1	12.9…
…	…	…	…	…	…	

Figure 2.2. An example of sampling fine-grained vehicle trajectory data (*one-phase* vs *multi-phase*).

2.2.3 Sanitization Framework. We now present the framework for our vehicle trajectory data sanitization with differential privacy (VTDP). The overview of the framework is shown in Figure 5.1. Sampling the dataset is a way to achieve differential privacy. Accordingly, we propose a novel multi-phase sampling mechanism that randomly generates true values from the original input in three phases. Notice that, multi-phase sampling could improve the output utility in two folds: (1) generating complete attributes in the output (identical to the input) which can be used for any analysis, and (2) more tuples can be retained while satisfying the same differential privacy guarantee (as demonstrated in Example 1 and Figure 2.2).

- **Phase I**: Sampling the combinations of V-ID [3]and Position (V_r, P_i) from the

[3]V-ID refers to vehicle pseudo identity in this paper.

input data D with the *specified output count for each position*. The optimal output counts of different positions will be derived (*maximizing the utility while satisfying the constraints of differential privacy guarantee for phase I*) to sample the V-IDs for each distinct position. Then, the output schema in phase I (denoted as O_1) is "V-ID, Position, Count".

- **Phase II**: Sampling the combinations of V-ID, Position and Moving (V_r, P_i, M_j) from the original *input data D* with the *phase I output* O_1. Then, the output schema in phase II (denoted as O_2) is "V-ID, Position, Moving, Count".

- **Phase III**: Sampling the original tuples (especially the timestamps) from the original *input data D* with the *phase II output* O_2. Then, the output schema in phase III (denoted as O_3) is "V-ID, Position, Moving, Timestamp".

Note that swapping the order of the three phases can also return a private output dataset but may result in reduced utility in vehicle trajectory data sanitization (*determined by the characteristics of the attributes*). For instance, if phase I samples V-IDs for timestamps, phase II samples positions based on the timestamps, and finally phase III samples moving values, the retained number of tuples would be less since many timestamps are associated with only a few vehicles during night time.

The following example based on Figure 2.2 further demonstrates the need for exerting the multi-phase sampling in scenarios when the dataset is fine-grained and diversified.

Example 1. *Figure 2.2 shows an excerpt of our vehicle trajectory dataset which goes under two different sampling mechanisms with the goal of achieving differential privacy. In the first mechanism, each record is independently sampled (all attributes together). As we use multinomial sampling in our scheme, all the records are going*

to be suppressed with a high probability (only those that have identical copies for all the attributes can be possibly retained).

In the second mechanism, by breaking down the dataset into three sets of attributes (corresponding to three phases of sampling), the number of copies for each unique attribute significantly increases, e.g., as illustrated in the figure, the coordinates in four records $(-72.2, 1181.4)$, $(-73.38, 1181.3)$, $(-75.54, 1181.2)$, $(-76.5, 1181.2)$ *are very close (which can be approximated as the same position; close coordinates that are approximated as the same position are marked with the same color). This enhances the utility of the scheme through increasing the chance of retaining individual records. Accordingly, vehicle 10 can be picked with probability of* 0.75 *in every single sampling as it includes the same location (with very close coordinates) for three times and vehicle 1001 also includes it.*

Subsequent phases of sampling are then applied to estimate the values for the remaining attributes in a utility preserving manner.

Note that, the multi-phase sampling is expected to randomly generate a subset of the original dataset, in which each of the phases satisfies *differential privacy* (also detailed in the upcoming sections).

Boosting Utility. Our sanitization framework includes the following components to improve the output utility.

- **Multi-phase sampling** improves the utility.
- **Multinomial sampling** in phase I and II generates counts by preserving their original distribution (e.g., the distribution of vehicles visiting the same position, the distribution of moving values at the same position) in the output.

- **Utility** is maximized in phase I for multinomial sampling with differential privacy.

- **Trajectory interpolation**. Since each fine-grained trajectory posed by one vehicle has an equal-length interval between consecutive timestamps (e.g., 0.1 second), if any vehicle has sampled tuples in the output O_3, its *complete output vehicle trajectory* (at all the times given in the input data) can be approximately *interpolated* with the properties of vehicle trajectories (i.e., the formulas between speed, acceleration and times). This further improves the output utility, and can be conducted by any untrusted data recipient (without affect the privacy guarantee).

2.3 Phase I: Sampling (V-ID, Position) VTDP in phase I, as shown in Figure 2.2, exerts sampling over the pair of vehicle IDs and their visited positions in the dataset. To preserve the distribution of vehicle IDs for each visited position, multinomial sampling is employed in phase I (as illustrated in Section 2.3.1). Next, as detailed in Section 2.3.2, we show that this notion of randomization can guarantee differential privacy with parameters ϵ and δ.

On the other hand, to boost the utility of the output, phase I in VTDP formulates a utility maximizing problem in which the optimal counts of specific positions emerge in the output, will be computed under the differential privacy constraints. Finally, we also ensure that no privacy violation from the optimization procedure occurs (see the discussion at the end of Section 2.3.3). Algorithm 2.3 presents the key steps for sampling phase I (denoted as $\mathcal{A}_1$).

2.3.1 Multinomial Sampling. Given any output count x_i for position P_i, multinomial sampling runs x_i *independent* trials to randomly pick V-IDs for P_i. Specifically, in every trial, a pair of V-ID and position (V_r, P_i) can be generated, and the

Table 2.1. Frequently used notations in phase I

Notions	Description
V_r	the rth vehicle ID, $r \in [1, n]$
Ω	the set of distinct positions
Φ	the set of distinct moving values
Ψ	the set of distinct timestampes
$P_i \in \Omega$	the ith position, $\forall i \in [1, \|\Omega\|]$
$M_j \in \Phi$	the jth moving value, $j \in [1, \|\Phi\|]$
$T_k \in \Psi$	the kth timestamp, $k \in [1, \|\Psi\|]$
D, O_1	input data and output of phase I
$\|D\|, \|O_1\|$	the size of D and O_1
c_i	count of position P_i in the input
x_i	count of position P_i in the output
c_i^r	count of pair of (V_r, P_i)

probability for generating (V_r, P_i) is $\frac{c_i^r}{c_i}$ where the total count of P_i is referred to $c_i = \sum_{r=1}^{n} c_i^r$. After all the x_i trials, we denote the count of V_r in the output as x_i^r where $\sum_{r=1}^{n} x_i^r = x_i$. For example, in the input data D, V_1 has visited P_1 for 6 times, V_2 has visited P_1 for 2 times, V_3 has visited P_1 for 5 times, and V_4 has not visited P_1. Then, while sampling V-IDs for position P_1, in any trial, the probabilities for sampling V_1, V_2, V_3, V_4 are $\frac{6}{6+2+5+0}, \frac{2}{6+2+5+0}, \frac{5}{6+2+5+0}$ and 0. With the properties of multinomial sampling, the portion of the output counts for different pairs of V-ID and position (e.g., (V_1, P_1), (V_2, P_1), (V_3, P_1), and (V_4, P_1)) lies similar to those in the

Data: input D, privacy budgets ϵ, δ
Result: output O_1 as (V-ID, Position, Count)
1 **for** *vehicle* $r \leftarrow 1$ **to** n **do**
2 **for** *position* $i \leftarrow 1$ **to** $|\Omega|$ **do**
3 retrieve count c_i^r from D
4 **end**
5 **end**
6 **for** $r \leftarrow 1$ **to** n **do**
  `// DP for the Sampling`
7 derive constraints with the privacy budgets (ϵ, δ) for variables $\forall i \in [1, |\Omega|], x_i$ (the output count of all the vehicles for position P_i)
8 **end**
9 compute the optimal counts $\forall i \in [1, |\Omega|], x_i^*$ (while satisfying the constraints in Line 6-8)
10 **for** *position* $i \leftarrow 1$ **to** $|\Omega|$ **do**
11 randomly sample x_i^* V-IDs for position P_i using multinomial distribution: x_i^* independent trials (randomly picking a V-ID in each trial), where the probability of picking V_r in each trial is $\frac{c_i^r}{\sum_{r=1}^{n} c_i^r}$
12 **end**
  `//` (V_r, P_i) `is sampled for` x_i^r `times`
13 return the output O_1 as (V_r, P_i, x_i^r)

Figure 2.3. Sampling phase I Algorithm $\mathcal{A}_1$

input data simply because $\forall r \in [1, |\Omega|]$, the expectation $E(x_i^r) = x_i \cdot \frac{c_i^r}{c_i}$ is proportional to $\frac{c_i^r}{c_i}$ (as the same x_i is applied).

2.3.2 Differential Privacy Guarantee. To investigate the differential privacy guarantee of multinomial sampling, we should explore the set of all possible outputs for any given input data D and any of its neighboring input data D' (differing in one vehicle V_r's vehicle trajectory Θ_r). As a result, we have two cases for neighboring inputs D and D': $D = D' \cup \Theta_r$ and $D' = D \cup \Theta_r$.

As discussed in Section 2.2.2, ϵ-differential privacy may not be achieved in the sampling since the probabilistic output O_1 may include item from D' yet cannot be from D (or vice-versa), e.g., V-ID V_r. Then, the relaxed notion (ϵ, δ)-differential privacy will be employed for phase I. Without loss of generality, we let $D = D' \cup \Theta_r$

(the other case $D' = D \cup \Theta_r$ will be discussed at the end of this subsection).

Now we divide the arbitrary output set $S \subseteq \text{Range}(\mathcal{A}_1)$ into S^+ and S^- where $V_r \in S^+$ and $V_r \notin S^-$ (note that S^+ *is formed with all the possible outputs with* V_r *while* S^- *does not include* V_r). Therefore, we can derive a sufficient condition for the randomization algorithm $\mathcal{A}_1$ and possible output O_1 (phase I) to bound $\frac{Pr[\mathcal{A}_1(D')\in S]}{Pr[\mathcal{A}_1(D)\in S]}$ and $\frac{Pr[\mathcal{A}_1(D)\in S]}{Pr[\mathcal{A}_1(D')\in S]}$ (Gotz et al. [46] also conducted similar study):

Theorem 1. *Given any neighboring inputs* D *and* D'*, if* $\forall O_1 \in S^-$*, inequality* $\frac{Pr[\mathcal{A}_1(D')=O_1]}{Pr[\mathcal{A}_1(D)=O_1]} \leq e^\epsilon$ *holds, then* $\frac{Pr[\mathcal{A}_1(D')\in S]}{Pr[\mathcal{A}_1(D)\in S]} \leq e^\epsilon$ *also holds.*

Proof. Since $\forall O_1 \in S^+$, $Pr[\mathcal{A}_1(D') = O_1] = 0$, we have

$$\begin{aligned} Pr[\mathcal{A}_1(D') \in S] &= \int_{\forall O_1 \in S^+} Pr[\mathcal{A}_1(D') = O_1] dO_1 + \int_{\forall O_2 \in S^-} Pr[\mathcal{A}_1(D') = O_2] dO_2 \\ &\leq e^\epsilon \int_{\forall O_2 \in S^-} Pr[\mathcal{A}_1(D) = O_2] dO_2 \\ &= e^\epsilon Pr[\mathcal{A}_1(D) \in S^-] \\ &\leq e^\epsilon Pr[\mathcal{A}_1(D) \in S] \end{aligned}$$

This completes the proof. □

Similarly, we can prove that $Pr[\mathcal{A}_1(D) \in S] \leq \delta + e^\epsilon Pr[\mathcal{A}_1(D') \in S]$. This shows that we can ensure differential privacy by letting $\forall O_1$ in any S^- (viz. any output in $\text{Range}(\mathcal{A}_1)$ without V_r),

$$\frac{Pr[\mathcal{A}_1(D') = O_1]}{Pr[\mathcal{A}_1(D) = O_1]} \leq e^\epsilon$$

in multinomial sampling, detailed as below.

Case (1): $\forall O_1 \in \text{Range}(\mathcal{A}_1)$ **where** $V_r \notin O_1$

Due to $V_r \notin O_1$, we have $Pr[A_1(D') = O_1] > 0$ and $Pr[A_1(D) = O_1] > 0$ (O_1 *can be generated from both* D *and* D' *with multinomial sampling*). At this time, we only need to sample V-IDs for the positions $\forall P_i \in D'$, which is a subset of D (otherwise, V_r might be included in O_1). Since sampling V-IDs for different positions is independent, we now examine two situations of all the positions in D'.

1. Position $P_i \in D' \setminus \Theta_r$. The probabilities for sampling any V-ID for P_i from D and D' are equal (since vehicle V_r's trajectory Θ_r does not include P_i). Thus,

$$\frac{Pr[(\mathcal{A}_1(D(P_i)) = O_1(P_i)]}{Pr[(\mathcal{A}_1(D'(P_i)) = O_1(P_i)]} = 1 \tag{2.1}$$

   where $D(P_i)$ and $O_1(P_i)$ denote the position P_i's share of the input D and output O_1.

2. Position $P_i \in D' \cap \Theta_r$. At this time, sampling V-IDs for position P_i should ensure that the probability of picking V-ID V_r (out of all the vehicle IDs in D' and V_r) with multinomial sampling is bounded. Since picking V-IDs for x_i times trials is independent using multinomial distribution, we have following equations:

$$\begin{aligned} &\frac{Pr[\mathcal{A}_1(D'(P_i)) = O_1(P_i)]}{Pr[\mathcal{A}_1(D(P_i)) = O_1(P_i)]} \\ =&\frac{1}{(1-\frac{c_i^r}{c_i})^{x_i}} = (\frac{c_i}{c_i - c_i^r})^{x_i} \end{aligned} \tag{2.2}$$

Given the output count $\forall i \in [1, |\Omega|], x_i$ for the ith position P_i, sampling vehicle IDs for each of the distinct position P_i is independent. For instance, while sampling for P_1, the sampling results can be "$(P_1, V_1, 5), (P_1, V_2, 3)$" where counts 5 and 3 are random. While sampling for P_2, the sampling results would be "$(P_2, V_1, 2), (P_2, V_2, 1)$"

where counts 2 and 1 are random. Therefore, the privacy budget can be allocated for each sampling with sequential composition [48]. As a result, for each of the $|\Omega|$ different multinomial sampling (w.r.t. $|\Omega|$ different positions, respectively), the following constraint can be generated:

$$\max_{\forall \Theta_r \in D} \prod_{\forall P_i \in \Theta_r} (\frac{c_i}{c_i - c_i^r})^{x_i} \leq e^{\epsilon} \tag{2.3}$$

Case (2): $\forall O_1 \in \text{Range}(\mathcal{A}_1)$ **where** $V_r \in O_1$

In this case, the output O_1 would include item(s) from D but not D', e.g., $V_r \in \Theta_r$. Then, δ is defined to bound such probability $Pr[\mathcal{A}_1(D) \in S] \leq \delta$, which means

$$Pr[\mathcal{A}_1(D) \in S] \leq \delta \implies Pr[V_r \in \mathcal{A}_1(D)] \leq \delta \tag{2.4}$$

We now examine the probability $Pr[\mathcal{A}_1(D) \in S]$ while $Pr[\mathcal{A}_1(D') \in S]$ equals 0 (output includes V_r) as applying multinomial sampling to D and D', respectively.

$$\max_{\forall \Theta_r \in D} \prod_{\forall P_i \in \Theta_r} [1 - (\frac{c_i - c_i^r}{c_i})^{x_i}] \leq \delta \tag{2.5}$$

Per Definition 3, while specifying a small number δ, our algorithm ensures ϵ-differential privacy with high probability $(1 - \delta)$. Thus, we can simply remove the vehicle trajectories whose data results in a violation of Equation 2.5. Then, satisfcation of Equation 2.3 can ensure (ϵ, δ)-differential privacy for our Phase I sampling $\mathcal{A}_1$.

Discussion. In case of $D' = D \cup \Theta_r$, adding any vehicle trajectory Θ_r to input D to generate D'. Similarly, as long as the given $\forall i \in [1, |\Omega|], x_i$ satisfy Equation 2.3 (now c_i is derived from D' and c_i^r is the count of P_i in Θ_r), differential privacy is guaranteed.

2.3.3 Optimal Differentially Private Sampling. As analyzed in Section 2.3.2, if the output counts for all the positions $\forall i \in [1, |\Omega|], x_i$ satisfy inequality 2.3 (*inequality 2.5 will be employed in data preprocessing for small* δ), then the multinomial sampling to generate the output with schema *(V-ID, Position, Count)* would satisfy (ϵ, δ)-differential privacy.

Theorem 2. *Sampling in Algorithm 2.3 (Line 10-12) is* (ϵ, δ)*-differentially private if and only if inequalities 2.3 hold for all* Θ_r*.*

Proof. It is straightforward to prove that the probabilities that results in Case (2) for all the vehicles and positions are bounded by δ if inequality 2.5 holds (by setting δ in the preprocessing). In addition, as analyzed in Case (1), if inequality 2.3 holds for all Θ_r, per Theorem 1, we have:

$$e^{-\epsilon} \leq \frac{Pr[\mathcal{A}_1(D) \in S]}{Pr[\mathcal{A}_1(D') \in S]} \leq e^{\epsilon} \tag{2.6}$$

where S represents any set of possible outputs (without data from Θ_r). This completes the proof. □

Notice that, in a special case $c_i = c_i^r$ (the position in Θ_r is unique, and cannot be found in D'), x_i should be 0, otherwise, inequality 2.3 cannot hold.

Therefore, we should look for the output counts $\forall i \in [1, |\Omega|], x_i$ that satisfy Equation 2.3. Note that $\forall i \in [1, |\Omega|], x_i$ should be derived from D (or D') by subjecting to:

$$s.t. \begin{cases} \forall \Theta_r \in D, \prod_{\forall P_i \in \Theta_r} (\frac{c_i}{c_i - c_i^r})^{x_i} \leq e^{\epsilon} \\ \forall \Theta_r \in D, \prod_{\forall P_i \in \Theta_r} [1 - (\frac{c_i - c_i^r}{c_i})^{x_i}] \leq \delta \\ \forall x_i \geq 0 \text{ and } x_i \text{ is an integer} \end{cases} \tag{2.7}$$

While satisfying differential privacy, $\forall i \in [1, |\Omega|], x_i$ can have many possible results. We now seek for the optimal output counts for the differentially private sampling. A generic way of evaluating the utility is to measure the difference between the count distribution of all the positions in the input and output using distance metrics, e.g., ℓ_1-norm or ℓ_2-norm. However, the utility optimization based on such metrics may generate biased results towards the frequently visited positions (and the diversity of the positions may not be effectively preserved) [49].

To address such limitation, we define a universal utility measure (for multiple applications) for all the variables $\forall i \in [1, |\Omega|], x_i$ by following the KL-divergence [4] [42, 51], which evaluates the entropy-based distance between all the positions' distributions in the input data $(\frac{c_1}{|D|}, \frac{c_2}{|D|}, \ldots, \frac{c_{|\Omega|}}{|D|})$ and the output data $(\frac{x_1}{|O_1|}, \frac{x_2}{|O_1|}, \ldots, \frac{x_{|\Omega|}}{|O_1|})$ where $|D|$ and $|O_1|$ represent the total number of records in the input and output $|D| = \sum_{i=1}^{|\Omega|} c_i$ and $|O_1| = \sum_{i=1}^{|\Omega|} x_i$.

$$D_{KL} = \sum_{i=1}^{|\Omega|} \frac{c_i}{|D|} \big[\log(\frac{c_i}{|D|} \cdot \frac{|O_1| + |\Omega|}{x_i + 1}) \big] \tag{2.8}$$

[4] Optimizing the utility with KL-divergence can address the count bias as an entropy-based measure [49]. The optimization can preserve more distinct positions in the output as well as minimize the deviation between the distributions of all the positions in the input and output (ensuring that the data distribution in the output still lies close to that in the input). Notice that, KL-divergence is also used as the distance metric in case of similar scenarios. For instance, Acs et al. [50] measure the distance of the two probability distributions (count distribution in the input and output histograms).

Recall that minimizing the KL-divergence can maximally preserve the distribution/portion of each position in the output. Then, with multinomial sampling, the distribution/portion of each combination of V-ID and position is expected to be preserved in the output as well. Since x_i may equal to 0 (in case of unique positions), the output counts in the KL-divergence are captured by approximating x_i with a close value $(x_i + 1)$ to avoid zero denominator.

Therefore, we can formulate an optimization problem to find the optimal multinomial sampling.

$$\min : \quad \sum_{i=1}^{|\Omega|} \frac{c_i}{|D|} \big[\log(\frac{c_i}{|D|} \cdot \frac{|O_1| + |\Omega|}{x_i + 1}) \big]$$
$$s.t. \begin{cases} \forall \Theta_r \in D, \prod_{\forall P_i \in \Theta_r} (\frac{c_i}{c_i - c_i^r})^{x_i} \leq e^{\epsilon} \\ \forall \Theta_r \in D, \prod_{\forall P_i \in \Theta_r} [1 - (\frac{c_i - c_i^r}{c_i})^{x_i}] \leq \delta \\ \forall x_i \geq 0 \text{ and } x_i \text{ is an integer} \end{cases} \tag{2.9}$$

We can solve the above nonlinear programming (NLP) problem using pairwise linear approximation by converting the objective function to linear (the constraints can be simply converted to linear constraints) [42].

Differential Privacy for the Optimization. While applying Algorithm 2.3 to D and D' (solving the optimization problem 2.9) to get two sets of output counts $\forall i \in [1, |\Omega|]$, x_i^*, and $\forall i \in [1, |\Omega|]$, $x_i^{*'}$, respectively. In case that $D = D' \cup \Theta_r$, $\forall i \in [1, |\Omega|], x_i^*$ can ensure (ϵ, δ)-differential privacy for multinomial sampling. In case that $D' = D \cup \Theta_r$, $\forall i \in [1, |\Omega|], x_i^{*'}$ can ensure (ϵ, δ)-differential privacy. Apart from such privacy guarantee, we also need to make such two computed set of counts *indistinguishable*.

Specifically, we can consider the problem solving process as a query over the

input data D or D', then the generic Laplace noise [52] $\frac{\Delta}{\epsilon'}$ can ensure ϵ'-differential privacy for the process of solving the problem [41,42], where ϵ' is an additional privacy budget for this step, and sensitivity $\Delta = \max_{\forall D,D'} |x_i^* - x_i^{*'}|$ [53, 54]. Due to space limit, we skip the details of such generic mechanism here. In summary, we have the differential privacy guarantee for Algorithm 2.3.

Theorem 3. *Phase I is $(\epsilon + \epsilon', \delta)$-differentially private.*

Proof. This can be proven by the sequential composition [48] of solving the optimal problem and sampling. □

2.4 Phase II: Sampling Moving

In this section, we present the sampling phase II of our VTDP framework: randomly generating moving values by breaking down the counts for different pairs of V-IDs and positions to the counts for the triplets of V-IDs, positions and moving values. Furthermore, we study the differential privacy for phase II. Note that the required notations for sampling phase II are listed in Table 2.2.

2.4.1 Dirichlet-Multinomial Sampling. Similar to sampling phase I, the pair of visited position and moving values (i.e., speed and acceleration) for each vehicle in the trajectory data can be sampled with multinomial distribution which is expected to preserve the distribution of moving values associated with each position. More specifically, in phase II, for each vehicle, each moving data should be sampled from each of its visited positions (generated in phase I). Note that any count value for vehicle V_r and for position P_i is sampled as x_i^r in phase I, then x_i^r moving values (may include duplicates) will be sampled using an individual multinomial sampling in phase II.

Given n vehicles in the original input, after phase I, we denote the number of

Table 2.2. Frequently used notations in phase II

Notation	Description
x_i^*	optimal count for Position P_i in phase I
x_i^r	sampled count of (V_r, P_i) in phase I
n'	number of vehicles in the output of phase I
γ^r	cardinality of sampled positions in phase I visited by V_r
$\theta_i(j)$	prior probability of sampling M_j for P_i (all vehicles)
θ_i	prior probability vector $\theta_i = (\theta_i(1), \ldots, \theta_i(\|\Phi\|))$
$\theta_i^r(j)$	posterior probability of sampling M_j for P_i and V_r
θ_i^r	posterior probability vector $\theta_i^r = (\theta_i^r(1), \ldots, \theta_i^r(\|\Phi\|))$
D_1, D_2	two datasets extract from the input D
$\lambda_i(j)$	count of (P_i, M_j) in D_1 (all the sampled vehicles)
$x_i^{r'}$	count of all the moving for (V_r, P_i) in D_2
$x_i^r(j)'$	count of (V_r, P_i, M_j) in D_2
O_2	output of phase II
$x_i^r(j)$	sampled count of (V_r, P_i, M_j) in phase II

vehicles in the output as n' where $n' \leq n$ (since some V-IDs might not be randomly picked). Denoting the number of unique positions sampled for V_r in phase I as γ^r, there are $\sum_{r=1}^{n'} \gamma^r$ independent multinomial sampling in phase II, each of which is allocated for a unique pair of vehicle and one of its visited position. While sampling any moving values $M_j \in \Phi$ at position P_i for vehicle V_r, x_i^r independent trials will be

tossed where the probabilities of possible sampling outcomes in every trial (denoted as *"probability vector"* $\theta_i^r = (\theta_i^r(1), \theta_i^r(2), \ldots, \theta_i^r(|\Phi|))$ will be learned from Dirichlet-Multinomial distribution [55] for the following reasons.

First, the distribution can integrate observations (drawn from the moving patterns posed by each vehicle in a particular position) and prior parameters (drawn from all the moving patterns at the same position). Therefore, considering the huge volume of moving patterns existed in the data, the posterior probability vector θ_i^r learned by the Dirichlet distribution would become significantly more accurate (e.g., in vehicle trajectory interpolation and analyses performed on the sanitized data). Second, sampling moving values with Dirichlet-Multinomial distribution does not result in *false moving values*. Specifically, if V_r has not visited P_i with moving value M_j, then the probability $\theta_i^r(j)$ would be 0 (since the corresponding observation is 0).

2.4.1.1 Probability Vector Learning. Before learning the probability vector, we extract two datasets from the input D (*which can minimize the privacy bound for phase II*, as illustrated in Section 2.4.2):

1. **Prior Data** D_1: a bipartite graph for every pair of position and moving (P_i, M_j) and the corresponding count $\lambda_i(j)$ – for deriving the prior distribution $\theta_i(j)$.

   The generation of D_1 includes two steps: (1) removing all the tuples inside each of the the unsampled trajectories (keeping only sampled data for n' vehicles), and (2) for every pair of position and moving (P_i, M_j), aggregating all the vehicles and timestamps' corresponding tuples to get count $d_i(j)$. Note that removing unsampled vehicles' data could ensure a tight privacy bound (e.g., $\epsilon = 0$) for phase II (as analyzed in Lemma 1).

2. **Observation Data** D_2: for each vehicle V_r, extracting its bipartite graph for each pair of its position and moving (P_i, M_j). Specifically, for each vehicle

V_r, we extract sampled positions of V_r in phase I (γ^r distinct positions) and the corresponding tuples in D (tuples including unsampled positions will be removed), and aggregate all the timestamps for the corresponding tuples for (V_r, P_i, M_j) to get $x_i^r(j)'$.

Then, $\forall j$, $\lambda_i(j)$ and $\theta_i(j)$ can be derived from data D_1 while $\forall j, \theta_i^r(j)$ can be derived from data D_2. Per the Bayes rule, we can learn the posterior distribution for the probability vector: for each Vehicle V_r and its position P_i.

$$
\begin{aligned}
& Pr(\theta_i^r | M_1, \ldots, M_{|\Phi|}) \propto P(M_1, \ldots, M_{|\Phi|} | \theta_i^r) Pr(\theta_i^r) \\
\propto & \frac{\Gamma(\sum_{j=1}^{|\phi|} \lambda_i(j))}{\prod_{j=1}^{|\phi|} \Gamma(\lambda_i(j))} \prod_{j=1}^{|\phi|} (\theta_i^r(j))^{\lambda_i(j)-1} \frac{n!}{x_1^r(1)! \cdots x_i^r(|\phi|)!} \prod_{j=1}^{\phi} \theta_i^{x_i^r(j)'}(j) \\
\propto & \prod_{j=1}^{|\phi|} \theta_i^r(j)^{\lambda_i(j)-1+x_i^r(j)'}
\end{aligned}
$$

where constant

$$
\frac{\prod_{j=1}^{|\phi|} \Gamma(\lambda_i(j))}{\Gamma(\sum_{j=1}^{|\phi|} \lambda_i(j))} = \frac{\Gamma(\lambda_i(1))\Gamma(\lambda_i(2)) \cdots \Gamma(\lambda_i(|\phi|))}{\Gamma(\lambda_i(1) + \lambda_i(2) + \cdots + \lambda_i(|\phi|))}
$$

and Gamma function

$$
\Gamma(\lambda_i(j)) = (\lambda_i(j) - 1)!
$$

Notice that, the same prior probability vector θ_i^r is adopted for position P_i for all the vehicles, thus θ_i and θ_i^r are interchangeable. In addition, for V_r, the prior and posterior probabilities for most of moving values $M_1, \ldots, M_{|\Phi|}$ are 0 in practice. For simplicity of notations, we still use $M_1, \ldots, M_{|\Phi|}$ to represent the moving values.

2.4.1.2 Sampling Algorithm (Phase II). We now present our sampling algorithm for phase II. First, the algorithm extracts D_1 and D_2 based on the output of phase I (O_1) and the original input D. Recall that,

1. D_1 is a bipartite graph with aggregated counts (in D) for every pair of (P_i, M_j) where the data of unsampled vehicles ($\forall V_r \in D \setminus D_2$) are not aggregated. Note that D_2 is the dataset including all the original tuples corresponding to the sampled output after phase I.

2. D_2 includes n' bipartite graphs (for n' sampled vehicles in O_1). Each vehicle's bipartite graph is extracted as its aggregated counts (in D) for every pair of (P_i, M_j) in O_1 (the output of phase I).

Second, the algorithm derives the prior probability vector of Dirichlet distribution and likelihood using D_1 and D_2. Thus, the posterior probability vector can be obtained using Bayes rule (using the expectation of the Dirichlet distribution [56]).

Finally, for each vehicle V_r and each of its visited position (e.g., P_i) in O_1, we apply multinomial sampling with its posterior probability vector and x_i^r trials. Algorithm 2.4 presents the details of sampling phase II.

2.4.2 Privacy Bound for Phase II. We now investigate the privacy bound for phase II, which samples x_i^r moving values for every pair of V-ID and position (V_r, P_i) where its count x_i^r is derived in phase I.

Lemma 1. *Phase II does not leak any additional information by sampling with the output of Phase I.*

Proof. We explore the privacy leakage by integrating phase I and II. Again, for two neighboring inputs D and D', w.l.o.g., we let $D = D' \cup \Theta_r$. In phase I, the probability of generating Case (2) (per Definition 3) is bounded by δ, which can be a negligible probability. Then, we only need to discuss Case (1) in phase I: $\forall O_1 \in Range(\mathcal{A}_1)$ where $V_r \notin O_1$, and investigate the privacy bound in phase II.

After phase I, the outputs (without V_r) derived from inputs D and D' are

Data: input D, phase I output O_1
Result: output O_2 as (V-ID, Position, Moving, Count)
extract D_1 and D_2 from D (using the Vehicles in O_1).
for $j \leftarrow 1$ **to** n **do**
    prior $P(\theta_i(j)) \leftarrow E[\theta_i(j)|\lambda_i(j)]$ where
      $E[\theta_i(j)|\lambda_i(j)][\theta_{ij}] = \frac{\lambda_i(j)}{\sum_{j=1}^{|\Phi|} \lambda_i(j)}$
    likelihood$\leftarrow \frac{x_i^{r'}(j)}{x_i^{r'}}$
end
Posterior $(\theta_i^r(j)) \leftarrow$ prior $(\theta_i(j)) \times$ likelihood $x_i^{r'}(j)/(x_i^{r'})$
foreach $V_r \in O_1$ **do**
    for $i \leftarrow 1$ **to** n **do**
        randomly sample x_i^r times moving values for vehicle V_r and position P_i using multinomial distribution: the probability of picking M_j in each trial is the posterior probability of $\theta_i^r(j)$
    end
end
return the output O_2 as $(V_r, P_i, M_j, x_i^r(j))$

Figure 2.4. Sampling phase II Algorithm $\mathcal{A}_2$

$(\epsilon + \epsilon')$-*indistinguishable*. Denoting the output for phase II as O_2, we first explore the multiplicative difference between probabilities $Pr[\mathcal{A}_2(D) = O_2]$ and $Pr[\mathcal{A}_2(D') = O_2]$. Specifically, as illustrated in Section 2.4.1, both D_1 and D_2 are extracted from D (or D' in the neighboring input case) in phase II (for learning the probability vector of multinomial sampling). In both D_1 and D_2, V-IDs is the baseline for extracting the tuples, whereas in the output of phase I: O_1, the position is the baseline. Since O_1 is indistinguishable for both inputs D and D' (both without V_r), each of two datasets D_1 and D_2 makes no difference in case of both D and D' (though D differs from D' in any vehicle trajectory Θ_r in phase I and II). Then, the probability vector would be indistinguishable for D and D', and thus we have:

$$\forall O_2 \in \text{Range}(\mathcal{A}_2), \frac{Pr[\mathcal{A}_2(D) = O_2]}{Pr[\mathcal{A}_2(D') = O_2]} = 1 \tag{2.10}$$

Note that even if a new vehicle trajectory Θ'_r is added to D at the beginning of phase II to form D', data in Θ'_r will be suppressed while generating D_1 and D_2 for sampling (due to O_1). In this case, Equation 2.10 still holds. Similar to Theorem 1, given any possible output set S in phase II, we have $\frac{Pr[\mathcal{A}_2(D')\in S]}{Pr[\mathcal{A}_2(D)\in S]} = 1$.

Therefore, Phase II ensures 0-indistinguishability to randomly generate the output O_2. □

2.5 Phase III: Sampling Timestamps

In this section, we discuss how to sample the timestamps based on phase II output O_2, which includes the output count $x_i^r(j)$ for each pairs of position and moving (P_i, M_j) for V_r. Then, the timestamps sampling for the triplet (V_r, P_i, M_j) in phase III will be based on count $x_i^r(j)$. Indeed, phase III is not the same as the previous two phases, due to the uniqueness of timestamps. Specifically, for each timestamp, there exists exactly only one vehicle at the same position (which has been validated in our experimental data). On the contrary, one vehicle may visit the same location every day or stay at the one position over a period, thus the triplet of (V_r, P_i, M_j) may have multiple unique timestamps $T_k \in \{T_1, T_2, ...T_{|\Psi|}\}$ in D (denoting such count as $c_i^r(j)$). We then present our algorithm $\mathcal{A}_3$ by considering the above facts. Similar to phase II $\mathcal{A}_2$, phase III also extracts a dataset D_3 from D based on O_2:

- For all vehicles $\forall V_r \in O_2$, extract trajectories Θ_r from D to generate D_3.

For each triplet (V_r, P_i, M_j), the algorithm in phase III randomly picks $x_i^r(j)$ timestamps out of $c_i^r(j)$ unique timestamps from D_3 (*note that* $c_i^r(j)$ *are identical in* D *and* D_3). However, since $x_i^r(j)$ was randomly generated with multinomial sampling in phase I and II, $x_i^r(j)$ may exceed $c_i^r(j)$, though the probability of generating such extreme case is fairly low. Thus, we have to handle such extreme case in our algorithm

Data: input D, phase II output O_2
Result: output O_3 as (V-ID, Position, Moving, Timestamp)
extract D_3 from D (using the Vehicles in O_2).
foreach $V_r \in O_2$ **do**
  for $i \leftarrow 1$ **to** n **do**
    if $x_i^r(j) \leq c_i^r(j)$ **then**
      randomly pick $x_i^r(j)$ unique timestamps from $c_i^r(j)$ in D_3
    else
      randomly pick $x_i^r(j)$ unique timestamps from $c_i^r(j)$ in D_3
      randomly picks $x_i^r(j) - c_i^r(j)$ timestamps from other tuples in D_3 which include position P_i and moving M_j (other vehicles)
    end
  end
end
return the output O_3 as (V_r, P_i, M_j, T_k)

Figure 2.5. Sampling phase III Algorithm $\mathcal{A}_3$

$\mathcal{A}_3$ in the following two situations.

- If $x_i^r(j) \leq c_i^r(j)$, the algorithm simply picks $x_i^r(j)$ timestamps out of $c_i^r(j)$ unique timestamps from D_3.
- If $x_i^r(j) > c_i^r(j)$. The algorithm first picks all $c_i^r(j)$ timestamps out of $c_i^r(j)$ unique timestamps from D, and then randomly picks $x_i^r(j) - c_i^r(j)$ timestamps from other tuples which include position P_i and moving M_j (other vehicles). Note that the associated V-IDs for the latter picked timestamps will not be V_r. This ensures that all the tuples randomly selected from D are true tuples.

2.5.1 Privacy Bound for Phase III. Similar to phase II, phase III also ensures indistinguishability for any neighboring inputs D and D'.

Lemma 2. *Phase III does not leak any additional information by sampling with the output of Phase II.*

Proof. Given two inputs data D and D' where $D = D' \cup \Theta_r$ (or $D' = D \cup \Theta_r$), similar to D_1 and D_2 in phase II, the datasets (denoted as D_3) extracted from D and D' for sampling are indistinguishable, since O_2 is indistinguishable for D and D' after phase I and II, and data in Θ_r is suppressed in D_3 in any case. Then, the probabilities of randomly picking any timestamp (tuple) from the D_3 of D and D', and the count $x_i^r(j)$ are indistinguishable for any neighboring inputs D and D'. Thus, we have:

$$\forall O_3 \in \text{Range}(\mathcal{A}_3), \frac{Pr[\mathcal{A}_3(D) = O_3]}{Pr[\mathcal{A}_3(D') = O_3]} = 1 \tag{2.11}$$

Similar to Theorem 1 and Lemma 1, given any possible output set S in phase III, we have

$$\frac{Pr[\mathcal{A}_3(D') \in S]}{Pr[\mathcal{A}_3(D) \in S]} = 1$$

Therefore, phase III also ensures 0-indistinguishability to randomly generate the output O_3. □

2.6 Discussions

To further improve the output utility of our three-phase sampling, we propose a vehicle trajectory interpolation procedure in the VTDP framework to *approximately estimate the missing values at different times.*

2.6.1 Vehicle Trajectory Interpolation. As shown in Figure 5.1, the vehicle trajectory interpolation can be conducted by the untrusted data recipients (without affecting the privacy guarantee). Specifically, the interpolation is executed based on every two consecutive sampled tuples in trajectory θ_r (for the missing tuples between them). For instance, at time T_1 and T_6, two tuples are sampled in θ_r are sampled: "$\ell, (x_1, y_1), v_1, a_1, T_1$" and "$\ell', (x_6, y_6), v_6, a_6, T_6$". Then, all the tuples at T_2, T_3, T_4, T_5

can be interpolated using the two tuples at T_1 and T_6 (all the timestamps have equal intervals) with the following rules.

- The lane number of the first half of the tuples between T_1 and T_6 (viz. T_2 and T_3 in this example) is assigned as ℓ (same as T_1) while the second half (viz. T_4 and T_5) is assigned as ℓ' (same as T_6). If there are odd number of timestamps between two consecutive sampled tuples, the timestamp in the middle is considered as the first half.

- The position (x, y) for timestamps T_2, T_3, T_4, T_5 will be interpolated with equal distance between any two adjacent timestamps: $(x_2, y_2) = (x_1 + \frac{x_6 - x_1}{6-1}, y_1 + \frac{y_6 - y_1}{6-1})$, $(x_3, y_3) = (x_1 + \frac{2(x_6 - x_1)}{6-1}, y_1 + \frac{2(y_6 - y_1)}{6-1})$, $\ldots$, $(x_5, y_5) = (x_1 + \frac{4(x_6 - x_1)}{6-1}, y_1 + \frac{4(y_6 - y_1)}{6-1})$.

- The interpolation for acceleration $a_2, \ldots, a_5$ follows the same way as position.

- Speed v for timestamps T_2, T_3, T_4, T_5 will be interpolated with the formula between speed, acceleration and moving time. Then, $v_2 = v_1 + a_1(T_2 - T_1)$, $v_3 = v_2 + a_2(T_3 - T_2)$, $\ldots$, $v_5 = v_2 + a_2(T_3 - T_2)$.

It is worth noting that the above examples for vehicle trajectory interpolation are illustrated in case of driving in the same lane. If vehicles make turns or switch lanes, the missing values in the output data can also be interpolated in a similar manner.

Privacy Analysis. For any neighboring inputs D and D', since the probabilities of generating any O_3 from D and D' are bounded, adversaries (e.g., untrusted data recipient) cannot identify whether any vehicle trajectory Θ_r is included in the input or not – *indistinguishability*. Since such trajectory interpolation is a deterministic procedure (after receiving the output O_3) without any additional information, the

adversaries cannot distinguish the interpolated outputs from D and D' either. Thus, the vehicle trajectory interpolation does not affect the differential privacy guarantee of our VTDP framework (and it can be performed by any untrusted data recipient).

2.6.2 Composition of Differential Privacy. Overall, the differential privacy for all the four major components of VTDP (computing optimal counts, sampling phase I, II and III) follows sequential composition [48]. We now discuss the composition and the privacy bounds step by step in our framework.

1. Computing the optimal counts (for sampling phase I): this step satisfies ϵ'-differential privacy.

2. Multinomial sampling to generate O_1 (sampling phase I). Sampling V-IDs for each position is independent but associates multiple positions with each V-ID. Thus, sampling phase I for each position follows sequential composition (as discussed in Section 2.3.2). This step satisfies (ϵ, δ)-differential privacy.

3. Dirichlet-Multinomial sampling to generate O_2 (sampling phase II). Sampling moving values for every pair of position and vehicle ID is independent (generating disjoint outputs), thus sampling phase II for every pair of position and vehicle ID follows parallel composition of differential privacy. This step has also been proven to satisfy 0-differential privacy (per Lemma 1).

4. Sampling timestamps to generate O_3 (sampling phase III). Similar to phase II, sampling timestamps for every pair of position and moving in θ_r also follows parallel composition of differential privacy. This step has also been proven to satisfy 0-differential privacy (per Lemma 2).

Theorem 4. *VTDP satisfies $(\epsilon + \epsilon', \delta)$-differential privacy.*

Proof. This can be proven by the sequential composition [48] of three sampling phases. □

2.6.3 Protection against Re-identification. We now discuss the re-identification attack to the sanitized dataset of VTDP. Assume that an adversary possesses arbitrary background knowledge on a specific vehicle V_r, e.g., knowing a large portion of places that the vehicle/driver has visited. While providing the differential privacy guarantee by VTDP, the probabilities of generating any output from D (with such vehicle's data) and D' (without such vehicle's data) are indistinguishable. Thus, the adversary cannot identify if such vehicle is included in the dataset from the output (since such output can also be obtained even if all the known places are not included in the input). At this time, knowing a large portion of places the vehicle/driver has visited cannot facilitate the re-identification.

2.6.4 Application to Sanitizing Other Datasets. Recall that phase I in our VTDP samples a probabilistic output with the attributes V-ID, position and count. Then, phase II samples the moving values to be associated with the V-ID and position. Finally, phase III samples the timestamps to be associated with the V-ID, position and moving values. The sanitization is not dependent on the number of fixed attributes. In other words, if more attributes are attached with the vehicle trajectories (e.g., distance to the traffic signal [1]), an output can be generated with the same number of tuples as the output of phase I. Following the above property of VTDP, we can apply our VTDP (via multi-phase sampling) to sanitize other datasets, such as generic microdata [57] and network data [58].

2.7 Experimental Results

We conduct experiments on the NGSIM dataset [59], which is a real world fine-grained vehicle trajectory data with "lane, coordinates (x, y), speed, acceleration,

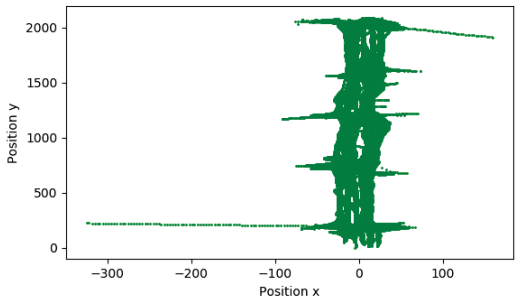
day, time". The experimental dataset includes 1,809 distinct vehicles, each of them consists of 479,763 tuples in an arterial road (Peachtree Street in Atlanta, GA). The time interval for collecting data from each vehicle is 0.1 second. Table 2.3 presents the characteristics of our experimental dataset.

Table 2.3. Characteristics of the dataset

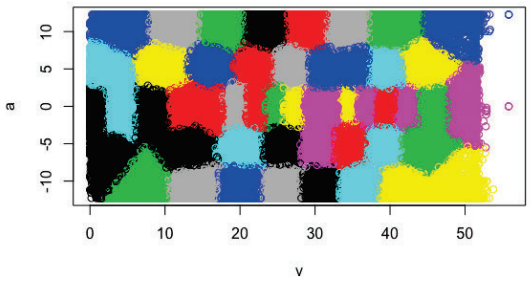
	Distinct #	Min	Max
Vehicles ID	1,809	n/a	n/a
Lane ID ℓ	7 (in multiple roads)	n/a	n/a
x (lateral)	66,336	-325.65	160.90
y (longitudinal)	372,003	0.0	2094.07
speed v	5,250	0.0	55.82
Acceleration a	2,451	-12.27	12.27
Day d	3	1	3
Time t	10,326	0.3	1,032.8

2.7.1 Experimental Setup. Due to the fine-grained property of vehicle trajectories, two different values of any attribute might be extremely close, and can be approximated as the same value. For instance, since the distance between two coordinates (-72.2,1181.4) and (-73.38,1181.3) is very small, they can be approximated as the same location. Furthermore, moving values (20ft/s, 1.2ft/s^2) and (22ft/s, 1.1ft/s^2) may represent very similar moving attributes on the road. Therefore, we preprocess such fine-grained dataset by approximating close values in the raw data.

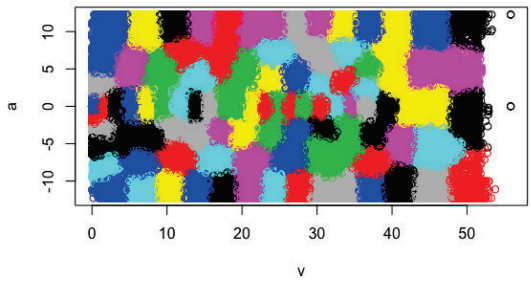
- First, all the *positions* (different combinations of ℓ, x, and y) can be partitioned with the equal size blocks (e.g., using the average length of vehicles), each of



(a) Coordinates of Positions



(b) 50 Clusters for Speed and Acceleration

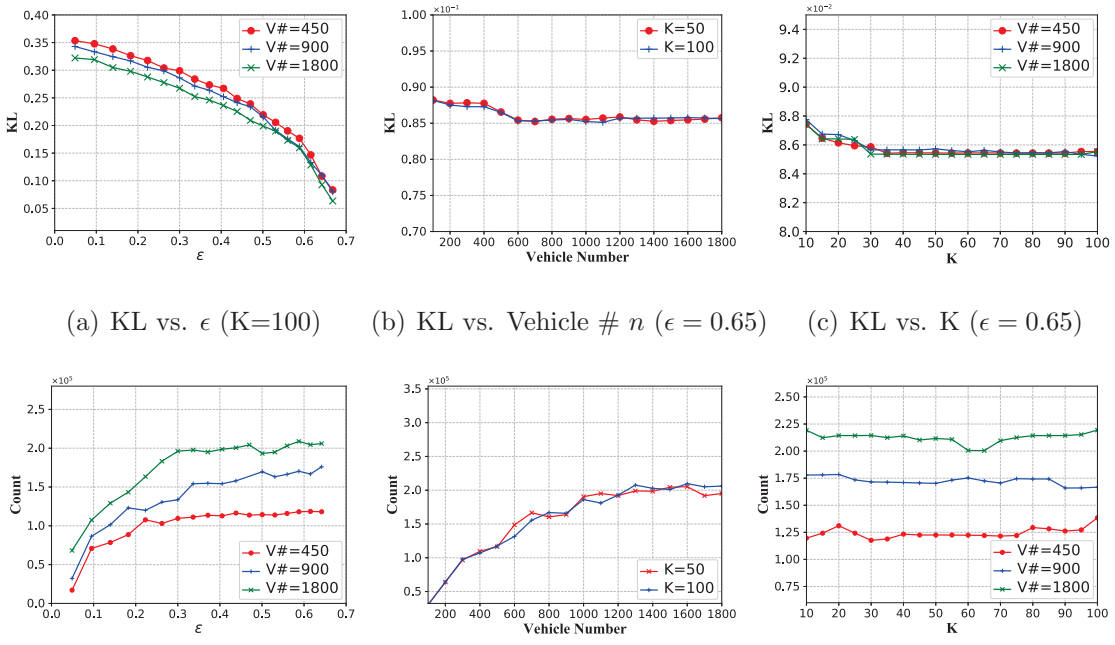


(c) 100 Clusters for Speed and Acceleration

Figure 2.6. Positions (x ft and y ft), speed (v ft/s) and acceleration (a ft/s^2) in the experimental data (Peachtree Street in Atlanta, GA)

which can be approximated as a distinct position. All the coordinates falling into each block share the same position (e.g., the centroid coordinates). Then, we denote such positions as $P_1, P_2, \ldots, P_{|\Omega|} \in \Omega$ where Ω represents the universe of positions and $|\Omega|$ represents its cardinality.

- Second, we can also cluster all the *moving* values (different combinations of v and a) to approximate the moving status of vehicles (e.g., identify K different groups of moving status using K-means clustering [60]). All the combinations of speed and acceleration in the same cluster share the same moving data (e.g., the mean of the cluster). Then, all the distinct approximated moving values are denoted as $M_1, M_2, \ldots, M_{|\Phi|} \in \Phi$ where Φ represents the universe of the approx-



(a) KL vs. ϵ (K=100) (b) KL vs. Vehicle # n ($\epsilon = 0.65$) (c) KL vs. K ($\epsilon = 0.65$)

(d) Output Count vs. ϵ (e) Output Count vs.Vehicle # (f) Output Count vs. K

Figure 2.7. Output utility vs. different parameters

imated combination of speed and acceleration, and $|\Phi|$ denotes its cardinality. Note that K can tune the granularity of the data in the approximation.

- Finally, the day and time are also fine-grained with equal length interval (e.g., 0.1 sec in the NGSIM data), then we consider them as the index of each vehicle's trajectories, and all the unique combinations of day and timestamp are denoted as $T_1, T_2, \ldots, T_{|\Psi|} \in \Psi$ where Ψ is the universe of day and time and $|\Psi|$ denotes its cardinality.

As a result, some representative positions are plotted in Figure 2.6(a) which demonstrates the traffic flow of the arterial road (note that many vehicles make turns at the intersections). For such fine-grained data, we approximate close values using clusters (described above). The coordinates of the positions are approximated by the equal size blocks (16.6ft×16.6ft) in coordinate axes. Since every pair of coordinates (x, y) can uniquely identify a position and the corresponding lane, we skip the lane in

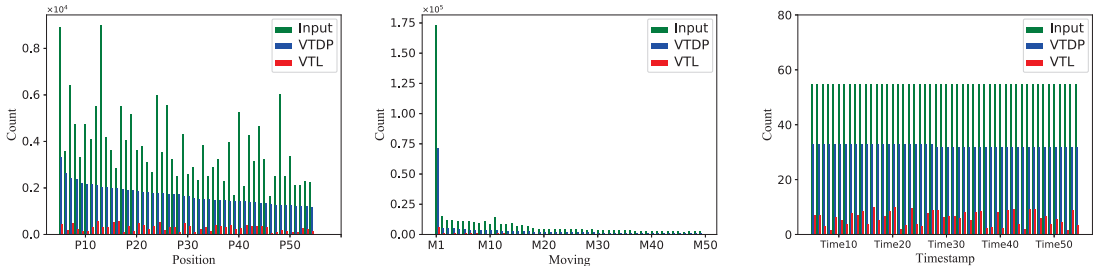
the plots. Furthermore, all the combinations of speed and acceleration are plotted in Figure 2.6(b) and 2.6(c) (clustered by K-Means where K=50 and 100 in the preprocessing while approximating each cluster as a distinct moving value) where the data points inside each cluster are marked with the same color.

We evaluate the utility of our VTDP technique with different privacy bounds for $(\epsilon + \epsilon', \delta)$-differential privacy. We set $\epsilon \in [0.05, 0.65]$ and $\delta = 0.01$. In addition, since ϵ'-differential privacy for computation of optimal counts in phase I follows generic Laplace mechanism [53], we do not evaluate the utility on different ϵ'. Instead, we set $\epsilon' = \ln(2)$. We also test the output utility on different number of vehicles, and different K used in approximating speed and acceleration. Then, we set vehicle number $n \in [100, 200, 300, \ldots, 1800]$ and K $\in [10, 15, \ldots, 95, 100]$.

All the programs were implemented in Python 3.6.4 and tested on an HP PC with Inter Core i7-7700 CPU 3.60GHz and 32G RAM running Windows 10 OS.

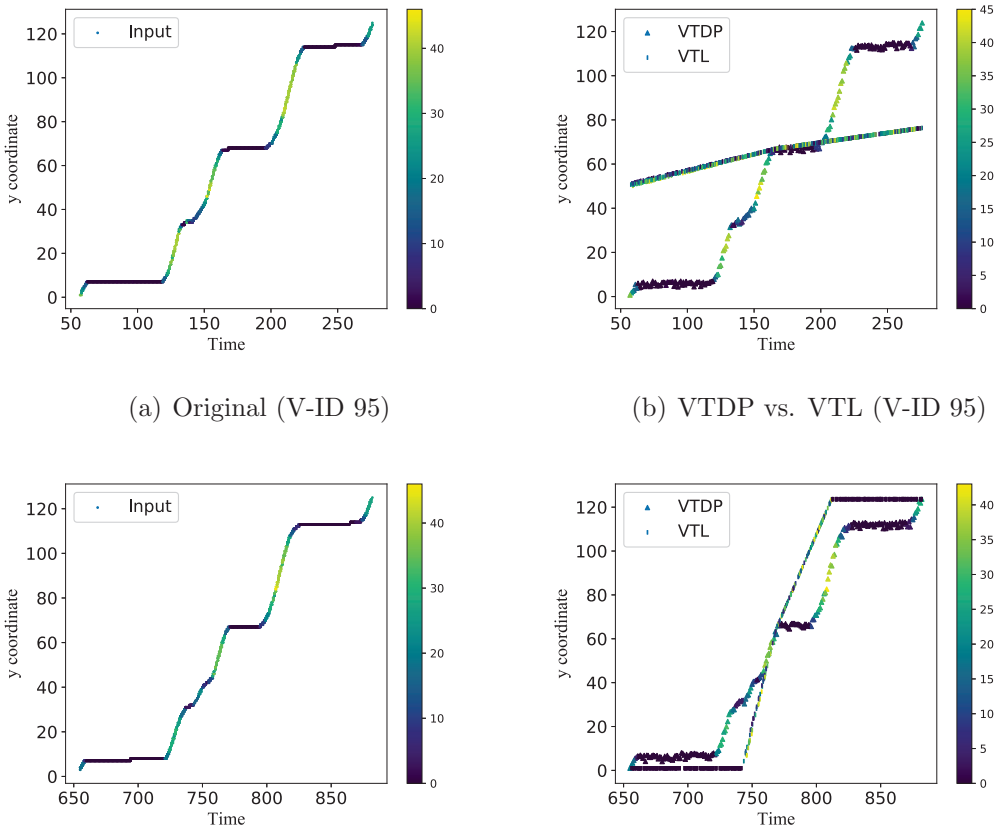
2.7.2 Utility Evaluation. . We first evaluate the output utility using the KL-divergence measure and the total output counts (after interpolation). Notice that, in our VTDP framework, phase I determines the V-IDs and the total output count for each vehicle in O_1, O_2, O_3 while phase II and III sample other attributes by expanding the full tuples based on the output of phase I and the data distribution in the input. Therefore, the minimized KL-divergence in phase I (the objective function of the optimization problem) can be an effective measure for the overall output utility.

Figure 2.7(a) shows the KL-divergence results on varying privacy bound ϵ (given δ and ϵ') in case of different size of the input (different number of vehicle trajectories). As the number of vehicles n increases (from 450 to 1800), the utility performs better given the same privacy bound. However, as large privacy bounds are given, the KL-divergence results are quite close for different number of vehicles



(a) Position Counts (b) Moving Counts (c) Timestamp Counts

Figure 2.8. Retained counts (top 50 frequent) in the output data (VTDP vs. VTL)



(a) Original (V-ID 95) (b) VTDP vs. VTL (V-ID 95)

(c) Original (V-ID 1170) (d) VTDP vs. VTL (V-ID 1170)

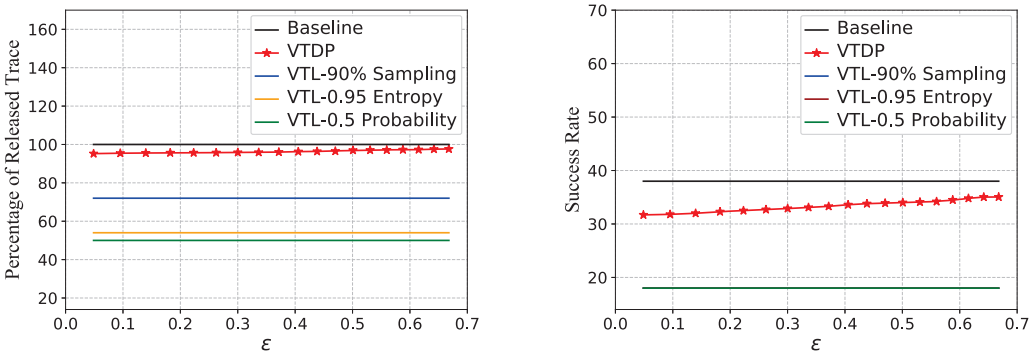
Figure 2.9. Output trajectory comparison (two representative vehicles)

(as shown in Figure 2.7(a) and 2.7(b)). We observe that the KL-divergence for approximating the speed and acceleration is almost steady when K changes (see Figure 2.7(c)).

Our VTDP framework can generate a large number of output tuples via data interpolation where most of the interpolated tuples can be close to the original tuples (since the sampled tuples span over the entire trajectory for most of the vehicles). In the same group of experiments as Figure 2.7(a)-2.7(c), we plot the corresponding output counts in Figure 2.7(d)-2.7(f). The total count of tuples increases as the privacy bound ϵ, and/or number of vehicles n increases. The parameter K for approximating moving values does not affect the output counts. Note that the output utility slightly fluctuates since our VTDP is a multi-phase randomization framework (though the results have been averaged for 5 times). It is worth noting that the total output count is not very close to the input count in case of strong privacy guarantee (i.e., small ϵ for differential privacy). This may also occur in many other high-dimensional data sanitization (e.g., search queries [41], and trajectories [4]) and the output counts can be further enlarged by relaxing the privacy budget due to the tradeoff between privacy and utility. Indeed, since the data distributions of the input and output can be close after the sanitization, the output can still accurately function many applications, as illustrated in Section 2.7.3.

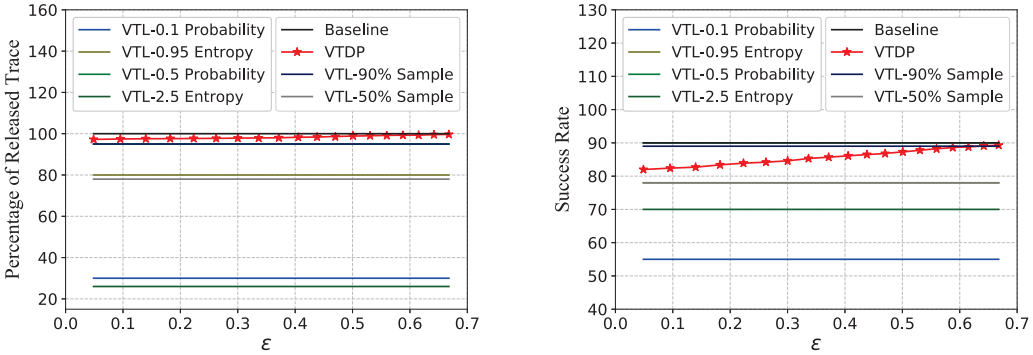
2.7.3 Trajectories Comparison. Besides quantitatively measuring the output utility, we also compare the utility of our VTDP technique with the existing privacy preserving approach ("VTL") [2, 8].[5] We perform two groups of comparisons. First, in Figure 2.8(a)-2.8(c), we plot the top 50 frequent distinct positions (coordinates), moving values (approximated combinations of speed and acceleration), and timestamps (day and time) in one of our experimental results (ϵ=0.65, n=1800, K=100). The counts of such positions, moving values, and timestamps are well preserved after interpolation in our VTDP framework. Note that the interpolation is based on

[5]The utility of VTDP and other differentially private trajectory sanitization techniques (e.g., [4–6]) are incomparable, since the output of VTDP is generated with *more attributes* (and not aggregated).



(a) Released Trace (50%)

(b) Success Rate (50%)



(c) Released Trace (100%)

(d) Success Rate (100%)

Figure 2.10. Queue length estimation – penetration rate (50% or 100%): percent of vehicles in data collection (e.g., by mobile sensors, traffic cameras)

timestamps in the input (considering timestamps as the index of tuples), the counts of all the distinct timestamps are quite close, but slightly smaller (compared to VTL) than that in the input simply because some of the tuples for each vehicles have not been sampled.

Second, we apply the same interpolation to both VTDP and VTL (discussed in Section 2.6.1), and then we compare the results obtained for the output data posed by each specific vehicle. More specifically, in Figure 2.9, we plot a part of the trajectories of two representative vehicles (e.g., Vehicle 95 and 1170) in the arterial road. The y axes in Figure 2.9 show the longitudinal coordinates of the positions (note that lateral

coordinates have negligible changes in the trajectories in that arterial road, we thus skip it for better visualization). The x axes in Figure 2.9 show the timestamps in sequence. The color bar presents the speed at different times. The results demonstrate that our VTDP technique can well preserve the trajectories and moving data (e.g., speed) – the trajectories for the two vehicles lie very close to the input compared to the interpolated results of VTL.

2.7.4 Comparison via Queue Length Estimation. We also evaluate that sanitized vehicle trajectories can still be effectively used for traffic modeling. Then, we apply real world traffic modeling applications, e.g., queue length estimation [2, 43] (which predicts the queue length at the traffic intersections) to our sanitized vehicle trajectories (generated by VTDP) and the output data generated from VTL techniques [8]. Thus, we compare the queue length estimation results derived from the VTDP output with the VTL outputs.

In literature, there are three different VTL techniques established based on different criteria (e.g., sampling, entropy, and probability) [8]. First, *sampling based VTL technique* randomly captures a portion of the traces (say 50%) at each VTL zone. Second, *probability based VTL technique* treats tracking probability as a privacy metric to generate the VTLs. It ensures that the released traces should have low tracking probability, e.g., 0.2 probability indicates that no more than one out of five vehicles can be successfully tracked. Third, *entropy based VTL technique* calculates the entropy value for a specific location trace for all possible vehicles that previously passed for specific VTL zones. Higher level of entropy gives higher confusion and better privacy.

In Figure 2.10, we demonstrate the queue length estimation results (VTDP vs. different VTLs) with measures *percentage of released trace* and *Success Rate*. The success rate of queue length estimation (also adopted in [2]) indicates the performance

of traffic modeling application, which defines as the percentage of cycles that the proposed algorithms can be successfully applied (i.e., cycles that have 2 or more samples of queued vehicles) [2]. Specifically, the "Baseline" results are captured with all the data around the VTL zones. While testing our sanitized data using queue length estimation application with different ratios of vehicles involved in the data collection (50% and 100% penetration rate), the results are very close to the baseline (as shown in Figure 2.10). Furthermore, compared to three different VTL techniques with different parameters (e.g., 90% sample, 50% sample, 0.5 probability, 0.1 probability, 2.5 entropy, 0.95 entropy), the vehicle trajectories generated by VTDP can provide better success rates for queue length estimation at signalized intersections. We can also observe that both the percentage of released trace and success rate lie closer to the baseline as ϵ increases (better utility with increased privacy budget).

Overall, our VTDP technique can generate vehicle trajectories with better utility than the state-of-the-art while ensuring stronger privacy guarantee. Recall that, applying some existing techniques (e.g., [4–6]) to fine-grained vehicle trajectories generates either incomplete attributes (suppressing moving values and timestamps) or aggregated data (e.g., for locations/positions). Thus, the results are incomparable with our VTDP technique.

2.7.5 Computational Costs. Since our VTDP algorithm has $O(n^2)$ complexity: $O(n^2)$ for optimal counts computation, $O(n)$ for three phases of sampling, and $O(n)$ for data interpolation, the vehicle trajectory data can be sanitized with high efficiency and scalability. Thus, we do not present such low computation costs due to space limit.

2.8 Related Work

Vehicle trajectory data generated in mobile apps, traffic monitoring cameras,

and GPS navigation system have great values to function intelligent transportation systems and smart cities. However, the privacy concerns in such data have received much attention, and have never been adequately addressed. In the prior work, some privacy techniques (including data sanitization [4–6, 61] and VTLs [2, 8]) are proposed to moderate the privacy issues. However, VTL techniques cannot fully protect the privacy (with provable guarantee) and existing data sanitization techniques cannot generate satisfactory fine-grained vehicle trajectory data for urban traffic modeling [36]. To address such limitations, our proposed VTDP technique satisfies the differential privacy with boosted utility.

Dwork et al. [9, 53] first proposed the rigorous privacy definition of differential privacy, which is a randomization algorithm which guarantees that for any two neighboring input datasets, the probabilities of generating any output from two inputs are bounded. This notion provides sufficient privacy protection for users regardless of the prior knowledge possessed by the adversaries. In the past decade, this has been extended to data release in different contexts. For instance, McSherry et al. [62] solved the problem of producing recommendations from collective user behavior while providing differential privacy for users. Wang et al. [63, 64] proposed a differentially private schemes for video analytics. In particular, some non-interactive differentially private data sanitization techniques [41, 57, 65] are very relevant to our work. Li et al. [57] identified the weakness of k-anonymity and proposed a privacy notion of safe k-anonymization to address such vulnerability by applying random sampling to meet k-anonymity and differential privacy. Bild et al. [65] proposed an approach for implementing the traditional data anonymization algorithm (k-anonymity) with differentially private components where k-anonymization was employed in order to reduce the added noise. Both techniques generate sanitized outputs for generic datasets while satisfying k-anonymity and differential privacy simultaneously. In addition, Hong et al. [41] proposed a multinomial sampling based approach to generate sanitized

search logs while maximizing the output utility. Phase I in our VTDP framework is inspired from such work where trajectory data (e.g., position coordinates) are substantially different from the search logs. Also, phase II and III (in VTDP) sample additional values (e.g., speed, acceleration and timestamps) based on the output of the multinomial sampling and the original input (whereas the timestamps in [41] are not published). Also, vehicle trajectory data provides properties to further improve the output utility via data interpolation.

Furthermore, previous work on preserving privacy in practical transportation systems is sparse. Hoh et al. [43] rely on a notion of privacy, k-anonymity, that is not particularly strong at preserving location privacy [66]. In particular, they focus on privacy for individual measurements, and thus do not directly offer formal protection for users transmitting time series such as location traces. Some research on privacy for location-based services, e.g., [34], can be considered somewhat related to our work. These works are typically concerned with perturbing GPS location traces to provide privacy while reconstructing some aggregate statistics, e.g, average density. However, they generally either do not rely on a formal definition of privacy, or consider simply the minimization of mutual information between the users' private data and the published data, which ignores the crucial issue of side information. In addition, Li et al. [67] has quantified the privacy leakage while sharing the locations in mobile social networks, and proposed a system-level solution (i.e., SmartMask) to prevent the location privacy breaches. Similar to our work, Ny et al. [38, 68] consider more traditional static sensors, e.g., single loop detectors. However, such techniques do not collect the fine-grained trajectory data, and fine-grained vehicle trajectories are not generated for output, either.

In intelligent transportation systems, privacy preserving VANET (Vehicular Ad-hoc Networks) applications [69, 70] may generate similar datasets. However, our

VTDP significantly differs from such works. Specifically, our VTDP focuses on the differentially private vehicle trajectory (including speed and acceleration) data sanitization. In such case, a data curator applies the proposed offline algorithm to generate a publishable dataset, which can be shared to any untrusted party. However, VANET focuses more on real-time communications between vehicles and/or infrastructure in a short range (e.g., real time computation/communication for road safety, and navigation) where privacy is generally ensured by cryptographic schemes [70].

2.9 Summary

As the rapidly growing deployment of intelligent transportation systems (ITS) and smart traffic applications, fine-grained vehicle trajectory datasets are generated from everywhere in our real life, e.g., GPS navigation systems, mobile applications, and urban traffic cameras. Although these data are extremely valuable for the ITS development, privacy risks also arise if such data are not properly sanitized before release for analysis. Recently, some researchers have proposed techniques to guarantee the privacy of vehicle trajectory data, but still have some limitations.

In this chapter, we take the first step to propose a differentially private vehicle trajectory data sanitization framework that can guarantee both strong privacy protection and high output utility. Differential privacy ensures the protection against inferences (whether any vehicle is involved in the input data) by the adversaries with arbitrary background knowledge. Our VTDP framework follows the sequential composition of multiple phases (parallel composition also exists in sampling phase II and phase III) but with limited overall bounds $(\epsilon + \epsilon', \delta)$. Our VTDP also greatly improves the output utility with the proposed vehicle trajectory interpolation based on the attributes of vehicle trajectory data. As validated in our experimental results, our VTDP framework generates fine-grained vehicle trajectory data with high utility, compared to the existing techniques (i.e., the VTL based techniques).

CHAPTER 3

SECURE MULTI-PARTY COMPUTATION (SMC) ON DIVISIBLE DOUBLE AUCTION

Auction mechanism generally requires a trusted-third party as the *market mediator* to coordinate bidding and resource allocation via collecting private data from the agents, which may arouse severe privacy concerns and high computation overheads. To address such issues, we propose a novel privacy-aware double auction framework (namely PANDA) by designing an efficient cryptographic protocol to privately execute double auction for divisible resources among all the agents. To ensure *privacy* and *truthfulness*, PANDA delicately co-designs VCG auction and cryptographic protocol, which is equivalent to a mediator for sealed-bid auction of divisible resources.

The work presented in this chapter have been published at [71] [6]

3.1 Background

In the past decade, divisible resources have been frequently exchanged in the electricity markets (e.g., electricity [72–75]), cloud markets (e.g., computation and storage resources [76]), financial markets (e.g., stock shares [77]), wireless networks (e.g., bandwidth [78]), among others. In such markets, each agent may sell resources with arbitrary amounts to any other buyers, and all the agents generally compete with each other by seeking for their maximum payoffs. Then, auction mechanisms have been extensively studied for exchanging such divisible resources to achieve the Nash Equilibrium [12, 13]. Since auctions request all the potential buyers to propose bid prices [13, 14] (in particular, double auction [15] requests both potential buyers and sellers to simultaneously submit their prices), a trusted-third party is established

[6] ©2020, IFAAMAS, with permission from Bingyu Liu, Shangyu Xie, and Yuan Hong. *PANDA: Privacy-Aware Double Auction for Divisible Resources without a Mediator, 2020.*

as the *market mediator* to coordinate the bidding and resource allocation in the auctions. The establishment of the mediator may result in high operational costs, extra charges to buyers/sellers, high computation burden, and high demand of trust on the mediator.

If directly eliminating the mediator in the auction, severe privacy concerns may occur since all the agents should disclose their local private data for completing the auction. In addition, some agents may try to win more payoffs in the auction by reporting untruthful bids, especially in sealed-bid auctions [16]. Even worse, agents (aka. potential buyers or sellers) may collect such information from their competitors [17], and misuse such private data, e.g., reselling the data (a mediator may also do so).

In this chapter, we propose a novel auction framework (namely PANDA) by designing an efficient cryptographic protocol among all the buyers and sellers to privately execute double auction for divisible resources. Specifically, we construct the cryptographic protocol with the fundamental cryptographic primitives: Homomorphic Encryption (HE) [18, 19] and Secure Function Evaluation (SFE) [20]. Then, the cryptographic protocol enables all the agents to securely communicate with each other and complete the transactions with limited information disclosure. Per the secure multiparty computation (MPC) theory [21, 22], the cryptographic protocol can be proven to be equivalent to a mediator. Furthermore, we design a double auction [11] based on the Vickrey-Clarke-Groves (VCG) [23, 24] mechanism in PANDA to ensure truthfulness.

3.2 Double Auction

We denote a set of buyers as $\mathcal{B}$ and sellers as $\mathcal{S}$ in the auction, where each buyer/seller submits a two-dimensional bid profile (bid price, and the maximum amount

to buy/sell) as follows: 1) buyer $m \in \mathcal{B}$: $b_m = (\alpha_m, d_m)$, and 2) seller $n \in \mathcal{S}$: $s_n = (\beta_n, h_n)$. Also, we denote the valuation function of each buyer m as $\widehat{V}_m(A_m)$ with its amount to buy A_m and the cost function of each seller n as $\widehat{C}_n(A_n)$ with its amount to sell A_n. Moreover, the valuation function $\widehat{V}_m$ follows a generic setting [23, 24]: (1) $\widehat{V}_m$ is differentiable and $\widehat{V}_m(0) = 0$, and (2) $\widehat{V}'_m$ is non-increasing and continuous.

We also denote the payoff function for buyer m and seller n as $f_m(r)$ and $f_n(r)$, respectively. In a VCG mechanism [23, 24], *transfer payment* is defined as the difference between all the agents' aggregated valuation if any agent is not in the auction minus the aggregated valuation if such agent is in the auction [11]. We denote the transfer payments for buyer m and seller n as $\rho_m(r)$ and $\rho_n(r)$, where r is the set of bid profiles. Thus, we have:

$$\rho_m(r) = \sum_{m \neq i} \alpha_m [A_m(0; r_{-i}) - A_m(r_i; r_{-i})] \tag{3.1}$$

$$\rho_n(r) = \sum_{n \neq j} \beta_n [A_n(0; r_{-j}) - A_n(r_j; r_{-j})] \tag{3.2}$$

Then, given the optimal allocation profile for buyers/sellers $A^*_m(r), A^*_n(r)$, we can get the payoff of the buyer/seller:

$$f_m(r) = \widehat{V}_m(A^*_m(r)) - \rho_m(r), \forall m \in \mathcal{B} \tag{3.3}$$

$$f_n(r) = -\widehat{C}_n(A^*_n(r)) - \rho_n(r), \forall n \in \mathcal{S} \tag{3.4}$$

Definition 4 (Nash Equilibrium in PANDA)**.** Given the bid profiles r, a Nash Equi-

librium (NE) holds such that:

$$\forall m \in \mathcal{B}, f_m(b^*_m, r^*_{-m}) \geq f_m(b_m, r^*_{-m}) \tag{3.5}$$

$$\forall n \in \mathcal{S}, f_n(s^*_n, r^*_{-n}) \geq f_n(s_n, r^*_{-n}) \tag{3.6}$$

3.3 Privacy-Aware Double Auction

The proposed protocol ensures that all the bid profiles are encrypted and privately computed in multiple iterations to achieve the best responses for the Nash Equilibrium (NE).

3.3.1 Overview of Framework. Figure 5.1 shows the major steps of the PANDA framework. In the initialization of each auction, PANDA first executes *Init()* to privately derive an initial bid profile while ensuring valid conditions for the auction via secure function evaluation (SFE) and *Aggre()*. Then, *IterUpdate()* is executed to privately update the potential amount C and *BestRespon()* is sequentially executed to privately compute the best response in each (current) iteration k. Finally, the auction reaches Nash Equilibrium after iteratively updating the potential amount and the best response. The details of each algorithm will be illustrated in the following section.

Properties of PANDA. First, PANDA inherits the properties of auction [79], i.e., budget balance, Pareto efficiency, and existence of NE. Our proposed framework works under semi-honest model that all the agents follow the protocol but may curious to infer others' private information. [21, 22]). While addressing the above threats, PANDA has the following properties.

1. **Decentralized**: no central market mediator or operator to coordinate agents to finish the auction.

2. **Privacy**: each agent's bid profile (the bid price and amount) is kept private; every pair of potential buyer and seller only know the amount in their transaction

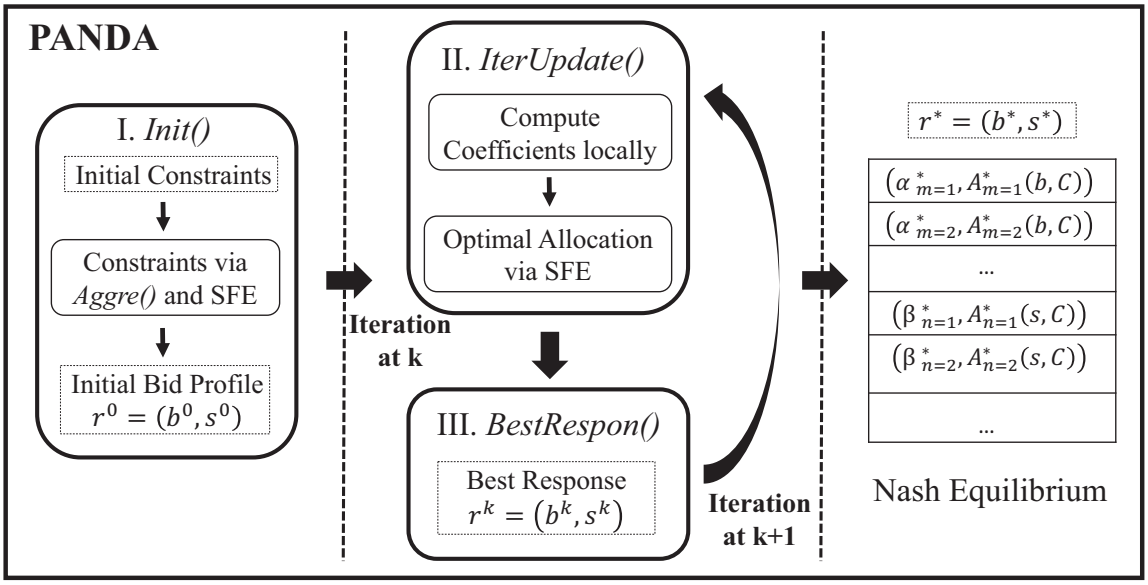


Figure 3.1. PANDA Framework

(and the *clearing price*).

3. **Truthfulness:** each agent truthfully participates in the auction would gain more payoff than the untruthful response.

3.3.2 Algorithms. In this section, we present three main procedures as below:

Init() PANDA first executes *Init()* to privately generate valid initial conditions. Specifically, secure function evaluation (SFE) is executed to privately ensure $\{\alpha_m\}_{max} < \{\beta_n\}_{min}$: such bid profiles would result in a valid auction (if not evaluated to be true, then all the agents execute it again). This step also calls another algorithm *Aggre()*, which is used to securely sum up the amounts of all the buyers and sellers. *Aggre()* mainly uses the additive property of Homomorphic Cryptosystem for the aggregation. Thus, the potential amount C (the common amount allocated in each side of the double auction) can also be determined. Note that C (which is initialized before the auction) is smaller than the total amounts. The auction moves to the next step once meeting the initial constraints.

IterUpdate() It privately updates the potential amount as $\widetilde{C}(r, C)$ with the following equation:

$$\widetilde{C}(r, C) = Q(r, C) + \frac{p_b(r, C) - p_s(r, C)}{\omega_{max} + \sigma_{max}} \tag{3.7}$$

ω_{max} and σ_{max} are denoted as the upper bound of the gradients. With the gradients of buyers' marginal valuations and sellers' marginal costs, the potential amount can reach the NE more efficiently. The minimum aggregated amounts of buyers and sellers $Q(r, C)$ can be obtained by SFE. It is assumed that the matched prices $p_b(r, C) = \min\{\alpha_i, A_i \geq 0\}$ and $p_s(r, C) = \max\{\beta_j, A_j \geq 0\}$ can be known to the other agents. Note that $\frac{p_b(r,C)-p_s(r,C)}{\omega_{max}+\sigma_{max}}$ is a private coefficient for the gradients of marginal valuations (costs).

Then, in iteration k, each agent locally updates the best response w.r.t. the bid profile of the others, and then jointly finds the optimal allocation using the SFE as below:

$$A_m^*(b, C) = \min\{d_m, \max\{[C - \sum_{i \in \mathcal{T}_m} d_i], 0\}\} \tag{3.8}$$

$$A_n^*(s, C) = \min\{h_n, \max\{0, [C - \sum_{j \in \mathcal{T}_n} h_j]\}\} \tag{3.9}$$

where $\mathcal{T}_m = \{i \in M; s.t.\ \alpha_i > \alpha_m\} \cup \{\alpha_i = \alpha_n \wedge i < m\}$ and $\mathcal{T}_n(s) = \{j \in N; s.t.\ \beta_j > \beta_n\} \cup \{\beta_i = \beta_m \wedge j < n\}$.

BestRespon() It is executed to derive the best response for buyer $m \in \mathcal{B}$ and seller $n \in \mathcal{S}$ (denoted as b_m^* and s_n^*, respectively). Then, we can calculate the

optimal profiles:

$$b_m^* = \arg\max\{f_m(b_m, b_{-m})\} \tag{3.10}$$

$$s_n^* = \arg\max\{f_n(s_n, s_{-n})\} \tag{3.11}$$

Recall that *IterUpdate()* iteratively returns the optimal allocation with SFE for every buyer/seller, then PANDA finally converges in the auction with the best responses of all the agents under Nash Equilibrium (NE). The matched prices from buyers and sellers eventually coverage to the *clearing price*.

3.4 Summary

In this chapter, we have proposed a novel framework PANDA that securely executes double auction for divisible resources by integrating the VCG mechanism and cryptographic protocol, which is equivalent to a market mediator. PANDA ensures privacy and truthfulness in the distributed computation among all the agents.

CHAPTER 4

TRUSTED EXECUTION ENVIRONMENT (TEE) ON DOUBLE AUCTION

Double auction mechanisms have been designed to trade a variety of divisible resources (e.g., electricity, mobile data, and cloud resources) among distributed agents. In such divisible double auction, all the agents (both buyers and sellers) are expected to submit their bid profiles, and dynamically achieve the best responses. In practice, these agents may not trust each other without a market mediator. Fortunately, smart contract is extensively used to ensure digital agreement among mutually distrustful agents. The consensus protocol helps the smart contract execution on the blockchain to ensure strong integrity and availability. However, severe privacy risks would emerge in the divisible double auction since all the agents should disclose their sensitive data such as the bid profiles (i.e., bid amount and prices in different iterations) to other agents for resource allocation and such data are replicated on all the nodes in the network. Furthermore, the consensus requirements will bring a huge burden for the blockchain, which impacts the overall performance. To address these concerns, we propose a hybridized TEE-Blockchain system (system and auction mechanism co-design) to privately execute the divisible double auction. The designed hybridized system ensures privacy, honesty and high efficiency among distributed agents. The bid profiles are sealed for optimally allocating divisible resources while ensuring truthfulness with a Nash Equilibrium. Finally, we conduct experiments and empirical studies to validate the system and auction performance using two real-world applications.

The work presented in this chapter have been published at [80][7] and [81][8]

---

[7]

[8]

4.1 Overview

Divisible resources (e.g., electricity, mobile data, and computation and storage resources in the cloud) have been frequently traded or allocated in a peer-to-peer mode. All the agents can purchase or sell any amount of the resources in such markets. Since all the agents generally compete with each other to maximize their payoffs, divisible double auction mechanisms [11] are designed to allow both buyers and sellers to dynamically submit their prices until convergence (e.g., achieving the Nash Equilibrium [12, 13]) and then complete the transaction with resource allocation. Recently, smart contracts (as decentralized and self-enforcing contracts) can be designed for distributed agents to trade divisible resources with digital agreements. The blockchain-based platform supports the execution of smart contracts for strong integrity and availability, which maintain the transparency, traceable and consensus properties.

However, severe privacy concerns may arise in both double auction [82] and blockchain-based systems [83]. For instance, during the auction, all the agents report their bidding profiles, including sensitive data such as their bidding amount and bidding prices. As rival agents, they may want to win competitive advantages in the market (more payoffs) by reporting untruthful bids if they know the others' bid profiles. Then, the market [16] would be deviated. Even worse, such private data might be collected and resold [82] to other untrusted parties.

To this end, it is desirable to propose a truthful divisible double auction mechanism while preserving all the agents' privacy (at least sealing all the bid profiles). Specifically, smart contracts on the blockchain system can be designed for the divisible double auction. However, the blockchain system has limitations on preserving privacy

---

for sensitive data and high performance execution. To complement the blockchain system, the Trusted Execution Environment (TEE) [28] could address such limitations by executing the core functionality (e.g., computation for the smart contract) in the *enclave*, which protects the data against malicious attacks. Compared with other types of secure and private solutions (e.g., Secure Multiparty Computation (SMC) [18, 25, 26]), TEE achieves stronger security and high efficiency for blockchain execution [27]. Thus, in this paper, we propose an efficient and privacy preserving divisible double auction with the TEE-Blockchain hybridized system (e.g., on the Intel SGX, which is a TEE supported by an architecture extension of Intel [28]). Then, the hybridized system is co-designed in three aspects.

- First, the blockchain-based platform is expected to ensure integrity and availability while it interacts with other components (i.e., TEE) for the transaction, which helps data/state recovery if the execution/protocol is broken or interrupted by accidents.
- Second, the smart contract can be loaded and executed within a protected environment in Intel SGX, (namely *enclave*) [84]. All the agents' sensitive data can be protected during the computation.
- Third, we propose an efficient, individually rational and weakly budget balanced double auction based on the Progressive Second Price (PSP) [23] auction, derived from the Vickrey-Clarke-Groves (VCG) [24] auction. The proposed divisible double auction ensures truthfulness for all the agents by achieving a Nash Equilibrium.

Furthermore, we conduct experiments for both *off-chain* procedures (executing the TEE program computation) and *on-chain* procedures (the interaction between the blockchain and TEE) in the hybridized system to evaluate the system and auction

performance using two real-world applications: (1) energy trading, and (2) wireless bandwidth allocation.

4.2 Background

4.2.1 Divisible Double Auction. In a divisible double auction, let $\mathcal{B}$ and $\mathcal{S}$ be the sets of buyers and sellers, respectively. The bidding information includes two-dimensional bid profiles, denoted as b_m for buyers and s_n for sellers. During the auction, the bid profiles are submitted as follows: (1) buyer $m \in \mathcal{B}$: $b_m = (\alpha_m, d_m)$ with bid price α_m and amount d_m to buy, and (2) seller $n \in \mathcal{S}$: $s_n = (\beta_n, h_n)$ with bid price β_n and amount h_n to sell. $b = (b_m, m \in \mathcal{B})$ denotes the bid profiles of all the buyers while $s = (s_n, n \in \mathcal{S})$ denotes the bid profiles of all the sellers. In addition, $r = (b, s)$ is defined as a strategy profile, which represents the bid profiles for all the agents. These are private information to be sealed amongst all the agents in the auction. A strategy profile without agent i is denoted as $r_{-i} = (r_1, ...r_{i-1}, r_{i+1}, ..., r_{|m+n|})$, then $r = (r_i; r_{-i})$.

From the global viewpoint, the main goal of the divisible double auction mechanism is to seek the maximum social welfare for optimal allocation. We use A_m and A_n to denote the allocation of buyer m and seller n, respectively. In the current iteration (k-th iteration) of the double auction, $A_m^{(k)}$ and $A_n^{(k)}$ represent the allocation for buyer m (*amount to purchase*) and seller n (*amount to sell*), respectively.

4.2.2 Trusted Execution Environment (TEE). TEE provides a fully isolated environment to prevent others (e.g., software, OS, and hosts) from tampering with or learning the state of an application running in it.

Intel SGX [85]: it is an instance of TEE that enables process execution in a protected address space *enclave*. The *enclave* ensures confidentiality and integrity for the process against attacks. An *enclave* is not allowed to make system calls, but can

read/write memory outside the *enclave* region. Thus, the isolated execution can be viewed as an ideal model which guarantees to be correctly executed with confidentiality. We denote the double auction program inside the *enclave* as Prog_x.

Remote Attestation: it allows to remotely verify if the pieces of code or program are running within the TEE or not. In Intel SGX, CPU can measure the trusted memory, cryptographically sign the computed results, and generate the signatures for the attesting party. The private key is only known to the hardware over the program. Group signatures (EPID) [86] are used for setting up a secure channel for *remote attestation*.

4.2.3 Smart Contract. Cryptocurrencies are traded on the decentralized network of peers which stores all the transactions via a public ledger. Through the consensus protocol, the ledger is stored as a chain of blocks with the agreement state. Smart contract is a machinery built on top of cryptocurrencies, and it defines and executes the contract on the blockchain. In other words, the smart contracts work as pieces of code or program digitally among distributed agents [87]. Based on the decentralized cryptocurrencies, the integrity and availability can be guaranteed. In our work, privacy will be ensured by TEE.

4.3 Auction Mechanism Design

We represent the strategy of each agent with a non-negative valuation function $\widehat{V}_m(\cdot)$ for buyers, which indicates the willingness to pay, or value for buyers to obtain the amount of divisible item. Similarly, we have negative cost function $\widehat{C}_n(\cdot)$ for sellers. In the auction design, we adopt generic assumptions [23, 24] for the valuation function $\widehat{V}_m(\cdot)$: (1) $\widehat{V}_m$ is differentiable, concave and $\widehat{V}_m(\emptyset) = 0$, and (2) $\widehat{V}'_m(\cdot)$ is non-increasing and continuous; for the cost function $\widehat{C}_n(\cdot)$: (1) $\widehat{C}_n$ is differentiable, convex and $\widehat{C}_n(\emptyset) = 0$, and (2) $\widehat{C}'_n(\cdot)$ is increasing and continuous;

In our settings, buyers have diminishing marginal utility while sellers have increasing marginal cost. This indicates that $\widehat{V}_m(A_m^k) > \widehat{V}_m(A_m^{k+1})$ $(\forall m \in \mathcal{B})$ where $A_m^k < A_m^{k+1}$ while $\widehat{C}_n(A_n^k) < \widehat{C}_n(A_n^{k+1})$ $(\forall n \in \mathcal{S})$ where $A_n^k < A_n^{k+1}$.

4.3.1 Problem Formulation. Assuming that each agent is selfish with the goal to maximize their own payoff. Therefore, they may untruthfully modify their bids in the auction. With the blockchain-based system to realize the smart contract for the auction, untruthful responses could be detected, and thus penalty will be applied to the cheating agent.

Thus, valuation function will be converted to $\widehat{V}_m(\cdot) - \mu_p(\cdot)$ where $\mu_p(\cdot)$ is a anti-monotonic function for measuring the penalty applied to the cheated amount for the buyers [88]. Note that $\mu_p(0) = 0$ means if the valuation is submitted and penalty will be exempted. Similarly, the cost function will be updated as $\widehat{C}_n(y_n) + \mu_p(\cdot)$ where $\mu_p(\cdot)$ is a monotonic function (and increasing derivative) for measuring the penalty applied to the sellers [88] and $\mu_p(0) = 0$ (exempting the penalty for truthful response of the sellers).

Payoff Functions. Then, the payoff functions are defined for buyer m and seller n as $f_m(r)$ and $f_n(r)$, to represent their payoffs w.r.t. the bid profiles of all the agents r. Specifically, ρ_m is the payment made by buyer m while ρ_n is the payment received by seller n. Moreover, $\rho(r_i, r_{-i})$ is defined as the difference between all the buyers' aggregated valuation if any buyer i is absent in the auction minus the aggregated valuation if i is included the auction [11, 23, 89]. Similarly, $\rho(r_j, r_{-j})$ is defined as the difference between all the sellers' aggregated cost if any seller j is absent minus the aggregated cost if j is included. Thus, we have:

$$\rho(r_i, r_{-i}) = \sum_{m \neq i} \alpha_m [A_m(0; r_{-i}) - A_m(r_i; r_{-i})]$$
$$\rho(r_j, r_{-j}) = \sum_{n \neq j} \beta_n [A_n(0; r_{-j}) - A_n(r_j; r_{-j})] \tag{4.1}$$

Then, given the optimal allocation profile for buyer $m \in \mathcal{B}$ and seller $n \in \mathcal{S}$ as A_m^* and A_n^*, we can define the payoffs for the buyer m and seller n as:

$$f_m(r) = \widehat{V}_m(A_m^*) - \rho(r_i, r_{-i}), \forall m \in \mathcal{B}$$
$$f_n(r) = \rho(r_j, r_{-j}) - \widehat{C}_n(A_n^*), \forall n \in \mathcal{S} \tag{4.2}$$

Definition 5 (*Individual Rationality*)**.** The divisible double auction mechanism achieves individual rationality if the following holds: $f_m(r) \geq 0$ and $f_n(r) \geq 0$.

It ensures that the all the agents obtain non-negative payoff while participating in the auction mechanism.

Definition 6 (*Incentive Compatibility*)**.** The divisible double auction mechanism achieves incentive compatibility if the following holds: $f_m(r) \geq f_m(\overline{r})$ and $f_n(r) \geq f_n(\overline{r})$ where r and $\overline{r}$ are denoted as the true bid profile and false bid profile.

It ensures that all the agents in the auction will obtain the maximum payoff if they report the truthful bid.

Definition 7 (*Weak Budget Balance*)**.** In the divisible double auction, for $\forall m \in B$ and $\forall n \in S$, if there exists: $\sum_{\forall m \in \mathcal{B}}(\alpha_m \cdot d_m) \geq \sum_{\forall n \in \mathcal{S}}(\beta_n \cdot h_n)$, then the auction mechanism satisfies weak budget balance.

It ensures "no budget deficit" in the auction.

Definition 8 (*Clearing Price*)**.** The price θ is defined as the *clearing price* for an optimal allocation $A^*(\cdot)$, if there exists a feasible and efficient allocation, such that, the best response is achieved for the maximum social welfare, denoted as $F(\cdot) = \sum_{m\in\mathcal{B}} \widehat{V}_m(A_m) - \sum_{n\in\mathcal{S}} \widehat{C}_n(A_n)$.

We say that the clearing price θ [90] supports the optimal allocation $A^*(\cdot)$ with the maximum social welfare.

Definition 9 (*Nash Equilibrium*)**.** In the divisible double auction, Nash Equilibrium holds if given the bid profile r^* such that:

$$\begin{aligned} f_m(b_m^*, r_{-m}^*) &\geq f_m(b_m, r_{-m}^*), \forall m \in \mathcal{B} \\ f_n(s_n^*, r_{-n}^*) &\geq f_n(s_n, r_{-n}^*), \forall n \in \mathcal{S} \end{aligned} \tag{4.3}$$

where $r_{-m} = \{r\} \setminus \{b_m\}$ is a bid profile for all the buyers excluding buyer m from $\mathcal{B}$ and $r_{-n} = \{r\} \setminus \{s_n\}$ is a bid profile for all the sellers except seller n from $\mathcal{S}$.

Our divisible double auction mechanism will find the optimal allocation for all the agents to achieve the maximum social welfare. Moreover, the truthfulness of bids will be ensured in the smart contract via *individual rationality* and *incentive compatibility*. To preserve privacy, all the agents' bid prices and amounts (bid profiles), as well as the valuation/cost functions can be protected in the auction. The clearing price and trading amount will only be disclosed to every pair of potential sellers/buyers at the end of the auction (after convergence).

4.3.2 Divisible Double Auction Mechanism. We now design the divisible double auction mechanism (DA), which will be executed as a smart contract inside the TEE. The procedures are detailed as below:

1 Initialization. Denoting the double auction program as $\mathsf{Prog}_{\mathsf{da}}$, while executing $\mathsf{Prog}_{\mathsf{da}}$ in the *enclave*, the decrypted bid profiles of all the agents will be checked if they satisfy the initial condition (i.e, $(\alpha_i)_{\max} \geq (\beta_j)_{\min}$). Otherwise, the state of auction will be turned from "active" into "fail". Then, it requires all the agents to update their bid profiles. Meanwhile, the potential amount of the resources C should be smaller than the overall demand/supply.[9] The auction will be active if and only if satisfying the above conditions.

2 Iteration. Once the iteration starts, the potential amount $\widehat{C}(r, C)$ is updated as below:

$$\widetilde{C}(r, C) = Q(r, C) + \frac{p_b(r, C) - p_s(r, C)}{\omega_{\max} + \sigma_{\max}} \tag{4.4}$$

where $Q(r, c) = \min\{\sum_{m \in \mathcal{B}} A_m^*,\ \sum_{n \in \mathcal{S}} A_n^*\}$, $p_b(r, C) = \min\ \{\alpha_i, A_i \geq 0\}$, $p_s(r, C) = \max\{\beta_j, A_j \geq 0\}$ and $\widehat{\mathcal{P}} = \frac{p_b(r,C) - p_s(r,C)}{\omega_{\max} + \sigma_{\max}}$.

We denote $Q(r, C)$ as the minimum value of total demand and total supply; A coefficient $\widehat{\mathcal{P}}$ is used for gradients of marginal valuations or costs; Two variables $p_b(r, C)$ and $p_s(r, C)$ are defined to stimulate the much faster coverage in each iteration with the updated potential amount. We use $\omega_{\max}$ and $\sigma_{\max}$ to denote the upper bound for buyers' valuations ($\omega_{\max} \geq \max \sup_{A_m} \{|\frac{\partial \widehat{V}_m(A_m)}{\partial A_m}|\}$) and the upper bound for sellers' costs ($\sigma_{\max} \geq \max \sup_{A_n} \{|\frac{\partial \widehat{C}_n(A_n)}{\partial A_n}|\}$). Note that the valuation function $\widehat{V}_m(A_m)$ and cost function $\widehat{C}_n(A_n)$ are updated with the penalty functions in the smart contract. The potential amount is expected to achieve a Nash Equilibrium quickly with the gradients of marginal valuations and costs.

The optimal allocation A_m^* and A_n^* are updated in each iteration, and agent

[9]The potential amount is used for computing and updating the allocation buyers and sellers in each iteration).

derive their best responses. Given (r, C), the optimal allocations (for buyers/sellers) are

$$A_m^* = \min\{d_m, \{[C - \sum_{i \in B_m(b)} d_i], 0\}_{\max}\}$$
$$A_n^* = \min\{h_n, \{0, [C - \sum_{j \in \mathcal{S}_n(s)} h_j]\}_{\max}\} \tag{4.5}$$

where d_m and h_n are the updated amount for buyer m to purchase and for seller n to sell, respectively; $B_m(b) = \{i \in \mathcal{B} | \alpha_i > \alpha_m\} \cup \{\alpha_i = \alpha_n \text{ and } i < m\}$ and $S_n(s) = \{j \in \mathcal{S} | \beta_j > \beta_n\} \cup \{\beta_i = \beta_m \text{ and } j < n\}$.

The updated potential amount $\widetilde{C}(r, C)$ can be iteratively derived based on the given potential amount C.

3 Best Response. We use b_m^* and s_n^* to represent the best response of buyer $m \in B$ and seller $n \in S$. With the bid profiles $r = (b, s)$ and a pair of potential amounts $(C, \widehat{C})$, the best response can be derived as below:

$$b_m^*(r, C, \widehat{C}) = \arg\max\{f_m(b_m, b_{-m})\}$$

$$s_n^*(r, C, \widehat{C}) = \{f_n(s_n, s_{-n})\} \tag{4.6}$$

In the divisible double auction program $\mathsf{Prog_{da}}$, the best response will be computed in each iteration and finally converge to a Nash Equilibrium. Notice that, in different applications (e.g., different divisible resources), the valuation and cost functions would be different, as discussed in Section 4.5.2. In this dynamic auction game, all the agents recompute their best response to the current strategies (bid profiles) of other agents.

Double Auction Mechanism $\text{Prog}_{da}(\mathcal{B}, \mathcal{S}, r)$

Initialize: $\forall m \in \mathcal{B}, \forall n \in \mathcal{S}, r = (b, s)$

Require: $(\alpha_i)_{\max} \geq (\beta_j)_{\min}$ and $C < \min\{\sum_{i \in B} d_i, \sum_{j \in S} h_j\}$

1 : set iteration $k := 1$

2 : **while** *true* **do**

3 : $A_m^*(b, C) := \min\{d_m, \{[C - \sum_{i \in \mathcal{B}_m(b)} d_i], 0\}_{\max}\}$

4 : $A_n^*(s, C) := \min\{h_n, \{[C - \sum_{i \in \mathcal{S}_n(s)} h_j], 0\}_{\max}\}$

5 : $Q(r, C) := \min\{\sum_{i \in \mathcal{B}} A_i^*(r, C), \sum_{j \in \mathcal{S}} A_j^*(r, C)\}$

6 : $\widehat{\mathcal{P}} := \dfrac{p_b(r, C) - p_s(r, C)}{\omega_{\max} + \sigma_{\max}}$

7 : $\widetilde{C}(r, C) := Q(r, C) + \widehat{\mathcal{P}}$

8 : $b_m^* = \arg\max\{f_m(b_m, b_{-m})\},\ m \in \mathcal{B}$

9 : $s_n^* = \arg\max\{f_n(s_n, s_{-n})\},\ n \in \mathcal{S}$

10 : **repeat until** convergence

11 : set iteration $k := k + 1$

12 : **endwhile**

13 : **return** $b_m^*(\forall m \in \mathcal{B}), s_n^*(\forall n \in \mathcal{S})$

Figure 4.1. Double Auction Mechanism

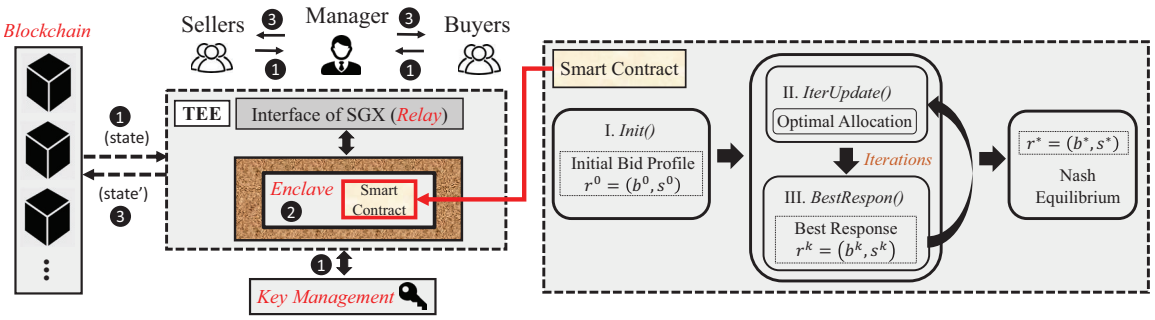


Figure 4.2. Hybridized TEE-Blockchain System

Theorem 5. *The divisible double auction (as program* $\mathsf{Prog}_{\mathsf{da}}$*) achieves individual rationality and incentive compatibility.*

Proof. First, suppose that the truthful bid profile provided by buyer $m \in B$, then we could obtain the non-negative payoff function $f_m(r) = \widehat{V}_m(A_m^*) - \rho(r_i, r_{-i})$. Correspondingly, given the truthful bid profile provided by seller $n \in S$, $f_n(r) = \rho(r_j, r_{-j}) - \widehat{C}_n(A_n^*)$, $\forall n \in \mathcal{S}$. Thus, the truthful bid profiles show that the non-negative payoffs are guaranteed for all the agents in the auction (individual rationality is proven).

Second, we define the A_m and A_n as allocation of buyer $m \in \mathcal{B}$ and seller $n \in \mathcal{S}$, separately. And A_m^k and A_n^k represent the allocation for k-iteration. We will verify the incentive compatibility for all buyers $m \in \mathcal{B}$ first for incentive compatibility. Suppose there is truthful bid profile $b_m = (\alpha_m, d_m^k)$ where $\alpha_m = \frac{\partial \widehat{V}_m(d_m^k)}{\partial d_m^k}$,which can make $f_m(b_m^k, r_{-m}) \geq f_m(b_m, r_{-m}), \forall m \in \mathcal{B}$, there are two cases involved: Case (A): assuming that $\alpha_m < \frac{\partial \widehat{V}_m(d_m)}{\partial d_m}$, if there is bid b_m^k, which makes $d_m^k = A_m \leq d_m$. Then, we could have $\alpha_m^k \geq \frac{\partial \widehat{V}_m(d_m)}{\partial d_m} > \alpha_m$, due to the diminishing marginal utility of the valuation function. Thus, we have $f_m(b_m^k, r_{-m}) \geq f_m(b_m, r_{-m}), \forall m \in \mathcal{B}$, since we have obtained the maximum social welfare; Case (B): suppose that $\alpha_m > \frac{\partial \widehat{V}_m(d_m)}{\partial d_m}$. If there is bid b_m^k, then we have $d_m^k = \frac{\partial \widehat{V}_m(d_m^k)}{\partial d_m^k} = d_m$, then we get $\alpha_m > \frac{\partial \widehat{V}_m(d_m)}{\partial d_m} = \alpha_m^k$. It is

known that $A_m^k \leq A_m$ for the maximum social welfare. If $A_m^k = A_m$, then we get $f_m(b_m^k, r_{-m}) = f_m(b_m, r_{-m}), \forall m \in \mathcal{B}$. If we have $A_m^k < A_m$, the below holds:

$$\begin{aligned} & f_m(b_m, r_{-m}) - f_m(b_m^k, r_{-m}) \\ = \quad & \widehat{V}_m(A_m) - \widehat{V}_m(A_m^k) + \rho(A_m^k, r_{-m}) - \rho(A_m, r_{-m}) \\ \leq \quad & \alpha_m^k(A_m - A_m^k) + F(r) - \alpha_m A_m - F(r^k) \quad + \alpha_m^k A_m^k \\ \leq \quad & \alpha_m^k(A_m - A_m^k) - \alpha_m^k(A_m - A_m^k) = 0 \end{aligned} \tag{4.7}$$

Thus, we could have $f_m(b_m^k, r_{-m}) \geq f_m(b_m, r_{-m})$, $\forall m \in \mathcal{B}$ with Case (A) and (B). Also, incentive compatibility can be proven for all the sellers $\forall n \in \mathcal{S}$ in a similar way. □

4.4 Hybridized System Design

4.4.1 Threat Model and Properties. To ensure data privacy and integrity for the auction computation, we use the TEE's attestation [91], where the computation is executed correctly inside the *enclave* trusted by all the agents. However, the remaining software stack outside the enclave and the hardware are not trusted. The adversary may corrupt any number of agents, assuming that honest agents will trust their own codes and platform (leakage resulted from its software bugs are out of the scope). Furthermore, we assume that all the agents do not trust each other in the auction while being potentially malicious, such as stealing the bid profiles information. During the execution, each agent may send, drop, modify and record arbitrary transactions. Note that the side-channel attacks against *enclave* and DoS attacks are not considered in this paper.

In our proposed hybridized TEE-Blockchain system, the TEE compensates for the privacy issue with respect to the smart contract, i.e., our system can address the privacy issue for the double auction by utilizing the TEE for isolating the contract

(auction process) execution inside the *enclave*, shielding it from potential malicious agents. From the system aspect, the following properties are addressed:

- **Correctness**. The correctness of computation in the TEE can be guaranteed and verified by the remote attestation based on the given state and inputs.
- **Privacy** and **Security**. Our system protect and verify the sensitive inputs (e.g., bid profiles) and outputs of all the agents.

4.4.2 Architecture. Figure 5.1 illustrates the main architecture of our hybridized system. There are five main components in hybridized system:

- *All the agents* $\mathcal{P}$ (buyers and sellers) are the end users of the smart contract. Th manager $\mathcal{P}_M$ is the delegation to compute all the incoming private agents' input and deliver results as the administrator. $\mathcal{P}_M$ further leverages *Relay* to trigger the *enclave* to be initialized for computation (will be explained as the following). Note that the manager $\mathcal{P}_M$ is considered to be malicious, which may collude with other agents or interrupt the computation.
- TEE ($\mathcal{T}$) is responsible to run the smart contact to processes the double auction computation among the agents (requested by the manager $\mathcal{P}_M$) in the *enclave* $\mathcal{E}$, which protects the privacy and integrity of computations. It also generates remote attestations (computation correctness) for state updates. To further improve the functionality and security of our system, we design the only interface component *Relay* $\mathcal{R}$ to provide indirect access to *enclave*. *Relay* can also provide the message passing with the *Blockchain*.
- *Blockchain* ($\mathcal{BC}$) maintains a distributed append-only ledger via running a consensus protocol. The state of $\mathcal{BC}$ and attestations are stored on the chain.

SGX Functionality Model: $\mathcal{F}_{SGX}[\mathsf{Prog_x}, \mathcal{R}]$

Initialize:

1 : Upon receiving (Init) from $\mathcal{R}$:

2 : Set outp := $\mathsf{Prog_x}$.**initalize()**

3 : $\psi_{att} := \sum_{sgx} \cdot \mathsf{Sig}(\mathsf{sk}_{sgx}, (\mathsf{Prog_x}, \mathsf{outp}))$

4 : Output $(\mathsf{outp}, \psi_{att})$

Resume:

5 : Upon receiving $\mathsf{Authen}(\mathsf{tx})$ from $\mathcal{R}$:

6 : Set outp := $\mathsf{Prog_x}$.**resume**()

7 : Output outp

Figure 4.3. Enclave Functionality Model

Moreover, the validity of state update are checked by the blockchain with the TEE attestations.

- *Key Management* ($\mathcal{KM}$) generates keys for both private agents' inputs and state encryption. All the agents and TEE can directly interact with the $\mathcal{KM}$ for the key pairs via a key distribution protocol.

4.4.3 Enclave Functionality Model. *Enclave* ($\mathcal{E}$) protects the code of program and data during the computation for the auction. Specifically, the program running inside the *enclave* is completely isolated from an adversarial OS as well as other processes on the host. We formalize and integrate the Intel SGX [92] as TEE in our hybridized sytem.

In order to model the ideal functionality channel with some proprieties such as privacy and authenticity, we utilize a global universal composability (UC) framework functionality [93] to instantiate the SGX Functions. More formally, we de-

note the program X which runs inside the SGX enclave as Prog_x, which can be Prog_{da} (Figure 4.1) for double auction. The SGX function can be expressed as $\mathcal{F}_{SGX}(\sum_{sgx})[\mathsf{Prog}_{\mathsf{x}}, \mathcal{R}]$, where $\sum_{sgx}$ is a group signature scheme and $\mathcal{R}$ is *Relay*. As shown in Figure 4.3, the program $\mathsf{Prog}_{\mathsf{x}}$ is loaded into *enclave* via the "init" call from *Relay*. When *Relay* calls "resume", the program is executed based on the incoming requests or inputs, denoted as inp, and computes the output with an attestation $\psi_{att} := \sum_{sgx} \cdot \mathsf{Sig}(\mathsf{sk}_{sgx}, (\mathsf{Prog}_{\mathsf{x}}, \mathsf{outp}))$. The signature under TEE hardware key sk_{sgx} and pk_{sgx} could be obtained from the SGX Functions ($\mathcal{F}_{SGX}$).

4.4.4 Workflow. Figure 4.4 depicts that the designed system is executed with three phases: (1) Initialization, (2) ExecProg, and (3) Finalization. We denote the input and output for the TEE as inp and outp, respectively. Also, regarding the deposit, we use $\widetilde{\xi_{b_m}}$, $\widetilde{\xi_{s_n}}$ and $\widetilde{\xi_{\mathcal{P}_M}}$ for all buyers, sellers and managers.

1 Initialization. Prior to the auction phase, all the agents (buyers and sellers) are supposed to prepare for their deposits $\widetilde{\xi_{b_m}}$, $\widetilde{\xi_{s_n}}$. Besides, the manager also needs to deposit $\widetilde{\xi_{\mathcal{P}_M}}$ (if the manager or any agent is identified to deviate the computation, then the deposit will be charged as penalty). Then the TEE will set state $:=$ *init* as confirming that the deposits in the blockchain. Otherwise, the TEE will set state $:=$ *abort* for preparing next auction and refund the deposit to the agents. For the auction computation, the TEE will fetch the key pair $(\mathsf{pk}_{\mathsf{sgx}}, \mathsf{sk}_{\mathsf{sgx}})$ from *Key Management* for attestation, where the key $(\mathsf{pk}_{\mathsf{sgx}})$ is bundled to the executing prog_x instance (auction) for checking the correctness of computation. Besides, the attestation with current state $[\mathsf{state}, \psi_{att}]$ are posted on the blockchain $\mathcal{BC}$ (as described in Section 4.4.3).

Next, to tackle the large inputs of agents, the manager $\mathcal{P}_M$ will handle the transaction $\mathsf{tx} := [\mathsf{Enc}_{\mathsf{pk}}(\mathsf{inp}), l_{id}, \widetilde{\xi_{b_m}}, \widetilde{\xi_{s_n}}]$ from all the agents where inp denotes the inputs of all the agents, and l_{id} represents a unique identifier (ID). Then, $\mathcal{P}_M$ will send tx to the *Relay* for executing the auction computation. Note that all the agents

send the transactions through secure channels. The tx is a transaction to deliver the input and output data among different system components.

2 ExecProg. To execute the auction requested from $\mathcal{P}_M$, the *Relay* will retrieve the state information from the *blockchain* and *Relay* will trigger TEE to execute the requested service (auction) with the "resume" call if the state can be verified. Then TEE first decrypts the input data (from the *Manager*) with the private key *sk* obtained from the *Key Management* and launch the auction smart contract code (Figure 4.1) as $\mathsf{Prog}_{\mathsf{x}}$ in the *enclave* (a sandboxed environment). Thus, a adversary cannot interrupt the execution or monitor data inside the *enclave* considering the natural merit of *enclave*. The final results output of the program (auction smart contract) $\mathsf{Prog}_{\mathsf{x}}$ will be securely returned to the manager.

3 Finalization. Once *manager* receives the final result outp from TEE and check the correctness with the *Blockchain*. If the result outp is accepted by the *Blockchain* via checking the new state state′, the auction result $(\mathsf{outp}, \psi_{sgx}, l_{id}, \widetilde{\xi_{b_m}}, \widetilde{\xi_{s_n}})$ will be delivered to all the agents via *Manager* and *Blockchain* will store the new state state′.

4.5 Discussions

4.5.1 Security. Based on the key feature of isolation in *enclave*, Intel SGX enables the program (data) to be executed inside the secure container (*enclave*) for confidentiality and integrity. The adversary cannot interrupt the computation executed in a sandboxed environment (*enclave*). Note that *enclave* is created in its virtual address space by an untrusted hosting application with OS support. Once *enclave* starts initialization, data and codes inside it will be isolated from the rest of the system. Note that the encrypted data are sent from agents to *enclave* through secure channels. However, other malicious servers cannot eavesdrop on the encrypted data and even

Execution Procedure ($\mathcal{P}$, $\mathcal{BC}$, $\mathcal{T}$)

InitRequest() :

1 : deposit $\widetilde{\xi_{b_m}}, \widetilde{\xi_{s_n}}$ and $\widetilde{\xi_{\mathcal{P}_M}}$

2 : invoke procedure *initEnclave*() via *request*

3 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\!\$\ \mathsf{KGen}(1^n)$

4 : publish pk to all agents for $\mathsf{Enc}_{\mathsf{pk}}(\mathsf{inp})$

InitEnclave() :

5 : receive($\mathsf{init}(), request$) to load $\mathsf{Prog}_{\mathsf{x}}$ inside E

6 : boost *enclave* with Prog_x.**initialize()**

7 : distribute $(\mathsf{pk}, \psi_{sgx})$ for attestation

InitAgent() :

8 : $\mathsf{tx} := (\mathsf{Enc}_{\mathsf{pk}}(\mathsf{inp}), l_{id}, \widetilde{\xi_{b_m}}, \widetilde{\xi_{s_n}})$ are sent to $\mathcal{P}_M$

9 : **if** receive $\mathsf{Authen}(\mathsf{tx})$ **then**

10 : $\mathcal{BC}.post(\mathsf{state}, \psi_{att}, l_{id})$

11 : **else**

12 : set $\mathsf{state} := \mathsf{fail}$*and* refund $\widetilde{\xi_{b_m}}, \widetilde{\xi_{s_n}}$ and $\widetilde{\xi_{\mathcal{P}_M}}$

ExecProg() :

13 : boost *enclave* with **resume()** to load and run Prog_x

14 : retrieve and decrypt the previous state from $\mathcal{BC}$

15 : decrypt encrypted inputs($\mathsf{Enc}_{\mathsf{pk}}(\mathsf{inp}), l_{id}$)

16 : load and compute Prog_x

17 : $\mathsf{outp} := \mathbf{TEE}(\mathsf{Prog}_{\mathsf{x}})$

Finalization() :

18 : Vf the correctness of outp with ψ_{att}

19 : $\psi_{sgx} := \sum_{sgx} \cdot \mathsf{Sig}(\mathsf{sk}_{sgx}, (\mathsf{Prog}_{\mathsf{x}}, \mathsf{outp}))$

20 : $\mathcal{BC}.post(\mathsf{state}', \psi_{att}, l_{id})$

Figure 4.4. Hybridized System Procedure

tamper with the communication.

During the execution, if any agents abort/skip this step or behave dishonestly during the initialization, the execution will be terminated and refunds to the honest agents within the time threshold T_1. Afterwards the computation starts, all agents send the encrypted inputs to the interface of SGX. In this phase, if no malicious behaviors are detected by the manager, the Relay $\mathcal{R}$ will forward the encrypted inputs to the *enclave* $\mathcal{E}$. However, it is hard to determine if the agents behave dishonest (i.e., fail to send message) or the Relay behave malicious (i.e, dropping message) during the execution if the *enclave* $\mathcal{E}$ does not receive any incoming requests. Thus, all agents $\mathcal{P}$ and Relay $\mathcal{R}$ both receive the challenge request (we denote as $\mathsf{request}_{chal}$). Within the certain time threshold T_2, if agents response with inputs and procedure will move to the next steps. Otherwise, the agents are proved to be malicious. Similarly, if the Relay $\mathcal{R}$ is proven to be the malicious one, the protocol is terminate and set up state is fail. In terms of the last phase Finalization, the TEE will return the final results to all the agents and publish the states on the blockchain. Note that during all the data flow, the deposits of malicious agents are not refunded for punishment.

4.5.2 Applications. In practice, divisible resources which could be privately traded using our system, e.g., electricity [94], cloud resources [95, 96], and wireless spectrum [97]. We now discuss two of them as representative applications. Note that different valuation/cost functions will be defined and implemented in different applications.

Energy Trading. There is demand from power grid for trading the excessive locally generated energy, e.g., the renewable energy resources [98, 99]. The proposed hybridized system is able to implement the privacy preserving divisible double auction for energy trading, due to the divisible of electricity resource. The valuation/cost functions are defined as $\widehat{V}_m(x_m) = \zeta_m \log(x_m + 1)$ and $\widehat{C}_n(y_n) = a_n y_n^2 + b_n y_n$ [100],

where ζ_m is a parameter leveraged by the behavior preference of buyer. The parameter of a_n and b_n are used for leveraging how much the sellers incline to sell. The valuation/cost functions follow the general assumption illustrated in Section 4.3.1. Eventually, hybridized system only generate the *clearing price* for the auction to all the agents. The energy amount of each pair of buyer and seller will only obtain the amount traded between them.

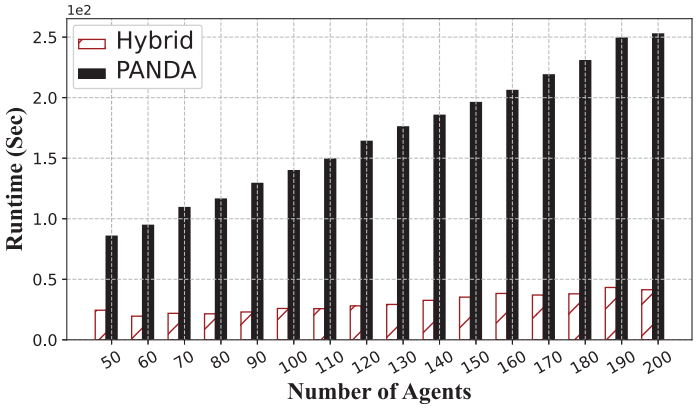
Wireless Bandwidth Allocation. We can model the wireless bandwidth allocation [78,101] based on our proposed hybridized system for network traffic and services. In terms of a MVNO (Mobile Virtual Network Operator), the valuation function for buyer m is defined as $\widehat{V}_m(x_m) = \zeta_m \ln(x_m)$ where ζ_m defined as a positive-valued parameter. This indicates that buyers willing to pay for the bandwidth. Meanwhile, the cost function of seller n, an InP (Infrastructure Provider) is denoted as $\widehat{C}_n(y_n) = \alpha_n e^{y_n}$, where y_n presents the bandwidth it can supply and α_n as another positive-valued parameter (bandwidth) for the seller n. As expected, the valuation/-cost functions are also follow the general assumptions in Section 4.3.1, and execute privately and truthfully via hybridized system for such divisible double auction.

4.6 Experimental Evaluations

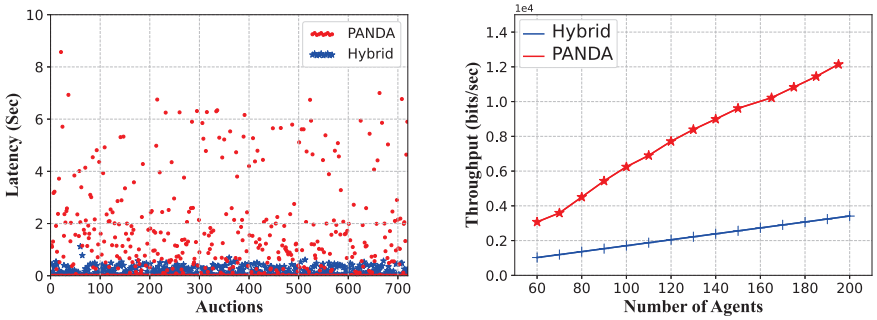
4.6.1 Experimental Setup. In this section, the system performance of both *off-chain* and *on-chain* procedures in the hybridized system will be evaluated. To support the smart contacts execution within the *enclaves*, we use Graphene[10] on the Microsoft Azure.[11] A *manifest* is adopted to support the *enclave* initialization, and it protects the smart contract execution in the host process. For the *on-chain* implementation,

[10]Graphene [102] is a lightweight guest OS, which replaces the Intel SDK for the *enclave* and host process.

[11]https://azure.microsoft.com/en-us/solutions/confidential-compute/



(a) Runtime vs. Number of Agents



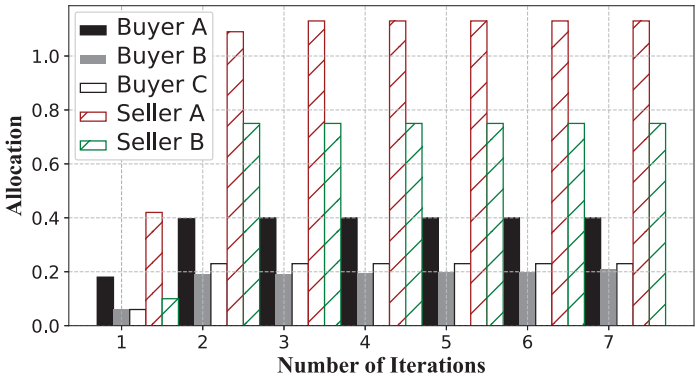
(b) Frequency vs. Latency (c) Throughput vs. Number of Agents

Figure 4.5. *Off-chain* System Performance Evaluation (1024-bit key)

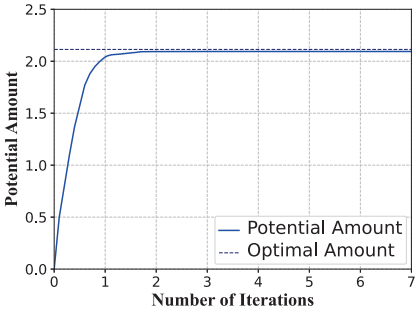
we use the Hyperledger Fabric[12], which is designed for distributed ledger technologies with multiple modules for the blockchain platforms. As a distributed ledger platform, it includes a highly modular and configurable architecture, which supports the smart contract execution. Note that the Hyperledger Fabric is deployed on the VM with Ubuntu 18.04 on the Microsoft Azure for the *on-chain* procedures.

Applications. We conduct experimental evaluations for two real-world applications: (1) energy trading/auction [98], and (2) wireless bandwidth allocation [78,101] among up to 200 agents. Each agent can be either a buyer or seller in the auction for both applications.

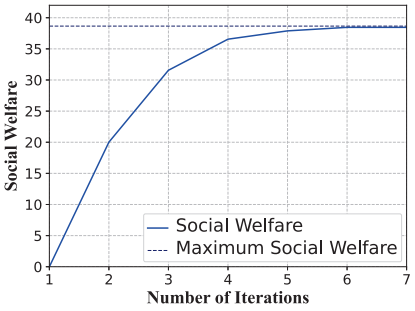
[12]https://hyperledger-fabric.readthedocs.io/en/latest/index.html



(a) Allocation vs. Iterations



(b) Potential Amount vs. Iterations

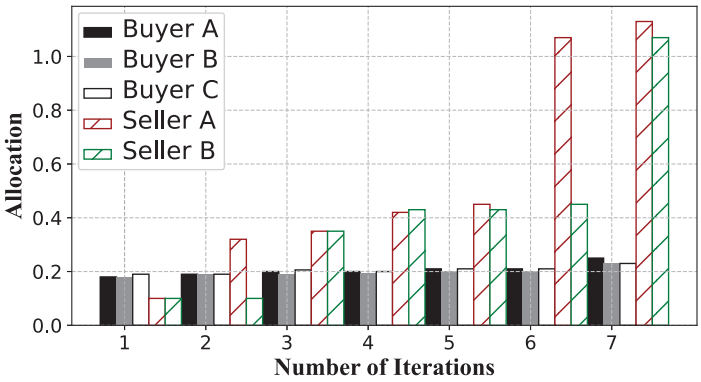


(c) Social Welfare vs. Iterations

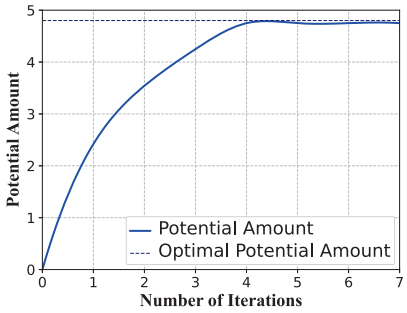
Figure 4.6. Energy Trading: *Off-chain* Double Auction Computation

In the experiments of energy trading/auction, we utilize the valuation function $\widehat{V}_m(x_m) = \zeta_m \log(x_m+1)$ and cost function $\widehat{C}_n(y_n)=a_n y_n^2+b_n y_n$, as detailed in [11]. We adopt the same parameters $\zeta_m = 50$, $a_n = 30$ and $b_n = 0$ as [11]. Similarly, wireless bandwidth allocation is implemented with the valuation function $\widehat{V}_m(x_m) = \zeta_m \ln(x_m)$ and cost function $\widehat{C}_n(y_n) = \alpha_n e^{y_n}$, where $\zeta_m = 50$ and $a_n = 2$ [78]. For the energy trading, real datasets from the UMASS Trace Repository [103] are adopted while synthetic datasets are generated per [78, 101] for the wireless bandwidth allocation.

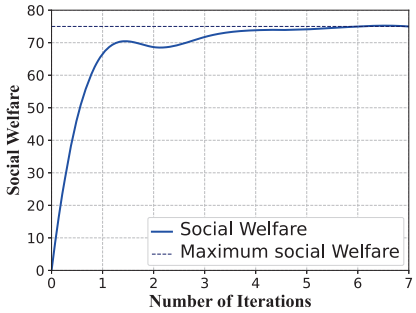
4.6.2 Off-Chain Performance Evaluation. We first evaluate system performance for securely perform computation for the auction (the off-chain computation for the optimal allocation). The system performances for the above two applications are quite similar, thus we only give the results for one application (w.l.o.g., the energy



(a) Allocation vs. Iterations



(b) Potential Amount vs. Iterations



(c) Social Welfare vs. Iterations

Figure 4.7. Wireless Bandwidth Allocation: *Off-chain* Double Auction Computation

trading). Figure 4.5(a) presents the total runtime of the off-chain computation while varying the number of agents (from 50 to 200) for the auction while setting the key size as 1024-bit. It shows that the runtime increases as the number of agents increases. Compared with the cryptographic protocol based double auction system [71] ("PANDA"), the runtime of our hybridized TEE-blockchain system ("Hybrid") has been significantly reduced. With a strong security guarantee, it takes only up to 5 minutes for 200 agents to perform the computation. In addition, Figure 4.5(b) presents the latency of 720 different auctions. The latency of our hybridized system is less than 1 second for most auctions, which is also significantly lower than the cryptographic protocols (PANDA). Finally, Figure 4.5(c) illustrates the throughput (bits/sec) of the system on a varying number of agents (1024-bit key). The through-

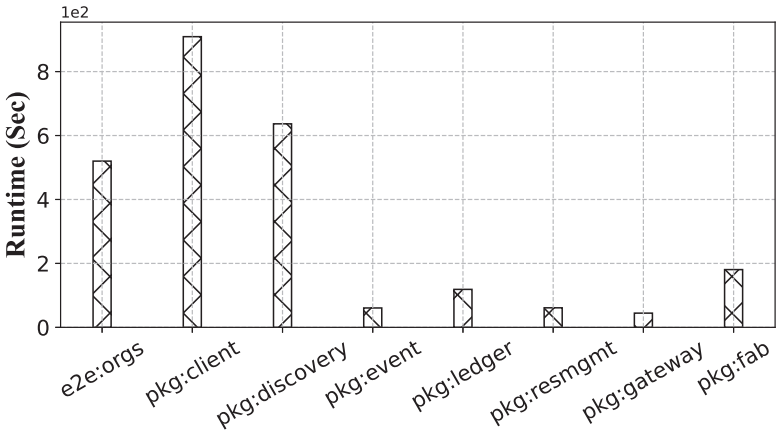
put of "Hybrid" increases slower than "PANDA" as the number of agents increases.

Furthermore, we conduct empirical studies for the two applications (energy trading and wireless bandwidth allocation), respectively, in case of 20 agents, including 12 buyers and 8 sellers. Figure 4.6 and 4.7 demonstrate the detailed results on (1) allocation, (2) potential amount, and (3) social welfare, for the two different applications.

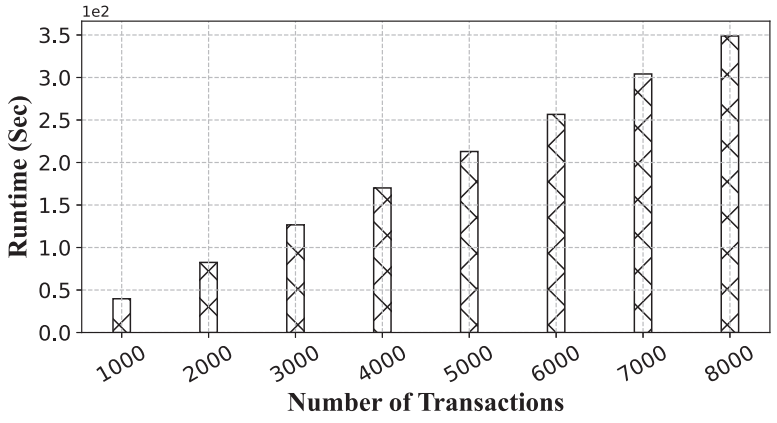
First, Figure 4.6(a) and 4.7(a) show the allocation for five randomly picked agents (three buyers and two sellers) in different iterations. The allocation of both buyers and sellers increase and finally achieve the optimally allocated amount after several iterations. In Figure 4.6(b) and 4.7(b), the potential amount of the auction (used for updating the allocation for buyers and sellers in each iteration) grows until convergence while moving to new iterations. Finally, the social welfare ($F(\cdot)$) is derived based on equation $F(\cdot) = \sum_{m\in\mathcal{B}} \widehat{V}_m(A_m) - \sum_{n\in\mathcal{S}} \widehat{C}_n(A_n)$. Figure 4.6(c) presents an increasing trend in multiple iterations and the social welfare of energy trading converges to the maximum value \$38 while the social welfare of the wireless bandwidth allocation in Figure 4.7(c) converges to the maximum value \$75.

4.6.3 On-Chain Performance Evaluation. Besides the off-chain computation, we demonstrate the performance of on-chain transactions. Figure 4.8(a) shows the runtime for different package functionalities. The functionality of the `pkg:client` takes 909.22 seconds (most of the on-chain runtime), the processes of functionality include preparing/creating channel and client context, and communicating with the Fabric network via the channel. Compare with the `e2e:orgs`, the `pkg:discovery` takes relatively longer time (636.58 seconds). It is implemented on the `DiscoveryFilterService` package and the discovery service with filter is returned. Also, `pkg:ledger` takes 118.77 seconds while `pkg:fab` takes around 180.67 seconds. Furthermore, `pkg:event` (60.63 seconds) works for users to receive

events such as block, filtered block, contracts, and transaction status events. The `pkg: resmgmt` (61.11 seconds) enables the creation and update of resources on a Fabric network, and it also allows the administrators to create/update channels, query peer for channels, and perform some operations, i.e., installing, instantiating and upgrading the smart contracts. Finally, `pkg: gateway` (44.49 seconds) enables users to update the application based on Hyperledger Fabric programming model. Finally, Figure 4.8(b) presents the on-chain runtime on a varying number of transactions. The runtime is only up to 350 seconds in case of 8000 transactions.



(a) Runtime vs. Functionality



(b) Runtime vs. Transaction

Figure 4.8. *On-Chain* Performance Evaluation

4.7 Related Work

There were some other auction mechanisms for allocating divisible resources, i.e., spectrum allocation [104, 105]. Combinatorial auctions [105] was discussed for cognitive radio networks. Strategy-proof mechanism for multi-radio spectrum buyers was proposed by Wu and Vaidya [104]. A sealed-bid reserve auction was modeled for the radio spectrum allocation problem. Hoefer et al. [106] investigated the combinatorial auctions with a conflict graph via an approximation algorithm (LP formulation). Other studies related to divisible resources auctions focused on the revenue maximization [107] or the efficiency of social maximization [105, 108].

The privacy concerns in auction mechanism for divisible resources have been raised in [109, 110]. In [111]cryptographic techniques were proposed for achieving the privacy and security in the auction game. A cryptographic scheme for one-side auctions was proposed in Huang et al. [112]. In addition, Ekiden [113] presented the complementary characters for blockchain and TEE, the rigorous security proofs are provided to support the confidentiality of the hybrid system. Also, the Hawk system [114] was designed as a decentralized smart contract framework for running the contracts off-chain while posting zero-knowledge proofs on-chain. Zhang et al. [115] proposed a system Town Crier that authenticates data feed using smart contracts supported by the Ethereum platform. It enables data fetching from existing HTTP-enabled data sources, and utilizes TEE to execute its core functionality and protect its data against malicious attackers.

In the context of double auction, a recent scheme was proposed to protect privacy for the bids [71]. However, it requires heavy computation burden by composing the cryptographic primitives. Instead, the proposed hybridized system can efficiently perform secure computation for the double auction.

4.8 Summary

In this chapter, we design a hybridized TEE-Blockchain system to securely execute divisible double auction among distributed agents within the *enclave* in a highly efficient way. Meanwhile, it interacts with the blockchain for validation and storage. The proposed divisible double auction mechanism guarantees *individual rationality*, *incentive compatibility*, *weak budget balance* and *pareto efficiency*. The input private data of all the agents in the divisible double auction can also be protected in the hybridized system. The experimental results have demonstrated both effectiveness and efficiency for the designed hybridized system to privately compute the optimal allocation and execute the divisible double auction.

CHAPTER 5

CRYPTOGRAPHIC INFERENCES FOR VIDEO DEEP NEURAL NETWORKS

Deep neural network (DNN) have been widely deployed for efficient and accurate learning in many different domains. For instance, a client may send its private input data (e.g., images, text messages and videos) to the cloud, which provides the inferences (e.g., classification and prediction) with the pre-trained DNN models. However, significant privacy concerns would emerge in such use cases due to data or model sharing with the cloud. *Secure inferences* with cryptographic techniques have been proposed to address such issues, and the system can perform *secure two-party inferences* between each client and cloud. However, most of the existing cryptographic systems only focus on DNNs for extracting 2D features for image inferences, which have major limitations on latency and scalability for extracting spatio-temporal (3D) features from videos for accurate inferences.To address such critical deficiencies on cryptographic DNN for video inferences, we design and implement the first cryptographic inference system, `Crypto3D`, which privately infers videos on 3D features with rigorous privacy guarantees.

The partial work in this chapter have been published at 2022 *ACM CCS* Poster session [13].

5.1 Overview

In the past decade, deep neural networks (DNNs) have been increasingly deployed in practical applications for object detection, image and action classification, anomaly detection, etc. In a client-server model for DNNs (e.g., deep learning as a service), the client may send its data to the cloud service provider, which provides

[13] ©2022, ACM, with permission from Bingyu Liu, Rujia Wang, Zhongjie Ba, Shanglin Zhou, Caiwen Ding and Yuan Hong, *Poster: Cryptographic Inferences for Video Deep Neural Networks*

the inference services (e.g., classification and prediction) using its pre-trained DNN models. However, severe privacy concerns may occur between the client and cloud. In the domain of video inferences, the users' videos involve considerable amounts of sensitive information (e.g., human face, identities, activities, and workspace). Directly disclosing them to the cloud would compromise the privacy of users. Indeed, the pre-trained DNN model should also be considered as the proprietary information for the cloud, which cannot be revealed to clients.

To eliminate such privacy risks, cryptographic protocols [21, 116, 117] are designed for *secure inferences* A secure inference protocol allows the client to send its private input data (encrypted), and privately obtain the learning result from the cloud. Neither party can learn anything regarding the model weights and private inputs from each other. Many existing works [116] use one or more cryptographic techniques such as Fully Homomorphic Encryption (FHE) [117], Garbled Circuits (GC) [21] and Secret Sharing (SS) to compose the protocols. FHE can provide higher privacy guarantees, but it brings expensive computational overheads. Moreover, some non-polynomial functionalities (e.g., Non-linear Activation Functions ReLU) cannot be supported. Garbled circuits support arbitrary functionality, but it results in significant computation and communication overheads for multiplication. In addition, Trusted Execution Environment (TEE) [118–120] provides secure *enclave* for the isolated sensitive computation with attestation. It ensures data privacy and integrity without provable guarantees, and it is also vulnerable to side-channels attacks [121]. Moreover, current TEEs are not scalable enough for processing large amount of data. Thus, directly using such protocols or systems are not ideal for secure DNN inferences.

The Delphi system [122] was recently proposed as one of the state-of-the-art efficient cryptographic inference systems. It outperforms other protocols in both latency and communication cost for image DNN with a hybrid cryptographic protocol.

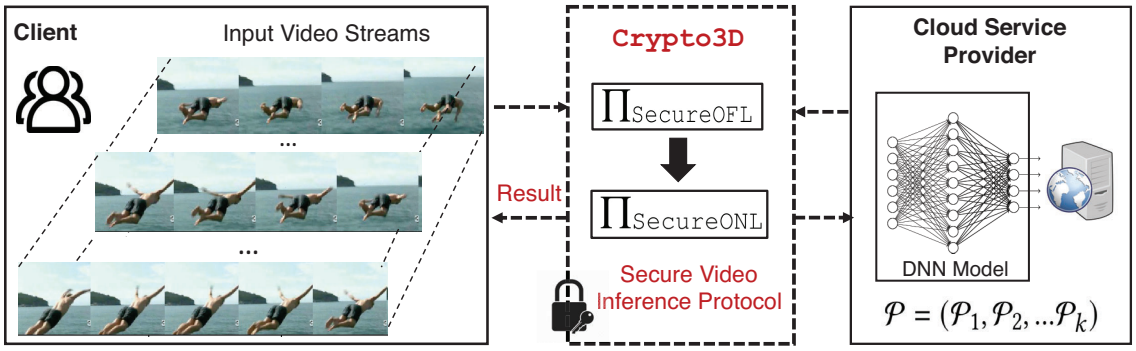


Figure 5.1. *Secure two-party inference* for private video classification between client $\mathcal{C}$ and cloud service provider $\mathcal{S}$.

It builds the framework based on the Gazelle [123] system, which contains the sub-protocols for linear and non-linear layers separately, and minimizes the computational and communication costs with offline. Unfortunately, *securely inferring images based on 2D features* by Delphi (the state-of-the-art) is far from enough for video-based applications in the real world. Compared with the 2D ConvNets, most 3D ConvNets have to infuse the temporal information of the videos after each convolution/pooling operations. Performing 3D convolution and pooling operations are supposed to deliver temporal information across all the neural network layers to the end. Integrated with both spatial and temporal information in each feature, 3D ConvNets (e.g., C3D and I3D networks) have proven to be more accurate on video inferences than 2D ConvNets [124, 125]. However, to our best knowledge, cryptographic inferences based on 3D features for video DNNs have not been studied yet in literature.[14]

To fill this gap, we design and implement the first cryptographic inference system (namely "`Crypto3D`") that privately infers videos based on 3D spatial-temporal

[14]Visor [121] provides confidentiality for analyzing video streams via a hybrid secure hardware-based Trusted Execution Environment (TEE) system. However, it still privately infers data (e.g., object detection and tracking) based on 2D features. PPVC [126] presents the private preserving on the video classification based on Secure Multiparty Computation, but it still utilizes the 2D ConvNets without fully preserving temporal information.

features (both C3D [124] and I3D [125]). It enables the client and cloud to privately perform the cryptographic inference for video classification, video action recognition and prediction, as well as visual anomaly detection. Also, we further boost the system efficiency with optimized matrix operations and ciphertext packing technique.

Specifically, as shown in Figure 5.1, in `Crypto3D`, the cryptographic network inference $\prod_{\texttt{SecureINF}}$ is first constructed by a offline protocol $\prod_{\texttt{SecureOFL}}$ and an online protocol $\prod_{\texttt{SecureOFL}}$ with C3D or I3D model. It enables the client to send private input video to the cloud, and then privately obtain the learning results from the cloud.

Contributions. Therefore, the major contributions of this paper are summarized as below:

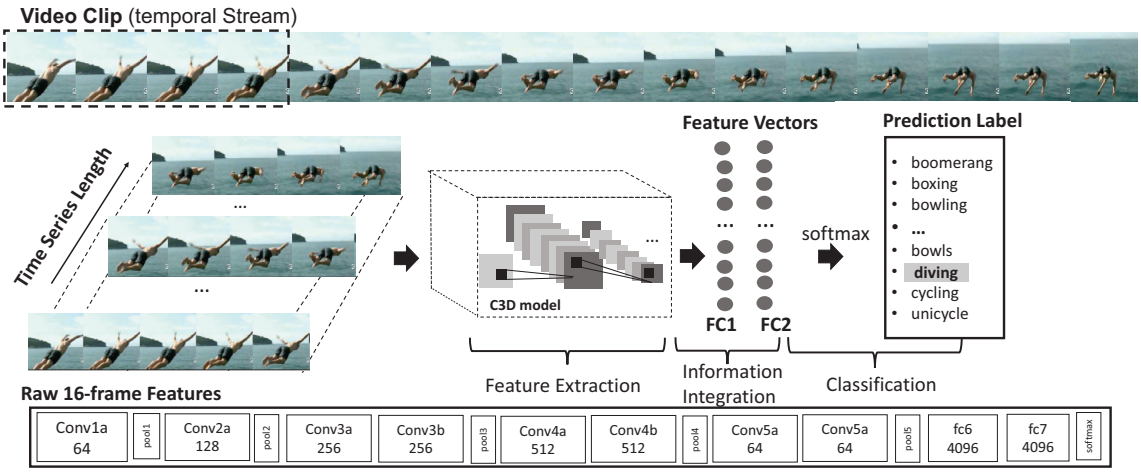


Figure 5.2. The flow diagram of the video inference based on C3D neural network. The upper bar represents video streams, and then it is uniformly divided into 16 frames as a clip for the feature extraction via C3D model.

- To our best knowledge, we design and implement the first cryptographic inference system for privately inferring videos with spatio-temporal (3D) features.
- We propose the mixed cryptographic system by co-designing and composing cryptographic primitives based on homomorphic encryption (HE), garbled circuit (GC), and secret sharing (SS). In addition, we optimize the matrix multi-

plication with HE and integrate ciphertext packing to further boost the overall system efficiency.

- We theoretical prove the security for `Crypto3D`, and conduct extensive experiments to evaluate the performance of `Crypto3D` for C3D and I3D inferences on UCF-101 and HMDB-51 datasets while benchmarking with the extended cryptographic inference systems.

- `Crypto3D` significantly outperforms existing systems. [15]

5.2 Background

5.2.1 Video Inferences as a Service. Nowadays, video application services are prevalent in all aspects of daily-lives with the influence of the internet. Specifically, there is an increasing interest in video analyses for human actions, such as action recognition, localization, and human behavior prediction. For example, the home monitoring security system aims at keeping the property and users' safety via video monitoring. The video streams are recorded and transmitted to the cloud server for further analysis. The user receives notifications once the system detects any anomalous behaviors. However, the privacy of the video contents, such as the home location, users' faces, activities and daily life might be exposed. Therefore, the video streams need to be confidential and not be revealed to any other parties. Such use cases motivate us to design a secure video analytics framework. As shown in the Figure 5.1, we propose a *secure two-party inference* framework for 3D video input

[15]Existing cryptographic inference systems with 2D features (e.g., Delphi, Gazelle, and HEANN3D) cannot be directly applied to privately infer videos with 3D features. We significantly modified most of the state-of-the-art methods (CryptoDL, HEANN, MP-SPDZ, E2DM, Intel SGX, and Gazelle) to privately infer videos with 3D features (C3D and I3D) as benchmarks. Note that Delphi cannot be extended to 3D inferences due to its end-to-end 2D architecture, and source codes for GALA are not available.

streams. Our model details can be found in Section 5.3.

Deep learning works as a powerful tool for modeling and predicting input data by learning the complex features with artificial neural networks. Deep neural networks (DNNs) consist of a number of layers. Each layer computes the input data and sends the output to the next layer as input. While there are many different types of DNNs, they share the similar model structure. In this paper, we adopt the C3D [124] model structure for 3D video input inference, as shown in the Figure 5.2. We also extend the cryptographic inference system based on the I3D model [125] for further performance evaluations.

Video Classification Models. The purpose of video classification is that a video $\mathcal{V}$ containing N frames $(F_0, F_1, ..., F_{N-1})$ needs to identify the classes to which the video belongs. Each of the m classes is denoted as $p(y_i|\mathcal{V})$. The following equation defines the probability of neural network prediction function:

$$P(y_i|\mathcal{V}) = f(F_0, F_1, ..., F_N - 1) \tag{5.1}$$

C3D Neural Network. Figure 5.2 shows a deep 3D CNN with a homogeneous architecture. It consists of $3 \times 3 \times 3$ convolutional kernels followed by $2 \times 2 \times 2$ pooling at each layer. The C3D model is trained on a large scale video dataset such as UCF101 [127] and Sports 1M [128]. With respect to the generic feature extraction, the 3D convolutions are able to extract both spatial and temporal components information in the videos, e.g., the motion of objects, human action and human-object interactions. It directly encodes the temporal structure with 3D convolutional network instead of 2D. The involved 3D kernel is able to extract information from both spatial and temporal dimensions and fuse them into the same feature [124]. Compared with 2D ConvNet, 3D ConvNet provides a better modeled temporal information with 3D

convolution and 3D pooling operations for more accurate video recognition.

5.2.2 Cryptographic Primitives. The mixed cryptographic protocol used for privacy-preserving DNN composes cryptographic primitives such as Homomorphic Encryption (HE), Garbled Circuit (GC), Secret Sharing (SS), etc. Before introducing the cryptographic primitives, we first define $\mathbb{R}$ as a finite ring. We denote a group of schemes for the HE = $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ as the linearly homomorphic encryption (LHE) over the plaintext space $\mathbb{R}$. We define the DNNs model owned by server privately as $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_{|l|})$ with l layers. Vector $\mathbf{x} \in \mathbb{R}^n$ represents the input from the client.

Homomorphic Encryption. A homomorphic encryption of x enables the computing encryption of $f(x)$ without any knowledge of the decryption key. A *Linearly homomorphic public-key encryption* [25,129] includes a set of probabilistic polynomial-time algorithms $\prod_{\mathrm{HE}} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$:

- HE.$\mathsf{KGen} \rightarrow (\mathsf{pk}, \mathsf{sk})$, where the pair of keys (pk, sk) is generated by the randomized algorithm.
- HE.$\mathsf{Enc}(\mathsf{pk}, m) \rightarrow c$. The ciphertext c is generated by the encryption algorithm HE.Enc with the pk and message m, where the message space is a finite ring $\mathcal{R}$.
- HE.$\mathsf{Dec}(\mathsf{sk}, c) \rightarrow m$. The message m can be obtained by the decryption algorithm HE.Dec with the sk.
- HE.$\mathsf{Eval}(\mathsf{pk}, c_1, c_2, \mathcal{L}) \rightarrow c_l$. The new ciphertext c_l is generated by the HE.Eval with the pk, two encrypted messages c_1 and c_2 and the linear function $\mathcal{L}$, where $\mathcal{L}$ maps (m_1, m_2) to $km_1 + m_2$ $(\exists k \in \mathbb{R})$.

It has the following properties:

Correctness: given the ciphertext c from
$\text{HE.}\mathsf{Enc}(\mathsf{pk}, m)$, it holds that $\mathsf{Dec}(\mathsf{sk}, (\mathsf{Enc}(\mathsf{pk}, m))) = m$.

Semantic Security: the $\{\mathsf{pk}, \text{HE.}\mathsf{Enc}(\mathsf{pk}, m)\} \approx_c \{\mathsf{pk}, \text{HE.}\mathsf{Enc}(\mathsf{pk}, m^{'})\}$ are required for $\forall m, m' \in M$, where two distributions are over the random choice of pk and the random coins of the encryption algorithm.

Function privacy: there exists a simulator Sim such that for $\forall$ adversary $\mathcal{A}$, linear function $\mathcal{L}$, and messages m_1, m_2, then we have computationally indistinguishable distributions as below:

$$\left\{\begin{array}{l} (r, r_1, r_2) \leftarrow \{0,1\}^\lambda \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \text{HE.}(1^\lambda; r) \\ c_1 \leftarrow \text{HE.}\mathsf{Enc}(\mathsf{pk}, m_1; r_1) \\ c_2 \leftarrow \text{HE.}\mathsf{Enc}(\mathsf{pk}, m_2; r_2) \\ c^{'} \leftarrow \text{HE.}\mathsf{Eval}(\mathsf{pk}, c_1, c_2, \mathcal{L}) \end{array}\right\} \approx_c \textsc{Sim}(1^\lambda, m_1, m_2, \mathcal{L}(m_1, m_2))$$

Secret Sharing. With Secret Sharing (SS) schemes, a secret can be securely shared among multiple parties. SS schemes ensure that each share does not reveal any information about the secret. A 2-of-2 additive secret sharing of x where $x \in \mathbb{Z}_p$ and p is a prime. Then, we have a pair $(x_1, x_2) = (x - r, r) \in \mathbb{Z}_p^2$, where a random $r \in \mathbb{Z}_p$, such that $x = x_1 + x_2$. Given the share of x_1 and x_2, the value x is perfectly hidden. The Beaver Multiplication Triples [130] extend the additive secret sharing for the Multiplication performance.

Beaver's multiplication: Beaver's multiplication triples are widely applied for secure computation. Let $\mathbb{F}$ be the finite field. A multiplication triple is a tuple $([a], [b], [c])$, where $a, b \in \mathbb{F}$ are random elements such that $c = a \cdot b$. We assume that the $[x]$ is an additive sharing of x and each party holds the share x_i, so that

$\sum_{i=1}^{n} x_i = x$. Then, the private multiplication (i.e., multiplying secret-shared values $[x \cdot y]$) can be performed by revealing these multiplication triples. Also, the triples generation procedure is a two-party protocol for triples (a, b, c) secret shares output. We have the $[a]_1, [b]_1, [ab]_1$ for the first party and $[a]_2, [b]_2, [ab]_2$ for the second party. In this paper, the linearly homomorphic encryption scheme will be used for Beaver's triples generation. We provide further details for the protocol `Crypto3D` in Section 5.3.

Garbled Circuits. It was proposed by Yao [21], which is the first two-party secure computation protocols. The garbled circuit generator (one party) prepares the encrypted circuit computing f while the garbled circuit evaluator (the other party) computes the output of the circuit without learning any intermediate values. Denoting the Boolean circuit as C, for the input $\mathbf{x}$, a *Garbling scheme* includes a group of algorithms $GS = (\textsc{Garble}, \textsc{Eval})$ as below:

- $GS.\textsc{Garble}(C) \rightarrow (\widetilde{C}, \{label_{i,0}, label_{i,1 i\in[n]})\}$, where $label_{i,b}$ is the assigned value $b \in \{0, 1\}$ to the $i-th$ input label. Given input a boolean circuit C, the Garble outputs a garbled circuit $\widetilde{C}$ and a set of labels $\{label_{i,0}, label_{i,1 i\in[n]})\}$.
- $GS.\textsc{Eval}(\widetilde{C}, \{label_{i,x_i}\}) \rightarrow y$. The Eval outputs $y = C(x)$ given labels $\{label_{i,x_i}\}$ with an input $x \in \{0, 1\}^n$ and input garbled circuit $\widetilde{C}$.

It has the following properties:

Completeness: the Eval output is equal to $C(x)$.

Privacy: given $\widetilde{C}$ and $\{label_{i,x_i}\}$, only the size of $|C|$ and the output $C(x)$ can be known by the evaluator.

Security: given input $1^\lambda, 1^{|C|}$ and $C(x)$, the outputs of $\widetilde{C}, \{label_i\}_{i\in[n]}$ is com-

putationally indistinguishable to $(\widetilde{C}, \{label_{i,x_i}\}$ generated the $GS.\textsc{Garble}$.

Oblivious Transfer. Oblivious Transfer (OT) [131] works as a fundamental cryptographic building block in MPC, which executes between the two parties client (called sender) and server (called receiver). Specifically, the sender has two inputs x_0 and x_1 while the receiver wants to obtain the x_b (a selection bit b) without any revealing b or learning anything to the server. In this setting, $(\perp; x_b) \leftarrow \mathsf{ObliviousTransfer}(x_0, x_1; b)$ is used to represent this functionality.

5.3 Secure Inference Protocol `Crypto3D`

We assume the generic *two-party secure inference* setting for our work: the client $\mathcal{C}$ and server $\mathcal{S}$. The pre-trained 3D neural network model is held by the server $\mathcal{S}$ while the input video that to be classified is held by the client $\mathcal{C}$. The *secure inference* can be achieved by the two-party interactions via the designed protocol. In this protocol, the video data from the client and model's parameters from the server are kept confidential, and results from the secure computation will be eventually sent back to the client. Neither party can learn or infer any knowledge based on the prediction results. In this section, we first define the threat model and the security guarantees for the secure inference protocol `Crypto3D`, and then illustrate the protocol design.

5.3.1 Threat Model and Security Definition.

In `Crypto3D`, we assume that the semi-honest client $\mathcal{C}$ and server $\mathcal{S}$ are honest to follow the protocol but curious to learn private information from each other. They can be corrupted by an adversary $\mathcal{A}$, but not at the same time. Also, this semi-honest adversary $\mathcal{A}$ will not deviate from the protocol but infer private information stead.

Each client holds its video streams and it expects not to disclose the content of video to the cloud or other video analytics services. We assume that computing the C3D and the DNN architecture are known to the public (i.e., dimensions and type of

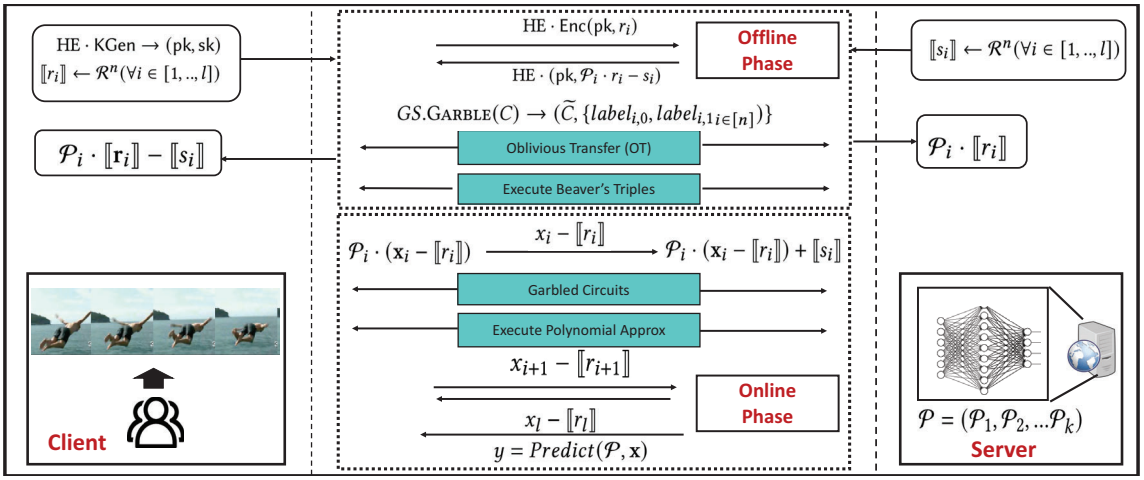


Figure 5.3. Overview of the `Crypto3D` framework including both offline and online phase. The operation are preformed in the left hand side of the figure are executed by the client while the right hand side are executed by the server.

each layer in the neural networks), except the parameter of model weights. Since it is the proprietary information to the cloud service provider, the model weights are not allowed to be revealed. Based on the proposed cryptographic protocols, the privacy of input video and model weights are guaranteed.

Definition 10. A cryptographic inference protocol $\prod_{\texttt{SecureINF}}$ between the client with an input feature vector $\mathbf{x}$ and the server with pre-trained model parameters $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2, ...\mathcal{P}_k)$ securely computes a function f, and satisfies the following properties.

- *Correctness*: For all set of model parameters $\mathcal{P}$ and all feature input vectors $\mathbf{x}$, the output at the end of protocol is the correct prediction in the cryptographic inference.
- *Security*:
  - Semi-Honest Server Security. There exists a simulator Sim_S such that $\mathsf{View}_S^{\prod} \approx_c \mathsf{Sim}_S(\mathcal{P})$, where $\mathsf{View}_S^{\prod}$ denotes the view of the server in the

protocol $\prod$. Sim_S is able to simulate a view of the semi-honest server without learning any private input vector $\mathbf{x}$ of client in polynomial time.

– Semi-Honest Client Security: There exists a simulator Sim_C such that $\mathsf{View}_C^{\prod} \approx_c \mathsf{Sim}_C(\mathbf{x}, \mathsf{output})$, where $\mathsf{View}_C^{\prod}$ denotes the view of the client in the protocol $\prod$ and output represents the results of inference. Sim_C is able to simulate a view of the semi-honest client without learning any pre-trained model parameters $\mathcal{P}$ in polynomial time.

As shown in Figure 5.3, `Crypto3D` is designed to secure the *two-party inference* between the client and the cloud service provider. Given the input data from the client, the cloud service provider executes the prediction with the neural network, and then sends the inference results back to the client. In `Cryto3D`, the two parties interact with each other for a secure inference together with the provided inputs. The cloud service provider provides the 3D neural network model used for computation, while the client offers the private input to be inferred by the 3D model.

5.3.2 Protocol Design. Figure 5.3 illustrates the *secure two-party protocol* `Crypto3D`. In the offline $\prod_{\texttt{SecureOFL}}$ phase, it can be executed independent of the inputs: this phase can be executed offline. The data pre-computed by the client and server can be used in the online execution for the $\prod_{\texttt{SecureONL}}$ phase. Without loss of generality, we present the design for the protocol based on the C3D [124], which can be extended.

Overview. As shown in the Figure 5.3, the `Crypto3D` protocol contains two phases: offline $\prod_{\texttt{SecureOFL}}$ and online inference/predication $\prod_{\texttt{SecureONL}}$ phase. Assume that the pre-trained DNN model from the server will not be changed and updated. The offline phase is supposed to be independent of the input data from the client. Once the offline $\prod_{\texttt{SecureOFL}}$ is completed, the input data given by the client will be sent to

the cryptographic protocol for executing the online phase. However, the arithmetic operations of the encrypted matrices are involved and it leads to the inefficiency for the high-dimensional data tensors computation. To mitigate this issue, our `Crypto3D` utilized the optimized matrix permutation [132] to efficiently perform the operation of matrix computation with the ciphertext packing and parallelism [133]. The operation of the matrix multiplication can be considered as the sum of component-wise products with the specific permutations of the matrices themselves. We assume that there are two square matrices with size $n \times n$, the n permutations of the matrix A via the followings symmetric permutations:

$$\sigma(A)_{i,j} = A_{i,i+j}, \tau(A) = A_{i+j,j}$$

$$\phi(A)_{j,j} = A_{i,j+1}, \psi(A) = A_{i+1,j} \tag{5.2}$$

where ϕ and ψ are denoted as the shifting functions for column and row, respectively. Then, the multiplication of two matrices (we denote A and B) with the order d can be computed as below:

$$A \cdot B = \sum_{k=1}^{d-1} (\phi^k \odot \sigma(A)) \times (\psi^k \odot \tau(B)) \tag{5.3}$$

where $\odot$ refers to the component-wise product and k is used to represent the number of times for perturbation. As such, we can efficiently compute the two matrix multiplications. In our protocol `Crypto3D`, we utilize the function $\mathsf{Permu}(\cdot)$ to represent the computation of the n permutation operations. To boost the efficiency, we also utilize the vectorable homomorphic encryption "Ciphetext packing". We use the $\mathsf{Encode}(\cdot)$ to refer to the matrix transformations, which transforms a matrix into a plaintext vector with encoding map functions. Similarly, $\mathsf{Decode}(\cdot)$ is used for

the plaintext vector transformations to the matrix. The equation 5.3 can be securely computed with the multiplicative property of HE. Our `Crypto3D` uses the optimized matrix multiplication and ciphertext packing [132] for the efficiency improvement. Since we can pack all the inputs into a single ciphertext and perform layer computation (e.g., convolutions) in parallel, we can enable the SIMD parallelism with the ciphertext packing.

At the last phase, the inference results will be privately obtained. The followings are the details for the procedures.

Offline Phase ($\prod_{\texttt{SecureOFL}}$). Our `Crypto3D` provides the offline phase execution, which can be executed before the input is known. First, $(\mathsf{pk}, \mathsf{sk})$ can be fetched via the KGen algorithm. The input value $\mathbf{x}$ is independent of the *offlinePhase()* execution. We denote $r_i \leftarrow \mathbb{R}^n, i \in [1, .., l]$ and $s_i \leftarrow \mathbb{R}^n, i \in [1, .., l]$ as the random masking vectors for the i-th layer. In the linear layer,the $\mathsf{Enc}(\mathsf{pk}, r_i)$ is sent to the server by the client. With the Eval procedure, the server computes the $\mathsf{Enc}(pk, (\mathcal{P}_i \cdot r_i - s_i))$ and send its back to the client. Then, the client decrypts and obtains decrypted value for all layers. Thus, the additive secret sharing of $\mathcal{P}_i \cdot r_i$ is held by both the client and the server before the online phase execution. Regarding the non-linear layer execution, the execution of activation function depends on what type of function. The garbled circuit is constructed via GC schemes. It helps to solve the ReLu function by exchanging the labels for input wires with r_{i+1} and $\mathcal{P}_i \cdot r_i - s_i$. On the other hand, the Beaver's triples protocol is used for the polynomial approximation functions.

Online Phase ($\prod_{\texttt{SecureONL}}$). Given the input $\mathbf{x}$, the server receives $\mathbf{x} - r_1$. At this time, the additive secret shares of $\mathbf{x}$ are held by the client and server, respectively. At the beginning of the i-th layer evaluation, $\mathbf{x}_i$ can be fetched from the first $(i-1)$ layers of the neural network. The client holds r_i while server holds $x_i - r_i$. For the evaluation

of the linear layer(s), the server computes $\mathcal{P}_i \cdot (\mathbf{x}_i - r_i)$, which ensures that the additive shared secrets of $\mathcal{P}_i \cdot \mathbf{x}_i$ are held by the client and server, respectively. Once the linear layer is completed, $\mathcal{P}_i \cdot (\mathbf{x}_i - r_i) + s_i$ and $\mathcal{P}_i \cdot r_i - s_i$ are held by the server and client, respectively. Similarly, we use the garbled circuits and Beaver's multiplication for evaluating the non-linear layers. For the Garbled Circuits evaluation, the client receives the garbled labels from the server, which is corresponding to the $\mathcal{P}_i \cdot (\mathbf{x}_i - r_i) + s_i$. With these labels, the garbled circuit is evaluated to return the output of one-time pad (OTP $(x_{i+1} - r_{i+1})$) to the server. The $x_{i+1} - r_{i+1}$ is obtained by the server with one-time pad key. On the other hand, the Beaver's multiplication procedure is executed for the polynomial approximation evaluation. The client and sever will hold the $[x_{i+1}]_1$ and $[x_{i+1}]_2$, separately after the Beaver's multiplication procedure. At this time, the client sends the results of the $[x_{i+1}]_1 - r_{i+1}$ to the server. The $x_{i+1} - r_{i+1}$ will be obtained by adding the $[x_{i+1}]_2$. Finally, the client learns the x_l.

5.4 Security Analysis

Theorem 6. *The secure two-party inference protocol* $\prod_{\mathit{SecureINF}}$ *for* `Crypto3D` *(including* $\prod_{\mathit{SecureOFL}}$ *and* $\prod_{\mathit{SecureONL}}$*) is secure against semi-honest adversaries.*

Proof. We prove this theorem by considering two cases separately: (1) the adversarial client C, and (2) the adversarial server S. Then, we build polynomial simulators to simulate the views of all the participants of the protocol under the secure multiparty computation theory, detailed as below. □

5.4.1 Adversarial Client C :.

In this case, assume that the simulator $\widetilde{S}$ exists with the given client C and input $\mathbf{x}$.

1. Client chooses an uniform random tape via Simulator $\widetilde{S}$.

2. In the offline $\prod_{\texttt{SecureOFL}}$ phase:

   (a) Simulator $\widetilde{S}$ receives pk and ciphertext $\mathcal{C}(r_i) \leftarrow \mathsf{Enc}(\mathsf{pk}, r_i)$. Then, simulator $\widetilde{S}$ sends ciphertext $\mathcal{C}(s_i)' \leftarrow \mathsf{Enc}(\mathsf{pk}, -s_i')$ with random $s_i' \in \mathcal{R}^n$ to the server.

   (b) Simulator $\widetilde{S}$ runs on 1^λ and $1^{|C|}$ and sets the random value for the circuit $\widetilde{S}_{GS}$ output. The $\widetilde{C}$ and $\{label_i\}$ are the outputs from $\widetilde{S}_{GS}$. In the $i-th$ Oblivious Transfer (OT) execution, $\widetilde{S}$ uses the $lable_i$ as the input and sends $\widetilde{C}$ to the client.

   (c) $\widetilde{S}$ runs the corresponding simulator with the Beaver's triples procedure under $\prod_{\texttt{SecureINF}}$.

3. Online phase: Simulator $\widetilde{S}$ receives $\mathbf{x} - r_1$ from the offline phase, sends $\mathbf{x}$ to the ideal functionality $\mathcal{F}$ and obtains the output y. The simulator $\widetilde{S}$ performs the corresponding evaluation as below:

   (a) Simulator $\widetilde{S}$ sends the simulated labels for GC layers.

   (b) Simulator $\widetilde{S}$ evaluates the polynomial approximation layer for Beaver's multiplication procedure.

4. Simulator $\widetilde{S}$ sends output $\mathbf{y} - r_l$ to the client.

We now present that the distribution of real world is computationally indistinguishable to the simulator $\widetilde{S}$ in the ideal world. We prove this by a sequence of random experiments as following. It shows that the successive random experiments are computationally indistinguishable. The server's model weights will not be used in the simulator $\widetilde{S}$ for the final simulated distribution, thus nothing except the prediction results and model architecture will be learned by the corrupted client.

- **Hybrid0**: it corresponds to real world distribution with the actual input matrices $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2, ...\mathcal{P}_k)$ from the honest server.

- **Hybrid1**: same as Hybrid0, except for a syntactic change. The simulator $\widetilde{S}$ sends $\mathbf{y} - r_k$ to the client in the output phase. With the knowledge of the client's random tape, the simulator $\widetilde{S}$ begins the evaluation of the $i-th$ layer with $\mathbf{x} - r_i$. The distribution on the view of the $\mathcal{A}$ for above is identical with this syntactic change.

- **Hybrid2**: in this hybrid, the inputs that server provides to each OT execution are changed, in which it acts as the sender. The server provides $label_{i,b}$ to replace the labels corresponding to 0 and 1 in each OT execution, where b is inputted from the client in that OT execution. In the semi-honest setting, b is a result of setting the random tape and learning the input of corrupted client. Hybrid2 is indistinguishable from Hybrid1 with the sender security of OT execution.

- **Hybrid3**: in this hybrid, we generate $\widetilde{C}$ using the $\widetilde{S}_{GS}$ on input 1^λ, $1^{|C|}$ and $C(z)$ where z is the input corresponding to the circuits evaluation by the client.Also, $C(z)$ is an one-time pad (OTP) encryption, which is distributed identically to a random string. Hybrid3 is indistinguishable from Hybrid2 with the followed security of the garbled circuits.

- **Hybrid4**: in this hybrid, the multiplication triples in the offline phase is generated with the corresponding simulator $\widetilde{S}$ for Beaver's protocol. This follows from the simulation security that Hybrid4 is indistinguishable from Hybrid3.

- **Hybrid5**: in this hybrid, we use simulator $\widetilde{S}$ for the Beaver's multiplication procedure for every quadratic approximation layer. Note that in this hybrid, the $x_i - r_i, s_i$ and matrix $\mathcal{P}_i$ are no longer used for $i-th$ layer evaluation. Similarly, this follows from the simulation security that Hybrid5 is indistinguishable from

Hybrid4.

- **Hybrid6**: the simulator $\widetilde{S}$ is used for the function privacy with respect to each homomorphic evaluation in the offline phase. Also, $\widetilde{S}$ only requires the $\mathcal{P}_i \cdot r_i - s_i$ for the homomorphically evaluated ciphertext generation. This follows the function privacy of HE in which Hybrid6 is computationally indistinguishable from Hybrid5.

- **Hybrid7**: in this hybrid, we set input $-s_i^{'}$ instead of the true value $(\mathcal{P}_i \cdot r_i - s_i)$. It is given to the $\widetilde{S}$ with randomly sampled $s_i^{'}$ from R^n. The s_i is chosen uniformly at random, this indicates that the Hybrid7 is identically distributed to Hybrid6. Eventually, we note that Hybrid7 is identically distributed to the simulator $\widetilde{S}$'s output.

This completes the proofs for the case of adversarial client.

5.4.2 Adversarial Server S **:.** In this case, we assume that the simulator $\widetilde{S}$ exists as below once given the inputs $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2, ...\mathcal{P}_k)$ from the server.

1. Server chooses a uniform random tape via simulator $\widetilde{S}$.
2. In the offline phase:
   (a) Simulator $\widetilde{S}$ sends $\mathsf{Enc}(\mathsf{pk}, 0)$ to the server with chosen pk and receives the ciphertext from the server.
   (b) Simulator $\widetilde{S}$ and works as the receiver from server and uses the fake input with "0" as receiver's choice bit.
   (c) Simulator runs the corresponding simulator $\widetilde{S}$ for Beaver's triples generating.

3. Online phase: simulator $\widetilde{S}$ sends r_1 from the offline phase with an uniformly chosen r_1. The simulator $\widetilde{S}$ performs the corresponding evaluation as below:

   (a) Simulator $\widetilde{S}$ sends the random value back to sever for GC layers.

   (b) $\widetilde{S}$ uses simulator for Beaver's multiplication to evaluate the polynomial approximation. The random value is sent back to the server at the final round.

We present that the distribution of real world is computationally indistinguishable to the simulated distribution by the following hybrid arguments. Since the user's input is not used by the simulator in the final simulated distribution, a corrupted server will not know anything in the real world.

- **Hybrid0**: the simulator $\widetilde{S}$ corresponds to the real world distribution with the actual input $\mathbf{x}$ from client.

- **Hybrid1**: same as Hybrid0, except for a syntactic change. With respect to the layer evaluation by the garbled circuits, we send the one-time pad encryption $\text{OTP}(x_{i+1} - r_{i+1})$ by the knowledge of $\mathbf{x}$, $\mathcal{P}_i$ and random tape of the server, instead of the circuits evaluation. Similarly, a share is sent in final round. Thus, when the server adds it with its own share, it gets $x_{i+1} - r_i$. Hybrid1 is identical to the Hybrid0 with this syntactic change.

- **Hybrid2**: in this hybrid, the inputs that client provides to each OT execution are changed, in which it acts as the sender. We provide the fake input with '0' to replace the real inputs. This follows the receiver security of obvious transfer protocol, and Hybrid2 is computationally indistinguishable from Hybrid1.

- **Hybrid3**: in this hybrid, we generate the multiplication triples in the offline phase with the simulator for Beaver's multiplication protocol. With the followed

simulation security, the Hybrid3 is computationally indistinguishable from Hybrid2.

- **Hybrid4**: in this hybrid, we use simulator $\widetilde{S}$ for the procedure of Beaver's multiplication, with respect to each quadratic approximation layer of the neural network. With the followed simulation security, the Hybrid4 is computationally indistinguishable from Hybrid3.

- **Hybrid5**: in this hybrid, we update the ciphertexts sent by the client in the offline phase. The client sends $\mathsf{Enc}(\mathsf{pk}, 0)$ instead of $\mathsf{Enc}(\mathsf{pk}, r_i)$. The Hybrid5 is computationally indistinguishable from Hybrid4 since this follows the semantic security of the encryption scheme.

- **Hybrid6**: in this hybrid, we make some changes. With respect to the layer evaluation by the garbled circuits, we send the one-time pad encryption OTP (r_{i+1}) with randomly chosen r_{i+1} to the server. Similarly, in terms of the each quadratic approximation layer, a share, which is chosen uniformly at random is sent at the final round. Furthermore, a uniformly chosen value r_1 in the offline phase will be sent. Eventually, we note that Hybrid6 is identically distributed to the simulator's $\widetilde{S}$ output.

This completes the proofs for the case of adversarial server.

5.5 Results

In this section, we evaluate `Crypto3D` with the UCF-101 and HMDB-51 human action recognition datasets. UCF-101 consists of the 13,320 videos from YouTube, with over 101 categories of human actions. HMDB-51 contains 6,849 video clips from 51 distinct action classes. More details can be found in Figure 5.1. The C3D weight model is generated from Sports-1M, which contains more than 1 mil-

lion YouTube videos annotated with 487 sports classes. The I3D ConvNet model is trained for action recognition with Kinetics-400, which includes 400 different actions.

In `Crypto3D`, we adopt the additively homomorphic encryption scheme of BFV (the scheme used in recent works of Gazelle [123] and Delphi [122]). Also, we adopt the optimized algorithms of Gazelle and [132] for homomorphic matrix-vector products and homomorphic convolutions. The BFV scheme uses the bathing optimization that enables operation on plaintext vectors over the field $\mathbb{Z}_n$.

Table 5.1. UCF-101 and HMDB-51 video datasets

	UCF-101 [127]	HMDB-51 [134]
Average Resolution	360 × 288	360 × 240
Pretrained	Sports-1M	Sports-1M
Accuracy	85.8%	54.9%
Category/Class	101	51

5.5.1 Evaluation Setup. Our `Crypto3D` is implemented with Rust, Python and C++. All the experiments are evaluated on a Ubuntu 20.04.2 LTS server with the NVIDIA-SMI 460.80 GPU. The CPU is Intel(R) Core(TM) i7-10700K at 3.80GHz featuring 8 cores with 64 GB. We evaluate `Crypto3D` with the following datasets and network architecture:

- **Datasets and 3D DNN Models**. We evaluate C3D and I3D features on the UCF-101 and HMDB-51 datasets, as described in Table 5.1. The C3D network contains 8 convolutions, 5 pooling layers and 2 fully connected layers. The first convolution layer (Convolution (1a)) has a size of 1x3x3 and it is followed

by a 1x2x2 pooling layer. This helps the temporal information preservation in the first layer and then builds higher level representation of the temporal information with the subsequent layers. In addition, other convolution and pooling layers have a size of 3x3x3 and 2x2x2 with the strides of 1 and 2, respectively. The fully connected layers have a size of 4096 dimensions with softmax outputs. The I3D model can further improve C3D via inflating 2D models. With I3D, we can reuse the 3D models' network architecture (e.g., ResNet, Inception) and bootstrap the weights of model from the 2D pre-trained model.

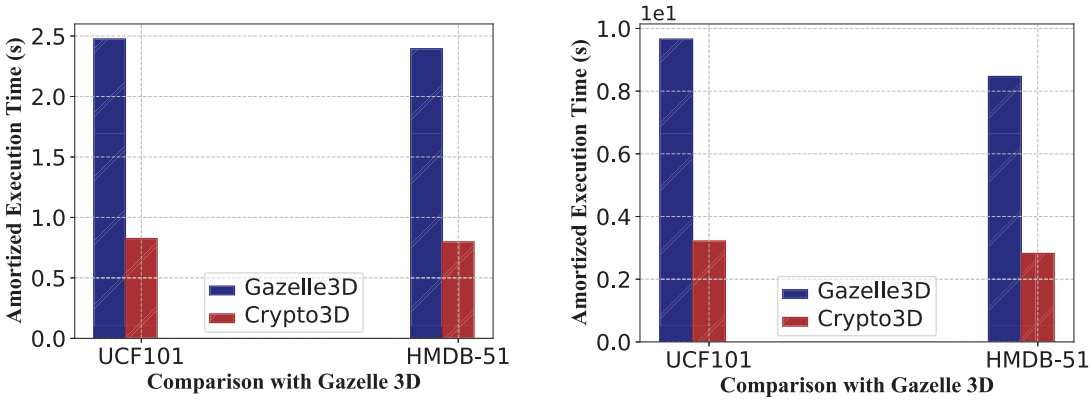


Figure 5.4. Gallze (3D) vs. Crypto3D

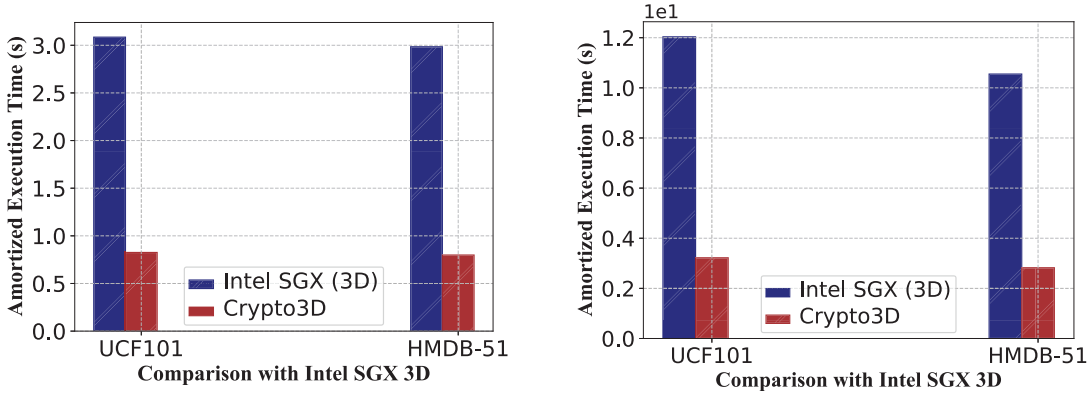


Figure 5.5. Intel SGX (3D) vs. Crypto3D

- **Comparison with Existing Systems**. We provide the performance com-

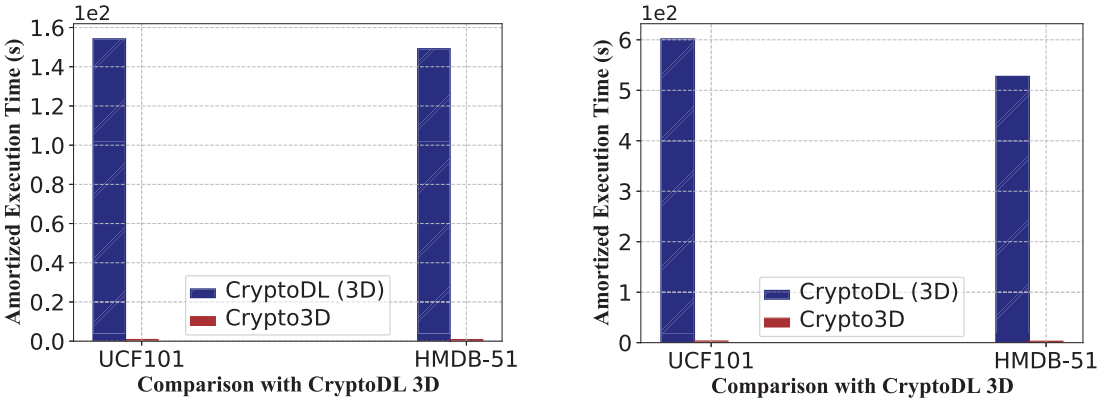


Figure 5.6. CryptoDL (3D) vs. Crypto3D

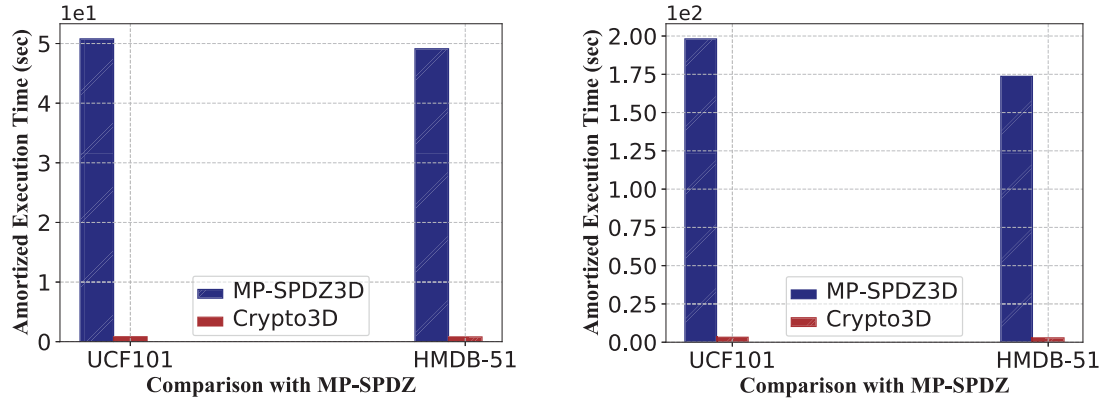


Figure 5.7. MP-SPDZ (3D) vs. Crypto3D

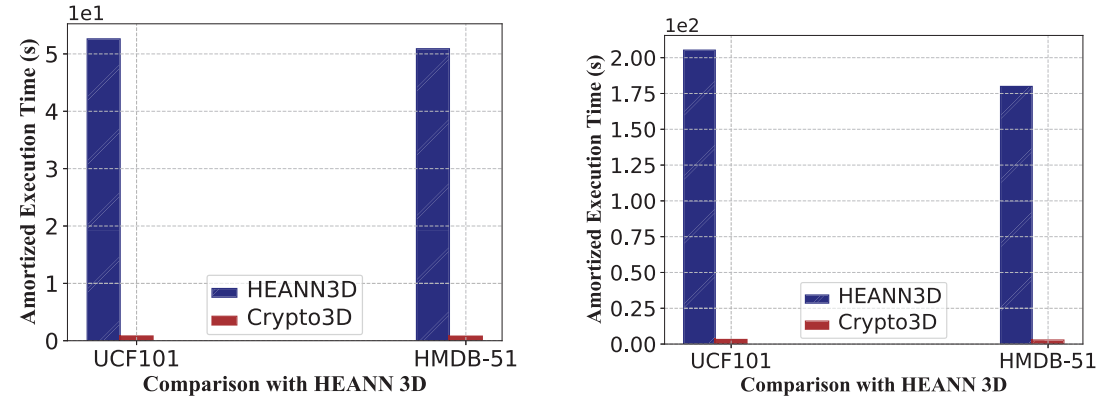


Figure 5.8. CryptoDL (3D) vs. Crypto3D

parison of `Crypto3D` and other privacy-preserving frameworks with 3D structure. All the benchmark systems cannot be directly applied to for video in-

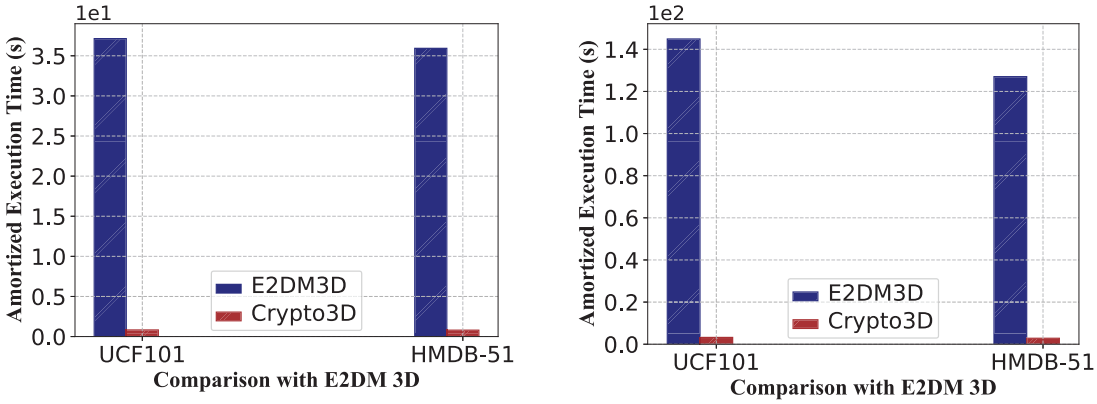


Figure 5.9. E2DM (3D) vs. Crypto3D

ferences based on the C3D model. We significantly extend them by modifying the 2D CNN network to embed with 3D architecture. With the 3D filters, the spatio-temporal features are able to be extracted. We re-implement the following systems on the C3D model: Gazelle (3D), Intel SGX (3D), MP-SPDZ (3D), CryptoDL (3D), HEANN (3D) and E2DM (3D). However, Delphi and GALA cannot be extended due to the 2D structure or lack of source codes. As we reported, it summarizes the cryptographic method, library, total execution time, speedup and amortized time. `Crypto3D` significantly outperforms all other benchmarks. The execution time of `Crypto3D` is over 186.89×, 63.75×, 61.52×, 45× 3.74× and 3× faster than CrytoDL (3D), HEANN (3D), MP-SPDZ (3D), E2DM (3D), Intel SGX (3D) and Gazelle (3D), respectively. These results show that `Crypto3D` is much more efficient in 3D privacy-preserving video input inference. Additionally, `Crypto3D` only takes 0.83 sec on average to process the secure inference for each frame, while other HE-based frameworks take much longer time because of the computational overhead. Note that the accuracy of the all other benchmarks is only less than 70% while `Crypto3D` can achieve the accuracy of 82.3%.

5.5.2 Performance of `Crypto3D` on C3D.

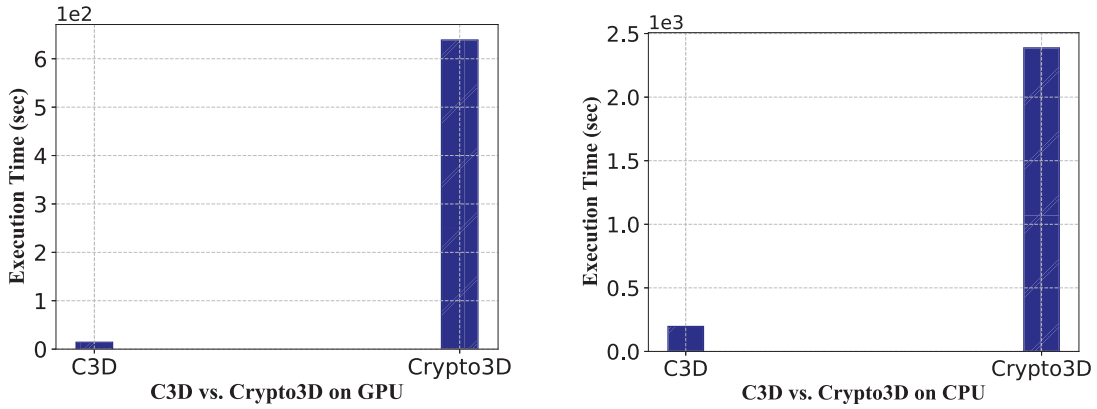


Figure 5.10. Execution time on GPU and CPU with (`Crypto3D`) and (C3D) without privacy support.

We first show the performance of our privacy-preserving `Crypto3D` system on CPU and GPU, and compare it with the native C3D that does not support privacy. The mixed cryptographic protocol introduces moderate performance overheads. On GPU, `Crypto3D` is 44.89× slower than C3D, and on CPU, `Crypto3D` is 12× slower than C3D. In `Crypto3D`, the inference runtime on GPU is 3.74× faster than CPU. Therefore, for the performance evaluation in the following sections, we run the experiments on GPU since it provides more parallelism and computation power. The mixed cryptographic protocol introduces moderate performance overheads. On GPU, `Crypto3D` is 44.89× slower than C3D, and on CPU, `Crypto3D` is 12× slower than C3D. In `Crypto3D`, the inference runtime on GPU is 3.74× faster than CPU. Therefore, for the performance evaluation in the following sections, we run the experiments on GPU since it provides more parallelism and computation power.

5.5.3 Comparison with C3D and I3D. For the generic human action recognition setting, we work on the microbenchmarks for the most representative two datasets (UCF101 and HMDB51). The pre-trained weight models we used are extracted from Sports-1M for C3D and kinetics400 for I3D, respectively. There are faster amortized

execution time for the HMDB-51 than UCF101 in both C3D and I3D along with related comparison benchmarks. C3D performs better than I3D on both UCF101 and HMDB-51 as well. Compared to the pre-trained model and DNNs architecture, we find that the dataset would not be the main factor for the performance impact under this case. It indicates that our system is able to the implemented based on different 3D models (e.g., C3D and I3D) with UCF101 and HMDB51 datasets.

5.5.4 Comparison with PPVC. We compare with PPVC [126], which is the current state-of-the art for 2-party secure video classification. It uses MPC technique for private classification. In Figure 5.3, we find that the PPVC takes slightly less time since it uses 2D ConvNets trained model for video processing instead of C3D model. The architecture of ConvNets is [(CONV-RELU)-POOL]-[(CONV-RELU)*2-POOL]*2-[FCRELU]*2-[FC-SOFTMAX] on the FER 2013 data. The construct of C3D network is similar to the common 2D ConvNets, but the main difference is that C3D uses 3D operations like Conv3D while 2D ConvNets are entirely 2D architecture. This data is used to extract facial features for emotion recognition. Note that our model C3D [124] aims at action recognition, which is trained on Sports-1M dataset [128]. Moreover, the dataset used in the PPVC is RAVDESS [135] for the emotion detection. Thus, PPVC cannot work well on private video inferences given the very low accuracy (e.g., 56%).

5.6 Related work

`Cryto3D` is essentially a *two-party secure inference* protocol. We review the privacy preserving on DNNs with different technique as well as secure two-party frameworks. Most of recent work contains these techniques: homomorphic encryption [136, 137], SMC [122, 123, 126] and differential privacy [138–140].

Homomorphic Encryption-based Protocols: Homomorphic encryption allows

the mathematical operations on ciphertext without knowing the unencrypted data. FHE enable the secure DNNs training and inference without any interaction. The CryptoNets [136] is proposed for HE-based secure neural network inference. The client can obtain the ciphertext results from the server once given the encrypted input without inferring any information. Other DNNs applications apply much faster HE [117, 141]. But it only supports limited depth of encryption and multiplication operations without bootstrapping [141]. Hesamifard et al. [137] propose CryptoDL, which is a privacy-preserving Machine Learning as a Service (MLaaS) with LHE. In CryptoDL, the polynomials are used to replace the complex nonlinear action function such as Sigmoid and tanh. However, HE is not ideal and practical for the efficiency, because of the computational overhead.

MPC-based Protocols. MPC enable multiple parties to evaluate the function without releasing any inputs information to each other except the results. Generally, existing works include Garbled Circuit(GC) [142–144], Secret Sharing [145–147] , and Mixed Protocol [123, 148, 149]. [143] and [142] propose a way to optimize the neural network activation functions with garbled circuit. In [142] , the practical data aggregation protocols are designed by Shamir's t-out-of-n secret sharing protocol [150]. SPDZ [147] is a secure computation protocol with additive secret sharing to against $n-1$ corrupted computation modes in the malicious model. XONN [143] and DeepSecure [144] both use circuit garbling schemes for neural network predictions. In [151], pure MPC-based solution is studied. It proposes a framework SecureML with three honest party for privacy preserving machine learning, which focuses on linear active function. MiniONN [116] develops a technique for turning a pretrained model into an oblivious one, which includes additive secret sharing and GC. It supports all operations commonly used by neural network with a lower overhead at prediction time.

However, the limitation of the secret sharing and garbled circuits will bring the computational overhead. Instead, the Mixed Protocol [123, 148, 149], use additive secret sharing or homomorphic encryption to perform linear operations while garbled circuits to address the non-linear computations. Gazelle [123] is a hybrid HE-MPC protocol, which contains additive HE for polynomial functions and garbled circuits for non-polynomial activation functions with boost efficiency. ABY3 [148] is able to support the secure evaluation of linear and non-linear operation via arithmetic or boolean circuit. Chameleon [149] is an extension of ABY framework, which is a secure two-party computation framework. It performs polynomial operations with arithmetic secret sharing and non-linear operations (ReLU) with boolean sharing protocols or GC. The Delphi [122] improves the gazelle based on the GC and quadratic polynomials for activation functions. However, these prior works are all based on the 2D ConNets, the inference results does not keep the temporal information for video. In the `Crypto3D`, the C3D model is used to perform the cryptographic inference for video classification. The temporal features is remained from the prediction results.

TEE-based Protocols. Trusted Execution Environments (TEE) [118–120, 152, 153] based provides secure *enclave*, the model/data owner is able isolate sensitive computations for DNNs models from the untrusted software stack. It can guarantee the both data privacy and integrity. Florian et al. [120] propose Slalom, which enables all linear layers in secure infernce with TEE (Intel SGX). It is executed by a faster but untrusted co-located processor. Also, it uses ZKP to attest their correct execution. Ghodsi et al. proposes SafetyNet [154], it converts the specific type of DNNs framework into an arithmetic circuit. The results are able to be verified by interactions with the servers. In [121], visor, a proposed system that enables privacy preserving video analytics services with a hybrid TEE architecture. It supports and guarantees strong confidentiality and integrity for video streams. Most TEE-based secure

cryprographic inference show the better performance than the protocols that rely on crypto tools. However, it requires trust in hardware for a weaker threat model and it needs to be implemented in the enclave. Also, side-channel attack also will be a more dangerous problem needs to be considered.

Differential Privacy-based Solutions. The differential privacy-based DNNs techniques aim at the reducing amount of sensitive information for the data carrying. And it is enable to reduce the negative impact of the addition of noise on training under the privacy budget. Shokri et al. [138] use the differential privacy for the deep learning model. It guarantees the data privacy is not compromised by sharing local parameters to the server. In other works [139, 155] propose different ways to handle the trade-off between the privacy and accuracy, (i.e., noise adding to the weights [139] , budget setting dynamically [140], etc.).

5.7 Summary

Many existing techniques are proposed to perform the *secure two-party inferences* with the cryptographic schemes for the deep neural networks. However, they cannot be directly applied to video inferences which extracts spatio-temporal (3D) features for more accurate video recognition. In this paper, we propose `crypto3D`, the first cryptographic neural network inference based on 3D features, which achieves significant performance by (i) privately inferring videos (e.g., action recognition, and video, and classification) on 3D spatial-temporal features with the C3D and I3D pretrained DNN models; (ii) involving an optimized matrix operations and ciphertext packing technique in `Crypto3D` for efficiency boosting. In addition, we significantly modify most of the state-of-the-art secure DNNs protocols (CryptoDL, HEANN, MP-SPDZ, E2DM, Intel SGX, and Gazelle) to privately infer videos with 3D features (C3D and I3D models) as the benchmarks.

APPENDIX A
DIFFERENTIAL PRIVACY

A.1 Proofs

A.1.1 Proof of Theorem 1.

Proof. Note that $\forall O_1 \in S^+$, $Pr[\mathcal{A}_1(D') = O_1] = 0$, then

$$\begin{aligned} Pr[\mathcal{A}_1(D') \in S] =& \int_{\forall O_1 \in S^+} Pr[\mathcal{A}_1(D') = O_1] dO_1 + \int_{\forall O_2 \in S^-} Pr[\mathcal{A}_1(D') = O_2] dO_2 \\ \leq& e^{\epsilon} \int_{\forall O_2 \in S^-} Pr[\mathcal{A}_1(D) = O_2] dO_2 \\ =& e^{\epsilon} Pr[\mathcal{A}_1(D) \in S^-] \\ \leq& e^{\epsilon} Pr[\mathcal{A}_1(D) \in S] \end{aligned}$$

This completes the proof. □

A.1.2 Proof of Theorem 2.

Proof. It is straightforward to prove that the probabilities that results in Case (2) for all the vehicles and positions are bounded by δ if inequality 2.5 holds (by setting δ in the preprocessing). In addition, as analyzed in Case (1), if inequality 2.3 holds for all Θ_r, per Theorem 1, we have:

$$e^{-\epsilon} \leq \frac{Pr[\mathcal{A}_1(D) \in S]}{Pr[\mathcal{A}_1(D') \in S]} \leq e^{\epsilon} \tag{A.1}$$

where S represents any set of possible outputs (without data from Θ_r). This completes the proof. □

A.2 Expectation of Dirichlet Distribution

In Dirichlet-Multinomial sampling, the expectation of Dirichlet distribution [56] can be derived as:

$$
\begin{aligned}
& E[\theta_i(j)|\lambda_i(j)] \\
=& \int_{\Delta} \theta_i(j) \frac{\Gamma(\sum_{\forall j} \lambda_i(j))}{\prod_{\forall j} \Gamma(\lambda_i(j))} [\theta_i(1)]^{\lambda_i(1)-1} \dots [\theta_i(|\Phi|)]^{\lambda_i(|\Phi|)-1} \mathrm{d}\theta_i \\
=& \frac{\Gamma(\lambda_i(j)+1)\Gamma(\sum_{\forall j} \lambda_i(j))}{\Gamma(\lambda_i(j))\Gamma(\sum_{\forall j} \lambda_i(j)+1)} \times \int_{\Delta} \theta_i(j) \frac{\Gamma(\sum_{j=1}^{\Phi} \lambda_i(j)+1)}{(\lambda_i(j)+1)\prod_{s \neq j} \Gamma(\lambda_i(s))} \prod_{s=1}^{|\Phi|} [\theta_i(s)]^{\lambda_i(s)-1} \mathrm{d}\theta_i \\
=& \frac{\Gamma(\lambda_i(j)+1)\Gamma(\sum_{\forall j} \lambda_i(j))}{\Gamma(\lambda_i(j))\Gamma(\sum_{\forall j} \lambda_i(j)+1)} = \frac{\lambda_i(j)}{\sum_{j=1}^{|\Phi|} \lambda_i(j)}
\end{aligned} \tag{A.2}
$$

The expectation of the Dirichlet distribution is adapted for learning the posterior probability vector in phase II.

APPENDIX B

CRYPTOGRAPHIC INFERENCES FOR VIDEO

B.1 Notation Table

Table B.1. Frequently used notations

Notation	Description
$\mathcal{S}$	cloud service provider
$\mathcal{C}$	client
$\mathbf{x}$	a feature vector DNNs inputs
$\mathcal{V}$	video streams given by client
N frames $(F_0, F_1, ..., F_{N-1})$	frame divided from a video
l	the number of layer for DNNs
$\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_{\|l\|})$	weight model for l layer
$\prod_{\texttt{SecureINF}}$	secure inference protocol
$\prod_{\texttt{SecureONL}}$	online phase for $\prod_{\texttt{SecureINF}}$
$\prod_{\texttt{SecureOFL}}$	offline phase for $\prod_{\texttt{SecureINF}}$
$f(.)$	functions for arbitrary inputs

B.2 C3D Network and Architecture

We consider the video clips with the size $c \times l \times h \times w$ where c denotes the number of channels, l as the length in number of frames, h and w represent the height and width of each frame, respectively. In terms of the network settings, the video clips are taken as inputs to predict the class labels (e.g., from 101 different actions while performing inferences using the C3D model trained on UCF101 dataset). All video frames are resized to 128×171 and split into non-overlapped 16-frames clips. The

network includes 8 convolution layers, 5 max-pooling layers, 2 fully connected layers and followed by one softmax layer for predicting the class label. The first pooling layer includes the kernel size $1 \times 2 \times 2$ while others include the kernel size $2 \times 2 \times 2$. The output after the fully connected layer includes 4096 vector entries. With the homogenous architecture with small kernel sizes of 3, the C3D model can boost for optimized implementation of these architectures for efficient inference.

B.3 Description Benchmarks

Libraries for FHE Many existing works use the privacy preserving computation based on the homomorphic encryption (HE). HE enables the computation on the encrypted data without decryption. However, it consists of many restrictions. Therefore, we benchmarks the different state-of-the-art *secure two party inference* frameworks for valuations by integrating different HE libraries. We discuss the details of the provided benchmark. For the benchmark of Gazelle 3D, we still use Brakerski-Fan-Vercauteren (BFV) scheme from the 2D CNN inference [123]. That supports the integer operations with the Lattice encryption library. And PALISADE is a framework that provides a general API for multiple FHE schemes including BFV, BGV, and CKK.

Microsoft SEAL is a HE library that enable additions and multiplications to be performed on encrypted integers or real numbers. Also, it comes with two different FHE schemes with different properties: BFV and CKK. The modular arithmetic can be performed on encrypted integers by the BFV scheme. And CKKS scheme allows additions and multiplications on encrypted real or complex numbers, however, the approximate results can be generated. The CKKS scheme would be the one of best options for the application such as calculating the total encrypted real numbers or evaluating machine learning models on encrypted data. The BFV scheme is the only option for the application, which requires the exact value.

With respect to the hardware-based protection TEE, the strong privacy and integrity can be guaranteed. In our evaluation, we use the Graphene [102] (a lightweight guest OS) as Intel SGX for the C3D inference execution. It can replaces the Intel SDK for the *enclave* and host process. Furthermore, the MP-SPDZ library is designed for the Secure Multiparty Computation (MPC) implementation. In [126], it uses the MP-SPDZ library for the private video classification based on the Secure MPC. The privacy preserving technique can be achieved and executed for the video classificiation in [126], it still utilizes single frame method for inference with 2D ConvNet instead of the 3D CNN model.

CryptoDL uses the HElib library, it supports the SIMD operations, however there are limitations. Firstly, the division of ciphertexts is not supported.Also, it may causes the incorrect decryption with the exceeded noise, since the additional noise will be added for every computation performed on the ciphertext. Thus, an arbitrary number of computations (i.e., Activation functions) can not be supported. In this case, we use the polynomials as activation functions. The fixed point arithmetics can be supported by HEAAN library. This library supports approximate operations between rational numbers. The approximate error depends on some parameters and almost same with floating point operation errors. The [133] use the scheme in this library. And the HEMat is a extension from the HEANN schemes, where it designed for performing a optimized matrix computation with homomorphic encryption.

B.4 Matrix-Vector Multiplication

We assume that the input matrix $\mathcal{P}$ has the size $n_0 \times n_i$, where n_i is smaller than the number of plaintext slots n_s. We denote the sub-matrices $\mathcal{P}_{ij}$ (where $0 \leq i < n_0$ and $0 \leq j < l$) with the size of $1 \times (n_i/l)$, which is splited from the $\mathcal{P}$. Next we pack the different matrics $(l \cdot n_s)/n_i$ into a single ciphertext, and the $n_c = (n_s/n_i)$ copies of the input vector r into a single cipertext. With the encoding n_c, the first diagonal

vectors of the matrix into a plaintext vector as below:

$$((\mathcal{P}_{0,0}|\mathcal{P}_{1,1}|...|\mathcal{P}_{l-1,l-1})|(\mathcal{P}_{l,0}|\mathcal{P}_{l+1,1}|...|\mathcal{P}_{2l-1,l-1})|...$$

$$(\mathrm{P}_{l\cdot(n_c-1),0}|\mathcal{P}_{l\cdot(n_c-1)+1,1}...|\mathcal{P}_{l\cdot(n_c-1)+l-1,l-1)}) \in \mathbb{R}^{n_s}$$

Each extended diagonal vector is encrypted in a single ciphertext and these ciphertexts are multiplied with l rotations of the encrypted vector r. Next we add together and the output (ciphertext) represents (n_i/l)- sized $(l \cdot n_c)$ chunks. With the $log(n_i/l)$ rotations, we get the ciphertext with the first $(l \cdot n_c)$ entries of $\mathcal{P} \cdot r \in R^{n_0}$. Finally, we get the $n_0/(l \cdot n_c)$ ciphertexts after repeating the procedures $n_0/(l \cdot n_c)$.

BIBLIOGRAPHY

[1] B. Xu, X. J. Ban, Y. Bian, J. Wang, and K. Li, "V2I based cooperation between traffic signal and approaching automated vehicles," in *IEEE Intelligent Vehicles Symposium, IV 2017, Los Angeles, CA, USA, June 11-14, 2017*, pp. 1658–1664, 2017.

[2] X. J. Ban, P. Hao, and Z. Sun, "Real time queue length estimation for signalized intersections using travel times from mobile sensors," *Transportation Research Part C: Emerging Technologies*, vol. 19, no. 6, pp. 1133–1156, 2011.

[3] N. Rizzo, E. Sprissler, Y. Hong, and S. Goel, "Privacy preserving driving style recognition," in *2015 International Conference on Connected Vehicles and Expo (ICCVE)*, pp. 232–237, Oct 2015.

[4] R. Chen, B. Fung, B. C. Desai, and N. M. Sossou, "Differentially private transit data publication: a case study on the montreal transportation system," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 213–221, ACM, 2012.

[5] G. Poulis, S. Skiadopoulos, G. Loukides, and A. Gkoulalas-Divanis, "Distance-based k^m-anonymization of trajectory data," in *2013 IEEE 14th International Conference on Mobile Data Management*, vol. 2, pp. 57–62, June 2013.

[6] M. Li, L. Zhu, Z. Zhang, and R. Xu, "Achieving differential privacy of trajectory data publishing in participatory sensing," *Inf. Sci.*, vol. 400, pp. 1–13, Aug. 2017.

[7] C. Cottrill and P. Thakuriah, "Protecting location privacy: Policy evaluation," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2215, pp. 67–74, 2011.

[8] X. Ban, R. Herring, P. Hao, and A. M. Bayen, "Delay pattern estimation for signalized intersections using sampled travel times," *Transportation Research Record*, vol. 2130, no. 1, pp. 109–119, 2009.

[9] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 486–503, Springer, 2006.

[10] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*, pp. 94–103, IEEE, 2007.

[11] S. Zou, Z. Ma, and X. Liu, "Resource allocation game under double-sided auction mechanism: Efficiency and convergence," *IEEE Trans. Automat. Contr.*, vol. 63, no. 5, pp. 1273–1287, 2018.

[12] R. T. Maheswaran and T. Başar, "Nash equilibrium and decentralized negotiation in auctioning divisible resources," *Group Decision and Negotiation*, vol. 12, no. 5, pp. 361–395, 2003.

[13] R. Johari and J. N. Tsitsiklis, "Efficiency loss in a network resource allocation game," *Mathematics of Operations Research*, vol. 29, no. 3, pp. 407–435, 2004.

[14] J. R. Correa, A. S. Schulz, and N. E. S. Moses, "The price of anarchy of the proportional allocation mechanism revisited," in *Web and Internet Economics - 9th International Conference, WINE 2013, Cambridge, MA, USA, December 11-14, 2013, Proceedings*, pp. 109–120, 2013.

[15] V. Kuleshov and A. Vetta, "On the efficiency of markets with two-sided proportional allocation mechanisms," in *International Symposium on Algorithmic Game Theory*, pp. 246–261, Springer, 2010.

[16] V. Krishna, *Auction theory*. Academic press, 2009.

[17] S. Xie, Y. Hong, and P.-J. Wan, "Pairing: Privately balancing multiparty real-time supply and demand on the power grid," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1114–1127, 2019.

[18] T. Okamoto and S. Uchiyama, "A new public-key cryptosystem as secure as factoring," in *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, pp. 308–318, 1998.

[19] C. Gentry, *A Fully Homomorphic Encryption Scheme*. PhD thesis, Stanford, CA, USA, 2009. AAI3382729.

[20] O. Goldreich, "On the foundations of cryptography," in *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*, pp. 411–496, 2019.

[21] A. C. Yao, "How to generate and exchange secrets (extended abstract)," in *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pp. 162–167, 1986.

[22] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game, or a completeness theorem for protocols with honest majority," in *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*, pp. 307–328, 2019.

[23] A. A. Lazar and N. Semret, "Design and analysis of the progressive second price auction for network bandwidth sharing," *Telecommunication Systems*, vol. 13, 2001.

[24] B. Tuffin, "Revisited progressive second price auction for charging telecommunication networks," *Telecommunication Systems*, vol. 20, no. 3-4, pp. 255–263, 2002.

[25] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology - EUROCRYPT '99, IACR, 1999, Proc.*, pp. 223–238.

[26] D. Naccache and J. Stern, "A new public key cryptosystem based on higher residues," in *Proc.of the ACM CCS*, pp. 59–66, 1998.

[27] P. Das, L. Eckey, T. Frassetto, D. Gens, K. Hostáková, P. Jauernig, S. Faust, and A. Sadeghi, "Fastkitten: Practical smart contracts on bitcoin," in *USENIX Security Symposium, 2019*, pp. 801–818.

[28] M. Hoekstra, R. Lal, P. Pappachan, V. Phegade, and J. del Cuvillo, "Using innovative instructions to create trustworthy software solutions," in *HASP@ISCA*, p. 11.

[29] B. Liu, S. Xie, H. Wang, Y. Hong, X. Ban, and M. Mohammady, "VTDP: privately sanitizing fine-grained vehicle trajectory data with boosted utility," *IEEE Trans. Dependable Secur. Comput.*, vol. 18, no. 6, pp. 2643–2657, 2021.

[30] X. J. Ban and M. Gruteser, "Towards fine-grained urban traffic knowledge extraction using mobile sensing," in *Proceedings of the ACM SIGKDD International Workshop on Urban Computing*, pp. 111–117, ACM, 2012.

[31] K. Wang, R. Chen, B. Fung, and P. Yu, "Privacy-preserving data publishing: A survey on recent developments," *ACM Computing Surveys*, 2010.

[32] C. D. Cottrill, "Location privacy: Who protects?," *URISA Journal-Urban and Regional InformationSystems Association*, vol. 23, no. 2, p. 49, 2011.

[33] R. Chen, B. C. M. Fung, N. Mohammed, B. C. Desai, and K. Wang, "Privacy-preserving trajectory data publishing by local suppression," *Inf. Sci.*, vol. 231, pp. 83–97, May 2013.

[34] N. Pham, R. K. Ganti, Y. S. Uddin, S. Nath, and T. Abdelzaher, "Privacy-preserving reconstruction of multidimensional data maps in vehicular participatory sensing," in *Wireless Sensor Networks* (J. S. Silva, B. Krishnamachari, and F. Boavida, eds.), (Berlin, Heidelberg), pp. 114–130, Springer Berlin Heidelberg, 2010.

[35] R. Chen, H. Li, A. K. Qin, S. P. Kasiviswanathan, and H. Jin, "Private spatial data aggregation in the local setting," in *32nd IEEE International Conference on Data Engineering, ICDE 2016, Helsinki, Finland, May 16-20, 2016*, pp. 289–300, 2016.

[36] P. Hao, X. Ban, K. P. Bennett, Q. Ji, and Z. Sun, "Signal timing estimation using sample intersection travel times," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 792–804, 2012.

[37] J. Le Ny, A. Touati, and G. J. Pappas, "Real-time privacy-preserving model-based estimation of traffic flows," in *ICCPS'14: ACM/IEEE 5th International Conference on Cyber-Physical Systems (with CPS Week 2014)*, pp. 92–102, IEEE Computer Society, 2014.

[38] M. Mohammady, *Differentially Private Event Stream Filtering with an Application to Traffic Estimation*. PhD thesis, École Polytechnique de Montréal, 2015.

[39] D. Leoni, "Non-interactive differential privacy: a survey," in *Proceedings of the First International Workshop on Open Data*, pp. 40–52, ACM, 2012.

[40] R. Chen, N. Mohammed, B. C. Fung, B. C. Desai, and L. Xiong, "Publishing set-valued data via differential privacy," *Proceedings of the VLDB Endowment*, vol. 4, no. 11, pp. 1087–1098, 2011.

[41] Y. Hong, J. Vaidya, H. Lu, and M. Wu, "Differentially private search log sanitization with optimal output utility," in *Proceedings of the 15th International Conference on Extending Database Technology*, pp. 50–61, ACM, 2012.

[42] Y. Hong, J. Vaidya, H. Lu, P. Karras, and S. Goel, "Collaborative search log sanitization: Toward differential privacy and boosted utility," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 5, pp. 504–518, 2015.

[43] B. Hoh, T. Iwuchukwu, Q. Jacobson, D. B. Work, A. M. Bayen, R. Herring, J. C. Herrera, M. Gruteser, M. Annavaram, and J. Ban, "Enhancing privacy and accuracy in probe vehicle-based traffic monitoring via virtual trip lines.," *IEEE Trans. Mob. Comput.*, vol. 11, no. 5, pp. 849–864, 2012.

[44] K. Jiang, D. Shao, S. Bressan, T. Kister, and K. Tan, "Publishing trajectories with differential privacy guarantees," in *Conference on Scientific and Statistical Database Management, SSDBM '13, Baltimore, MD, USA, July 29 - 31, 2013*, pp. 12:1–12:12, 2013.

[45] B. Hoh, M. Gruteser, R. Herring, J. Ban, D. Work, J.-C. Herrera, A. M. Bayen, M. Annavaram, and Q. Jacobson, "Virtual trip lines for distributed privacy-preserving traffic monitoring," in *Proceedings of the 6th international conference on Mobile systems, applications, and services*, pp. 15–28, ACM, 2008.

[46] M. Gotz, A. Machanavajjhala, G. Wang, X. Xiao, and J. Gehrke, "Publishing search logs—a comparative study of privacy guarantees," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, pp. 520–532, March 2012.

[47] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "Random-data perturbation techniques and privacy-preserving data mining," *Knowl. Inf. Syst.*, vol. 7, no. 4, pp. 387–414, 2005.

[48] F. McSherry, "Privacy integrated queries: an extensible platform for privacy-preserving data analysis," in *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2009, Providence, Rhode Island, USA, June 29 - July 2, 2009*, pp. 19–30, 2009.

[49] H. Deng, I. King, and M. R. Lyu, "Entropy-biased models for query representation on the click graph," in *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009* (J. Allan, J. A. Aslam, M. Sanderson, C. Zhai, and J. Zobel, eds.), pp. 339–346, ACM, 2009.

[50] G. Ács, C. Castelluccia, and R. Chen, "Differentially private histogram publishing through lossy compression," in *12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, December 10-13, 2012* (M. J. Zaki, A. Siebes, J. X. Yu, B. Goethals, G. I. Webb, and X. Wu, eds.), pp. 1–10, IEEE Computer Society, 2012.

[51] Y. Kanzawa, Y. Endo, and S. Miyamoto, "Kl-divergence-based and manhattan distance-based semisupervised entropy-regularized fuzzy c-means," *JACIII*, vol. 15, no. 8, pp. 1057–1064, 2011.

[52] B. Anandan and C. Clifton, "Laplace noise generation for two-party computational differential privacy," in *13th Annual Conference on Privacy, Security and Trust, PST 2015, Izmir, Turkey, July 21-23, 2015*, pp. 54–61, 2015.

[53] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*, pp. 265–284, Springer, 2006.

[54] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pp. 75–84, ACM, 2007.

[55] N. Zamzami and N. Bouguila, "Text modeling using multinomial scaled dirichlet distributions," in *Recent Trends and Future Technology in Applied Intelligence - 31st International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2018, Montreal, QC, Canada, June 25-28, 2018, Proceedings*, pp. 69–80, 2018.

[56] T. P. Minka, "Estimating a dirichlet distribution," tech. rep., 2000.

[57] N. Li, W. Qardaji, and D. Su, "On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy," in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, pp. 32–33, ACM, 2012.

[58] M. Mohammady, L. Wang, Y. Hong, H. Louafi, M. Pourzandi, and M. Debbabi, "Preserving both privacy and utility in network trace anonymization," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018* (D. Lie, M. Mannan, M. Backes, and X. Wang, eds.), pp. 459–474, ACM, 2018.

[59] "https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm."

[60] X. Yi and Y. Zhang, "Equally contributory privacy-preserving k-means clustering over vertically partitioned data," *Inf. Syst.*, vol. 38, no. 1, pp. 97–107, 2013.

[61] L. Ou, Z. Qin, S. Liao, Y. Hong, and X. Jia, "Releasing correlated trajectories: Towards high utility and optimal differential privacy," *IEEE Trans. Dependable Secur. Comput.*, vol. 17, no. 5, pp. 1109–1123, 2020.

[62] F. McSherry and I. Mironov, "Differentially private recommender systems: Building privacy into the netflix prize contenders," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 627–636, ACM, 2009.

[63] H. Wang, S. Xie, and Y. Hong, "Videodp: A universal platform for video analytics with differential privacy," *CoRR*, vol. abs/1909.08729, 2019.

[64] H. Wang, Y. Hong, Y. Kong, and J. Vaidya, "Publishing video data with indistinguishable objects," in *Proceedings of the 23rd International Conference on Extending Database Technology, EDBT 2020, Copenhagen, Denmark, March 30 - April 02, 2020* (A. Bonifati, Y. Zhou, M. A. V. Salles, A. Böhm, D. Olteanu, G. H. L. Fletcher, A. Khan, and B. Yang, eds.), pp. 323–334, OpenProceedings.org, 2020.

[65] R. Bild, K. A. Kuhn, and F. Prasser, "Safepub: A truthful data anonymization algorithm with strong privacy guarantees," *Proceedings on Privacy Enhancing Technologies*, vol. 2018, no. 1, pp. 67–87, 2018.

[66] R. Shokri, C. Troncoso, C. Diaz, J. Freudiger, and J.-P. Hubaux, "Unraveling an old cloak: k-anonymity for location privacy," in *Proceedings of the 9th annual ACM workshop on Privacy in the electronic society*, pp. 115–118, ACM, 2010.

[67] H. Li, H. Zhu, S. Du, X. Liang, and X. S. Shen, "Privacy leakage of location sharing in mobile social networks: Attacks and defense," *IEEE Trans. Dependable Secur. Comput.*, vol. 15, no. 4, pp. 646–660, 2018.

[68] J. L. Ny, A. Touati, and G. J. Pappas, "Real-time privacy-preserving model-based estimation of traffic flows," *2014 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, pp. 92–102, 2014.

[69] F. Dötzer, "Privacy issues in vehicular ad hoc networks," in *Privacy Enhancing Technologies, 5th International Workshop, PET 2005, Cavtat, Croatia, May 30-June 1, 2005, Revised Selected Papers* (G. Danezis and D. M. M. Jr., eds.), vol. 3856 of *Lecture Notes in Computer Science*, pp. 197–209, Springer, 2005.

[70] K. Sampigethaya, M. Li, L. Huang, and R. Poovendran, "AMOEBA: robust location privacy scheme for VANET," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 8, pp. 1569–1589, 2007.

[71] B. Liu, S. Xie, and Y. Hong, "PANDA: privacy-aware double auction for divisible resources without a mediator," in *AAMAS*, pp. 1904–1906, 2020.

[72] D. An, Q. Yang, W. Yu, X. Yang, X. Fu, and W. Zhao, "SODA: strategy-proof online double auction scheme for multimicrogrids bidding," *IEEE Trans. Systems, Man, and Cybernetics: Systems*, vol. 48, no. 7, pp. 1177–1190, 2018.

[73] Q. Yang, D. An, W. Yu, X. Yang, and X. Fu, "On stochastic optimal bidding strategy for microgrids," in *34th IEEE International Performance Computing and Communications Conference, IPCCC 2015, Nanjing, China, December 14-16, 2015*, pp. 1–8, 2015.

[74] S. Xie, Y. Hong, and P. Wan, "A privacy preserving multiagent system for load balancing in the smart grid," in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, pp. 2273–2275, 2019.

[75] Y. Hong, H. Wang, S. Xie, and B. Liu, "Privacy preserving and collusion resistant energy sharing," in *ICASSP*, pp. 6941–6945, IEEE, 2018.

[76] Z. Zhao, F. Chen, T.-H. H. Chan, and C. Wu, "Double auction for resource allocation in cloud computing.," in *CLOSER*, pp. 273–280, 2017.

[77] G. Matvos and A. Seru, "Resource allocation within firms and financial market dislocation: Evidence from diversified conglomerates," *The Review of Financial Studies*, vol. 27, no. 4, pp. 1143–1189, 2014.

[78] Z. Feng, C. Qiu, Z. Feng, Z. Wei, W. Li, and P. Zhang, "An effective approach to 5g: Wireless network virtualization," *IEEE Communications Magazine*, vol. 53, no. 12, pp. 53–59, 2015.

[79] S. Parsons, J. A. Rodríguez-Aguilar, and M. Klein, "Auctions and bidding: A guide for computer scientists," *ACM Comput. Surv.*, vol. 43, no. 2, pp. 10:1–10:59, 2011.

[80] B. Liu, Y. Yang, R. Wang, and Y. Hong, "Poster: Privacy preserving divisible double auction with A hybridized tee-blockchain system," in *41st IEEE International Conference on Distributed Computing Systems, ICDCS 2021, Washington DC, USA, July 7-10, 2021*, pp. 1144–1145, IEEE, 2021.

[81] B. Liu, S. Xie, Y. Yang, R. Wang, and Y. Hong, "Privacy preserving divisible double auction with a hybridized tee-blockchain system," *Cybersecur.*, vol. 4, no. 1, p. 37, 2021.

[82] F. Brandt, T. Sandholm, and Y. Shoham, "Spiteful bidding in sealed-bid auctions," in *Proc. of IJCAI, 2007*, pp. 1207–1214.

[83] K. Wüst, L. Diana, K. Kostiainen, G. Karame, S. Matetic, and S. Capkun, "Bitcontracts: Adding expressive smart contracts to legacy cryptocurrencies," *IACR Cryptol.*, vol. 2019, p. 857.

[84] C. Tsai, D. E. Porter, and M. Vij, "Graphene-sgx: A practical library OS for unmodified applications on SGX," in *USENIX ATC 2017*, pp. 645–658.

[85] V. Costan and S. Devadas, "Intel SGX explained," *IACR Cryptol.*, vol. 2016, p. 86, 2016.

[86] E. Brickell and J. Li, "Enhanced privacy ID from bilinear pairing," *IACR Cryptol. ePrint Arch.*, vol. 2009, p. 95.

[87] M. S. Miller, C. Morningstar, and B. Frantz, "Capability-based financial instruments," in *Proc. FC 2000*, vol. 1962, pp. 349–378.

[88] N. Li and J. R. Marden, "Decoupling coupled constraints through utility design," *IEEE Trans. Autom. Control.*, 2014.

[89] F. Kojima and T. Yamashita, "Double auction with interdependent values: Incentives and efficiency," *Theoretical Economics*, vol. 12, no. 3, pp. 1393–1438, 2017.

[90] G. Brero, S. Lahaie, and S. Seuken, "Fast iterative combinatorial auctions via bayesian learning," in *AAAI, 2019*, pp. 1820–1828.

[91] R. Yuan, Y. Xia, H. Chen, B. Zang, and J. Xie, "Shadoweth: Private smart contract on public blockchain," *J. Comput. Sci. Technol.*, vol. 33, no. 3, pp. 542–556, 2018.

[92] E. Shi, F. Zhang, R. Pass, S. Devadas, D. Song, and C. Liu, "Trusted hardware: Life, the composable universe, and everything," *manuscript*, 2015.

[93] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in *FOCS, 2001*, pp. 136–145.

[94] Y. Wang, W. Saad, Z. Han, H. V. Poor, and T. Basar, "A game-theoretic approach to energy trading in the smart grid," *IEEE Trans. Smart Grid*, vol. 5, no. 3, pp. 1439–1450, 2014.

[95] A. Jin, W. Song, and W. Zhuang, "Auction-based resource allocation for sharing cloudlets in mobile cloud computing," *IEEE Trans. Emerging Topics Comput.*, vol. 6, no. 1, pp. 45–57, 2018.

[96] I. Fujiwara, K. Aida, and I. Ono, "Applying double-sided combinational auctions to resource allocation in cloud computing," in *Tenth Annual International Symposium on Applications and the Internet, SAINT 2010, Seoul, Korea, 19-23 July, 2010, Proceedings*, pp. 7–14, 2010.

[97] H. Kebriaei, B. Maham, and D. Niyato, "Double-sided bandwidth-auction game for cognitive device-to-device communication in cellular networks," *IEEE Trans. Vehicular Technology*, vol. 65, no. 9, pp. 7476–7487, 2016.

[98] D. E. Aliabadi, M. Kaya, and G. Şahin, "An agent-based simulation of power generation company behavior in electricity markets under different market-clearing mechanisms," *Energy Policy*, vol. 100, pp. 191–205, 2017.

[99] M. N. Faqiry and S. Das, "Double-sided energy auction in microgrid: Equilibrium under price anticipation," *IEEE Access*, vol. 4, pp. 3794–3805, 2016.

[100] E. Bompard, Y. Ma, R. Napoli, and G. Abrate, "The demand elasticity impacts on the strategic bidding behavior of the electricity producers," *IEEE Transactions on Power Systems*, vol. 22, pp. 188–197, Feb 2007.

[101] D. Zhang, Z. Chang, F. R. Yu, X. Chen, and T. Hämäläinen, "A double auction mechanism for virtual resource allocation in sdn-based cellular network," in *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–6, IEEE, 2016.

[102] C. Tsai, K. S. Arora, N. Bandi, B. Jain, W. Jannen, J. John, H. A. Kalodner, V. Kulkarni, D. A. S. de Oliveira, and D. E. Porter, "Cooperation and security isolation of library oses for multi-process applications," in *EuroSys*, pp. 9:1–9:14, 2014.

[103] S. Barker, A. Mishra, D. Irwin, P. Shenoy, and J. Albrecht, "SmartCap: Flattening peak electricity demand in smart homes," *PerCom 2012*.

[104] F. Wu and N. H. Vaidya, "SMALL: A strategy-proof mechanism for radio spectrum allocation," in *INFOCOM 2011. 30th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 10-15 April 2011, Shanghai, China*, pp. 81–85, 2011.

[105] M. Dong, G. Sun, X. Wang, and Q. Zhang, "Combinatorial auction with time-frequency flexibility in cognitive radio networks," in *Proceedings of the IEEE INFOCOM 2012, Orlando, FL, USA, March 25-30, 2012*, pp. 2282–2290, 2012.

[106] M. Hoefer, T. Kesselheim, and B. Vöcking, "Approximation algorithms for secondary spectrum auctions," *ACM Trans. Internet Techn.*, vol. 14, no. 2-3, pp. 16:1–16:24, 2014.

[107] J. Jia, Q. Zhang, Q. Zhang, and M. Liu, "Revenue generation for truthful spectrum auction in dynamic spectrum access," in *Proceedings of the 10th ACM Interational Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2009, New Orleans, LA, USA, May 18-21, 2009*, pp. 3–12, 2009.

[108] A. Gopinathan and Z. Li, "Strategyproof wireless spectrum auctions with interference," in *Proceedings of the Global Communications Conference, 2010. GLOBECOM 2010, 6-10 December 2010, Miami, Florida, USA*, pp. 1–5, 2010.

[109] Z. Chen, L. Huang, L. Li, W. Yang, H. Miao, M. Tian, and F. Wang, "PS-TRUST: provably secure solution for truthful double spectrum auctions," in *2014 IEEE Conference on Computer Communications, INFOCOM 2014, Toronto, Canada, April 27 - May 2, 2014*, pp. 1249–1257, 2014.

[110] H. Huang, X. Li, Y. Sun, H. Xu, and L. Huang, "PPS: privacy-preserving strategyproof social-efficient spectrum auction mechanisms," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 5, pp. 1393–1404, 2015.

[111] K. Suzuki and M. Yokoo, "Secure generalized vickrey auction using homomorphic encryption," in *Financial Cryptography, 7th International Conference, FC 2003, Guadeloupe, French West Indies, January 27-30, 2003, Revised Papers*, pp. 239–249, 2003.

[112] Q. Huang, Y. Tao, and F. Wu, "SPRING: A strategy-proof and privacy preserving spectrum auction mechanism," in *Proceedings of the IEEE INFOCOM 2013, Turin, Italy, April 14-19, 2013*, pp. 827–835, 2013.

[113] R. Cheng, F. Zhang, J. Kos, W. He, N. Hynes, N. M. Johnson, A. Juels, A. Miller, and D. Song, "Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts," in *IEEE EuroS&P*, pp. 185–200, 2019.

[114] A. E. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *IEEE S&P*, pp. 839–858, 2016.

[115] F. Zhang, E. Cecchetti, K. Croman, A. Juels, and E. Shi, "Town crier: An authenticated data feed for smart contracts," in *ACM CCS*, pp. 270–282.

[116] 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, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017* (B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, eds.), pp. 619–631, ACM, 2017.

[117] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," *ACM Trans. Comput. Theory*, vol. 6, no. 3, pp. 13:1–13:36, 2014.

[118] L. Hanzlik, Y. Zhang, K. Grosse, A. Salem, M. Augustin, M. Backes, and M. Fritz, "Mlcapsule: Guarded offline deployment of machine learning as a service," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2021, virtual, June 19-25, 2021*, pp. 3300–3309, Computer Vision Foundation / IEEE, 2021.

[119] S. Tople, K. Grover, S. Shinde, R. Bhagwan, and R. Ramjee, "Privado: Practical and secure DNN inference," *CoRR*, vol. abs/1810.00602, 2018.

[120] F. Tramèr and D. Boneh, "Slalom: Fast, verifiable and private execution of neural networks in trusted hardware," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, OpenReview.net, 2019.

[121] R. Poddar, G. Ananthanarayanan, S. Setty, S. Volos, and R. A. Popa, "Visor: Privacy-preserving video analytics as a cloud service," in *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020* (S. Capkun and F. Roesner, eds.), pp. 1039–1056, USENIX Association, 2020.

[122] P. Mishra, R. Lehmkuhl, A. Srinivasan, W. Zheng, and R. A. Popa, "Delphi: A cryptographic inference service for neural networks," in *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020* (S. Capkun and F. Roesner, eds.), pp. 2505–2522, USENIX Association, 2020.

[123] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "GAZELLE: A low latency framework for secure neural network inference," in *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018* (W. Enck and A. P. Felt, eds.), pp. 1651–1669, USENIX Association, 2018.

[124] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pp. 4489–4497, IEEE Computer Society, 2015.

[125] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 4724–4733, IEEE Computer Society, 2017.

[126] S. Pentyala, R. Dowsley, and M. D. Cock, "Privacy-preserving video classification with convolutional neural networks," in *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 8487–8499, PMLR, 2021.

[127] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," *CoRR*, vol. abs/1212.0402, 2012.

[128] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pp. 1725–1732, IEEE Computer Society.

[129] T. E. Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. 31, no. 4, pp. 469–472, 1985.

[130] D. Beaver, "Efficient multiparty protocols using circuit randomization," in *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings* (J. Feigenbaum, ed.), vol. 576 of *Lecture Notes in Computer Science*, pp. 420–432, Springer.

[131] M. O. Rabin, "How to exchange secrets with oblivious transfer," *IACR Cryptol. ePrint Arch.*, p. 187, 2005.

[132] X. Jiang, M. Kim, K. E. Lauter, and Y. Song, "Secure outsourced matrix computation and application to neural networks," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018* (D. Lie, M. Mannan, M. Backes, and X. Wang, eds.), pp. 1209–1222, ACM, 2018.

[133] J. H. Cheon, A. Kim, M. Kim, and Y. S. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I* (T. Takagi and T. Peyrin, eds.), vol. 10624 of *Lecture Notes in Computer Science*, pp. 409–437, Springer, 2017.

[134] H. Kuehne, H. Jhuang, E. Garrote, T. A. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," in *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011* (D. N. Metaxas, L. Quan, A. Sanfeliu, and L. V. Gool, eds.), pp. 2556–2563, IEEE Computer Society, 2011.

[135] S. R. Livingstone and F. A. Russo, "The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english," *PloS one*, vol. 13, no. 5, p. e0196391, 2018.

[136] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. E. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016* (M. Balcan and K. Q. Weinberger, eds.), vol. 48 of *JMLR Workshop and Conference Proceedings*, pp. 201–210, JMLR.org, 2016.

[137] E. Hesamifard, H. Takabi, M. Ghasemi, and R. N. Wright, "Privacy-preserving machine learning as a service," *Proc. Priv. Enhancing Technol.*, vol. 2018, no. 3, pp. 123–142, 2018.

[138] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015* (I. Ray, N. Li, and C. Kruegel, eds.), pp. 1310–1321, ACM, 2015.

[139] N. Phan, X. Wu, H. Hu, and D. Dou, "Adaptive laplace mechanism: Differential privacy preservation in deep learning," in *2017 IEEE International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, November 18-21, 2017* (V. Raghavan, S. Aluru, G. Karypis, L. Miele, and X. Wu, eds.), pp. 385–394, IEEE Computer Society, 2017.

[140] L. Yu, L. Liu, C. Pu, M. E. Gursoy, and S. Truex, "Differentially private model publishing for deep learning," in *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pp. 332–349, IEEE, 2019.

[141] H. Zhou and G. W. Wornell, "Efficient homomorphic encryption on integer vectors and its applications," in *2014 Information Theory and Applications Workshop, ITA 2014, San Diego, CA, USA, February 9-14, 2014*, pp. 1–9, IEEE, 2014.

[142] M. Ball, B. Carmer, T. Malkin, M. Rosulek, and N. Schimanski, "Garbled neural networks are practical," *IACR Cryptol. ePrint Arch.*, p. 338, 2019.

[143] M. S. Riazi, M. Samragh, H. Chen, K. Laine, K. E. Lauter, and F. Koushanfar, "XONN: xnor-based oblivious deep neural network inference," in *28th USENIX Security Symposium, USENIX Security 2019, Santa Clara, CA, USA, August*

14-16, 2019 (N. Heninger and P. Traynor, eds.), pp. 1501–1518, USENIX Association, 2019.

[144] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, "Deepsecure: scalable provably-secure deep learning," in *Proceedings of the 55th Annual Design Automation Conference, DAC 2018, San Francisco, CA, USA, June 24-29, 2018*, pp. 2:1–2:6, ACM, 2018.

[145] K. A. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," 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.), pp. 1175–1191, ACM, 2017.

[146] M. Yamane and K. Iwamura, "Secure and efficient outsourcing of matrix multiplication based on secret sharing scheme using only one server," in *IEEE 17th Annual Consumer Communications & Networking Conference, CCNC 2020, Las Vegas, NV, USA, January 10-13, 2020*, pp. 1–6, IEEE, 2020.

[147] 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* (R. Safavi-Naini and R. Canetti, eds.), vol. 7417 of *Lecture Notes in Computer Science*, pp. 643–662, Springer, 2012.

[148] P. Mohassel and P. Rindal, "Aby^3: A mixed protocol framework for machine learning," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018* (D. Lie, M. Mannan, M. Backes, and X. Wang, eds.), pp. 35–52, ACM, 2018.

[149] M. S. Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar, "Chameleon: A hybrid secure computation framework for machine learning applications," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security, AsiaCCS 2018, Incheon, Republic of Korea, June 04-08, 2018* (J. Kim, G. Ahn, S. Kim, Y. Kim, J. López, and T. Kim, eds.), pp. 707–721, ACM, 2018.

[150] J. Halpern and V. Teague, "Rational secret sharing and multiparty computation," in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pp. 623–632, 2004.

[151] 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*, pp. 19–38, IEEE Computer Society, 2017.

[152] X. Chen, J. Ji, L. Yu, C. Luo, and P. Li, "Securenets: Secure inference of deep neural networks on an untrusted cloud," in *Proceedings of The 10th Asian Conference on Machine Learning, ACML 2018, Beijing, China, November 14-16, 2018* (J. Zhu and I. Takeuchi, eds.), vol. 95 of *Proceedings of Machine Learning Research*, pp. 646–661, PMLR, 2018.

[153] J. Keuffer, R. Molva, and H. Chabanne, "Efficient proof composition for verifiable computation," in *Computer Security - 23rd European Symposium on Research in Computer Security, ESORICS 2018, Barcelona, Spain, September 3-7, 2018, Proceedings, Part I* (J. López, J. Zhou, and M. Soriano, eds.), vol. 11098 of *Lecture Notes in Computer Science*, pp. 152–171, Springer, 2018.

[154] Z. Ghodsi, T. Gu, and S. Garg, "Safetynets: Verifiable execution of deep neural networks on an untrusted cloud," 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.), pp. 4672–4681, 2017.

[155] D. Wang, M. Ye, and J. Xu, "Differentially private empirical risk minimization revisited: Faster and more general," *CoRR*, vol. abs/1802.05251, 2018.