Revisiting Location Privacy in MEC-Enabled Computation Offloading

Jingyi Li, Wenzhong Ou, Bei Ouyang, Shengyuan Ye, *Graduate Student Member, IEEE*, Liekang Zeng, Lin Chen, *Member, IEEE*, and Xu Chen, *Senior Member, IEEE*

***Abstract*—Mobile Edge Computing (MEC) revolutionizes real-time applications by extending cloud capabilities to network edges, enabling efficient computation offloading from mobile devices. In recent years, the location privacy concern within MEC offloading has been recognized, prompting the proposal of various methodologies to mitigate this concern. However, this paper demonstrates that the prevailing privacy protection methods exhibit vulnerabilities. First, we analyze the shortcomings of current methodologies through both system modeling and evaluation metrics. Then, we introduce a Learning-based Trajectory Reconstruction Attack (LTRA) to expose the weaknesses, achieving up to 91.2% reconstruction accuracy against the state-of-the-art protection method. Further, based on w-event differential privacy, we propose an ℓ-trajectory differentially private mechanism, i.e., OffloadingBD. Compared to the existing works, OffloadingBD provides more flexible and enhanced protection with sound privacy theoretical guarantee. Lastly, we conduct extensive experiments to evaluate LTRA and OffloadingBD. The experiment results show that LTRA has good generalization ability and OffloadingBD showcases a superior balance between privacy and utility compared with baselines.**

***Index Terms*—Edge computing, privacy-preserving computation offloading.**

I. Introduction

MOBILE Edge Computing (MEC) has become a promising paradigm for enabling real-time and low-latency applications, e.g., Virtual Reality (VR), Telemedicine, and Internet of Vehicles (IoV), in mobile networks [1], [2]. It extends cloud computing capabilities to the edge of the network, enabling data processing and analyzing to be performed close to mobile devices. Based on MEC, computation offloading is widely studied [3], [4]. It is a process of delegating computation-intensive tasks from mobile devices to nearby edge servers to optimize their performance, battery life, energy efficiency, etc.

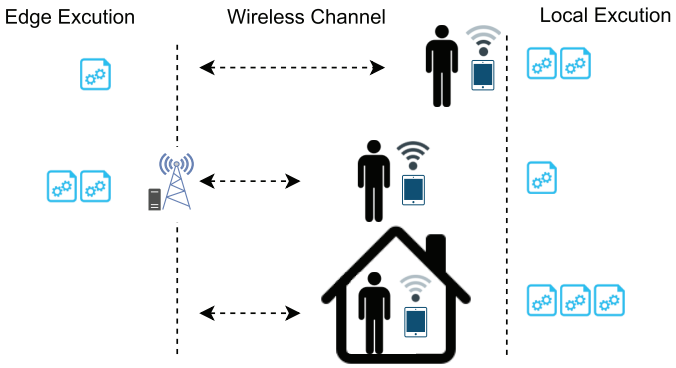


Fig. 1. Schematic diagram of location privacy risks in edge offloading. The quality of wireless channel conditions leads to different optimal offloading decisions for mobile users. The wireless channel condition is closely related to distance and other geo-information. Therefore, an edge server with prior environmental knowledge may compromise the users' location privacy.

Furthermore, privacy protection is one of the critical topics in today's digital era. For the first time, He et al. revealed that the offloading decisions of a user may expose the location information. The privacy threats are described in Fig 1. They designed an online algorithm to perturb the offloading decisions [5]. Subsequently, a cascade of works, e.g., [6], [7], [8], [9], built upon this seminal work, with several studies focusing on devising more sophisticated models and optimization algorithms to strike optimal balances between privacy protection and other performance measures such as energy efficiency, latency, and throughput. These collective efforts have significantly advanced the field in terms of both theoretical modeling and practical algorithmic development. However, we reveal the vulnerabilities of prevailing privacy protection methods

To begin with, we revisit the privacy-preserving methods in MEC offloading. We analyze why existing methods are vulnerable to attacks and why this vulnerability has not been identified. First, we summarize and generalize the modeling of existing works. We find that the constraints in the existing

Received 11 May 2024; revised 14 November 2024 and 11 March 2025; accepted 31 March 2025. Date of publication 10 April 2025; date of current version 1 May 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 42222106, in part by Guangdong Basic and Applied Basic Research Foundation under Grant 2023B1515120058, in part by Guangzhou Basic and Applied Basic Research Program under Grant 2024A04J6367, and in part by the Program for Guangdong Introducing Innovative and Entrepreneurial Teams under Grant 2017ZT07X355. The work of Lin Chen was supported in part by the National Natural Science Foundation of China under Grant 62172455, in part by the Department of Science and Technology of Guangdong Province under Grant 2022A0505050028, and in part by the Pearl River Talent Program under Grant 2019QN01X140. The associate editor coordinating the review of this article and approving it for publication was Prof. Albert Levi. *(Corresponding authors: Xu Chen; Lin Chen.)*

Jingyi Li, Wenzhong Ou, Bei Ouyang, Shengyuan Ye, and Xu Chen are with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510275, China (e-mail: lijy573@mail2.sysu.edu.cn; ouwzh3@mail2.sysu.edu.cn; ouyb9@mail2.sysu.edu.cn; yeshy8@mail2.sysu.edu.cn; chenxu35@mail.sysu.edu.cn).

Liekang Zeng is with the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong, SAR, China (e-mail: lkzeng@cuhk.edu.hk).

Lin Chen is with the Engineering Research Centre of Applied Technology on Machine Translation and Artificial Intelligence, Macao Polytechnic University, Macau, SAR, China (e-mail: lchen@mpu.edu.mo).

Digital Object Identifier 10.1109/TIFS.2025.3558593

works neglect correlation among points on the offloading trajectory, which leads the proposed event-level perturbation scheme, e.g., injecting random noises into the offloading ratio at every single point without considering higher-level formal privacy guarantees, provides limited protection for users' location privacy. Second, our empirical findings notably contradict earlier assertions made in works like [5] and [10], where it was hypothesized that consistent task size patterns during offloading could render the system vulnerable to inference attacks. Based on the hypothesis, they use the difference between the number of offloaded tasks and generated tasks as the metric to evaluate the effectiveness of their framework on inference attacks. We reveal that this metric is invalid for measuring the defense capability of inference attacks.

Further, we devise a learning-based inference attack. *Our intuition is that a mobile user moves within the coverage of the edge server and forms an offloading trajectory. For the trajectory, multiple single-point perturbations may have little impact on the overall trajectory or a set of successive time slots.* To validate the weakness of event-level perturbation methods, we propose a Learning-based Trajectory Reconstruction Attack (LTRA). We build Seq2Seq models to transfer the offloading pattern into the predicted channel state pattern. We evaluate LTRA by targeting a state-of-the-art (SOTA) single-server privacy-preserving offloading approach, i.e., OffloadingGuard [11]. The evaluation results show that the learning-based approach can achieve a maximum reconstruction accuracy of 91.2%, which means the event-level perturbation methods do not preserve location privacy very well.

What is more, we refine privacy-preserving offloading with an ℓ-trajectory differential privacy mechanism. We propose OffloadingBD, a budget distribution mechanism inspired by w-event differential privacy. The mechanism ensures ϵ-differential privacy for the offloading ratio trajectory within any ℓ-length window, suitable for MEC-enabled computation offloading in infinite streams. Evaluation results affirm that OffloadingBD outperforms event-level and user-level mechanisms in striking a balance between privacy and utility. It is worth noting that even some event-level-based existing works define a form of a total budget over the whole trajectory, but the privacy guarantee they provided is still the event-level [10], [11]. Then, we verify that the proposed OffloadingBD is able to provide a better privacy-utility balance in terms of average system cost and reconstruction accuracy experiment.

The contributions of this paper are summarized as follows:

- We expose vulnerability in prevailing event-level perturbation techniques for safeguarding user location privacy in MEC-enabled offloading systems. We revisit privacy-preserving methods in MEC offloading, highlighting vulnerabilities due to overlooked correlations among trajectory points and pointing out that existing event-level perturbation schemes lack robust privacy guarantees. Our empirical findings challenge assertions that consistent task size patterns render systems vulnerable to inference attacks. Metrics like task discrepancy fail to accurately assess defense against such attacks.
- We propose a learning-based trajectory reconstruction attack (LTRA). LTRA is build upon Seq2Seq models and is able to map the offloaded tasks trajectory to a normalized movement (bandwidth) pattern. The experimental outcomes reveal that our learning-based attack attains a peak reconstruction accuracy of 91.2% against SOTA on location privacy-preserving MEC offloading.
- We further refine privacy-aware offloading by introducing an ℓ-trajectory differential privacy mechanism, i.e. OffloadingBD. Unlike the existing event-level privacy guarantees, OffloadingBD provides a ℓ-trajectory ϵ-differential privacy, i,e, ℓ-trajectory privacy. It outperforms event-level privacy mechanisms by offering stronger privacy preservation while inflicting lesser damage to data utility compared to user-level mechanisms. Notably, its independence from specific data patterns enables the application of infinite trajectories, making it a more adaptable and suitable choice for diverse scenarios.

The rest of the paper is organized as follows. We begin by delving into the background of the study in Sec. II. Then, we revisit privacy-preserving MEC offloading in Sec. III. Following that, we articulate the attack methods in Sec. IV. Subsequently, we introduce our proposed enhanced defense mechanisms in Sec. V. Afterward, we proceed to the evaluation section in Sec. VI. Finally, we conclude our paper and discuss the future work in Sec. VIII.

II. Background

A. Cloud and Edge Offloading

The evolution of distributed computing has witnessed a profound shift from the centralized paradigm of cloud computing to a more decentralized model, prominently featuring edge computing. Initially, cloud computing revolutionized the landscape by offering a remotely accessible, seemingly infinite pool of computational resources and storage capacities [15]. Applications could offload their heavy processing loads and data storage needs to distant cloud servers, relieving end-user devices of these burdens and enabling scalability unseen before [16]. However, as technology advanced and the demand for real-time processing escalated, the limitations of cloud-centric architectures became evident, particularly concerning latency and bandwidth constraints. This gave rise to the advent of edge computing, which extends the cloud's capabilities to the periphery of the network, closer to where data is generated and consumed [17].

Edge offloading, a cornerstone of this new paradigm, redefines how computational tasks are managed. By delegating processing responsibilities to edge devices or nearby servers, it significantly reduces latency, enhancing responsiveness and user experience [18]. This approach capitalizes on the spatial distribution of edge resources to minimize network hops and data transmission delays, making it particularly suited for latency-sensitive applications like autonomous vehicles, augmented reality, and industrial IoT [19].

B. Privacy-Preserving Edge Offloading

Several studies [20], [21], [22] have acknowledged that user privacy can be compromised when offloading data during the computation process. Then, He et al. further pointed out

TABLE I
SUMMARY OF EXISTING WORK

Paper	Privacy guarantee	Threat model	Task generation	Network Setup	Algorithm
[12]	Event-level (ϵ, δ)-DP	Honest-but-curious	Uniform	Simulation	RL
[10]	Event-level (ϵ, δ)-DP	Honest-but-curious	Uniform	Simulation	Genetic algorithm
[11]	Event-level ϵ-DP	Honest-but-curious	Uniform	Simulation	RL
[6]	–	Honest-but-curious	Uniform	WiFi	RL
[8]	–	–	Normal	Simulation	RL
[7]	–	–	Fixed	Simulation	RL
[5], [13], [14], [8]	–	–	Uniform	Simulation	RL
Ours	ℓ-trajectory privacy	Refined honest-but-curious	Uniform	WiFi/5G/Simulation	OffloadingBD

that user location can be inferred through his task offloading strategy and proposed a Q-learning base method to jointly optimize privacy and other metrics [5]. Following that, related works were springing up and most of them used RL-(RL)-based schemes [6], [7], [8], [10], [11], [14]. For instance, Lan et al. considered computing offloading in a satellite scenario and proposed a Proximal Policy Optimization (PPO) algorithm to optimize the offloading policy [14]. Nguyen et al. modeled a blockchain system and a temporal difference (TD) approach is employed to allow the agent to learn offloading policies without requiring the state transition probability.

None of the above works have any privacy guarantees. Then, Pang et al proposed OffloadGuard and gave a ϵ-differential privacy guarantee [11]. However, even though OffloadingGuard has a formal total privacy constraint on the entire trajectory, its total privacy budget is obtained through parallel composition instead of sequential composition. Hence, there is only one event-level guarantee. The guarantee is weak for a trajectory [23]. More recently, Wang et al. considered the relaxed differential privacy guarantee in the scenario of multiple servers [10]. Then, followed by [10], [12] and [24] provide a similar argument. However, these guarantees are event-level yet.

We summarize the existing work in TABLE I.

C. Differential Privacy

Differential privacy is a concept in the field of privacy-preserving data analysis and data mining. It provides a rigorous mathematical framework for quantifying and ensuring the privacy of individuals whose data is used for statistical analysis while allowing useful information to be extracted from the data. The following is the formal definition of event-level (user-level) ϵ-differential privacy of streaming data.

Definition 1 [23]: (event-level (user-level) ϵ-differential privacy of streaming data) Let Λ be a mechanism that takes as input a stream prefix of arbitrary size. Let $\mathcal{O}$ be the set of all possible outputs of Λ. Then, Λ is event-level (user-level) ϵ-differentially private if for all sets $O \subseteq \mathcal{O}$, all event-level (resp. user-level) adjacent stream prefixes $S(t)$, $S'(t)$, and all t, it holds that

$$\Pr[\Lambda(S(t)) \in O] \leq e^{\epsilon} \cdot \Pr[\Lambda(S'(t)) \in O]. \tag{1}$$

In other words, ϵ-differential privacy ensures that the presence or absence of any individual in the dataset has a limited impact on the output of the algorithm, thus providing strong privacy guarantees for individuals (events or users) in the dataset. And, the parameter ϵ is the so-called privacy budget.

The Laplace mechanism is one of the fundamental techniques used to achieve differential privacy. It is employed to add controlled noise to the output of a function, such as a query or statistical analysis, in order to preserve privacy. Suppose we have a function f that takes a dataset as input and produces a real-valued output. To make f differentially private, we can add Laplace noise to its output. The amount of noise added depends on the sensitivity of f to changes in the input dataset and the desired privacy level ϵ. Given a dataset D, the Laplace noise η is drawn from a Laplace distribution with scale $\frac{\Delta f}{\epsilon}$, and the output of the differentially private function $\tilde{f}(D)$ is calculated as:

$$\tilde{f}(D) = f(D) + \text{Lap}\left(\frac{\Delta f}{\epsilon}\right). \tag{2}$$

Laplace mechanism satisfies ϵ-differential privacy. It is worth noting that a modified Laplace mechanism was developed in [11] to force the output of the mechanism within $[0, 1]$ but still holds ϵ-differential privacy. We denote the mechanism as MLap.

D. Threat Model

Existing literature implicitly assumes that the edge computing service provider and the base station operator are separate entities. We emphasize that this assumption should be explicit, otherwise, the base station manager/operator could directly obtain user location/distance information through channel conditions or other easily accessible means, rendering the problem meaningless. The main privacy threat we consider is that the server can access the amount of the offloaded task at each time slot and then extract the location privacy from the offloading trajectory. Meanwhile, we assume that the attacker (server) can not access the total amount of the task at each time slot.

Further, both single-server models and multi-server models can be found in existing works. If multiple servers are owned by the same entity, implying interactive information between them is allowed, it is equivalent to having multiple anchors for localization. In such cases, multi-server models are more prone to violating user location privacy compared to single-server models. The simulation results in [10] also support this hypothesis. If multiple servers are owned by different entities, there are much more spaces for the user to devise the privacy enhancement strategy. In this paper, our primary focus is on the single-server model.

III. Revisiting Privacy Protection in MEC Offloading

In this section, we delve into the reasons behind the susceptibility of existing methodologies. First, we embark on modeling and meticulously scrutinize the constraints within its optimization framework, revealing inherent limitations. Furthermore, our examination of real-world systems has unveiled a fundamental flaw: the metric employed to gauge resilience against interference attacks in existing approaches is inherently flawed, fostering a deceptive sense of protection efficacy.

A. Modeling

In this section, we summarize and generalize the modeling forms in existing work, and analyze their limitations.

The specific modeling in different existing works is varied. Some consider the equal-length discrete local tasks modeling and the decisions are the count of tasks to offload, e.g., [5], [10], and some build a continuous task model and the decisions are offloading ratios, e.g., [11]. Nevertheless, the core principle behind these models is consistent. Discrete modeling can be seen as a special case of continuous modeling. Consider that there is an edge server and a mobile user. Both the server and the user are equipped with computing resources. To minimize cost, a mobile user can offload part of their task to the MEC servers via wireless channel for computation and complete the remaining tasks locally. The user's offloading strategy typically involves determining the appropriate balance between offloading tasks to the MEC server and completing them locally. The key factor in this decision is the offloading ratio of the size of generated tasks. Further, adding noise to the offloading ratio is allowed. We summarize the cost and privacy modeling as follows.

At each time slot t, a computation task with the size of s_t (in bits) is generated at the mobile device of a user. A user can access a MEC server, and the offloading ratio of the total task of the user on the edge servers $\alpha_t \in [0, 1]$. The task executed on the user's local device and edge server is $s_t^e = s_t\alpha_t$ and $s_t^l = s_t(1-\alpha_t)$, respectively. Further, the offloading ratios are allowed to be perturbed to preserve privacy. A metric called Pattern Chaos (PC) [5], [6], [11] quantifying the difference between the user's optimal offloading ratio α_t^* and the perturbed offloading ratio $\overline{\alpha}_t$. Mathematically, PC can be defined as:

$$PC_t = \left|\alpha_t^* - \overline{\alpha}_t\right|. \tag{3}$$

The objective of these works is to enable efficient task offloading to the MEC servers while ensuring user privacy and minimal computation costs. In the offloading process, which spans T time periods, the goal is to minimize total computation cost for each user's tasks, while ensuring that a pre-defined total privacy protection level of each user meets their privacy requirement Γ., e.g.,

$$\min_{\alpha_t} \quad C = \sum_{t=1}^{T} C_t \tag{4}$$

$$\text{s.t.} \quad \alpha_t \in [0, 1], \tag{4a}$$

$$\sum_{t=1}^{T} PC_t \geq \Gamma, \tag{4b}$$

where C is the total cost function and C_t is the cost function in time slot t. Generally, the cost functions take into account performance factors such as latency, energy consumption costs, etc. Further, one assumption is that the optimal solution of (4), $\alpha^*(t) \propto d_{sd}$, where d_{sd} is the geographical distance from the server to the user.

Most existing works model the movement of the user as a Markov Decision Process (MDP) and solve (4) by RL-based methods. In particular, the SOTA work in a single server setting, i.e., OffloadingGuard [11], provides ϵ-event-level differential privacy by adding a modified Laplacian noise to increase PC_t.

Constraint (4b) merely specifies the total amount of noise added along the trajectory but does not impose any restrictions on how the noise is distributed throughout the trajectory, which can be seen in Fig. 3 and Fig 4. We point out that this is limiting, potentially leading to a situation where the addition of noise fails to achieve the desired privacy protection. Specificly, this kind of scheme has two significant drawbacks:

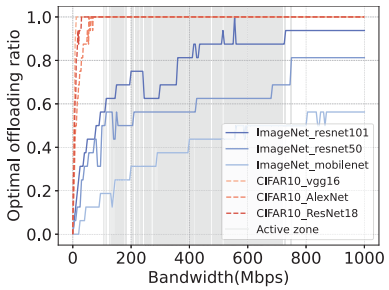
- The schemes are trajectory-oblivious. They only consider how to minimize the cost of the system given an upper privacy loss limit, but do not take into account the distribution of the added noises over the time slots. At some slots, the agent adds very large noise, but at some other slots, the agent may do nothing.
- Except [10], [11], [12], other works do not provide any provable theoretical privacy guarantees. Also, [10], [11], [12] derive differential privacy guarantees for every single point, but they apply parallel composition instead of sequential composition [25]. In other words, they are event-level differential privacy guarantees. For trajectory data with location privacy concerns, event-level guarantees can be weak [23].

These two drawbacks lead us to get the intuition that a mobile user moves within the coverage of the edge server and forms an offloading trajectory. For the trajectory, multiple single-point position perturbations may have little impact on the overall trajectory or a set of successive time slots.

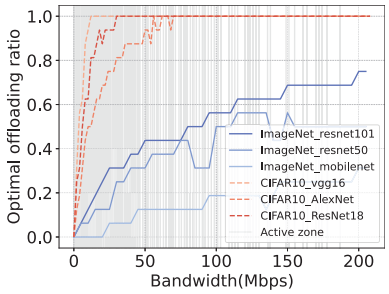
B. Emprical Study

In this subsection, we build a practical system and show that the metric used in the existing works to measure the invulnerability of inference attacks is not appropriate.

First of all, we introduce our experiment configuration. We use Dell t5820 as our edge server, which is equipped with Intel(R) Xeon(R) W-2145 CPU @ 3.70GHz, NVIDIA 3090Ti GPU and 32GB RAM. We use Jetson Xavier NX as our mobile device. We have set up two types of networks. For 5G, we collect data around a standard commercial 5G base station with coordinates (1**.*****4, 3*.*****1)(Anonymized). For WiFi, we use a Netgear R9000 for signal transmission and reception. The task we deploy is edge-device collaborative convolutional neural network inference. In each time slot, a batch of data inference requests is generated during movement. Users have the option to split their batch data into two sub-batches, performing one locally and sending the other to the edge server for execution. Specifically, we deploy MobileNet



(a) Tested under 5G.



(b) Tested under WiFi.

Fig. 2. Relationship between optimal offloading ratio and bandwidth. The active zone (marked in gray) encapsulates the fluctuating bandwidth values experienced by the mobile device throughout its movement.

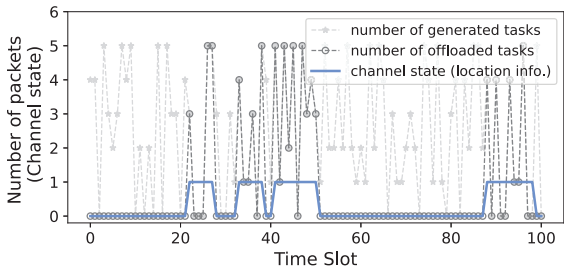


Fig. 3. Optimal privacy-oblivious offloading scheduling.

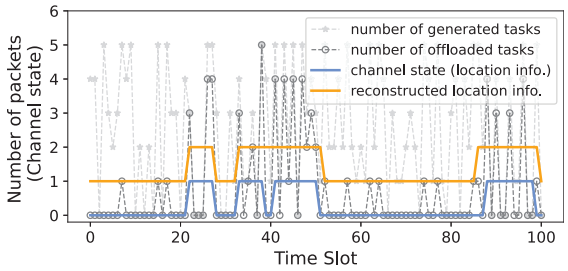


Fig. 4. Privacy-aware offloading scheduling in [5].

V2, ResNet50, ResNet101 for ImageNet, VGG16, AlexNet, and ResNet18 for CIFAR-10. The cost we considered is the total execution delay.

Moreover, we noticed that in some existing works, i.g., [5], [10], evaluate their methods against inference attacks. Our question is why their experimental results demonstrate excellent defense against inference attacks yet remain vulnerable to our attack. It is said that if the offloaded tasks or the number of offloaded tasks and generated tasks consistently approximate a specific value in a fixed period, it indicates that the offloading process is susceptible to inference attacks and uses this conclusion to build privacy leakage metrics, which is called **task discrepancy**. The quantitative description of the metrics *TD*:

$$TD = \sum_{t=1}^{T} \left| (N_t^{Off} - N_t^{Gen} \right|, \tag{5}$$

where N_t^{Off} and N_t^{Gen} refers to the number of offloaded tasks and generated tasks at time slot t. In our experiment, it is observed that all intersection points between the active zone and the optimal offloading curve share identical values (1.0), as exemplified by the curve corresponding to CIFAR10 in Figure 2(a). This implies that every location in the user's trajectory during this period experiences excellent network speed, and the low communication overhead makes it cost-effective to offload all tasks at any position along this trajectory. In such scenarios, users can achieve optimal offloading without sacrificing any privacy. The experimental results above demonstrate the unreliability of the metrics. In numerous cases, the entire offloading decision is almost exclusively in a none-or-all situation. On the other hand, this metric does not take the relationship between consecutive time slots. Under such circumstances, using the TD as an indicator for privacy leakage is also not applicable, which can be seen in Figs. 3 and 4. We highly recommend using more reasonable metrics for evaluating vulnerability. The metric should: 1) Account for the sequential nature of offloading decisions and their correlation over time. This can be achieved by analyzing the offloading trajectory over sliding windows of time slots. 2) Directly quantify the amount of location information that can be inferred from the offloading trajectory. This can be achieved by using attack success rates as a proxy for privacy leakage.

IV. Attacking Method

In this section, to reveal the vulnerability of the scheme, we give a motivating example that we can reconstruct the channel state trajectory even by direct observation. Further, we propose a learning-based trajectory reconstruction attack (LTRA), which is able to map the offloaded tasks trajectory to a normalized movement (bandwidth) pattern.

A. A Motivating Example

The goal of this subsection is to raise a motivating example to show the vulnerability of the event-level perturbation-based scheme. For simplicity and without loss of generality, we reproduce the results of the first paper of this series work, i.e., [5] wherein they model the task of a user's local device s_t as discrete equal-sized packages. The number of generated tasks is uniformly random in $[1, 5]$ at each time slot. The decision made by the user at each time slot is the number of offloading tasks out of the maximum total number of 5. The reproduced results are shown in Fig. 3 and 4. The channel state equal to 1 means the channel state performance is good and equal to 0 is the opposite. While we use [5] as a motivating example to illustrate the problems and introduce our attack, we also discuss other works in Sec. VI.

Specifically, Fig. 3 (Fig. 2 of [5]) is the optimal privacy-oblivious offloading scheduling. It can be seen that the channel state can be easily inferred by comparing the number of offloaded tasks in different time slots because the offloading decision is highly related to the channel state. Recall that the channel state contains information about the distance between the user and the edge server. Thus, user location privacy poses a risk of leakage. Based on the above observation, they take an event-level perturbation method into account and design a Q-learning-based algorithm to pursue a better utility-privacy trade-off. Also, we reproduce the improved results by the proposed algorithm in Fig. 4 (Fig. 4 of [5], but the orange line is additional). We can see that the pattern of the number of offloaded tasks (red dotted line) is more chaotic than privacy-oblivious scheduling.

However, we find that it is insecure yet. Even though a lot of points have been perturbed, one can observe the whole offloading trajectory and induce the binary channel state. We can recover the channel state trajectory by a two-step method: 1) *Thresholding*. Let the time stamp whose number of offloaded tasks is greater than 1 map to the good channel state, and that is less than or equal to 1 map to the bad channel state. 2) *Smoothing*. When a certain state only lasts for less than or equal to 2 time slots, set the corresponding time slot to another state.

The recovered channel state trajectory is shown in Fig. 3 (orange line). It turns out that the shape of the orange line is almost the same as the light green line. Further, following the settings in [5], we repeat the test 100 times, and our **average reconstruction error**, measured by the number of timestamps of the correct output over the total number of timestamps, is **only 3.1%**. It means that this event-level protection does not achieve the desired privacy protection.

B. Learning-Based Trajectory Reconstruction Attack

The motivating example reveals the vulnerability of the event-level perturbation-based scheme preliminarily, but may not be sufficient to support the conclusion. The model and parameters in [5] are special and the threshold and length of the smoothing window in the reconstruction method above are to be set manually. Hence, we will promote a more general learning-based trajectory reconstruction attack (LTRA). The MEC server can first simulate users' movement in its coverage to generate distance and offloaded task sequence pair as training data. Then it can train a Seq2Seq model to build the mapping from received offloaded tasks to wireless communication bandwidth pattern. Finally, the server would use the well-trained model to launch a reconstruction attack.

Technically, we collect M pairs of T-length input and output sequences $\{\mathbf{x}_i, \mathbf{b}_i\}$, $i \in [M]$. Here, $x_i \in [0, s_t]^T$ is the sequence of offloaded task sizes and $\mathbf{b}_i \in \mathbb{R}^T$ is the sequence of the distances between the base station (WiFi AP) and the mobile device. Our aim is not to meticulously reconstruct precise distances based on offloading patterns; rather, our focus lies in normalized bandwidth pattern. Thus, we perform the following normalization operation on $\mathbf{b}_i$:

$$\bar{x}_{ij} = \frac{x_{ij}}{\max \mathbf{x}_i}, \quad \forall i \in [N], \ j \in [T], \tag{6}$$

Then, our aim is to train a parameterized Seq2Seq model $\pi(\cdot;\theta)$, which can be used to map the offloading pattern to discrete distance information. Here, θ represents trainable parameters.

It is crucial to emphasize that the normalization process in our learning-based methodology holds significant importance. Given the substantial variations in server and mobile device capabilities, coupled with the disparate communication latencies across different systems, establishing a precise correlation between offloading volume and bandwidth without exhaustive scenario-specific training data is unfeasible. Nevertheless, through normalization, our model acquires the capability to discern the inherent, relative connection between distance and bandwidth within the confines of a consistent motion path. This enhancement enables a more nuanced understanding of the dynamics at play, transcending the need for exhaustive, individualized data collection for every potential scenario.

One consideration is whether the normalized and discretized reconstructed pattern is indeed a threat for location privacy. However, upon closer examination, even if such a pattern cannot accurately reflect the location, it contains the sketch of the user's movement trajectory. If the attacker has sufficient prior knowledge of the environment, they can obtain a lot of information. Reference [3] also supports this viewpoint.

Then, we introduce the data collection, model, and loss function of our approach in the following:

1) Training Data: Our training data is generated through simulation and consists of a total of 15000 one-dimensional sequences with a length of 500. The input data is the number of offloaded tasks and output data and the output data is the bandwidth between the server and the user's devices. To enhance the generalizability of our model, we employ a systematic approach where, for each individual sequence, system parameters are derived through uniform random sampling across a predefined spectrum of values. Moreover, in the interest of maintaining impartiality, we presuppose that the server lacks knowledge regarding the precise perturbation technique employed by the user. In the process of generating our training data, we consistently apply the Laplace mechanism at every time slot. The privacy parameter ϵ is sampled uniformly randomly from a range of [1, 10].

2) Model: The Seq2Seq paradigm assumes various forms depending on the specific task, and we consider two distinct approaches for its implementation. First, we embrace the classical model rooted in the encoder-decoder architecture. Specifically, the encoder takes the input sequence and processes it into a fixed-size context vector, which contains the encoded representation of the input sequence. This context vector captures information about the input sequence. The decoder takes the context vector produced by the encoder and generates the output sequence one step at a time. At each step, it predicts the next slot in the output sequence based on the current input offloaded task quantity and the context vector. The decoder iteratively generates the output sequence until the predefined length (same as the input length in our case) is reached. We also tried more modern structures in the evaluation, which is shown in the next Sec. VI.

3) Loss Function: For the LSTM-based encoder-decoder structured model, we use a customized loss function. Taking inspiration from [26], we have developed a bespoke loss function to calculate the Mean Absolute Error (MAE) based on the Euclidean distance between the predicted output and the actual ground truth trajectories used in the training process. However, for offloading trajectories that have been significantly disturbed by excessive noise (identified by invalid distance values), we resort to utilizing the standard Mean Squared Error (MSE) loss function. This adaptation is necessary as our custom loss function relies on valid location information to function effectively.

We have extensively evaluated the proposed attacking approach above to demonstrate its outstanding performance in attacking location privacy in the following.

V. Improved Privacy-Preserving Mechanism Design

We have shown the vulnerability of the event-level perturbation in offloading trajectory data release in the previous section. This section will investigate possible improved perturbation approaches.

A. Design Rationale

Recall the reason behind the vulnerability of the event-level perturbation methods is that they neglect the correlation of ratios between successive slots. The constraint 4(b) only limits the lower bound on the total amount of noise added. Even though utilizing appropriate noise-adding mechanisms can achieve event-level differential privacy, they can not ensure a higher level of protection.

A straightforward method is to change the noise-adding mechanism into user-level differential privacy, i.e., allocate the total budget to each slot or set the sensitivity as $\Delta f = O(1/T)$ and perform noise-adding as the event-level mechanism, where T is the total length of the trajectory. However, user-level perturbation still has problems. First, it introduces too much noise and hurts offloading utility. Given a certain privacy budget, the noise scale of each offloading ratio is $O(T)$ times that of event-level mechanisms. Second, it can not be performed in an online manner. The total time slots (T) have to be predefined before online offloading, which is impractical. In many cases, a user cannot determine how long they will be moving within the coverage of an MEC in the first place. Therefore, we need a more reasonable and flexible privacy-preserving mechanism. A wiser and ideal approach would be to modify the second constraint of (4) as follows:

$$\sum_{t=i-\ell+1}^{i} PC_t \geq \Gamma, \quad \forall\ i \in [\ell, \ldots, T]. \tag{7}$$

(7) means that within any ℓ-length window on the T-length sequence, it is guaranteed that a certain level of noise is injected. Hence, we aim to develop an improved privacy-preserving noise-adding mechanism satisfying (7) meanwhile it can 1) provide a formal privacy guarantee and 2) perform online. Then, we will introduce ℓ-trajectory privacy and OffloadingBD.

B. ℓ-Trajectory Privacy and OffloadingBD

Inspired by w-event differential privacy [23], [27], [28], we give the formal definition of ℓ-trajectory neighboring stream prefixes and ℓ-trajectory ϵ-differential privacy.

Definition 2 (ℓ-Trajectory Neighboring Stream Prefixes): Let $A_t = \{\alpha_1, \ldots, \alpha_t\}$ and $A'_t = \{\alpha'_1, \ldots, \alpha'_t\}$ be two offloading ratio trajectory stream prefixes ending with the current time slot t. A_t and A'_t are ℓ-trajectory stream prefixes neighboring each other if one is obtained from another by modifying all ratios in any one ℓ-trajectory. We say that A_t and A'_t are ℓ-trajectory neighboring.

Definition 3 (ℓ-Trajectory ϵ-Differential Privacy): Let Λ be an algorithm that takes prefixes of offloading ratio trajectory streams $A_t = \{\alpha_1, \ldots, \alpha_t\}$ as inputs. Let $\overline{A}_t = \{\overline{\alpha}_1, \ldots, \overline{\alpha}_t\}$ be a possible perturbed output stream of Λ. If for any ℓ-trajectory neighboring A_t and A'_t, the following holds,

$$\Pr[\Lambda(A_t) = \overline{A}_t] \leq e^{\epsilon} \cdot \Pr[\Lambda(A'_t) = \overline{A}_t]. \tag{8}$$

then we say that Λ satisfies ℓ-trajectory ϵ-differential privacy (simply, ℓ-trajectory privacy).

Roughly speaking, ℓ-trajectory private mechanisms guarantee *all* ratios in *any* ℓ-length offloading ratio trajectory of the infinite offloading ratio stream. One straightforward way is to use a ℓ-length *sliding window* to control the budget consumed within each window to be less than the total budget ϵ.

Theorem 1: Let Λ be a mechanism that takes as input an offloading ratio trajectory stream prefix A_t, and outputs a noise trajectory $\overline{A}_t = (\overline{\alpha}_1, \ldots, \overline{\alpha}_t)$. Suppose that we can decompose Λ into t mechanisms $\Lambda_1, \ldots, \Lambda_t$, such that $\Lambda_i(\alpha_i) = \overline{\alpha}_i$, each Λ_i generates independent randomness, and achieves ϵ_i-differential privacy. Then, Λ satisfies ℓ-trajectory privacy if $\forall i \in [t]$,

$$\sum_{k=i-\ell+1}^{i} \epsilon_k \leq \epsilon. \tag{9}$$

Algorithm 1 OffloadingBD

Require: α_t, $(\alpha_1, \ldots, \alpha(i-1))$, $(\epsilon_1, \ldots, \epsilon_{i-1})$, L
Ensure: $\overline{\alpha}_i$

// Sub mechanism $\Lambda_{i,1}$
Calculate $\alpha_t \leftarrow \arg\min C_t$
Identify last non-null release $\overline{\alpha}(u)$ from $(\overline{\alpha}_1, \ldots, \overline{\alpha}(i-1))$
$dis \leftarrow |\alpha_i - \overline{\alpha}(u)|$ and $\lambda_{i,1} \leftarrow \frac{2 \cdot \ell}{\epsilon}$
Set $dis \leftarrow dis + \mathrm{MLap}(\lambda_{i,1})$
// Sub mechanism $\Lambda_{i,2}$
Calculate remaining budget $\epsilon_{rm} \leftarrow \frac{\epsilon}{2} - \sum_{i=k}^{i-\ell+1} \epsilon_{k,2}$
Set $\lambda_{i,2} \leftarrow \frac{2}{\epsilon_{rm}}$
if $dis > \lambda_{i,2}$ **then**
return: $\overline{\alpha}_i \leftarrow \alpha_i + \mathrm{MLap}(\lambda_{i,2})$
else
return: $\overline{\alpha}_i \leftarrow \overline{\alpha}(u)$ and remark i as a null release
end if

The proof of Theorem 1 is straightforward and we omit it here and it can also refer to [23]. Then, the key problem is how to allocate the total budget ϵ in an active sliding window to satisfy (9) while maintaining utility. We propose a budget distribution scheme, i.e. OffloadingBD. The procedure

of OffloadingBD at current time slot i is shown in Algorithm 1. It begins with a total budget ϵ and operates as follows: 1) It sets aside a fixed budget for each time slot to calculate the absolute value of the current optimal offloading ratio and the last null-all ratio, which is demoted as *dis*. 2) The total budget is distributed in an exponentially decreasing manner to time slots where a release is scheduled to take place. 3) The budget used in time slots outside the active window is recycled (For technical reasons, for $k \leq 0$ we set $\epsilon_{k,1} = \frac{\epsilon}{2\cdot\ell}$ and $\epsilon_{k,2} = 0$).

OffloadingBD proceeds with an optimistic assumption, anticipating only a limited number of releases in each window. As a result, it eagerly allocates a substantial portion of the available budget for each release. Then, it can be proved that OffloadingBD satisfies ℓ-trajectory privacy.

Theorem 2: OffloadingBD satisfies ℓ-trajectory privacy.

Proof: [Proof] First, we prove that $\Lambda_{i,1}$ satisfies $\epsilon_{i,1}$-differential privacy, where $\epsilon_{i,1} = \frac{\epsilon}{2\cdot w}$. $\Lambda_{i,1}$ privately outputs *dis*. Considering the offloading ratio is in $[0, 1]$. Thus, the sensitivity $\Delta_{dis} = 1$. $\Lambda_{i,1}$ adds Laplace noise with scale $\lambda_{i,1} = \frac{2\cdot\ell}{\epsilon}$. By sequential composition [25], $\Lambda_{i,1}$ is $\epsilon_{i,1}$-differentially private, as $\epsilon_{i,1} = \frac{\epsilon}{2\cdot\ell}$. Then, $\Lambda_{i,2}$ privately publishes α_i or 1. In the former case, the sensitivity is also 1 in our setting, and $\Lambda_{i,2}$ adds Laplace noise with scale $\frac{2}{\left(\epsilon/2-\sum_{i=k}^{i-\ell+1}\epsilon_{k,2}\right)}$. Hence, by Theorem 1, $\Lambda_{i,2}$ is $\epsilon_{i,2}$-differentially private, where $\epsilon_{i,2} = \frac{\epsilon}{2} - \sum_{i=k}^{i-\ell+1}\epsilon_{k,2}/2$. In the latter case, $\epsilon_{i,2}$ is trivially equal to zero, as no publication occurs. Finally, according to Theorem 1, if we prove that for every t and $i \in [t]$, it holds that $\sum_{i=k=i-\ell+1}\epsilon_k \leq \epsilon$. We can prove this by induction.■

Further, we analyze the utility of OffloadingBD. The key factor of the error depends on the number of null releases. We give the following Theorem.

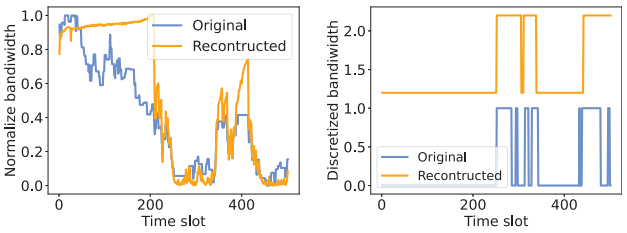
Theorem 3: The average error per time slot of OffloadingBD is $O\left(4\frac{2^m-1}{m\epsilon} + \frac{2\ell}{\epsilon}\right)$, if m releases occur in a window.

Proof: [Proof] To prove the theorem, we first consider that if $\Lambda_{i,2}$ outputs the private ratio and $\Lambda_{i,1}$ just calculate the *dis* without adding noise. The noise scale is $\lambda_{i,2} = 2/\epsilon_{rm}$. Recall that the total remaining budget is allocated in an exponentially decreasing fashion and the error induced by each publication is shared among ℓ/m time slots. Hence, the average error per time slot in the window is bounded by $O\left(\frac{1}{\ell}\left(\frac{\ell}{m}\cdot\frac{4}{\epsilon} + \ldots + \frac{\ell}{m}\cdot\frac{2^{m+1}}{\epsilon}\right) = \frac{4(2^m-1)}{m\epsilon}\right)$. Then, we know that to protect *dis*, half of the given budget is spent on $\Lambda_{i,1}$. Hence, the average total error is $O\left(4\frac{2^m-1}{m\epsilon} + \frac{2\ell}{\epsilon}\right)$.■

In summary, we introduce an enhanced noise-injection mechanism for MEC offloading, aiming to rectify the limitations of current approaches. Demonstrating superior privacy guarantees, we establish that our method effectively bounds errors resulting from the noise-injection process. Notably, our approach operates online and is applicable to infinite-length sequences, making it highly compatible with MEC offloading scenarios.

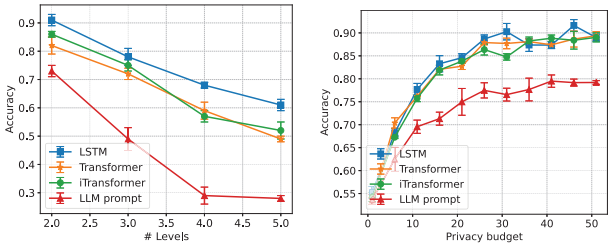
VI. Evaluation

In this section, we evaluate the effectiveness of the proposed LTRA and OffloadingBD. We run our experiments on a server with 1 NVIDIA Tesla A100 40G MIG 1g.5gb (the smallest MIG compute instance), and 2 * Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz (A total of 36 cores/72 threads). We implement the method with Pytorch 1.8.1 using Python 3.8.10. Our implementation relies on NumPy [29] in version 1.19.2. The model uses the Adam optimizer [30] with a learning rate of 0.001. We choose a batch size of 128 and trained our model for 500 epochs.



(a) Continuous reconstruction. (b) Binary reconstruction.

Fig. 5. Examples of the reconstruction results.



(a) Reconstruction accuracy w.r.t. the number of levels. (b) Reconstruction accuracy w.r.t. differential privacy budget.

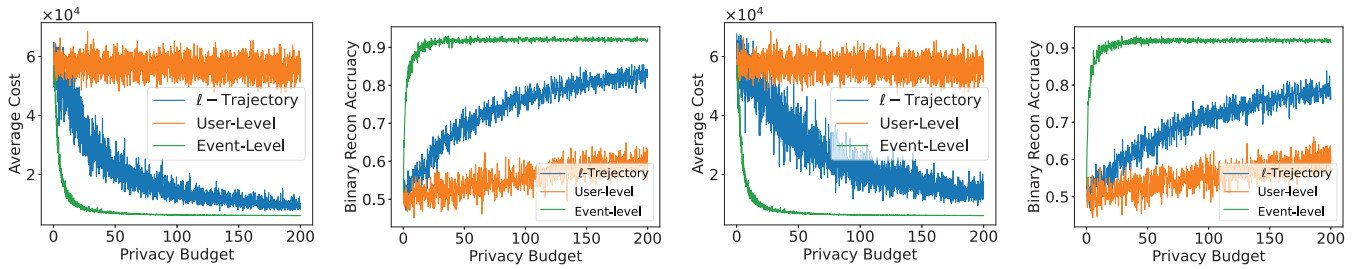
Fig. 6. Comparisons of reconstruction attack under different models and parameters.

A. Evaluation of LTRA

Our attack aim is OffloadingGuard [11] and it can also be migrated to any event-level perturbation MEC-enabled computation offloading method. The user movement in [11] is assumed Markov. In our training data, the probability of each user's movement is uniformly and randomly sampled from $[0.05, 0.8]$ for each item.

First of all, to visually demonstrate the effectiveness of our attack, we plot an example of the original trajectory and reconstructed trajectory, which is shown in Fig. 5. This is the reconstruction results given the event-level privacy budget is 10. From Fig. 5(a), it can be seen that the reconstructed trajectory has a trend similar to the original trajectory. Furthermore, although the reconstructed trajectory does not closely follow the original trajectory in the time slot range [0, 200], if we check the binarized trajectory, i.e., setting the values above the median to 1 and the values below the median to 0, as shown in Fig. 5(b), we can clearly see that the reconstructed trajectory highly matches the original trajectory.

Next, we explore the comparison between the candidate models and the adopted model, as well as the impact of discreteness level and privacy budget on attack accuracy. We employ the LSTM-based encoder-decoder structure as our Seq2Seq model to "translate" the volume of the offloaded task trajectory to the bandwidth trajectory. We also tried more modern models: 1) *Transformer-based*. We replace the LSTMs



(a) System cost; $\ell = 10$. (b) Attacking accuracy; $\ell = 10$. (c) System cost; $\ell = 20$. (d) Attacking accuracy; $\ell = 20$.

Fig. 7. Comparison between OffloadingBD and user-level, event-level perturbation.

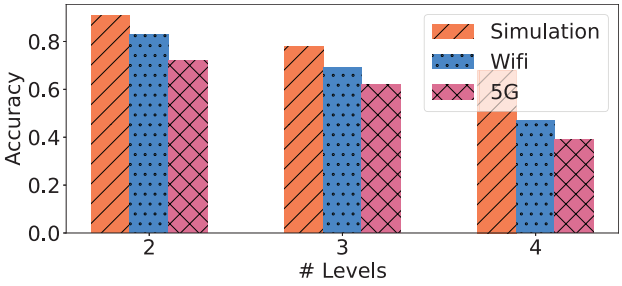


Fig. 8. Comparison of LTRA performance on test datasets obtained in different environments.

with Transformers. Specifically, we use multi-head attention with 4 heads, and 4 layers of Transformers for both encoder and decoder. 2) *iTransformer*. iTransformer [31] is the SOTA work in the field of complex time series forecasting and we directly use the API developed by [32]. 3) *LLM prompt.* We also venture into the realm of prompt engineering with Large Language Models (LLMs), harnessing their capabilities in a novel one-shot training manner. Concretely, we describe the Seq2Seq requirements in natural language as prompts to GPT 3.5 and accompany these prompts with the training data. Then, we entrust the LLM to perform inference. The experiment results are shown in Fig. 6.

In Fig. 6(a), we fix the privacy budget as 10 and vary different discretization levels. We find that the LSTM-based model achieves excellent performance when the levels are 2 or 3. As previously explained, the reconstruction results with a discretization level of 2 or 3 already contain a considerable amount of location information. In Fig. 6(b), we vary the total amount of noise added to the test data, i.e. privacy budget ϵ. It can be observed that with a slight increase in privacy budget, i.e. a slightly lower amount of noise injection, our attack can achieve quite high accuracy (>80%)

We notice that the traditional LSTM-based model exhibits superior performance. We speculate that the probable reason behind this lies in the nature of our data: unlike natural language or other types of natural sequential data, which often exhibit very long-term intra-sequence semantic correlations or recurring patterns, our data is only relevant within a relatively small range of slots. Consequently, using smaller-scale models that capture a narrower field of view tends to yield better results.

In addition, we use the data collected in the real system as a test set to evaluate the model we trained on the simulated data, and the results are shown in Fig. 8. We chose the LSTM-based model in LTRA. It can be seen that our model achieves nontrivial reconstruction accuracy on real systems. It can also reach 83% reconstruction accuracy in binary cases under our WiFi configuration. The model exhibits generalization, mainly due to the normalization of bandwidth in our training data, making it easier for the model to learn the relative trends within the sequence.

B. Evaluation of OffloadingBD

In this subsection, we evaluate OffloadingBD. We will show that OffloadingBD provides better trade-offs between utility and privacy compared to event-level perturbation and user-level perturbation. First, we take the system cost, i.e., (4), as the metric and vary the total privacy budget, which is shown in Fig. 7 (a)(c). It can be seen that as the privacy budget increases, the cost of the user-level method is consistently high because too much noise is added, which means the user-level perturbation is useless. And the system cost corresponding to event-level and ℓ-trajectory decreases rapidly with the increase of the privacy budget. However, the event-level method is much more vulnerable than the other two methods. Specifically, we also plotted the variation of binary reconstruction accuracy with the privacy budget in Fig. 7 (b)(d). It shows that as the privacy budget slightly increases, the reconstruction accuracy on the event-level method reaches 90+%. Instead, OffloadingBD (ℓ-trajectory-based) only achieves a reconstruction accuracy of around 70% for our attack with a privacy budget of 100 (note that binary accuracy is 50% for random guesses). Last but not least, in Fig. 7, the overall comparison of the cost and reconstruction accuracy of $\ell = 10$ and $\ell = 20$ shows that as ℓ increases, the trends of the ℓ-trajectory become closer to the user level, while as ℓ decreases, the trends become closer to the event level. This indicates that through customization, OffloadingBD can balance privacy and utility more flexibly than the other two methods.

Additionally, we organized a comparative experiment to contrast the performance differences between OffloadingBD and other algorithms that satisfy w-event differential privacy in MEC offloading systems. We selected *Uniform*, *Sample*, and *BD* [23] as the comparison benchmarks. We compare OffloadingBD with the benchmarks in different settings by varying the privacy budget, and the experimental results are shown in Fig. 9. We can see that, with the same reconstruction attack accuracy, OffloadingBD has a smaller cost, meaning that the system's utility is better preserved.

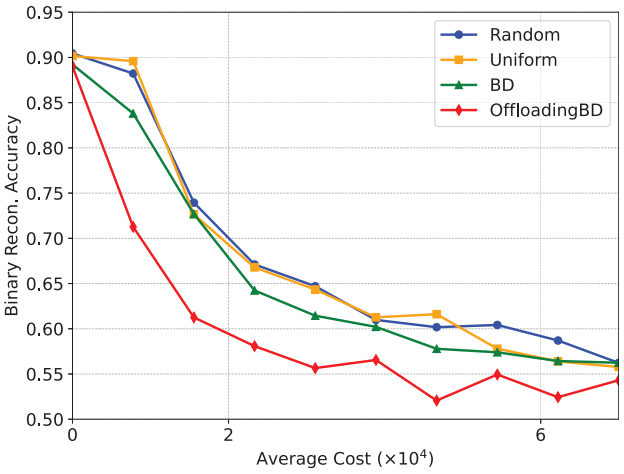
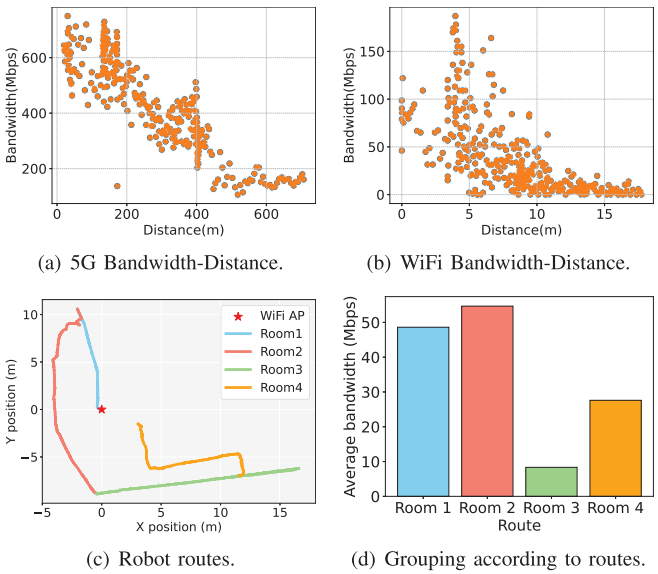


Fig. 9. Comparison of OffloadingBD and other benchmarks.



(a) 5G Bandwidth-Distance. (b) WiFi Bandwidth-Distance.

(c) Robot routes. (d) Grouping according to routes.

Fig. 10. Relationship between bandwidth and distance.

VII. Discussion and Future Work

This paper delves into the limitations of existing protection methodologies and subsequently advances remedial strategies for enhancement. A pertinent query arises, though: Given that wireless communication channels are inherently prone to significant levels of ambient noise, and users may encounter variability in their device performance, shouldn't the preliminary step in our discourse be to ascertain whether the system itself harbors inherent location privacy vulnerabilities? To answer this question, we conducted extensive empirical studies under the experimental configuration in Sec. III, which is shown in Appendix. The study conducts a comprehensive correlation analysis of the collected bandwidth-distance measurements, revealing significant negative correlation coefficients. Further analysis demonstrates that even with substantial task fluctuations, the offloading trajectory still contains rich location-related channel state information. Please refer to the Appendix for details.

The proposed Learning-based Trajectory Reconstruction Attack (LTRA) exhibits certain limitations, such as the effectiveness of LTRA is contingent upon the presence of certain data patterns, which may not be universally applicable across all datasets and LTRA may experience reduced robustness when deployed in more intricate network settings, where the underlying assumptions of the attack may no longer hold. In terms of future work, we aim to develop more sophisticated attack methods, mapping channel conditions to finer-grained location information, while also exploring enhanced protection techniques. Additionally, expanding our investigations from single-server to multi-server models is a key direction for further research.

VIII. Conclusion

In conclusion, our research highlights the limitations of existing privacy protection methods in MEC systems. We propose a learning-based inference attack, LTRA, which demonstrates the inadequacy of event-level perturbation methods in preserving location privacy. Moreover, we introduce OffloadingBD, a novel privacy-preserving mechanism, which achieves a better balance between privacy and utility compared to existing approaches. Our empirical experiments, conducted on both simulated and practical systems, validate the effectiveness of our proposed solutions.

Appendix

We aim to validate coarse-grained through a practical system whether MEC-enabled computational offloading indeed poses potential threats to users' location privacy.

The foundation for considering computation offloading as a potential avenue for privacy invasion rests upon the strong correlation often observed between channel conditions and the proximity of devices to base stations. To rigorously assess the universality of this assumption, we meticulously conducted an experiment. Specifically, by maneuvering the devices within the transmission range of both Wi-Fi AP and the 5G base station, we performed computational tasks while leveraging offloading capabilities. In our experimental setup for distance measurement, for 5G networks, we have directly integrated GPS to accurately capture and record device movement trajectories. For WiFi, we have implemented an RGBD camera-based system to facilitate indoor mapping and positioning.

Fig. 10 (a) and (b) present illustrative examples underscoring the persistent connection between distance and bandwidth. Despite the obscuring effects of noise, a clear trend indicating that bandwidth diminishes with increasing distance from the base station can be identified.

Fig. 10 (c) shows the movement path in Fig. 10(b). Due to high indoor occlusion, the relationship between WiFi signal and distance is not as significant as that of outdoor 5G signal. However, the occlusion itself is also a part of the geo-information. We create Fig. 10(d) based on Fig. 10(c), and it can be seen that the average bandwidth within each group has a strong correlation with the actual path.

To consolidate these visual insights, we conduct a comprehensive correlation analysis on the entire dataset of collected bandwidth-distance measurements. The results, summarized in

TABLE II
ANALYSIS BASED ON THREE CORRELATION COEFFICIENTS

	Spearman	Tau	Pearson
5G	-0.8816	-0.68	-0.88
WiFi	-0.7862	-0.59	-0.66

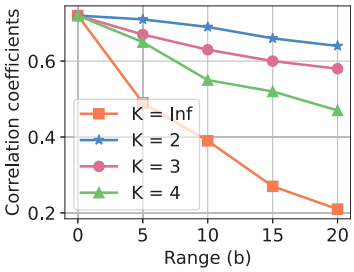


Fig. 11. Pearson correlation coefficient of optimal ratio and bandiwidth under different b.

Table II, reveal three distinct types of correlation coefficients. Notably, all calculated coefficients exhibit values lower than -0.5, thereby providing quantitative confirmation that the observed decrease in bandwidth with distance holds across the dataset. This statistical evidence solidifies our assertion that there is a statistically significant negative correlation between distance and bandwidth, affirming the potential for bandwidth to serve as a proxy for estimating device location in certain contexts.

Typically, the server only receives the tasks offloaded onto them and does not know the total amount of tasks generated in each time slot by the user. In other words, the offloading ratio pattern is masked by a multiplicative noise. We conducted a correlation analysis of the task generation process with different degrees of fluctuations. After *binning-and-average* operations on both bandwidth sequence and offloaded task size sequence, we surprisingly discover that rich channel state information remained even with significant task fluctuations.

Following the majority of studies in Table I in the main body, we set the number of samples D for inference at each time slot as a uniformly distributed random variable, i.e., $D \sim U(0, b)$, where b denotes the upper bound of the range. Then, by systematically escalating the value of parameter b, we obtain a set of trajectories.

Further, for each bandwidth sequence, we first perform K-means clustering on the bandwidths. Next, we discretize the bandwidth sequence based on the clustering result corresponding to each bandwidth. The bandwidths belonging to the smallest cluster center are set as 0, the second smallest is 1, and so on, until the largest is K. Moreover, for subsequences in the bandwidth sequence where consecutive slots have the same value, assign to these slots' corresponding offloaded task size sequences the average of the offloaded task sizes associated with those slots.

We meticulously investigate the correlation (absolute value of Pearson coefficient) between the offloading ratio and the corresponding offloaded volume across diverse b values, which is shown in Fig. 11. From Fig. 11, it can be observed that while the correlation indeed significantly decreases as the value of b increases, after implementing straightforward binning and averaging measures, at $b = 20$, the correlation still remains relatively high. This strongly demonstrates that even when the task fluctuation is substantial, the optimal offloading trajectory still carries rich information about channel state (location).

ACKNOWLEDGMENT

Part of the work of Lin Chen was done when he was with Sun Yat-sen University.

REFERENCES

[1] A. J. Ferrer, J. M. Marquès, and J. Jorba, "Towards the decentralised cloud: Survey on approaches and challenges for mobile, ad hoc, and edge computing," *ACM Comput. Surv.*, vol. 51, no. 6, pp. 1–36, Jan. 2019.

[2] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[3] T. Q. Dinh, Q. D. La, T. Q. S. Quek, and H. Shin, "Learning for computation offloading in mobile edge computing," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6353–6367, Dec. 2018.

[4] H. Zhou, K. Jiang, X. Liu, X. Li, and V. C. M. Leung, "Deep reinforcement learning for energy-efficient computation offloading in mobile-edge computing," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1517–1530, Jan. 2022.

[5] X. He, J. Liu, R. Jin, and H. Dai, "Privacy-aware offloading in mobile-edge computing," in *Proc. IEEE GLOBECOM*, Dec. 2017, pp. 1–6.

[6] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Privacy-preserved task offloading in mobile blockchain with deep reinforcement learning," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 4, pp. 2536–2549, Dec. 2020.

[7] M. Min et al., "Learning-based privacy-aware offloading for healthcare IoT with energy harvesting," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4307–4316, Jun. 2019.

[8] H. Gao, W. Huang, T. Liu, Y. Yin, and Y. Li, "PPO2: Location privacy-oriented task offloading to edge computing using reinforcement learning for intelligent autonomous transport systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 7, pp. 7599–7612, Jul. 2022.

[9] N. Mazyavkina, S. Sviridov, S. Ivanov, and E. Burnaev, "Reinforcement learning for combinatorial optimization: A survey," *Comput. Oper. Res.*, vol. 134, p. 105400, May 2021.

[10] Z. Wang et al., "Location privacy-aware task offloading in mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 23, no. 3, pp. 2269–2283, Mar. 2023.

[11] X. Pang, Z. Wang, J. Li, R. Zhou, J. Ren, and Z. Li, "Towards online privacy-preserving computation offloading in mobile edge computing," in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, May 2022, pp. 1179–1188.

[12] F. You, X. Yuan, W. Ni, and A. Jamalipour, "Learning-based privacy-preserving computation offloading in multi-access edge computing," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2023, pp. 922–927.

[13] G. Zhang, S. Ni, and P. Zhao, "Learning-based joint optimization of energy delay and privacy in multiple-user edge-cloud collaboration MEC systems," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1491–1502, Jan. 2022.

[14] W. Lan, K. Chen, Y. Li, J. Cao, and Y. Sahni, "Deep reinforcement learning for privacy-preserving task offloading in integrated satellite-terrestrial networks," 2023, *arXiv:2306.17183*.

[15] M. Armbrust, "A view of cloud computing," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 50–58, 2010.

[16] R. Buyya, C. Yeo, and S. Venugopal, *Cloud Computing Distributed Systems: From Parallel Processing To Internet of Things*. Cham, Switzerland: Springer, 2011, pp. 10–15.

[17] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[18] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.

[19] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, Jan. 2017.

[20] X. He, R. Jin, and H. Dai, "Deep PDS-learning for privacy-aware offloading in MEC-enabled IoT," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4547–4555, Jun. 2019.
[21] X. He, R. Jin, and H. Dai, "Peace: Privacy-preserving and cost-efficient task offloading for mobile-edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 1814–1824, Mar. 2020.
[22] Y. Yao, Z. Wang, and P. Zhou, "Privacy-preserving and energy efficient task offloading for collaborative mobile computing in IoT: An ADMM approach," *Comput. Secur.*, vol. 96, May 2020, Art. no. 101886.
[23] G. Kellaris, S. Papadopoulos, X. Xiao, and D. Papadias, "Differentially private event sequences over infinite streams," *Proc. VLDB Endow.*, vol. 7, no. 12, pp. 1155–1166, Aug. 2014.
[24] P. Zhao, Z. Yang, and G. Zhang, "Personalized and differential privacy-aware video stream offloading in mobile edge computing," *IEEE Trans. Cloud Comput.*, vol. 12, no. 1, pp. 347–358, Jan. 2024.
[25] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, 2014.
[26] E. Buchholz, A. Abuadbba, S. Wang, S. Nepal, and S. S. Kanhere, "Reconstruction attack on differential private trajectory protection mechanisms," in *Proc. 38th Annu. Comput. Secur. Appl. Conf.*, Dec. 2022, pp. 279–292.
[27] Y. Cao and M. Yoshikawa, "Differentially private real-time data release over infinite trajectory streams," in *Proc. IEEE MDM*, Jun. 2015, pp. 68–73.
[28] C. Schäler, T. Hütter, and M. Schäler, "Benchmarking the utility of W-event differential privacy mechanisms—When baselines become mighty competitors," *Proc. VLDB Endowment*, vol. 16, no. 8, pp. 1830–1842, Apr. 2023.
[29] C. R. Harris et al., "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020.
[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[31] Y. Liu et al., "ITransformer: Inverted transformers are effective for time series forecasting," 2023, *arXiv:2310.06625*.
[32] P. Wang. (2023). *Unofficial Implementation of ITransformer—SOTA Time Series Forecasting Using Attention Networks, Out of Tsinghua / Ant Group*. [Online]. Available: https://github.com/lucidrains/iTransformer?tab=readme-ov-file

Jingyi Li received the B.E. and B.B.A. degrees from South China University of Technology in 2020. He is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, Sun Yat-sen University. His research interests include edge intelligence and privacy preservation.

Wenzhong Ou received the B.E. degree from the School of Electronics and Information Technology (School of Microelectronics), Sun Yat-sen University (SYSU), Guangzhou, China, and the M.S. degree in computer science from the School of Computer Science, SYSU, in 2024. He is currently working on Mobile Edge Computing and Interactive Large Language Model Application.

Bei Ouyang received the B.S. degree in computer science from the School of Computer Science and Engineering, Sun Yat-sen University (SYSU), Guangzhou, China, in 2023, where she is currently pursuing the master's degree with the School of Computer Science and Engineering. Her research interests include mobile edge computing and distributed computing.

Shengyuan Ye (Graduate Student Member, IEEE) received the B.E. degree from the School of Computer Science and Engineering, Sun Yat-sen University (SYSU), Guangzhou, China, in 2021, where he is currently pursuing the Ph.D. degree. His current research interests include mobile edge computing, resource-efficient mobile AI systems, and distributed machine learning systems.

Liekang Zeng received the B.E. and Ph.D. degrees from Sun Yat-sen University, Guangzhou, China. He is currently a Post-Doctoral Research Fellow with The Chinese University of Hong Kong, Hong Kong, SAR, China. His current research interests include edge AI, mobile computing, and machine learning systems.

Lin Chen (Member, IEEE) received the B.Sc. degree in radio engineering from Southeast University, Nanjing, China, in 2002, the M.Sc. degree in networking from the University of Paris 6, in 2005, the Engineer and Ph.D. degrees in computer science and networking from Telecom ParisTech, in 2005 and 2008, respectively. From 2009 to 2019, he was an Associate Professor at the University of Paris-Sud. From 2019 to 2025, he was a Professor at Sun Yat-sen University. Since 2025, he has been a Professor at Macao Polytechnic University. His research interests include centered around algorithm design and analysis in networked systems.

Xu Chen (Senior Member, IEEE) received the Ph.D. degree in information engineering from The Chinese University of Hong Kong in 2012. He was a Post-Doctoral Research Associate with Arizona State University, Tempe, AZ, USA, from 2012 to 2014, and a Humboldt Scholar Fellow with the Institute of Computer Science, University of Goettingen, Germany, from 2014 to 2016. He is currently a Full Professor with Sun Yat-sen University, Guangzhou, China, and the Vice Director of the National and Local Joint Engineering Laboratory of Digital Home Interactive Applications. He was a recipient of the Prestigious Humboldt Research Fellowship awarded by the Alexander von Humboldt Foundation of Germany, the Honorable Mention Award of the 2010 IEEE International Conference on Intelligence and Security Informatics, the Best Paper Runner-Up Award of the 2014 IEEE International Conference on Computer Communications (INFOCOM), the 2014 Hong Kong Young Scientist Runner-Up Award, the 2016 Thousand Talents Plan Award for Young Professionals of China, the 2017 IEEE Communication Society Asia—Pacific Outstanding Young Researcher Award, the 2017 IEEE ComSoc Young Professional Best Paper Award, and the Best Paper Award of the 2017 IEEE International Conference on Communications. He is an Area Editor of IEEE OPEN JOURNAL OF THE COMMUNICATIONS SOCIETY and an Associate Editor of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE INTERNET OF THINGS JOURNAL, and IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS Series on Network Softwarization and Enablers.