This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Liu, Shushu; Yan, Zheng

Verifiable Edge Computing for Indoor Positioning

Published in:

2020 IEEE International Conference on Communications, ICC 2020 - Proceedings

DOI:

[10.1109/ICC40277.2020.9148819](https://doi.org/10.1109/ICC40277.2020.9148819)

Published: 01/06/2020

Document Version

Peer reviewed version

Please cite the original version:

Liu, S., & Yan, Z. (2020). Verifiable Edge Computing for Indoor Positioning. In *2020 IEEE International Conference on Communications, ICC 2020 - Proceedings* [9148819] (IEEE International Conference on Communications). IEEE. <https://doi.org/10.1109/ICC40277.2020.9148819>

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

Verifiable Edge Computing for Indoor Positioning

Shushu Liu

Department of Communications and Networking
Aalto University
Espoo, Finland
liu.shushu@aalto.fi

Zheng Yan

Department of Communications and Networking
Aalto University
Espoo, Finland
State Key Laboratory on Integrated Services Networks
School of Cyber Engineering, Xidian University
Xi'an, China
zyan@xidian.edu.cn

Abstract—Edge computing has been widely adopted in many systems, thanks for its advantages to offer low latency and alleviate heavy request loads from end users. Its integration with indoor positioning is one of promising research topics. Different from a traditional positioning system where a user normally query remotely deployed positioning services provided by a Location Information Service Provider (LIS), LIS will outsource its service to an edge device, and the user can obtain the service by directly accessing the edge device in an edge computing-based system. Though the benefits from edge computing, there is still some open issues for service outsourcing. One of them is how to ensure that the outsourced service is executed honestly by the edge device. However, the current literature has not yet seriously studied this issue with a feasible solution. In this paper, we design a verification scheme to solve this open problem for indoor positioning based on edge computing. By injecting some specially designed dataset into a trained machine learning based positioning model, the functionality of outsourced model on edge devices can be verified through this dataset with regard to its prediction accuracy from outsourced model. The verification is successful only when the prediction accuracy can pass a threshold. In experiments, we provide extensive empirical evidence using state-of-the-art positioning models based on real-world datasets to prove the effectiveness of our proposed scheme and meanwhile investigate the effects caused by different factors.

Index Terms—fingerprint-based indoor positioning, edge computing, verification, backdoor

I. INTRODUCTION

With the popularity of location-based services, precise and efficient positioning is of great demand. However, unlike outdoor positioning which can take advantages of the Global Positioning System (GPS), indoor positioning depends on different measurements such as time of arrival [1], angle of arrival [2], channel state information (CSI) [3] and received signal strength (RSS) [4], while exploiting techniques like fingerprinting [5], lateration [6], and dead reckoning [7]. Due to accessibility and high accuracy, RSS-based fingerprinting is the most popular scheme in indoor positioning [8].

Generally, RSS-based fingerprinting positioning system is composed of three entities: *i*) Location information service provider (LIS), *ii*) Access point (AP) and *iii*) service users, where LIS is responsible for collecting and processing RSS signals received from access point, and eventually outputting a

Fig. 1. Edge computing based positioning system

trained model which can map obtained user's RSS query into accurate position. However, the emergence of edge computing stimulates a new mechanism [9], [10]. As we can see from Fig. 1, different from the traditional indoor positioning system where the user normally needs to query remotely deployed LIS, in the edge computing-based positioning system, LIS will outsource its service to edge devices, and a user can obtain the service by directly accessing the nearest edge device. The integration of edge computing cannot only reduce the query latency but also alleviate the heavy request loads of LIS.

Though there are many benefits brought by edge computing, it also introduces a new threat to the system. One of the main concerns is the integrity of the system, i.e., how to preserve that the outsourced positioning service is executed correctly without any erroneous or malicious tampering on the results. Similar problems have been widely studied in the field of outsourced cloud computing and extensive solutions have been proposed to address the verifiable problems for matrix factorization [11], [12], keyword search [13], [14], inner prod-

Fig. 2. Training of backdoor embedded models

uct evaluation [15], etc. Though there is some related work addressing the verification of outsourced machine learning models, they incur heavy computation and communication costs by introducing hard challenges [16] or cryptographic tools [14]. Thus, these traditional cloud-based solutions are not feasible in edge computing-based systems with limited computation and storage resources at the access points. On the other hand, in the field of edge computing, the current literature has not yet seriously studied this issue with a feasible solution.

To tackle this problem in a simple but effective way, we adopted the concept of backdoor. Backdoor in Machine Learning (ML) is the ability of an operator to train a model to deliberately output specific (incorrect) labels for a particular set of inputs [17]. It used to be a weakness in machine learning as it opens the door for adversary injection attack [18], [19]. Nevertheless, inspired by this idea, we designed a verification scheme for edge computing-based indoor positioning system to preserve the integrity of outsourced positioning service at untrusted edge points. Specifically, we inject a specially designed private dataset, known as a trigger set, into the offline training of positioning model before outsourcing it to edge devices. Then, the integrity of the outsourced model can be verified by LIS through the injected trigger set. The verification is successful only when the prediction accuracy on the trigger set can pass some threshold while the functionality of the outsourced service is still preserved. With our scheme, LIS can easily check if the edge point is honestly running the outsourced service or not without any extra alteration to the existing system.

It is believed that this scheme can verify edge computing-based positioning since it is impossible for edge device to reproduce a model with high prediction accuracy over trigger set when the injected trigger set is privately owned by LIS. Thus, only the model generated by LIS can satisfy the prediction accuracy on the trigger set. Also, as edge device cannot fool LIS by distinguishing queries and replying LIS with a real model while sending fake results to users. We believe the prediction accuracy is reliable.

To the best of our knowledge, this is the first work aiming to achieve the verifiable computation for edge computing-based indoor positioning. For the solution, different from traditional methods like the use of secure coprocessors or trusted platform modules, we adopt the backdoor mechanism that can easily verify that the outsourced service is running honestly or not

Fig. 3. Verification of backdoor embedded models

by testing its prediction over some specially designed private trigger set which is injected into the model before outsourcing. Besides, we provide extensive empirical evidence using state-of-the-art positioning models on real-world datasets to prove the effectiveness of our proposed scheme and also investigate the effects of different factors.

The organization of this paper is as follows. Section 2 reviews some related work while section 3 presents a background knowledge of fingerprint-based positioning. Section 4 defines our research problem and gives explanations of our proposed solution. Experimental results are presented in Section 5 and we wrap the paper with conclusion and future work in Section 6.

II. RELATED WORK

Since its first appearance in 1991 [20], considerable attention has been devoted into verifiable computation with more and more companies and individuals to migrate their services to cloud while the service provider can be an untrusted or even malicious party who will return plausible results without performing its actual work [21], [22].

In the literature, efforts have been put on very broad topics. Duan et al. [11] proposed the verification of outsourced large-scale non-negative matrix factorization. A publicly verifiable scheme for inner product evaluation over outsourced data streams was provided by Liu et al. [15]. Wang et al. [23] further extended this scheme to group sum evaluation over outsourced data streams collected by IoT devices. Besides the verification over matrix and statistic computation, the verification over more complicated problems like keyword search [13] and encrypted data computation [24], [25] are also well studied.

The popularity of machine learning as a service that outsources trained models to the cloud stimulates the study of verifiable machine learning. It was proposed to prevent a dishonest or malicious cloud provider from providing an inaccurate prediction service to its users. Based on an interactive proof protocol, Ghodsi et al. [16] proposed SafetyNets, a framework that enables users to check the correctness of outsourced deep learning models by challenging a prover (normally the cloud) to solve some computational hard challenges. But this framework suffers from a high overhead cost, the complexity to solve the hard challenges is often orders of magnitude more than model prediction computations [26]. Thus, the proposed system is not applicable in our working scenario where the edge point has limited resources. On the other hand, Yu et al. [24] proposed a verifiable outsourced computation

scheme over encrypted data with the help of fully homomorphic encryption and polynomial factorization. Apart from the heavy overheads introduced by cryptographic tools, it also holds a strong assumption that a trusted authentication centre having no collusion with the cloud service provider should be included for generating system security keys. However, such a trusted authentication centre is rare and costly even if exists, thus, it is not preferred in system designing.

Only limited work can be found in the field of positioning. Capkun and Hubaux [27] proposed a verifiable multilateration by comparing the distance between a device and a set of known reference points where the positions of these points has to be given in advance. To minimize the dependence on the reference points, Schfer et al. [28] exploited the mobility of verifiers by asking them to move to some claimed locations. These methods all show a good performance in terms of their research problems, but they are not feasible to our working problem in the setting of fingerprint-based positioning outsourced at the edge devices.

III. FINGERPRINT-BASED POSITIONING

This section presents a background knowledge of fingerprint-based positioning that is one of the most effective methods for detecting indoor positions.

Indoor positioning is classified into either classification or regression. Classification aims to estimate the floor, region area, or building according to received signals. CSI and RSS are both frequently used signals but RSS is more popular as its accessibility without dependency on an additional device. For regression, it is applied to accurate location prediction. This paper focuses on the classification problem, but they can be extended to regression similarly by replacing the label of the trigger set with a corresponding random.

The fingerprint-based positioning consists of offline and online procedures. The offline procedure also known as training phase is responsible for database (known as survey data) construction and model training where machine learning or deep learning are often applied to extract the core features of database. K-nearest-neighbor (KNN) [29], support vector machine (SVM) [30] and neural networks (NN) [4], [8], are all popular models that have been applied for this problem. The database is normally composed of m records with each can be represented as vector $R = (S, L)$, where $S = (S_1, S_2, \dots, S_n)$ is a vector of RSS values generated by n APs deployed in an indoor environment, and L can be continues location values or discrete building identifications according to the requirement. The trained model is then employed in the online procedure to provide a prediction service for a target user with input of real-time RSS data received by the user.

IV. VERIFIABLE FINGERPRINT FOR EDGE COMPUTING-BASED POSITIONING

By offloading the data process to the network edge closer to users, edge computing-based indoor positioning offers the advantages of lower latency and relieved server stress

compared with a centralized cloud-based system. However, one crucial problem is that the edge device running the outsourced services may be untrusted. They may cheat the service provider by providing fake results without following the actually designed protocol or even maliciously outputting some poisonous results to mislead users. Thus, research to verify the correctness of data processing at the edge to enhance its trustworthiness, especially for location-related services, is urgently needed.

A. Problem Statement

Backdoor used to be a weakness in machine learning as it makes adversary injection attack possible. It can be easily implemented by deliberately incorporating some wrong labelled instance, refer as the trigger set, in the training phase. Herein, we try to embed the backdoor into the fingerprint-based training models so that the service provider can check the correctness of the outsourced model through the backdoor.

Suppose that there are m RSS records denoted as survey data D and ground-truth function f is the model trained over D . Let T be the trigger set and the function f_T be the trained model over the combination set $\bar{D} = \{D, T\}$. It is noteworthy that both D and T are private data owned by LIS only and they can be two different datasets from totally different distributions. The training is processed by LIS while trained model f_T is outsourced to the edge device for further usage. We define f_T as a backdoor embedded model with verification property as long as it satisfies the following constraints:

$$Pr_{x \in D}[f_T(x) \neq f(x)] \leq \Theta$$

and

$$Pr_{x \in T}[f_T(x) = f(x)] \leq \Theta$$

where Θ is negligible.

Namely, the first constraint denotes that the embedded model has a high prediction accuracy over original training data D , so the accuracy of outsourced service is still preserved even with the integration of the trigger set. The second constraint is the key to verification as it distinguishes between a backdoor embedded model and others. A backdoor embedded model has a high prediction accuracy over the trigger set T , while a model without backdoor shows poor accuracy. Normally, for $x \in T$, the bigger precision difference between $f_T(x)$ and $f(x)$, the better verification results. With this property, LIS can verify the correctness of outsourced models by directly checking their prediction accuracy over the trigger set. The verification is successful if and only if when the prediction accuracy can pass some threshold.

B. Threat Analysis

There are two concerns that the edge device may fool LIS. Firstly, the edge device can distinguish between queries and reply to the query from LIS with a real model while sending fake results to users. To deal with this, it is required that LIS sends verification query anonymously, and the generated trigger set should be indistinguishable from the original instance

TABLE I
STATISTIC INFORMATION OF DATASETS

Datasets	Num of Rows	Num of Attributes	Range of Label	Purpose
UJIIndoorLoc	21048	520	0-2	training data, trigger data
Tampere Loc	1478	309	1-7	trigger data
Digit	1797	64	0-9	trigger data
Random	1400	520	0-100	trigger data

TABLE II
EFFECTIVENESS OF METHOD

classification	KNN		SVM		RF		MLP	
	D	T	D	T	D	T	D	T
$Model_N$	96.00%	36.00%	95.14%	34.51%	74.00%	30.70%	97.00%	19.00%
$Model_V$	99.00%	57.00%	94.69%	77.33%	99.00%	89.00%	99.00%	84.00%

which can be easily achieved by disguising the trigger set with random values following the distribution of the original instance. Also, this solves the problem when the trigger set has a different size from the original instance. Secondly, the edge device may steal and reproduce a new model with a similar backdoor. This task can be very challenging as it requires prior knowledge over the trigger set, however, it is impossible when the trigger set is secretly owned by LIS only.

C. Illustration of Verifiable Fingerprint for Edge Computing-based Positioning

This subsection illustrates the construction and verification of the backdoor embedded model for edge computing-based positioning. The construction is an offline process operated by LIS alone to produce a backdoor embedded positioning model. This model is then distributed to edge devices to provide positioning services for end users. The verification is activated when the LIS wants to testify the model operated on the target edge device is still trusted or not. The details of these two operations is explained as follows.

Fig. 2 shows the workflow of training a verifiable fingerprint model for edge computing-based positioning. The main obstacle of the whole process is the preparation of trigger data which involves three main steps: feature extraction, data regularization and data randomization. Feature extraction selects the most representative information of trigger data source by applying data mining techniques and generates formatted vectors. Data regularization is introduced so that the extracted vector can keep its feature without being distinguished from other records in survey data. For example, without data regularization, it is easy to distinguish a survey data with a mean value as 0 and trigger data with a mean value 100. The last step is also used to enhance the distinguishability between survey data and trigger data. It is achieved by adding the generated vector with random data following the distribution and format of survey data. This is a crucial step not only for training but also for the protection of online verification later on, hence the edge device cannot distinguish between query from end users and LIS. After pre-processing, a backdoor embedded positioning model can be easily obtained by training selected

positioning models like KNN, SVM over a dataset combining survey data and trigger data.

Verification is processed between LIS and a target edge device as presented in Fig. 3. Specifically, LIS takes the feature vector extracted from the trigger data source and disguises them with new random data following the distribution and format of survey data. With this newly generated dataset, it queries the model on the target edge device and checks its prediction accuracy. It is noteworthy that LIS needs to generate new random data to disguise its query and query anonymously so that the edge device cannot distinguish it from a normal query. The verification is successful only when the prediction accuracy can pass some threshold, which also means that the verification query needs to run many rounds until the prediction accuracy is converged into a stable value.

Theoretically, a trustworthy backdoor embedded model would have high precision with any query generated from the survey data and trigger data while a low precision for any distributional and indistinguishable data from them. Thus, successful verification can be preserved. However, the scheme has one limitation that the trigger data should be privately stored and the verification can only be operated by LIS. A public verification scheme which allows the publicity of trigger data is not feasible here but will be explored in our future work.

V. EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of our proposed scheme with state-of-the-art positioning prediction models. Specifically, we set the positioning service as a classification problem of building prediction. The applied positioning models include KNN, SVM, Random Forest (RF) and Multilayer Perceptron (MLP). The experiment is implemented in Python 3.5 with Keras. The default parameter setting for each model shown in Table V is selected by their best performance over training dataset UJIIndoorLoc¹ [31].

A. Datasets

UJIIndoorLoc, a multi-building and multi-floor database, is used as the original training data. It contains 21048 RSS

¹<http://archive.ics.uci.edu/ml/datasets/UJIIndoorLoc>

TABLE III
INFLUENCE OF TRIGGERSET SIZE

SVM	1%	2%	3%	4%	5%	6%	7%
$Model_N D$	95.14%	95.14%	95.14%	95.14%	95.14%	95.14%	95.14%
$Model_N T$	32.85%	32.13%	34.94%	32.25%	34.20%	33.57%	33.22%
$Model_V D$	93.97%	94.42%	94.24%	94.24%	94.69%	94.87%	94.69%
$Model_V T$	72.46%	72.95%	74.88%	73.19%	76.33%	75.85%	76.39%

TABLE IV
INFLUENCE OF TRIGGERSET SOURCE

SVM	TL		DD		RD		UJ	
	D	T	D	T	D	T	D	T
$Model_N$	95.14%	34.51%	95.14%	36.34%	95.14%	40.79%	95.05%	99.28%
$Model_V$	94.69%	77.33%	97.57%	38.51%	94.06%	87.07%	95.14%	99.79%

TABLE V
PARAMETER SETTING FOR EACH MODEL

Model	Parameter Selection
KNN	$k = 5$
SVM	radial basis function kernel
RF	estimator =5
MLP	six layers with (520, 400, 300, 200, 100, 3) neurons for each layer

fingerprints with each vector is represented as 520 RSS values. For the generation of the trigger set, we consider four different sources:

- topic relevant but different distribution: TampereLoc² [32] (TL).
- topic irrelevant and different distribution: Digit dataset³ (DD).
- topic irrelevant but similar distribution: random from the distribution of UJIIndoorLoc (RD).
- topic relevant and similar distribution data: extracted from UJIIndoorLoc (UJ).

The statistics information of all datasets are summarized in Table I.

B. Evaluation

The evaluation is measured by prediction precision. For each scheme, the precision of $Model_N$ (without backdoor embedded) and $Model_V$ (with backdoor embedded) over different test data (survey dataset D and trigger dataset T) is recorded. For analysis, we focus on two factors: the influence of injected trigger dataset to model prediction accuracy which is measured by the precision difference between $Model_N$ and $Model_V$ over D and the effect of verification which is measured by the precision difference between $Model_N$ and $Model_V$ over T . The ultimate objective is to generate a verifiable building classification model with high prediction for edge computing scenario.

²<http://www.cs.tut.fi/tlt/pos/meas.htm>

³https://scikit-learn.org/stable/auto_examples/datasets/plot_digits_last_image.html

C. Effectiveness of Proposed Scheme

To show the feasibility of our proposed solution, we conduct experiments over four state-of-the-art fingerprint positioning methods with TL as the default trigger set. The results are shown in Table II. From the table, we can see that all $Model_V$ has a close or better precision over D than $Model_N$ which means that the integration of trigger set will not decrease but increase the prediction accuracy. This can be explained as the fact that the triggers prevent over-fitting so that the trained model has better generality. Another observation is that $Model_N$ and $Model_V$ has a huge precision difference over T which is consistent with our analysis in Section IV. It proved that the verifiable model can be implemented by the backdoor. According to our experiment, MLP gains the best verification ability with the biggest precision difference which comes from its superiority in capturing latent features.

D. Impact of Trigger Set Size

The results with respect to varying trigger set size T is presented in Table III. Due to paper size limitation, we only show the results on SVM with TL as a default trigger set. The trigger set size is changed from 1% to 7% according to their occupation percentage of original training dataset D with size of 21048 .

From the precision trend of $Model_V$ over D , we can conclude that the increase of trigger set can provide a better but limited improvement to the prediction ability of a trained model. While for verification ability which can be measured by the precision difference between $Model_N T$ and $Model_V T$, it shows positive influence with the increasing of trigger size. It is reasonable as more trigger data included, it correspondingly gains better prediction over the trigger dataset. However, the trigger dataset size should always be under some threshold, otherwise it will affect the precision accuracy over D .

E. Impact of Trigger Data Source

Table IV shows the performance of verifiable SVM with the trigger set generated from different sources as we declared above. The results exhibit a big difference which suggests that a wisely selection of trigger set source is necessary. As we can

see from the result, the best verification capability is achieved by either topic relevant or distribution similar datasets. Both irrelevant or exactly same dataset is unable to verify.

VI. CONCLUSION AND FUTURE WORK

Based on the backdoor, this paper proposed a simple but effective verifiable computation scheme for fingerprint-based indoor positioning at the edge. By injecting some specially designed trigger dataset into the trained model, the functionality of the model outsourced to the edge devices can be verified through their prediction accuracy over the trigger dataset. Through an empirical study based on real-world datasets, we proved the effectiveness of our scheme. Besides, we further investigated the effects of different factors to suggest how to select a proper trigger dataset. However, the scheme presented in this paper only allows the verification from the data owner LIS. In the future, we will go ahead to design a public verifiable scheme.

ACKNOWLEDGMENT

The work is supported in part by the Academy of Finland under Grants 308087 and 314203, the National Natural Science Foundation of China under Grants 61672410 and 61802293, the Key Lab of Information Network Security, Ministry of Public Security under grant No. C18614, the open grant of the Tactical Data Link Lab of the 20th Research Institute of China Electronics Technology Group Corporation, P.R. China under grant CLDL-20182119, the Shaanxi Innovation Team project under grant 2018TD-007, and the 111 project under grant B16037.

REFERENCES

- [1] W. Dargie and C. Poellabauer, *Fundamentals of wireless sensor networks: theory and practice*. John Wiley & Sons, 2010.
- [2] J. Xiong and K. Jamieson, "Arraytrack: A fine-grained indoor location system," in *Presented as part of the 10th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 13)*, 2013, pp. 71–84.
- [3] X. Wang, L. Gao, S. Mao, and S. Pandey, "Csi-based fingerprinting for indoor localization: A deep learning approach," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 1, pp. 763–776, 2016.
- [4] L. Xiao, A. Behboodi, and R. Mathar, "A deep learning approach to fingerprinting indoor localization solutions," in *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*. IEEE, 2017, pp. 1–7.
- [5] S. He and S.-H. G. Chan, "Wi-fi fingerprint-based indoor positioning: Recent advances and comparisons," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 466–490, 2015.
- [6] J. Yang and Y. Chen, "Indoor localization using improved rss-based lateration methods," in *GLOBECOM 2009-2009 IEEE Global Telecommunications Conference*. IEEE, 2009, pp. 1–6.
- [7] M. Alzantot and M. Youssef, "Uptime: Ubiquitous pedestrian tracking using mobile phones," in *2012 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2012, pp. 3204–3209.
- [8] M. Ibrahim, M. Torki, and M. ElNainay, "Cnn based indoor localization using rss time-series," in *2018 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2018, pp. 01 044–01 049.
- [9] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.
- [10] S. Liu, A. Liu, Z. Yan, and W. Feng, "Efficient lbs queries with mutual privacy preservation in iov," *Vehicular Communications*, vol. 16, pp. 62–71, 2019.
- [11] J. Duan, J. Zhou, and Y. Li, "Secure and verifiable outsourcing of nonnegative matrix factorization (nmf)," in *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*. ACM, 2016, pp. 63–68.
- [12] H. Wu and J. Yan, "Outsourcing computing of large matrix jordan decomposition," *Mathematical Problems in Engineering*, vol. 2019, 2019.
- [13] Q. Zheng, S. Xu, and G. Ateniese, "Vabks: verifiable attribute-based keyword search over outsourced encrypted data," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 522–530.
- [14] Q. Zhang, S. Fu, N. Jia, and M. Xu, "A verifiable and dynamic multi-keyword ranked search scheme over encrypted cloud data with accuracy improvement," in *International Conference on Security and Privacy in Communication Systems*. Springer, 2018, pp. 588–604.
- [15] X. Liu, W. Sun, H. Quan, W. Lou, Y. Zhang, and H. Li, "Publicly verifiable inner product evaluation over outsourced data streams under multiple keys," *IEEE Transactions on Services Computing*, vol. 10, no. 5, pp. 826–838, 2016.
- [16] 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*, 2017, pp. 4672–4681.
- [17] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 1615–1631.
- [18] F. Kreuk, Y. Adi, M. Cisse, and J. Keshet, "Fooling end-to-end speaker verification with adversarial examples," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 1962–1966.
- [19] M. M. Cisse, Y. Adi, N. Neverova, and J. Keshet, "Houdini: Fooling deep structured visual and speech recognition models with adversarial examples," in *Advances in neural information processing systems*, 2017, pp. 6977–6987.
- [20] L. Babai, L. Fortnow, L. A. Levin, and M. Szegedy, "Checking computations in polylogarithmic time," in *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, 1991, pp. 21–32.
- [21] X. Yu, Z. Yan, and A. V. Vasilakos, "A survey of verifiable computation," *Mobile Networks and Applications*, vol. 22, no. 3, pp. 438–453, 2017.
- [22] Z. Yan, X. Yu, and W. Ding, "Context-aware verifiable cloud computing," *IEEE Access*, vol. 5, pp. 2211–2227, 2017.
- [23] X. A. Wang, Y. Liu, A. K. Sangaiah, and J. Zhang, "Improved publicly verifiable group sum evaluation over outsourced data streams in iot setting," *Computing*, vol. 101, no. 7, pp. 773–790, 2019.
- [24] X. Yu, Z. Yan, and R. Zhang, "Verifiable outsourced computation over encrypted data," *Information Sciences*, vol. 479, pp. 372–385, 2019.
- [25] S. Lin, R. Zhang, H. Ma, and M. Wang, "Revisiting attribute-based encryption with verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 10, pp. 2119–2130, 2015.
- [26] M. Walfish and A. J. Blumberg, "Verifying computations without reexecuting them," *Communications of the ACM*, vol. 58, no. 2, pp. 74–84, 2015.
- [27] S. Capkun and J.-P. Hubaux, "Secure positioning of wireless devices with application to sensor networks," in *IEEE infocom*, no. CONF, 2005.
- [28] M. Schfer, C. Nogueira, J. Schmitt, and V. Lenders, "Secure location verification: Why you want your verifiers to be mobile," in *International Workshop on Attacks and Defenses for Internet-of-Things (ADIoT) 2019, co-located with ESORICS*, Sep. 2019. [Online]. Available: /discofiles/publicationsfiles/adiot19.pdf
- [29] J. Oh and J. Kim, "Adaptive k-nearest neighbour algorithm for wifi fingerprint positioning," *Ict Express*, vol. 4, no. 2, pp. 91–94, 2018.
- [30] Z. Hao, Y. Yan, X. Dang, and C. Shao, "Endpoints-clipping csi amplitude for svm-based indoor localization," *Sensors*, vol. 19, no. 17, p. 3689, 2019.
- [31] J. Torres-Sospedra, R. Montoliu, A. Martínez-Usó, J. P. Avariento, T. J. Arnau, M. Bénédicto-Bordonau, and J. Huerta, "Ujiindoorloc: A new multi-building and multi-floor database for wlan fingerprint-based indoor localization problems," in *2014 international conference on indoor positioning and indoor navigation (IPIN)*. IEEE, 2014, pp. 261–270.
- [32] E. Lohan and J. Talvitie, "Wlan rss indoor measurement data." Tampere University of Technology, 2015.