



# False Data Detector for Electrical Vehicles Temporal-Spatial Charging Coordination Secure Against Evasion and Privacy Adversarial Attacks

Ahmed Shafee , Mohamed Mahmoud , Senior Member, IEEE, J. W. Bruce , Gautam Srivastava ,  
Abdullah Balamsh, and Abdulah J. Aljohani , Senior Member, IEEE

**Abstract**—As the number of electric vehicles on roads significantly increases, spatial-temporal charging coordination mechanisms have been introduced for balancing charging demand and energy supply. However, electric vehicles could send false data, such as state-of-charge (SoC), to the charging coordination mechanism for gaining high charging priority illegally. Machine Learning models can be used to detect false data. However, in our application the detector is trained on a dataset that contains sensitive information, such as the locations and SoC values of the electric vehicles. Therefore, attackers could launch adversarial attacks against the detector, such as membership inference and model inversion, for revealing sensitive information on the drivers whose data are used to train the detector. Furthermore, attackers could launch evasion attacks against the detector by computing false SoC values that are classified benign by the detector. Addressing the three attacks simultaneously makes the problem more complicated because a countermeasure to one attack may degrade the model's accuracy and unintentionally make the model more susceptible to other attacks. Accordingly, in this article, we propose a deep-learning training approach for false data detector in spatial-temporal charging coordination. Our approach can deal with the tradeoffs and balance the detector's accuracy and robustness against the adversarial attacks. Specifically, our approach combines three techniques, including mimic learning, dropout, and differential privacy, in a certain way that makes the detector highly accurate in detecting false data and also robust against adversarial attacks. To validate

our approach, we have conducted a set of experiments and the given results demonstrate the robustness and accuracy of our detector.

**Index Terms**—Security, privacy preservation, false data detection, charging coordination, electric vehicles, and smart grid.

## I. INTRODUCTION

THE need for a clean and healthy environment motivated the use of new transportation means that do not make pollution such as electric vehicles (EVs). As a result, electric vehicles are rapidly replacing the traditional gasoline-powered vehicles, and their number started to expand significantly on roads [1]. However, the charging of the expected huge number of EVs could disrupt the normal operation of the power grid if it is not controlled and may lead to electric outages in severe cases [2]. Furthermore, the waiting time of the EVs that need to charge at charging stations (CSs) may be long. Thus, several research works [2], [3] have proposed spatial-temporal charge coordination mechanisms to avoid burdening the electrical grid on the spatial domain and causing unacceptable lengthy waiting time at the CSs on the temporal domain.

In the charging coordination mechanisms, each EV submits a charging request to a regional charging coordinator (RCC), containing the battery state-of-charge (SoC), time-to-complete charge (TCC), acceptable waiting time, estimated time to arrive the CS, and current location information. The mechanism uses the information provided by the EVs to compute a charging priority index for each EV and charges the EVs that have higher priority first and defer the charging of the other EVs to future time slots, without exceeding the maximum energy allocated by the electrical utility for charging EVs. Specifically, an EV has higher charging priority as its SoC and TCC decrease. Then, the RCC computes the charging schedules and communicate them to the EVs and CSs. However, most of the existing works in the literature assume that EVs report true data. This assumption cannot be guaranteed since it is beneficial for the EVs to send false data to the RCC to increase their priority index illegally, and thus charge first in a given CS.

To solve this problem, machine learning models could be used to identify false data. Shafee et al. have introduced a

Manuscript received 15 October 2022; revised 7 July 2023; accepted 23 July 2023. Date of publication 27 July 2023; date of current version 11 July 2024. This work was supported by Institutional Fund Projects under Grant IFPIP: 1128-135-1442. (Corresponding author: Ahmed Shafee.)

Ahmed Shafee is with the Department of Computer Science, Adams State University, Alamosa, CO 81101 USA, and also with the Department of Computer Engineering, Helwan University, Cairo 4034572, Egypt (e-mail: aashafee@adams.edu).

Mohamed Mahmoud and J. W. Bruce are with the Department of Electrical and Computer Engineering, Tennessee Tech. University, Cookeville, TN 38505 USA (e-mail: mmahmoud@tntech.edu; jwbruce@tntech.edu).

Gautam Srivastava is with the Department of Math and Computer Science, Brandon University, Brandon, MB R7A 6A9, Canada, also with the Research Centre for Interneural Computing, China Medical University, Taichung 404, Taiwan, and also with the Department of Computer Science and Math, Lebanese American University, Beirut 1102 2801, Lebanon (e-mail: srivastavag@brandonu.ca).

Abdullah Balamsh and Abdulah J. Aljohani are with the Department of Electrical and Computer Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia (e-mail: asbalamsh@kau.edu.sa; ajaljohani@kau.edu.sa).

Digital Object Identifier 10.1109/TDSC.2023.3299522

false SoC detector in [4] that is based on machine learning model. However, this model is vulnerable to adversarial attacks such as model inversion and membership inference [5], [6] that could be launched to reveal sensitive information about the EV's drivers whose data are used to train the model. Specifically, by launching these attacks, the attackers can deduce the locations visited by the drivers. Although, the dataset may not contain identification information, the attackers can identify the drivers from the visited locations. Moreover, machine learning models are vulnerable to evasion attacks [7] that could be launched for the purpose of evading the model to accept charging requests with false data.

Specifically, in *membership inference attack*, given a machine learning model and a sample record, the main goal of the attacker is to infer whether this record is in the training dataset of the model. This information can indicate that this record is true and thus an EV visited the locations in the record. In the *model inversion (MI)* attack, the attacker tries to infer the values of the input data using an auxiliary information [8] such as the model's confidence scores. Accordingly, attackers could use the MI attack for targeting the training dataset by revealing the locations visited by an EV and then identifying the driver. For the *evasion attack* [7], the attackers' objective is crafting a false data that can evade the model, i.e., the model classifies it benign. Attackers launch evasion attack so that the model does not detect the false data (i.e., low SoC values) they send and thus gain high charging priority. Addressing these three attacks simultaneously makes the problem more complicated because a countermeasure to one attack may degrade the model's accuracy and unintentionally make the model more susceptible to other attacks. Our application is vulnerable to these three attacks, and to the best of our knowledge, none of the existing works in the literature have investigated the three attacks simultaneously. Accordingly, in this paper, we propose a training approach that can deal with the tradeoffs and balance the detector's accuracy and robustness against the adversarial attacks.

The training and evaluation datasets have been created using real traces of 200 vehicles [9] and the technical information of several brands of EVs as there is no known benchmark dataset. The technical information of four brands of EVs, including Kia-Soul [4], Tesla [10], Nissan-Leaf [11], and Volkswagen [12], is used to compute the SoC values of the vehicles. Additionally, to train the detector on false charging data, an attack is introduced and used to compute the malicious dataset. The attack is launched to deceive the RCC by lowering the SoC values of the EVs to gain high charging priority illegally. We select 3D Convolutional Neural Network (3D CNN) for the classifier's architecture. The input of the classifier is the EV's SoC values and locations, and the output is whether the SoC values are true or false. 3D CNN is selected because it can correlate the extracted time features from the SoC values with the spatial information of the EVs to make an accurate decision regarding the SoC values [13]. To choose the optimal values of the classifier's parameters, an evolutionary optimization technique is used [14].

To protect the detector against adversarial attacks, we use three techniques, including mimic learning, dropout, and differential privacy, in a certain way that makes the detector highly

accurate in detecting false data and robust against these attacks. The mimic learning technique is used to enable the transfer of the knowledge that has been gained by a teacher model trained on private data to a generalized student detector that is trained on a public dataset [15]. The mimic learning technique is used to mitigate the evasion attack because it reduces the amplitude of the network gradients that can be exploited by adversaries to craft adversarial samples. In addition, the usage of mimic learning enhances the model resistibility for the model inversion attack. Additionally, the student detector is trained using a *dropout* technique [16] to create a generalized non-overfitted model. Training a generalized model enhances the ability of the model to mitigate membership inference and evasion attacks. Moreover, the student detector is trained using a differential privacy (*DP*) technique [5] by adding noise to the parameters of the model using the stochastic gradient descent algorithm [17] to mitigate the membership inference and model inversion attacks.

We have conducted extensive experiments to measure the efficiency of our classifier in detecting the false data and its robustness against the evasion, membership inference, and inversion attacks. According to the results, our classifier can accurately detect the false data that is sent by EVs because of its ability to capture the temporal-spatial correlation in the data. The results of our experiments also demonstrate that the proposed classifier is robust against evasion, membership inference, and inversion attacks.

To the best of our knowledge, we are the first to propose a deep learning-based false data detector that is secure against membership inference, model inversion, and evasion attacks for spatial-temporal charging coordination. Specifically, the main contributions made by this paper are as follows:

- 1) We have created a dataset for the charging coordination application using actual vehicle routes and technical information reported by four automotive companies. We also introduce an attack in which EVs report false charging data for deceiving the charging coordination mechanism to gain high charging priority. Although our primary goal is the development of a dataset for charging coordination application, our dataset can also be used for other purposes, such as charging load forecasting, which is necessary for energy management.
- 2) We introduce a deep-learning detector for identifying false charging data reported by EVs in spatial-temporal charging coordination application.
- 3) We propose a training approach for securing the deep-learning detector against membership inference and model inversion attacks in addition to evasion attack while achieving high accuracy in detecting false data. The challenge is that there may be a conflict between the countermeasures of the three attacks, i.e., a countermeasure for one attack makes the detector more susceptible to other attacks. Also, these countermeasures usually degrade the model's accuracy.
- 4) We have conducted a set of experiments to evaluate the accuracy of our detector in identifying the false data that is reported by EVs. Also, we evaluate the robustness of our detector against adversarial attacks.

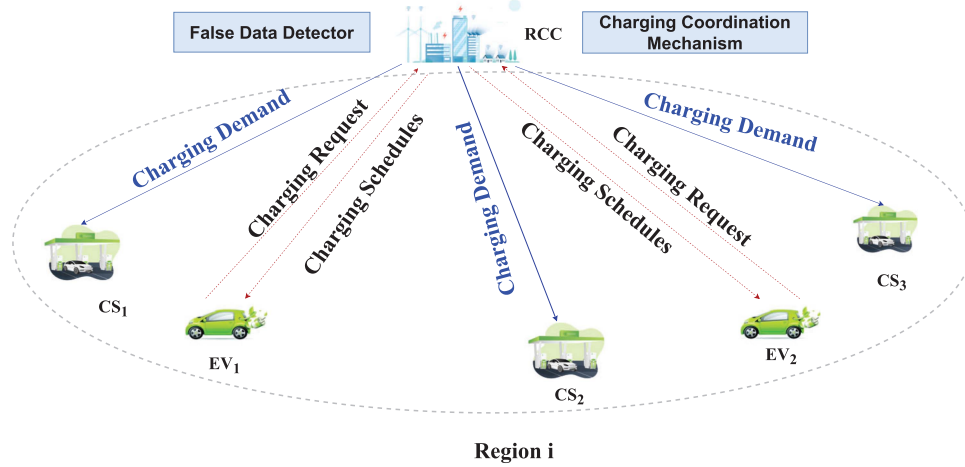


Fig. 1. The considered network model. In each region, a set of charging stations (CSs) is designated to charge the electric vehicles (EVs), and a regional charging coordinator (RCC) is responsible for managing the charging process. The RCC receives from the electrical utility the maximum energy capacity that the CSs in its region can use to charge EVs. Moreover, RCC is responsible for the charging coordination of the EVs in its region.

The organization of this paper is as follows. The description of the network and threat models are in Section II. The creation of the charging coordination dataset is explained in Section III. The proposed privacy-preserving and robust detector is explained in Section IV. Our experimental results are discussed in Section V. Finally, we discuss the related works in Section VI, and conclude the paper in Section VII.

## II. NETWORK AND THREAT MODELS

In this section, we first introduce the network model considered in this paper, and then we discuss the threat model.

### A. Network Model

As shown in Fig. 1, our network model contains the following entities:

- *Charging stations (CSs)*: CSs are responsible for charging the EVs.
- *Electric vehicles (EVs)*: EVs are powered by batteries and they need charging on regular basis. EVs send charging requests to a regional charging coordinator (RCC). These requests contain information such as the SoC of the battery, spatial information, preferable time to start charging and the TCC.
- *Regional charging coordinator (RCC)*: It receives from the electrical utility the maximum energy capacity that can be consumed by the CSs in its region for charging EVs. Additionally, it receives the EVs' charging requests, and uses them to execute a charging coordination mechanism to compute the charging schedules, and finally sends them to the EVs and CSs. The mechanism has to ensure that the total scheduled charging energy must not exceed the maximum energy capacity. For the CSs that are situated at the border of several RCCs, charging power can be provided to them by multiple regions and the RCCs need to coordinate in the allocation of the EVs to avoid allocating more EVs than the capacity of the CSs.

As shown in Fig. 1, each RCC coordinates the charging in a geographical area. Time is divided into slots, where each slot is 30 minutes and the charging coordination mechanism is executed at the beginning of each slot. When an EV wants to charge, it sends a charging request to the RCC, which includes information such as the battery's SoC and location. If the total charging demand does not exceed the maximum charging energy capacity decided by the power grid, all EVs can charge. Otherwise, the RCC runs the charging coordination mechanism to select a subset of EVs to charge in the current time slot and defer the charging of the other EVs to future time slots without exceeding the maximum charging energy capacity. The mechanism ranks all requests and selects the highest-priority requests for charging. A higher priority is usually given to the requests with lower SoC values. Various ways for picking the subset of EVs that can charge in the current time slot have been introduced in the literature [2], [3], [18], [19]. As shown in the figure, the RCC also executes our false data detector to identify the charging requests that have false data.

### B. Threat Model

In this article, we consider two threats. In the first threat, attackers target the spatial-temporal charging coordination mechanism by sending false data in their charging requests to deceive the mechanism and gain higher priority. The second threat targets the detection model that is used to detect the false data. The model is attacked to reveal sensitive information on the dataset used to train it and also to evade the model by crafting false data that is classified benign. For privacy attacks, we consider membership inference and model inversion. For the evasion attack, the charging requests should contain low SoC values so that the malicious EV can gain high charging priority. Additionally, the attacker has full knowledge of the detector (i.e., white-box setting). White-box attacks are much more successful comparing to black-box attacks because attackers have full knowledge of the model's architecture, parameters, and training data, allowing them to understand the weaknesses and

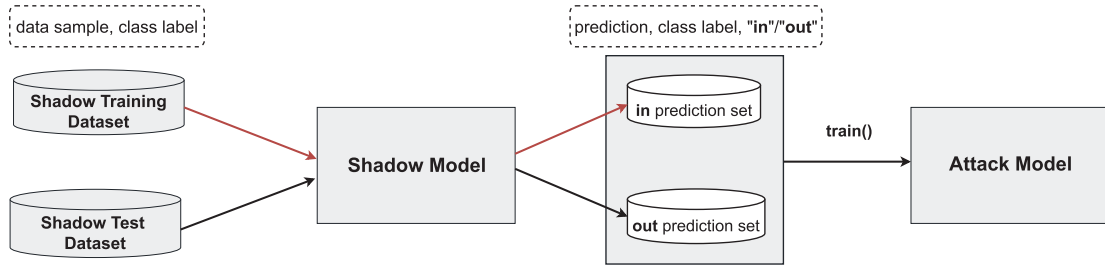


Fig. 2. The membership inference attack [5]. The attacker trains an attack model using a shadow model that replicates the behavior of the target model. The shadow model generates a dataset used to train the attack model, with labels indicating whether each sample is in or out of the training dataset. The shadow model is trained assuming the attacker can sample a dataset from the same distribution as the original dataset used to train the target model.

TABLE I  
PARAMETERS DEFINITION FOR THE MEMBERSHIP INFERENCE  
ATTACK ALGORITHM

Algorithm Parameters	Definition
ShadowModelTrainDataset	Training dataset for shadow model.
ShadowModelTestDataset	Testing dataset for shadow model.
attackModel	The attack model used by the attacker to decide whether the sample is in the training dataset of the original model or not.
MembershipDataset	Set of data samples that the attack model identifies as being used during the training process of the target model.

vulnerabilities of the model, and thus launching successful attacks [21]. White-box attack is a serious attack, and accordingly, it has been considered in several works in the related works such as [20], [21], [22], and [6]. Practically, there are several ways that can be used by the attackers to launch while-box attacks, as follow.

- *Insider Threats* [23]: An attacker with an access to the utility's network, systems, and data, such as a current or former employee, could potentially obtain sensitive information about the machine learning model.
- *Cyberattacks* [24]: Cyberattack could be launched to hack the network and gain access to the data and the machine learning model's parameters.
- *Model Reverse Engineering* [25]: Attackers could reverse engineer the machine learning model and retrieve its parameters and architecture, e.g., by launching model extraction attacks.

The attack methodology of each of the three attacks are discussed in the following subsections.

1) *Membership Inference Attack*: Fig. 2 shows the membership inference attack model. The attack is illustrated in Algorithm 1 and the attack parameters are defined in Table I. The attack steps could be described as follows [5]:

- The attacker trains an attack model that can decide whether the input data is in the training dataset or not (line 15–17 in algorithm).

#### Algorithm 1: Membership Inference Attack Used in [5].

```

1 function MembershipInferenceAttack
  (ShadowModelTrainDataset, ShadowModelTestDataset)
2   ShadowModelDatasetLabel ←
    TargetModel(ShadowModelTrainDataset)
3   shadowModel ←
    TrainClassifier(ShadowModelTrainDataset,
    ShadowModelDatasetLabel)
4   ShadowAttackTrainDataset ← split
    ShadowModelTrainDataset
5   AttackModelDatasetLabel_in ←
    shadowModel(ShadowAttackTrainDataset)
6   AttackModelDatasetLabel_out ←
    shadowModel(ShadowModelTestDataset)
7   AttackModelDatasetLabel ←
    AttackModelTrainDatasetLabel_in +
    AttackModelTrainDatasetLabel_out
8   AttackModelDataset ← ShadowAttackTrainDataset +
    ShadowModelTestDataset
9   attackModel ← TrainClassifier(AttackModelDataset,
    AttackModelDatasetLabel)
10  Return attackModel
11
12 function AttackerMain ()
13   ShadowModelTrainDataset ← {x1, ..., xm}
14   ShadowModelTestDataset ← {w1, ..., wm}
15   AttackModelTestDataset ← {z1, ..., zm}
16   attackModel ←
    MembershipInferenceAttack(ShadowModelTrainDataset,
    ShadowModelTestDataset)
17   ResultDataset ← targetModel(AttackModelTestDataset)
18   MembershipDataset ← attackModel(ResultDataset)
  
```

- A shadow model that replicates the behavior of the target model is used to generate the dataset needed to train the attack model, where the label of each sample in the dataset is either "in" or "out" when the sample is in the training dataset or not, respectively (lines 5–6 in algorithm). As a result, the attack model is trained to classify the input data to in/out using supervised learning (lines 7–9 in the algorithm).
- To train the shadow model, we assume that the attacker is able to sample a dataset from the same distribution of the original dataset used to train the target model (lines 12–13 in the algorithm).
- 2) *Model Inversion Attack (MI)*: Given a detection model with function, defined as  $f$ , that takes a feature vector  $\{x_{i,1}, x_{i,2}, \dots, x_{i,d}\}$ , where  $i$  is either dimension 0 which represents the SoC values or dimension 1 which represents the



**Algorithm 2: Model Inversion Attack Using Confidence Values.**


---

```

1 function ModelInversionAttack (targetDataSample)
  // targetDataSample is the sample that contains
  // the sensitive information which the attacker
  // wants to reveal
2 for  $x_{1,i}, \dots, x_{1,j} \in \text{targetDataSample}$  :
3   targetDataSample  $\leftarrow$ 
4    $\{x_{0,1}, \dots, x_{0,48}, x_{1,0}, \dots, x_{1,i}, \dots, x_{1,j}, \dots, x_{1,48}\}$ 
5   ConfidenceValues  $\leftarrow$ 
6   targetModel(targetDataSample)
7   if ConfidenceValues  $\geq$  threshold
8     Return targetDataSample
9 end

```

---

**Algorithm 3: Model Inversion Attack Using Membership-Inference Attack.**


---

```

1 function ModelInversionAttack (targetDataSample)
  // targetDataSample is the sample that contains
  // the sensitive information which the attacker
  // wants to reveal
2 attackModel  $\leftarrow$  MembershipInferenceAttack()
3 for  $x_{1,i}, \dots, x_{1,j} \in \text{targetDataSample}$  :
4   targetDataSample  $\leftarrow$ 
5    $\{x_{0,1}, \dots, x_{0,48}, x_{1,0}, \dots, x_{1,i}, \dots, x_{1,j}, \dots, x_{1,48}\}$ 
6   Decision  $\leftarrow$  attackModel(targetDataSample)
7   if Decision = True
8     Return targetDataSample
9 end

```

---

location values and  $d$  is the vector length per dimension which is equal to 48, the output  $y$  can be described by the equation  $y = f(x_{0,1}, \dots, x_{0,d}, x_{1,0}, \dots, x_{1,d})$ . The MI attack aims to infer the values of the location features  $\{x_{1,0} \dots x_{1,48}\}$ , which are the sensitive features, i.e., the locations visited by an EV's driver. As illustrated in Algorithms 2 and 3, the attacker tries different values for the location features (line 3 in Algorithm 2 and line 4 in Algorithm 3) and then uses two different methods to determine whether the location features the attacker guessed are true. In the first method, the attacker analyzes the confidence scores of the detector to determine whether the inputs are in the training dataset as depicted in Algorithm 2. Note that, usually the confidence scores of machine learning models are higher when the inputs are in the training dataset (lines 5–8 in the algorithm). The second method uses the membership inference attack model to determine whether the input sample guessed by the attacker is in the dataset or not as illustrated in Algorithm 3, the attacker tries several different values for the location features and then use the membership inference attack model to confirm whether the inferred sample was in the training dataset of the target model or not (lines 4–9 in the algorithm).

3) *Evasion Attack*: In the evasion attack (which is illustrated in Algorithm 4), the attacker's main goal is to evade the detector to classify false SoC values as benign. The attack crafts a malicious sample that not only evades the detector, but also has lower SoC value to gain higher charging priority. This is done by multiplying a true sample by a parameter called,  $\beta$ , as follows:  $x^* = x \times \beta$ , where  $x^*$  is the false sample and  $x$  is the correct

**Algorithm 4: Evasion Attack.**


---

```

1 function EvasionAttack (maliciousSample)
2    $\beta \leftarrow 0.1$ 
3   while decision  $\neq$  benign :
4     maliciousSample  $\leftarrow$  maliciousSample  $\times \beta$ 
5     decision  $\leftarrow$  attackModel(maliciousSample)
6      $\beta \leftarrow \beta + 0.01$ 
7   Return maliciousSample
8 end

```

---

**Algorithm 5: Benign SoC Dataset Creation.**


---

```

1 function SoC_Computation (Distance)
2   SoC0  $\leftarrow$  0
3   for  $i \in \text{Distance}$ 
4     SoCi+1  $\leftarrow$  SoCi - (ConsumptionRate  $\times$ 
5     Distance)/MaxCapacity
6     if SoCi+1 < 0.15
7       CSnearest  $\leftarrow$  EV
8       SoCi+1  $\leftarrow$  SoCi + (ConsumptionRate  $\times$ 
9       Distance)/MaxCapacity
10    end
11  end
12  Return SoC

```

---

**Algorithm 6: Benign LoC Dataset Creation.**


---

```

1 function LoC_Computation (Longitude, Latitude)
2   Cell  $\leftarrow \{cell_1, \dots, cell_{22000}\}$ 
3   for  $i \in \text{Longitude}$ 
4     longitudei  $\leftarrow$  Interpolate(longitudei+1 +
5     longitudei-1, TimeDifference(i + 1, i - 1))
6   end
7   for  $i \in \text{Latitude}$ 
8     latitudei  $\leftarrow$  Interpolate(latitudei+1 +
9     latitudei-1, TimeDifference(i + 1, i - 1))
10  end
11  for  $i \in \text{Longitude and Latitude}$ 
12    Distancei  $\leftarrow$ 
13    DistanceComputes(longitudei, latitudei,
14    longitudei+1, latitudei+1)
15    if Longitude and Latitude  $\in$  celli
16      LoCi  $\leftarrow$  celli
17    end
18  end
19  Return LoC, Distance

```

---

one (line 4 in the algorithm). The initial value of  $\beta$  is 0.1 (line 2 in the algorithm) and then the attacker increases it gradually by 0.01 (line 6 in the algorithm) until the classifier changes its classification to benign (line 3 in the algorithm).

**III. DATASET**

A benchmark dataset that could be used in charging coordination of electric vehicles is not available. Accordingly, in this subsection, we describe how we created the dataset needed for training our false-data detector using real vehicles' routes and technical data supplied by EVs manufacturers [26]. Despite the fact that our primary goal is the creation of a dataset for charging coordination, the dataset can also be used for other purposes such as training a charging load predictor that could

TABLE II  
PARAMETERS DEFINITION FOR ALGORITHMS 5 AND 6

Algorithm Parameters	Definition
Longitude	Vector that contains the longitude values of EV
Latitude	Vector that contains the latitude values of EV
Distance	Vector that contains the distance values between each two consecutive points (represented by longitude and latitude) of EV
LoC	Vector that contains the locations (represented by cells) of an EV at different times
SoC	Vector of computed SoC values of an EV at different times

TABLE III  
TECHNICAL DATA OF THE SELECTED FOUR EVS BRANDS USED IN OUR EXPERIMENTS

Brand	Kia	Tesla	Volks	Nissan
Model	Soul	Model 3	ID.3 Pure	Leaf
Battery Capacity (kwh)	64	40	45	36
Avg. Electric Range (km)	370	265	280	220
Max. Charge Rate (kw/min)	1.45	2.06	1.0	1.02
Power Consumption Rate (Wh/km)	170.87	149	161	164

predict the amount of energy needed in a given region for energy management [19].

#### A. Benign Dataset

The dataset is created using real routes of 536 vehicles in San Francisco, California, published in [9] that include timestamps, latitudes, and longitudes. Our application needs a dataset that contains the SoC values and the locations of EVs every 30 minutes. To compute these values, we use the technical parameters of four EVs brands that are published by their manufactures. These brands are Kia Soul [4], Tesla [10], Volkswagen [12], and Nissan-leaf [11] and their technical data are given in Table III. These parameters are used to compute the SoC values at different times for 200 EVs, including 50 EVs from each brand as illustrated in Algorithm 5. Table II defines the parameters used in this algorithm. Because we compute the SoC values over short time intervals, we use a linear model similar to the existing works in the literature such as [27], [28], [29], [9]. As the SoC value changes linearly with respect to the distance driven by the EV [4], the following equation is used to compute the SoC value (line 3 in the algorithm).

$$SoC_{new} = SoC_{old} - \frac{ConsumptionRate \times Distance}{MaxCapacity} \quad (1)$$

where *ConsumptionRate* is the power consumption rate of the EV, *MaxCapacity* is the maximum capacity of the EV's battery, and *Distance* is the distance traveled by the EV in Km.

Furthermore, the EVs drive to the nearest CS when their SoC values become less than or equal to 15% of the maximum capacity (lines 5–6 in the algorithm). We used the fast-charging method for charging the EVs. In this method, the EVs charge from the grid using direct current (DC) in a small time scale using fast charging protocols such as Combined Charging System (CCS) and Tesla Supercharging [10]. At charging time, the SoC value of the EV increases according to the following formula (line 7 in the algorithm):

$$SoC_{new} = SoC_{old} + \frac{ChargeRate \times Time}{MaxCapacity} \quad (2)$$

where *ChargeRate* is the power charging rate of the EV and *Time* is the charging time of the EV in minutes.

The EV's location is expressed by the longitude and latitude in the dataset of real routes of vehicles [9]. Since the GPS reporting is not uniform [9], the longitude and latitude values in [9] are not periodic. To solve this problem we use the interpolation method to compute the missing longitude and latitude values in the dataset [30] as illustrated in Algorithm 6 (lines 3–8 in the algorithm). Table II defines the parameters used in this algorithm. However, using the longitude and latitude values to represent the EVs' locations is inefficient especially when it is used as an input for a machine learning model. Accordingly, the longitude and latitude of each EV is mapped into a cell number. The vehicles were mainly in the San Francisco city, and accordingly, we divided the city into 22,000 equal area regions, called cells, and each cell has a unique identifier (line 2 in the algorithm). Then, the longitude and latitude values of the EVs are mapped into cell identifiers (line 10–12 in the algorithm). Each EV is represented in the dataset by 20 rows that give the EV's driving pattern across 20 days. Each row has 48 elements for one day where each element gives the EV's SoC value and its cell number (spatial information) every 30 minutes. As a result, the benign dataset has 4000 (200 × 20) data samples.

To evaluate the periodicity of the data, the results of the auto-correlation function (ACF) of the SoC values of four randomly selected EVs are plotted in Fig. 3.

The ACF measures the correlation between a time-series data and a lagged version of itself. In other words, the ACF measures how a time-series data is correlated to its past values. The ACF is typically plotted as a function of lag *k*, where the lag represents the time difference between the current value of a time-series data and the lagged value being correlated. The ACF ranges from -1 to +1, where +1 indicates perfect positive correlation, 0 indicates no correlation, and -1 indicates perfect negative correlation. Positive correlation indicates that the time-series values tend to be in the same direction, while negative correlation indicates that the time-series values tend to be in opposite directions. Zero correlation indicates that there is no relationship between the data and lagged values. In the ACF plot, the X-axis represents the lag or the time difference between the values being compared. If autocorrelation value is positive at lag *z*, it indicates that there is a positive correlation

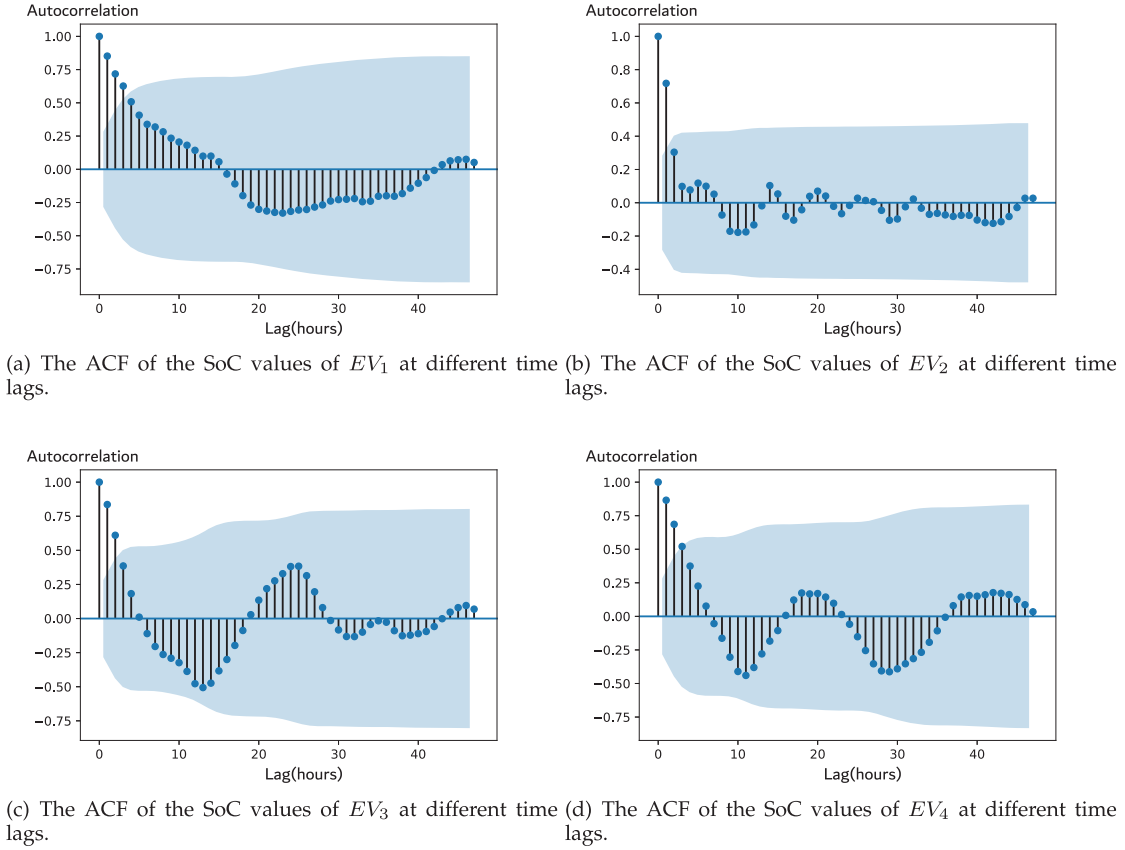


Fig. 3. The ACF of the SoC values of four EVs at different time lags. According to the acf diagrams of the four EVs, there is a positive correlation at 48-time lag which reflects the periodicity of the daily driving pattern of these EVs.

TABLE IV  
THE CROSS-CORRELATION BETWEEN THE SPATIAL INFORMATION  
AND THE SoC VALUES

	EV1	EV2	EV3	EV4
<b>Corr</b>	0.127	0.202	0.115	0.151

with the time-series values that are  $z$  time units apart. In other words, as a time-series data increases (or decreases), the lagged  $z$  data-series data also increases (or decreases). The bounds of the ACF depend on the data being analyzed and the lag being considered.

In our case, the ACF of our time-series data that represents the SoC values of a given EV is measured over a lag range of 48 hours, with each data point representing a one-hour sample interval. The blue shaded area of the ACF plot indicates the 95% confidence interval of the ACF values. According to the figure, there is a positive correlation at 48-time lag. This reflects the periodicity of the daily driving pattern of these EVs. In addition, Table IV gives the spatial-temporal correlation between the spatial information and the SoC values of the four EVs using the cross-correlation function. The function measures the change of two time series data vectors with respect to each other. The cross-correlation values in Table IV indicate that *there is a correlation between the SoC values and the spatial information of each EV*. Therefore, the results given in Fig. 3

TABLE V  
PARAMETERS DEFINITION FOR ALGORITHM 7

Algorithm Parameters	Definition
SoC	Vector of the computed SoC values of a given EV
Distance	Vector of the distances the EV has moved between two consecutive time periods
ReportedSoC	Vector of the data reported from a given EV

and Table IV indicate that there is a correlation between the data of the charging requests of the EVs. Therefore, *to obtain high performance, our detector should learn this correlation* so that it can identify false data which deviates from the correlation learned by the detector.

### B. Malicious Dataset

To create the malicious dataset, an attack methodology is explained in this section to enable the attacker to send false SoC values for gaining high charging priority without being detected easily as illustrated in Algorithm 7. Table V defines the parameters used in this algorithm. The attack scenario was formulated to ensure that the low SoC values reported by an EV are consistent with its driving pattern; otherwise, the attacks

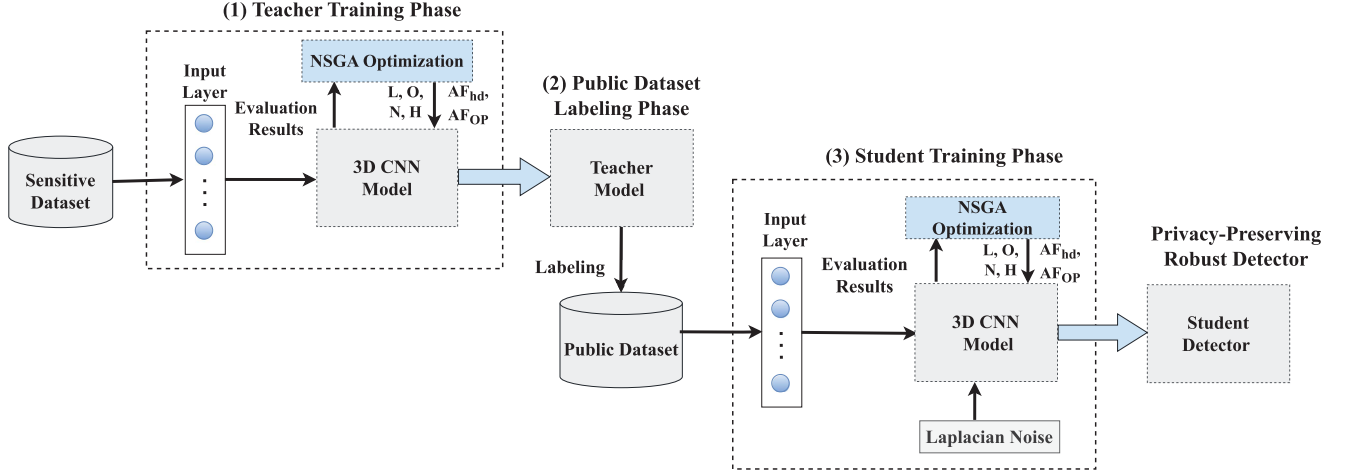


Fig. 4. The proposed approach for training a privacy-preserving and robust false-data detector. A 3D CNN model is used for capturing the spatial-temporal correlations that are in the data. Three techniques are used in the training process of the detector to be able to thwart the adversarial attacks. These techniques are mimic learning, dropout generalization, and differential privacy.

could be easily detected. To do that, we made sure that the difference between two consecutive SoC values reported by the EV is consistent with the distance driven by the vehicle. In this attack, the EV multiplies the SoC value in the beginning of each stopping period by a day-dependent factor  $\alpha_i(d, t)$  whose value is less than one, where  $RS_i(d, t) = \alpha_i(d, t)S_i(d, t)$ ,  $RS_i(d, t)$  is the reported SoC value of  $EV_i$  to the RCC at time  $t$  on day  $d$  and  $S_i(d, t)$  is the real SoC value (line 4 in the algorithm). If  $EV_i$  is honest, then  $RS_i(d, t)$  is equal to  $S_i(d, t)$ . After that, the rest of the reported values change similar to the changes in the real SoC values as given in the following equation (line 7 in the algorithm):  $RS_i(d, t) = RS_i(d, t-1) + \Delta S_i(d, t)$ , where  $\Delta S_i(d, t) = S_i(d, t) - S_i(d, t-1)$ . However, since the malicious EV reports SoC value that is decreased by the factor of  $\alpha_i(d, t)$  at the beginning of each stop, the malicious EV should ensure that this value does not become negative at some point of time due to decreasing it as the EV is moving. Accordingly, the EV needs to report an initial SoC value that ensures that the SoC value is positive until the next stop. This can be done by ensuring that the initial SoC value is enough for the trip.

This attack was applied on the daily driving pattern of the EVs' dataset resulting in a malicious dataset with 4000 records. By combining both datasets (benign and malicious), we have a dataset with 8000 records that was used to train and evaluate the false-data detector. Additionally, we assume that the EVs report correct locations all the time. This is because it is easy to detect the EVs that report false locations, e.g., by using location verification techniques [31].

#### IV. THE PROPOSED DETECTOR

Fig. 4 shows the proposed approach for training a privacy-preserving and robust false-data detector. We use a 3D CNN model for capturing the spatial-temporal correlations that are in the data to increase the detection efficiency.

In addition, we use three techniques during the training of the detector to be able to thwart the adversarial attacks. These

#### Algorithm 7: Malicious Dataset Creation.

```

1 function Malicious_Attack (SoC, Distance)
2   for  $t \in T$ 
3     if  $Distance_t == 0$ 
4       |  $ReportedSoC(t) \leftarrow \alpha_i(t) \times SoC_t$ 
5     end
6     if  $Distance_t \neq 0$ 
7       |  $ReportedSoC(t) \leftarrow$ 
8         |  $ReportedSoC(t-1) + (SoC_t - SoC_{t-1})$ 
9     end
10  Return  $ReportedSoC$ 

```

techniques include mimic learning, dropout generalization, and differential privacy. In the following, we explain how these techniques can secure our detector.

In *mimic learning*, the knowledge is transferred from the original model that is trained on the original sensitive dataset (called *teacher model*) to other shareable model trained on insensitive data (called *student detector*). The knowledge is transferred from the teacher model to the student detector by using the teacher model to annotate insensitive dataset that is used to train the student detector. We use the mimic learning technique to encounter the evasion attack because it reduces the amplitude of the network gradients that can be exploited by adversaries to craft adversarial samples. In addition, the usage of mimic learning technique can also help mitigate the model inversion attack.

*Dropout generalization technique* creates a generalized non-overfitted model. Generalization is very important to avoid overfitting during the model training to produce an accurate model that could give good performance when encountering a data that is seen for the first time. Additionally, generalization can mitigate the membership inference attack [5] as overfitting can give the attacker a good indication about the membership of a given sample in the training dataset. In another words, if a model classifies a given sample with high confidence score, then this indicates that the sample has been seen before during



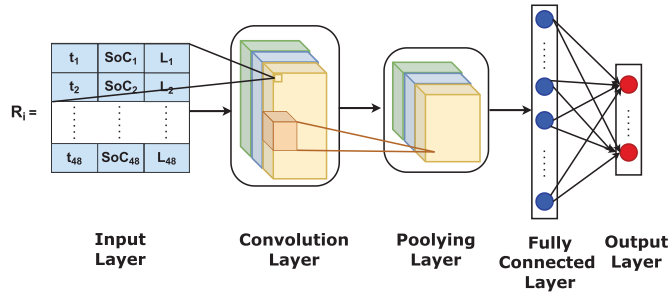


Fig. 5. An abstract view of the 3D CNN architecture. The model has input, convolutional, pooling, fully connected, and Softmax layers. The input to the CNN model is a 3D vector  $R_i$  that contains the reported SoC values and spatial information of EV  $i$  during day  $d$  with a vector size of 48.

the training phase. Accordingly, avoiding overfitting is an effective technique to mitigate the membership inference attack. Moreover, according to [32], generalization is important to mitigate the evasion attack because if the model is generalized, it can perform well on the test data (the data that has not been seen before) and it becomes hard to evade it easily. During the training of a generalized model, the models' gradients are smoothed which allows the model to classify the unseen samples correctly. Accordingly, crafting a malicious sample to evade the generalized model becomes hard. In this article, we choose to use the *Dropout generalization* technique for its efficiency in training generalized deep-learning models [16].

In *differential privacy (DP)*, noise is added to the models' parameters to preserve the privacy of the training dataset. *DP* is effective to mitigate the membership inference attack because adding noise hinders the ability of the attacker to detect the membership of a given sample from the output of the model. Additionally, *DP* is an effective technique for thwarting the model inversion attack. However, there is a trade-off between increasing the amount of noise for increasing the robustness of the model and the accuracy.

In the following subsections, our proposed technique is explained in more details.

#### A. Teacher Model

We select CNN for our detector because it can learn the spatial-temporal correlation in the SoC and location information reported by each EV which results in detecting the false data accurately. As shown in Fig. 5, the model has input, convolutional, pooling, fully connected, and Softmax layers. The input to the CNN model is a 3D vector  $R_i$  that contains the reported SoC values and spatial information of EV  $i$  during day  $d$  with a vector size of 48. The *Convolutional layer* is a feature extraction layer that uses a series of filters to traverse a given array and produce a feature (activation) map. The model extracts spatial-temporal correlation using several filters. To move over the input, each filter employs a pre-defined stride length. The provided 3D vector is then transformed into representative characteristics. By traversing through the input data to learn significant features, the convolution process preserves the spatial-temporal relationship of the reported 3D vector. The resulting feature map depicts

the locations where the feature resides in the array by using the linear combination between the filter and the array values. For each feature, each filter explores and generates a feature map.

The *pooling layer* performs a non-linear down-sampling. The transfer of information across the convolution layers is made easier by ignoring numerous representations of the same element. To combine surrounding values into a single representation, many strategies are utilized, such as taking the maximum value of a group of adjacent cells or averaging them. Finally, the *fully connected (dense) layer* makes the classification. It is a standard feed-forward layer made up of a single layer of neurons, each with its own activation function. This layer's input is flattened and the output is fed to the *Softmax* layer for classification. The last layer produces a vector having the form  $(1, 0)^T$  to indicate true data, while it produces a vector having the form  $(0, 1)^T$  to indicate false data. To train the model, we use the dataset discussed in Section III which has data for 200 EVs.

Additionally, determining the optimal model parameters is a hard optimization problem that could be solved using an optimization technique. The parameters that need to be tuned for the 3D CNN model are the number of layers ( $L$ ), the activation function of the hidden layers ( $AF_{hd}$ ), the activation function of the output layer ( $AF_{op}$ ), optimization algorithms ( $O$ ), loss function ( $H$ ), number of filters ( $NF$ ), and kernel size ( $KS$ ). In this article, we select the Non-dominated Sorting Genetic Algorithm (NSGA-II) multi-objective optimization technique [4] to find the optimal parameters of the model that maximize the overall objective function ( $DR - FA$ ), where  $DR$  is the detection rate and  $FA$  is the false alarm.

In this subsection, we have explained how the teacher model is trained, but this model is vulnerable to privacy and evasion attacks. As explained in Section II, the attacker could launch privacy attacks for retrieving sensitive information regarding the dataset that has been used to train the detector. In our application, the visited locations by an EV's driver are sensitive information that should not be revealed as the EV's driver can be identified from these locations. In the next subsection, we explain how the teacher model is used to train a shareable student detector that is privacy-preserving and robust against evasion attacks.

#### B. Privacy-Preserving and Robust Student Detector

As discussed in Section IV, to train a robust and privacy-preserving detector, we combine and optimize three techniques including mimic learning, dropout generalization, and differential privacy. In this subsection, we explain our training approach.

1) *Mimic Learning Approach*: We use the dataset discussed in Section III and the teacher model discussed in Section IV-A to create a student detector using mimic learning. The idea is that the teacher model is used to annotate and convert publicly available and unlabeled data into labeled data. The *student detector* is then created by training a new 3D CNN model on the newly labelled data. The annotation process is the method by which the teacher model transfers its knowledge (learned from the sensitive dataset) to the student detector. The mimic learning approach is used to thwart the evasion attack as it reduces the amplitude of network gradients that can be exploited by

adversaries to craft adversarial samples. This forces the attacker to add a large amount of perturbations to the crafted samples to be able to evade the model, which makes the process of crafting a malicious sample that could be classified benign by the model harder. Mimic learning also helps to thwart the model inversion attack as the attacker infers the model inputs of the insensitive public data samples instead of inferring the sensitive data samples.

2) *Dropout Technique*: Overfitting is a term used in data science to describe a machine learning model that fits its training data perfectly [16]. When this occurs, the model is unable to function accurately against unseen data. Several techniques have been introduced for overcoming the over-fitting problem, and the most common one is the dropout technique [33]. Dropout is a generalization technique that has been introduced to encounter the over-fitting problem in neural networks. It is a generalization approach for simulating the training of a large number of neural networks with different architectures concurrently and combining (or averaging) them to produce the final architecture. This is done by ignoring (or dropping out) a group of randomly chosen neurons temporally during the training phase. Dropping neurons means ignoring the output of their activation function during the forward pass and the weights of their links are not updated during the backward pass of the neural network training phase. In other words, it simulates having a reduced network without the existence of these nodes and their in and out links. Accordingly, this produces a final generalized (non-overfitted) neural network architecture that is a combination of all of these reduced networks.

Dropout technique is used for its efficiency in training generalized deep-learning models that are able to thwart membership inference attack. The randomness introduced by the dropout technique reduces the confidence score of the seen samples and makes it difficult for the attacker to differentiate between the confidence scores of seen and unseen samples. Additionally, a generalized deep-learning model that is trained using dropout is able to thwart the evasion attack. During the training process, the dropout technique smooths the gradients which reduces the rate of change of the model output with respect to the input which means that to change the class of an adversarial sample, a large amount of perturbations should be added to the input sample. Accordingly, the difference between the malicious sample and the evasion one may not be large which may not lead to a big increase in the charging priority of the EVs.

3) *Differential Privacy (DP)*: Differential privacy [34] is originally introduced for preserving privacy in the applications that retrieve aggregated statistics from databases. The DP's main goal is to ensure that the responses of a database's queries do not exhibit significant statistical disparities comparing to a similar database that has one different record. In case of machine learning models, the datasets are equivalent to databases and adjacent datasets differ in only one sample [35]. The DP definition is as follows: a randomized mechanism  $M:D \rightarrow R$  with domain  $D$  and range  $R$  satisfies  $(\epsilon, \delta)$ -DP, if for any two adjacent inputs  $d, d' \in D$  and for any subset of outputs  $S \subseteq R$ ,

$$Pr[M(d) \in S] \leq e^\epsilon Pr[M(d') \in S] + \delta \quad (3)$$

where  $\epsilon$  is the privacy budget that controls the level of privacy protection provided by an algorithm. Specifically,  $\epsilon$  is a measure for the ability of distinguishing between two outputs of the algorithm when it is applied on two different datasets  $d$  and  $d'$  that differ by only one individual's data. In other words, it measures the ability of an attacker to infer information about a specific individual by observing the output of the algorithm. A smaller value of  $\epsilon$  indicates a stronger level of privacy protection because it indicates that the probability of observing any given output of the algorithm is more evenly distributed across all possible inputs. This, in turn, makes it harder for an attacker to infer information about a specific individual's data from the output of the algorithm. However, a smaller value of  $\epsilon$  also indicates that the algorithm may need to add more noise to the data in order to achieve the desired level of privacy protection. This may reduce the accuracy of the algorithm. On the contrary, a larger value of  $\epsilon$  may result in more accurate outputs but indicates weaker privacy protection, and thus, attackers may be able to infer more information about specific individuals from the output of the algorithm.  $\delta$  is the probability with which this privacy guarantee may not hold [5].

In this article, we add noise to the models' parameters during the training process using differentially private stochastic gradient descent (dpSGD) algorithm [17]. The dpSGD technique has two important phases which are the privacy accountant and the sanitizer. The sensitivity of each sample, which is how the model's output is affected by the sample's existence, is limited in the sanitizing phase by clipping the gradient of the sample  $\nabla L(\theta_t)$  and then adding perturbation to the gradient in batches before parameters are updated at time  $t + 1$  using the following equation:  $\theta_{t+1} = \theta_t - \eta \nabla L(\theta(t))$ , where  $\theta_t$  is a vector of the model's parameters at time  $t$  and  $\eta$  is the learning rate. The training privacy loss, which is the amount of change in the model's output due to the existence of a training sample, is monitored during the privacy accountant phase.

The noise distribution that we used for providing the DP to our student classifier is the Laplace distribution that has been proven to provide  $\epsilon$ -differential privacy [17]. The privacy guarantee is affected by  $\epsilon$ , where large  $\epsilon$  leads to a strong privacy guarantee but with reduced detector's performance. DP can mitigate the membership inference attack because adding noise makes it difficult to differentiate between the samples that are in the training dataset and the unseen samples by analyzing the model's outputs. Specifically, adding noise makes it difficult to differentiate between the confidence scores of the seen and unseen samples. Additionally, DP is used to mitigate the model inversion attacks as adding noise makes it difficult for the attacker to infer the values of the sensitive features of the input sample.

Furthermore, the privacy budget of the DP mechanism is reduced with the increase of the number of operations (queries) that are done on the dataset. The privacy budget is the maximum amount of privacy loss that is allowed in a system while still guaranteeing the preservation of privacy [17]. When a query is made to a system (model), noise (randomness) is intentionally introduced into the system (model) response. The main purpose of this noise is to obfuscate the true values or information

related to individuals in the dataset, thereby protecting their privacy. However, as the number of operations (queries) increases, the available privacy budget per query decreases as more information regarding the dataset could be deduced from the responses. This reduction in the per-query privacy budget can result in a higher level of noise being added to each response. Consequently, the increased noise may affect the accuracy of the results obtained from the system (model). Several theories have been introduced to monitor the privacy budget of a given system. The composition theorem defines the overall comparable level of privacy that can be achieved for a given model that is responding to a set of queries (or doing a set of computations) to be the union of the privacy level that is preserved for each of these queries (computations). However, the composition theorem allows for only a few iterations over the training dataset before exhausting the privacy budget due to its relaxed limitation on privacy usage [17]. The composition theorem overestimates the privacy loss or risk of revealing sensitive information for each operation (query) which results in a more noise is added to the query responses, even though a lower level of noise could have sufficed to maintain the desired privacy guarantees. In other words, after few iterations, the *DP* mechanism will start adding more noise to the results of the system to keep protecting the privacy of the information related to the dataset, which will cause the accuracy of the model to be reduced. Moment account is a method that is based on the composition theorem that provides a more tighter bound on the protection that is used to protect the result of each query [17]. It monitors the accumulative privacy loss over a series of queries computations to accurately estimate and manage the privacy budget in a more precise manner. Moment account adds noise to the training dataset in batches satisfying a  $(O(qT\epsilon), qT\delta) - DP$  for each training step, where  $L$  is the lot size,  $T$  is the number of rounds and  $q = L/N$  is the sampling rate [17].

## V. EVALUATIONS

The experiments conducted to measure the performance of the proposed detector is discussed in this section, followed by a discussion of the experimental results.

### A. Experiments

The dataset that is explained in Section III is split into training dataset that contains 6400 samples (160 EVs) and test dataset that contains 1600 samples (40 EVs). The training dataset is split into a training dataset for the teacher model and a training dataset for the student detector, each with 3200 samples (80 EVs, 20 from each brand). First, the teacher model is trained using the first training dataset. The training of the teacher model does not include using the dropout technique or the *DP* because the model is not shared and thus we do not need to secure this model against adversarial attacks. Then, the second training dataset is labeled using the teacher model, where each sample is labeled either 0 for benign or 1 for malicious. After that, the student detector is trained using the second (labeled) training dataset. The training of the student detector includes the use of the dropout technique and the *DP* for securing the

TABLE VI  
THE OPTIMAL ARCHITECTURES OF THE *TEACHER* MODEL AND  
*STUDENT* DETECTOR

	Teacher	Student
<b>L</b>	4	3
<b>NF</b>	32	16
<b>KS</b>	3	3
<b><math>AF_{hd}</math></b>	<i>Relu</i>	<i>Relu</i>
<b><math>AF_{OP}</math></b>	<i>Softmax</i>	<i>Softmax</i>
<b>O</b>	<i>Adams</i>	<i>Adams</i>
<b>H</b>	<i>MSE</i>	<i>MSE</i>
<b>D</b>	0.0	0.5

detector against adversarial attacks. Finally, the performance of the teacher model and the student detector is evaluated using the test dataset. Additionally, the NSGA [4] is adopted to find the optimal parameters of the teacher model and student detector from the following parameters:

- $NF = \{64, 128, 256, 512, 768, 1024\}$ .
- $KS = \{3, 4, 5, 6\}$ .
- $AF_{hd} = \{relu, linear, sigmoid, tanh, softsign\}$ .
- $AF_{op} = \{sigmoid, softmax, linear\}$ .
- $O = \{rmsprop, adam, sgd, adagrad, nadam\}$ .
- $H = \{CCE, MSE\}$ .

The optimal architectures of the teacher model and the student detector are given in Table VI.

To evaluate the performance of the detector, the following metrics are used [4]:

- Detection Accuracy (ACC) is the summation of the positive and negative samples to the total number of samples.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

$TP$  is the number of false samples that are correctly classified as false.  $TN$  is the number of true samples that are classified as true.  $FP$  is defined as the number of true samples that are classified as false.  $FN$  is defined as the number of false samples that are classified as true.

- False Alarm (FA) is defined as the ratio of false positives to the total number of true samples.

$$FA = \frac{FP}{TN + FP}$$

- Detection Rate (DR) is the ratio of the number of true positives to the total number of false samples.

$$DR = \frac{TP}{TP + FN}$$

- Area Under the Curve (AUC) is the area computed under the receiver operating characteristic (ROC) curve. The ROC curve is a visualization tool for evaluating the performance of a given classifier. It depicts the relationship between the FA and the DR.

We use the following metric to evaluate the robustness of our detector against membership inference attack [5]:

TABLE VII  
THE PERFORMANCE OF THE TEACHER MODEL AND THE STUDENT DETECTOR

Model	ACC (%)	DR (%)	FA (%)
Teacher	97.20	97.51	2.9
Student_w/o_DP	96.81	96.70	4.9
Student_w_DP (NM = 0.8)	95.31	96.82	5.6
Student_w_DP (NM = 1.3)	87.10	79.53	5.9

The results illustrate the detection accuracy of the model before and after adding the privacy-preserving techniques (including adding different amount of noise).

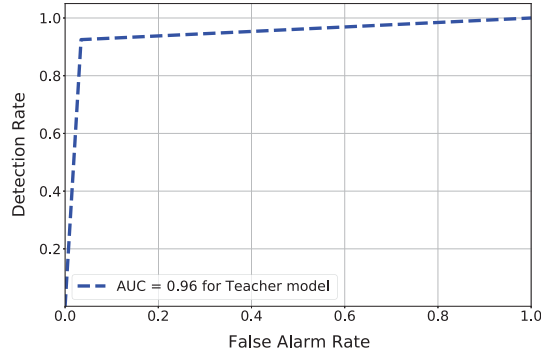


Fig. 6. The ROC of the teacher model. The teacher model has high performance in detecting the false-charging data due to the CNN architecture that is able to extract the spatial-temporal correlation in the data.

- *Attack success rate* is the ratio of the correct membership predictions to the total number of predictions made by the attacker.

The following metric is used to evaluate the robustness of the detector against model inversion attacks [36]:

- *Attack success rate* is the number of samples the attacker is able to find the missing values successfully to the total number of samples.

The following metric is used to evaluate the robustness of the detector against the evasion attack [21]:

- *Model Resistibility (MR)* is the maximum amount of reduction in the true SoC values of a sample to evade the detector, i.e., to be classified as benign.

## B. Results

The detection accuracy of the teacher model is given in Table VII and in Fig. 6. According to the results, the teacher model has high performance in detecting the false-charging data with accuracy of 97.20%, DR of 97.51%, FA of 2.9%, and AUC of 0.96, because of the ability of the CNN architecture to extract the spatial-temporal correlation in the data.

To evaluate the student detector, we first measure the detection accuracy, and then we measure the model's robustness against the membership inference, model inversion, and evasion attacks.

1) *Detection Accuracy*: We measure the detection accuracy of two student detectors (called Student\_w/o\_DP and Student\_w\_DP) trained without and with DP, respectively to

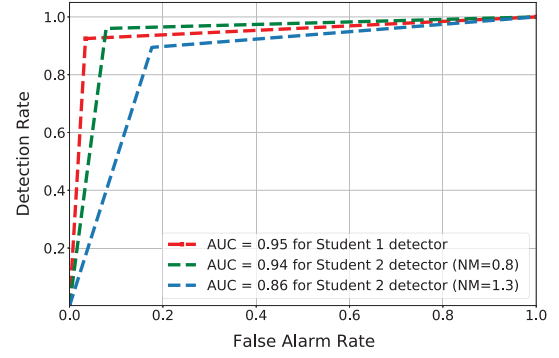


Fig. 7. The ROC of the student detector. The results illustrate that the model's efficiency is slightly affected by the mimic learning but as more noise is introduced, the accuracy decreases for student detector using the DP technique, with a greater decrease observed for higher levels of noise.

study the impact of adding noise to the model. Additionally, during the training of Student\_w\_DP, we trained two detectors for studying the effect of increasing the amount of noise that is added to the model on the accuracy. We used the noise multiplier (NM) parameter to control the amount of noise added to the model's gradients. The NM is a number that represents how much noise is sampled and added to the models' gradients during the update process of the networks weights and generally the higher the number, the more noise is added to models' gradients. The noise multiplier has a direct impact on the  $\epsilon$ -parameter. The higher the value of the noise multiplier (i.e., the lower the value of  $\epsilon$ ), the higher the robustness of the model against privacy attacks but with reduced accuracy. So, we conduct two experiments to train two student detectors, with noise multipliers of 0.8 and 1.3.

As given in Table VII and Fig. 7, the accuracy of Student\_w/o\_DP is 96.81% with DR of 96.70%, FA of 4.9%, and AUC of 0.95. Comparing to the performance of the teacher model, the results indicate that *the mimic learning has a slight impact on the efficiency of the model*. However, when we add noise to the student detector using the DP technique, the accuracy decreases and it decreases more as more noise is used. Student\_w\_DP gives 95.31% accuracy with DR of 96.82%, FA of 5.6%, and AUC of 0.94 with noise multiplier of 0.8, while it gives 87.10% accuracy with DR of 79.53%, FA of 5.9%, and AUC of 0.86 with noise multiplier of 1.3.

2) *Robustness Against Membership Inference Attack*: For evaluating the membership inference attack, experiments have been conducted using a separate dataset that is created from the data of 40 EVs to produce a dataset with 800 samples to train the shadow model needed in the attack, as discussed earlier. These EVs are different from the 200 EVs that have been previously used to create the training and testing datasets of the detector. Additionally, we use the data of another 20 EVs to create unseen dataset with 400 samples. Then, to create the dataset of the membership inference attack model, the shadow model, which is a 3D CNN model with the same architecture as the target model, is queried using a dataset that contains 400 samples from the shadow model's training dataset to produce the training data of the class "in" (the data that is in the training dataset) and the 400 unseen samples to produce the training



TABLE VIII

THE EVALUATIONS OF THE MEMBERSHIP INFERENCE ATTACK AGAINST THE TEACHER AND THE STUDENT MODEL

Model	Success rate (%)
Teacher	91.67
Student_w/o_DP	72.12
Student_w_DP (NM = 0.8)	59.1
Student_w_DP (NM = 1.3)	50.3

The results reflect the effectiveness of the attack on the model before and after adding the privacy-preserving techniques (including adding different amount of noise).

data for the class "out" (the data that is not in the training dataset). Finally, to evaluate the success rate of the membership inference attack against the proposed detector, a separate test dataset with 400 samples has been used. The test dataset contains 200 samples from the training dataset of the detector and 200 unseen samples, i.e., samples from the test dataset of the detector. It should be mentioned that all the datasets are benign.

According to the results given in Table VIII, the success rate of the membership inference attack is too high for the teacher model that does not have a countermeasure, but the success rate significantly decreases in case of the student detectors. Student\_w\_DP is more robust than Student\_w/o\_DP because of using the DP technique. Additionally, the robustness of Student\_w\_DP with NM of 1.3 is higher than the robustness of Student\_w\_DP with NM of 0.8 because the extra noise added increases the robustness of the model but with reduced accuracy as given in Table VII. In the next subsections, we evaluate Student\_w/o\_DP and Student\_w\_DP with NM of 0.8 against evasion and model inversion attacks. We excluded Student\_w\_DP with NM of 1.3 to keep the accuracy of the detector high.

3) *Robustness Against Model Inversion Attack*: The robustness against the model inversion attack is evaluated empirically by launching the attack against the teacher model and the student detector. As discussed in Section II, in the model inversion attack, the attacker has a partial information on a sample and his goal is to find the missing information. To clarify, in membership inference attack, the attacker has a complete sample and he wants to know whether it is in the training dataset, but in model inversion attack, the attacker has an incomplete sample and he wants to complete it and determines whether it is in the training dataset. For instance, the attacker may know the first location in a sample (the victim's home) and some locations such as the victim's workplace and he wants to launch the attack to find the remaining locations that are visited.

We use two different approaches to launch model inversion attacks. In the first approach, the attacker tries different values for the missing information of a sample and then uses the information provided by the model (e.g., the confidence values of the model predictions) to determine if the sample is in the training dataset. In the second approach, the model inversion attack is launched using the membership inference attack model to determine whether a given sample is located in the training dataset. In each of the aforementioned approaches, we consider two cases in which three and five locations are missing.

TABLE IX

THE EVALUATIONS OF THE MODEL INVERSION ATTACKS USING TWO APPROACHES (THE CONFIDENCE VALUES AND THE MEMBERSHIP ATTACK MODEL)

Model	Approach 1		Approach 2	
	3 missing features	5 missing features	3 missing features	5 missing features
Teacher	97.5	94.4	90.5	88.2
Student_w/o_DP	58.5	56.5	70.0	66.3
Student_w_DP	52.5	42.5	60.0	50.0

The results reflect the effectiveness of the attack on the model before and after adding the privacy-preserving techniques (including adding different amount of noise) when different numbers of features are missing.

Additionally, the evaluation is done using 400 samples from the benign samples of the detector's training dataset.

Table IX displays the results of launching the model inversion attack against the teacher model and the student detector. In case that three features are missing, the success rate is decreased from 97.5% in case of the teacher model to 58.5% in case of Student\_w/o\_DP and 52.5% in case of Student\_w\_DP. In case that five features are missing, the success rate is reduced from 94.4% in case of teacher model to 56.5% and 42.0% in cases of Student\_w/o\_DP and Student\_w\_DP, respectively. It can also be seen that comparing the case of three missing features to the case of five missing features, launching the model inversion attack becomes harder when there are more missing features. The reason is that there are more possible values for the missing features which leads to having more candidate samples that could have close confidence scores which confuses the attacker and thus he mistakenly decides that a wrong sample is in the training dataset. In case of using membership inference attack model to launch inversion attack, Table IX shows that the success of the model inversion attack decreases from 90.5% in case of the teacher model to 70.0% and 60% in cases of Student\_w/o\_DP and Student\_w\_DP respectively, in case of three missing features. The table also shows that the success of the model inversion attack decreases from 88.2% in case of the teacher model to 66.3% and 50% in cases of Student\_w/o\_DP and Student\_w\_DP respectively, in case of five missing features. The results given in Table IX confirm that our approach can significantly reduce the success rate of the model inversion attacks.

4) *Robustness Against Evasion Attack*: To evaluate the robustness of our detector against the evasion attack, experiments have been conducted. The attack is launched by multiplying the original SoC values by parameter  $\beta$  to decrease their values. The initial value of  $\beta$  is 0.1 and its value is incremented by 0.01 each time the sample fails to evade the detector. The more the attacker increases  $\beta$ , the closer the reported SoC values become to the true values and the less the effectiveness of the attack because the attacker can not increase its charging priority. To compute the resistibility of our detector, we compute the ratio of  $\beta$  at which the sample evades the model to the initial  $\beta$  (or  $\beta_i$ ) using the following equation:  $MR = \frac{\beta}{\beta_i}$ , where  $\beta_i$  is equal to 0.1 in our experiments. Additionally, the evaluation is done using 400 samples from the benign samples of the detector's testing dataset.

TABLE X  
THE EVALUATIONS OF THE MODEL EVASION ATTACK

Model	MR (%)
Teacher	405
Student_w/o_DP	548
Student_w_DP	650

The results illustrates the amount by which  $\beta$  should be increased to evade the teacher and the student models.

As shown in Table X, the attacker needs to increase  $\beta$  by 135% for evading the teacher model. After using the mimic learning approach combined with the dropout technique, the attacker needs to increase  $\beta$  by 183% to deceive Student\_w/o\_DP. Finally, after adding the noise during the training process, the attacker needs to increase  $\beta$  by 217% for deceiving Student\_w\_DP.

The evaluation results given in Tables VII, VIII, IX, and X and Figs. 6 and 7, demonstrate the efficiency of our methodology for training a robust classifier that can mitigate membership inference, model inversion, and evasion attacks with a slight impact on the detection accuracy.

## VI. RELATED WORK

Although significant attention has been paid to security and privacy of smart metering system [37], [38], [39], [40], [41], [42], [43], [44], little work has focused on EVs. The existing research on spatial-temporal charging coordination mechanisms is presented first in this section. After that, the existing work on cyber attacks against electrical power grids and their countermeasures is illustrated. Then, the existing works on the evasion, model inversion, and membership inference attacks are discussed. Finally, we discuss the research gap that motivated this article.

### A. Spatial-Temporal Charging Coordination Mechanisms

The simultaneous and uncontrolled charging of electric vehicles has the potential to overload the power grid which may lead to blackouts in severe cases [4]. As a result, charging coordination mechanisms are introduced to control the charging load in terms of time and space. The existing mechanisms can be classified into temporal and/or spatial coordination [2]. The purpose of the temporal charging coordination [18] is to find the optimal time slot to serve the charging request of a given EV, while the main goal of the spatial-temporal coordination is to choose the best charging station to charge the EV to avoid overloading the grid and long waiting time at charging stations [2].

Yu et al. [2] have proposed a centralized spatial-temporal charging coordination mechanism using bi-level optimization technique. The objective of the proposed mechanism is to distribute the charging load of the EVs requests spatially (across multiple CSs) and temporally (across a set of time slots in each CS). Regarding the temporal part, the main goal of the mechanism is to optimize the cost of charging and the degree of the driver's satisfaction. From the spatial perspective, the mechanism's main goal is to minimize the load variance. In

the introduced mechanism, the bi-level technique is used for finding an optimized solution to the spatial-temporal problem by converting it into a two-stage hierarchical optimization problem.

Zhang et al. [3] have proposed a distributed spatial-temporal charging coordination mechanism which has two steps. In the first step a decentralized scheme using a bucket sort algorithm determines if a certain EV is authorized to charge at a given CS and current time slot. The second step decides whether the EVs that are not allowed to charge in the current time slot in a given CS are temporally shifted to a later time slot in the current CS or they are spatially shifted to another CS.

López et al. [19] have proposed a charging coordination mechanism using machine learning models that are trained on optimized historical data. The historical data is labeled using Markov Decision Process (MDP) framework that is based on dynamic programming. After that, the models make decisions regarding the charging of the EVs. Machine learning models have the ability to learn the prediction function automatically from the data and removes the need for hand-crafted analytic decision functions that are needed in the optimization cases.

### B. Cyber-Physical Attacks

Several studies have explored cyber attacks on the electrical power grid and proposed solutions to secure both the infrastructure and communication of its components [45].

Inayat et al. [46] conducted a survey on potential cyber attacks on IoT systems and introduced several machine learning based intrusion detection systems. They also compared these methods in regards to the types of attacks they can detect, feature selection methods used by each method, and the datasets used for evaluation.

Ashraf et al. [47] studied the Denial-of-Service (DoS) attack on electrical power grid substations, with a focus on the digital substation automation system. The study found that a DoS attack on the automation system server can cause significant delays in communication between substation components, and in extreme cases, block standard communication protocol messages from reaching their intended recipients.

Musleh et al. [48] presented a real-time signal processing solution to encounter false data injection attacks on digital measurements from various sensors in electrical power grids. The solution utilizes a multi-sensor temporal prediction wide area monitoring controller to ensure the accuracy of the readings. The temporal prediction aspect of the approach effectively addresses false data injection attacks by precisely estimating and controlling voltage magnitude readings that are read from the sensors.

Khalid et al. [49] proposed a detection algorithm for false data injection attacks that target electrical power grids. In the proposed approach the false measurement readings are detected using Bayesian-based detector that is distributed amongst several nodes. The distributed nature of the detector enables it to detect the false data by monitoring the interaction between different grid components.

Shafee et al. [4] have proposed using a machine learning model to detect EVs that send false SoC values in temporal

charging coordination. A Gated Recurrent Unit (GRU) deep learning network is used to extract the temporal features of the charging requests to detect anomalies which indicate the existence of false data. The work in [4] does not study adversarial attacks. Comparing to [4], this article focuses on temporal-spatial charging coordination and aims to secure the detector against privacy and evasion adversarial attacks.

### C. Adversarial Machine Learning

1) *Evasion Attack*: Numerous studies have addressed the evasion attacks and proposed countermeasures. Ayub et al. [20] have investigated launching evasion attack against network intrusion detection model (IDS) that is based on the Multilayer Perceptron (MLP). The Jacobian-based Saliency Map Attack (JSMA) is used for launching the evasion attack against the IDS model. In JSMA, the attacker adds noise to a specific set of input features to evade the model. The experimental results indicate that evasion attack drops the IDS model accuracy against two network standard datasets. Gu, et al. [50] have introduced an evasion attack against deep neural networks that classify images. The attack is based on the iterative gradient shielding (IGS) technique, which allows for gradient-based attacking and region selection. The attack is based on selecting the key parts (features) in the image that have a great impact on the classification decision and add perturbation to those parts only. Adding noise to the key parts of the image samples and ignoring the other parts generates an adversarial sample that can evade the classification model with a small amount of noise.

Xu et al. [21] have studied evasion attacks that could be launched in remote sensing applications that need to classify remote sensing images into several categories using machine learning. The attacker aims to evade the model by adding a small amount of noise to change the category of a specific image. The attack could be targeted in the sense that the label of the image changes to another specific label or it could be untargeted in the sense that the label of the image changes to any other label. Moreover, an adversarial training defense strategy is proposed to mitigate evasion attacks. The idea is that the model is trained on both original data and adversarial samples. Additionally, the efficiency of the defense technique is evaluated when the model is inaccessible by the attackers by studying the transferability of the adversarial samples, which measures the success rate of adversarial samples crafted for a substitute model. According to the evaluation results, the adversarial samples that are generated for one model could evade other deep-learning models and even the traditional shallow classifiers.

Papernot et al. [32] use a distillation technique to counter evasion attacks. Distillation is a knowledge transfer technique that is done between two models that have the same architecture. The class knowledge that is acquired by the first model after the training process is transferred to the second model by labeling the same training dataset using the output probabilities of the first model, and then this dataset is used to train the second model. Papernot et al. [51], have proposed an evasion attack that is launched against remote machine learning models without knowing the internal details of the targeted models.

The proposed attack reduces the efficiency of the distillation methodology proposed in [32].

Yang et al. [52] have proposed a weighted random forest technique for training a robust model against evasion attacks by using the concept of randomization. Machine learning models such as tree-based models use some deterministic equations for selecting the most suitable set of features to perform their classification efficiently. The number of these features is usually small but they have a big impact on the classification decision. Accordingly, if attackers change the values of these features slightly, the model can be evaded easily. Hence, using models that present randomization such as random forest in the classification process is the solution that has been presented in [52]. The idea is that, to classify a given sample the decisions of a randomly selected subset of the decision-tree models are considered. Thus, the attacker has to change the values of a large set of features to evade the classifier, and thus the attack becomes harder.

2) *Membership Inference Attack*: The membership inference attack has been studied in several papers in the literature. The attack makes use of the discrepancy in the output of a model when the input data sample is present in the training dataset versus when it is not. This attack creates a machine learning model that identifies whether an input sample is in the dataset of the model or not [5].

Salem et al. [16] launch membership inference attack using only one shadow model and without knowing the architecture of the target model and the distribution of the dataset. The experimental results show that the attack has a high success rate. Nasr et al. [22] targeted deep neural networks using a white-box version of the membership inference attack in which the attacker has full knowledge about the target model including the architecture, parameters, and weights. The attack exploits the impact of each sample on the models' parameters. Each sample changes a specific group of models' gradients (called gradient features) to minimize the classification error of the model during the training process. The proposed attack trains an attack model to compute the membership probability of a given sample using the extracted gradient features from the target model. Song et al. [53] have investigated the robustness of a countermeasure to an evasion attack such as adversarial training against the membership inference attack. The experimental results *demonstrate that the countermeasure of the evasion attacks makes the model more susceptible to the membership inference attack.*

3) *Model Inversion Attack*: Fredrikson et al. [8] used the confidence levels of a machine learning models' outputs to find the missing values in the input sample when it presents in the training dataset. Maximum a posterior (MAP) estimator is used by attackers to choose the values of sensitive features successfully. However, Fredrikson et al. [6] have demonstrated that the model inversion attack using the MAP estimator is impractical in the applications that have large domains such as face recognition, and a modification is proposed to the attack to find the missing values of the sensitive features using the gradient descent algorithm. Hidano et al. [36] have proposed a model inversion attack that targets a set of sensitive features without any partial knowledge about the other insensitive features. This can be done by inserting adversarial samples during the training



process of the target model [51] to make the model vulnerable to inversion attack without reducing the accuracy of the model.

#### D. Research Gap

Most of the existing works in the literature assume that EVs report true data in their charging requests, but EVs are motivated to report false data to the RCC to charge first in a specific time slot and a given CS. A machine learning model can be used to identify false data, but *in our application, the model is vulnerable to evasion, inversion, and membership inference attacks*. However, the literature usually focuses only on one attack, and mitigating the three attacks simultaneously is not an easy task because of the tradeoff between the countermeasures and the model accuracy and also the conflict between the countermeasures themselves as concluded by [53] where a countermeasure to the evasion attacks makes the model more susceptible to membership inference attacks. Accordingly, we propose a privacy-preserving and robust technique for training a machine learning model to accurately identify false data while being able to thwart the membership inference, model inversion, and evasion attacks.

### VII. CONCLUSION AND FUTURE WORK

In this article, we have proposed a robust and privacy-preserving deep-learning model to detect false data sent by EVs to increase their charging priority. Our detector is based on 3D CNN model because of its ability to correlate the temporal features of the SoC values with the spatial information of EVs to make accurate decisions. For training the detector, we have used three techniques to mitigate membership inference, model inversion, and evasion attacks. Mimic learning technique is used to mitigate the evasion and model inversion attacks using the knowledge transfer concept. Dropout generalization is used to thwart the evasion and membership inference attacks by training a generalized model. Differential privacy is used to mitigate the membership inference and model inversion attacks by adding noise to the model gradients during the training process. The results of our experiments show the superiority of our proposed training approach compared to the baseline 3D CNN teacher model. According to the results, the success rate of the membership inference attack against the teacher model is 91.67 %, and it drops to 50.3 % after applying our countermeasure. Regarding the model inversion attack, the success of the attack against the teacher model in case of three missing features and five missing features are 97.5% and 94.5% respectively, using the first attack approach and 90.5% and 88.2% respectively in case of using the membership inference model to launch the inversion attack. These success rates have been reduced after applying our countermeasures to 52.5% and 42.5% respectively in case of the first attack approach and to 60.0% and 50.0% respectively in case of the second attack approach. In the model evasion attack, to evade the teacher model, the attacker needs to increase  $\beta$  by 135%, while  $\beta$  needs to be increased by 217% to evade the model after applying the proposed countermeasure. Accordingly, the results of our experiments demonstrate that our methodology can train a detector that has the ability to encounter the membership

inference, model inversion, and evasion attacks with a slight impact on its accuracy in detecting false data.

In this article, we have investigated the white-box attacks which are more challenging than other attacks such as black-and gray-box. In our future work, we will investigate the black-box and the gray-box attacks. In the black-box attacks, the attackers are assumed to have no prior knowledge about the detection model, while in the grey-box attacks, the attackers have partial knowledge about the model. To launch these attacks, the attacker might need to build a shadow model either by training it using public data or by launching model extraction attack to approximate the parameters of the original model. We will also investigate mitigation techniques while achieving high detection accuracy.

#### ACKNOWLEDGMENT

The authors gratefully acknowledge technical and financial support from the Ministry of Education and King Abdulaziz University, DSR, Jeddah, Saudi Arabia.

#### REFERENCES

- [1] H. Tu, H. Feng, S. Srdic, and S. Lukic, "Extreme fast charging of electric vehicles: A technology overview," *IEEE Trans. Transport. Electrific.*, vol. 5, no. 4, pp. 861–878, Dec. 2019.
- [2] L. Yu, T. Zhao, Q. Chen, and J. Zhang, "Centralized bi-level spatial-temporal coordination charging strategy for area electric vehicles," *CSEE J. Power Energy Syst.*, vol. 1, pp. 74–83, Dec. 2015.
- [3] J. Zhang, Y. Pei, J. Shen, L. Wang, T. Ding, and S. Wang, "Charging strategy unifying spatial-temporal coordination of electric vehicles," *IEEE Access*, vol. 8, pp. 74 853–74 863, 2020.
- [4] A. A. Shafee, M. Fouda, M. Mahmoud, A. J. Aljohani, W. Alasmay, and F. Amsaad, "Detection of lying electrical vehicles in charging coordination using deep learning," *IEEE Access*, vol. 8, pp. 179 400–179 414, 2020.
- [5] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Secur. Privacy*, San Jose, CA, USA, 2017, pp. 3–18.
- [6] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2015, pp. 1322–1333.
- [7] B. Biggio et al., "Evasion attacks against machine learning at test time," *Mach. Learn. Knowl. Discov. Databases*, vol. 8190, pp. 387–402, Sep. 2013.
- [8] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart, "Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing," in *Proc. 23rd USENIX Conf. Secur. Symp.*, Berkeley, CA, USA, 2014, pp. 17–32.
- [9] H. Akhavan-Hejazi, H. Mohsenian-Rad, and A. Nejat, "Developing a test data set for electric vehicle applications in smart grid research," in *Proc. IEEE 80th Veh. Technol. Conf.*, Vancouver, BC, Canada, 2014, pp. 1–6.
- [10] Tesla, "Tesla features," Accessed: Jan. 2022. [Online]. Available: <https://ev-database.org/car/1320/Tesla-Model-3-Standard-Range-Plus>
- [11] Nissan-Leaf, "Nissan-leaf features," Accessed: Jan. 2022. [Online]. Available: <https://ev-database.org/car/1106/Nissan-Leaf>
- [12] Volkswagen, "Volkswagen features," Accessed: Jan. 2022. [Online]. Available: <https://ev-database.org/car/1127/Volkswagen-ID3-Pure>
- [13] Y. Kwak, K. Kong, W. J. Song, B. Y. Min, and S. E. Kim, "Multilevel feature fusion with 3D convolutional neural network for EEG-based workload estimation," *IEEE Access*, vol. 8, pp. 16 009–16 021, 2020.
- [14] G. R. Harik, F. G. Lobo, and D. E. Goldberg, "The compact genetic algorithm," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 287–297, Nov. 1999.
- [15] A. Shafee, M. Baza, D. A. Talbert, M. Fouda, M. Nabil, and M. Mahmoud, "Mimic learning to generate a shareable network intrusion detection model," in *Proc. IEEE 17th Annu. Consum. Commun. Netw. Conf.*, 2019, pp. 1–6.



- [16] A. Salem, Y. Zhang, M. Humbert, M. Fritz, and M. Backes, "ML-leaks: Model and data independent membership inference attacks and defenses on machine learning models," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, San Diego, CA, USA, 2019.
- [17] J. Zhao, Y. Chen, and W. Zhang, "Differential privacy preservation in deep Learning: Challenges, opportunities and solutions," *IEEE Access*, vol. 7, pp. 48 901–48 911, 2019, doi: [10.1109/ACCESS.2019.2909559](https://doi.org/10.1109/ACCESS.2019.2909559).
- [18] N. Arias, J. Franco, M. Lavorato, and R. Romero, "Metaheuristic optimization algorithms for the optimal coordination of plug-in electric vehicle charging in distribution systems with distributed generation," *Electric Power Syst. Res.*, vol. 142, pp. 351–361, Jan. 2017.
- [19] K. L. López, C. Gagné, and M. Gardner, "Demand-side management using deep learning for smart charging of electric vehicles," *IEEE Trans. Smart Grid*, vol. 10, no. 3, pp. 2683–2691, May 2019.
- [20] A. M. Ayub, W. A. Johnson, D. A. Talbert, and A. Siraj, "Model evasion attack on intrusion detection systems using adversarial machine learning," in *Proc. IEEE 54th Annu. Conf. Inf. Sci. Syst.*, Princeton, NJ, USA, 2020, pp. 1–6.
- [21] Y. Xu, B. Du, and L. Zhang, "Assessing the threat of adversarial examples on deep neural networks for remote sensing scene classification: Attacks and defenses," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 2, pp. 1604–1617, Feb. 2021.
- [22] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. IEEE Symp. Secur. Privacy*, San Francisco, CA, USA, 2019, pp. 739–753.
- [23] B. Li, R. Lu, G. Xiao, H. Bao, and A. A. Ghorbani, "Towards insider threats detection in smart grid communication systems," *Inst. Eng. Technol.*, vol. 13, pp. 1728–1736, Jul. 2019.
- [24] X. Wang, C. Fidge, G. Nourbakhsh, E. Foo, Z. Jadidi, and C. Li, "Anomaly detection for insider attacks from untrusted intelligent electronic devices in substation automation systems," *IEEE Access*, vol. 10, pp. 6629–6649, 2022.
- [25] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction APIs," in *Proc. 25th USENIX Conf. Secur. Symp.*, Austin, TX, USA, 2016, pp. 601–618.
- [26] Shafee et al., "EV Dataset," Accessed: Jul. 2023. [Online]. Available: <https://drive.google.com/file/d/1KGfOrmE1dHpRAvATrBnRDyMREKAY91DA/view?usp=sharing>
- [27] J. Meng, D. I. Stroe, M. Ricco, G. Luo, and R. Teodorescu, "A simplified model-based state-of-charge estimation approach for lithium-ion battery with dynamic linear model," *IEEE Trans. Ind. Electron.*, vol. 66, no. 10, pp. 7717–7727, Oct. 2019.
- [28] M. Zhang and X. Fan, "Review on the state of charge estimation methods for electric vehicle battery," *World Electric Veh. J.*, vol. 11, no. 1, 2020, Art. no. 23.
- [29] D. Castanho et al., "Method for SoC estimation in lithium-ion batteries based on multiple linear regression and particle Swarm optimization," *Energies*, vol. 15, no. 19, 2022, Art. no. 6881.
- [30] J. Fan, P. Zhang, J. Chen, B. Li, L. Han, and Y. Zhou, "Quantitative estimation of missing value interpolation methods for Suomi-NPP VIIRS/DNB nighttime light monthly composite images," *IEEE Access*, vol. 8, pp. 199 266–199 288, 2020.
- [31] T. Tithi, B. Deka, M. R. Gerdes, C. Winstead, M. Li, and K. Heaslip, "Analysis of friendly jamming for secure location verification of vehicles for intelligent highways," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7437–7449, Aug. 2018.
- [32] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proc. IEEE Symp. Secur. Privacy*, San Jose, CA, USA, 2016, pp. 582–597.
- [33] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.
- [34] T. Zhu, D. Ye, W. Wang, W. Zhou, and P. S. Yu, "More than privacy: Applying differential privacy in key areas of artificial intelligence," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 6, pp. 2824–2843, Jun. 2022.
- [35] M. Nabil, M. Ismail, M. Mahmoud, W. Alasmari, and E. Serpedin, "PPETD: Privacy-preserving electricity theft detection scheme with load monitoring and billing for AMI networks," *IEEE Access*, vol. 7, pp. 96 334–96 348, 2019.
- [36] S. Hidano, T. Murakami, S. Katsumata, S. Kiyomoto, and G. Hanaoka, "Model inversion attacks for prediction systems: Without knowledge of non-sensitive attributes," in *Proc. IEEE 15th Annu. Conf. Privacy Secur. Trust*, Calgary, AB, Canada, 2017, pp. 115–126.
- [37] M. Ibrahim, M. Mahmoud, M. Fouda ++, F. Alsolami, W. Alasmari, and X. Shen, "Privacy-preserving and efficient data collection scheme for AMI networks using deep learning," *IEEE Internet of Things J.*, vol. 8, no. 23, pp. 17 131–17 146, Dec. 2021, doi: [10.1109/IJOT.2021.3077897](https://doi.org/10.1109/IJOT.2021.3077897).
- [38] M. Ibrahim, M. Nabil, M. Fouda, M. Mahmoud, W. Alasmari, and F. Alsolami, "Efficient privacy-preserving electricity theft detection with dynamic billing and load monitoring for AMI networks," *IEEE Internet of Things J.*, vol. 8, no. 2, pp. 1243–1258, Jan. 2021, doi: [10.1109/IJOT.2020.3026692](https://doi.org/10.1109/IJOT.2020.3026692).
- [39] A. Takiddin, M. Ismail, M. Nabil, M. Mahmoud, and E. Serpedin, "Detecting electricity theft cyber-attacks in AMI networks using deep vector embeddings," *IEEE Syst. J.*, vol. 15, no. 3, pp. 4189–4198, Sep. 2021, doi: [10.1109/JSYST.2020.3030238](https://doi.org/10.1109/JSYST.2020.3030238).
- [40] M. Badr, M. Ibrahim, M. Mahmoud, M. Fouda ++, F. Alsolami, and W. Alasmari, "Detection of false-reading attacks in smart grid net-metering system," *IEEE Internet of Things J.*, vol. 9, no. 2, pp. 1386–1401, Jan. 2022, doi: [10.1109/IJOT.2021.3087580](https://doi.org/10.1109/IJOT.2021.3087580).
- [41] M. J. Abdulaal et al., "Real-time detection of false readings in smart grid AMI using deep and ensemble learning," *IEEE Access*, vol. 10, pp. 47 541–47 556, 2022, doi: [10.1109/ACCESS.2022.3171262](https://doi.org/10.1109/ACCESS.2022.3171262).
- [42] M. Badr et al., "Privacy-preserving and communication-efficient energy prediction scheme based on federated learning for smart grids," *IEEE Internet of Things J.*, vol. 10, no. 9, pp. 7719–7736, May 2023.
- [43] M. Ibrahim, M. Mahmoud, F. Alsolami, W. Alasmari, A. AL-Ghamdi, and X. Shen, "Electricity-theft detection for change-and-transmit advanced metering infrastructure," *IEEE Internet of Things J.*, vol. 9, no. 24, pp. 25 565–25 580, Dec. 2022.
- [44] M. M. Badr, M. M. E. A. Mahmoud, M. Abdulaal, A. J. Aljohani, F. Alsolami, and A. Balamsh, "A novel evasion attack against global electricity theft detectors and a countermeasure," *IEEE Internet of Things J.*, vol. 10, no. 12, pp. 11 038–11 053, Jun. 2023.
- [45] M. S. Mahmoud, H. M. Khalid, and M. M. Hamdan, *Cyber-Physical Infrastructures in Power Systems: Architectures and Vulnerabilities*, 1st ed. New York, NY, USA: Elsevier — Academic, 2021.
- [46] U. Inayat, M. F. Zia, S. Mahmood, H. M. Khalid, and M. Benbouzid, "Learning-based methods for cyber attacks detection in IoT systems: A survey on methods, analysis, and future prospects," *Electronics*, vol. 11, no. 9, pp. 1–20, May 2022.
- [47] S. Ashraf, M. H. Shawon, H. M. Khalid, and S. M. Mueen, "Denial-of-service attack on IEC 61850-based substation automation system: A crucial cyber threat towards smart substation pathways," *Sensors*, vol. 21, no. 19, pp. 1–19, Sep. 2021.
- [48] A. S. Musleh, H. M. Khalid, S. M. Mueen, and A. Al-Durra, "A prediction algorithm to enhance grid resilience towards cyber attacks in WAMCS applications," *IEEE Syst. J.*, vol. 13, no. 1, pp. 710–719, Mar. 2019.
- [49] H. M. Khalid and J. C.-H. Peng, "A Bayesian algorithm to enhance the resilience of WAMS applications against cyber attacks," *IEEE Trans. Smart Grid*, vol. 7, no. 4, pp. 2026–2037, Jul. 2016.
- [50] Z. Gu, W. Hu, C. Zhang, H. Lu, L. Yin, and L. Wang, "Gradient shielding: Towards understanding vulnerability of deep neural networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 921–932, Second Quarter 2021.
- [51] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, New York, NY, USA, 2017, pp. 506–519.
- [52] F. Yang, Z. Chen, and A. Gangopadhyay, "Using randomness to improve robustness of tree-based models against evasion attacks," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 969–982, Feb. 2022.
- [53] L. Song, R. Shokri, and P. Mittal, "Membership inference attacks against adversarially robust deep learning models," in *Proc. IEEE Secur. Privacy Workshops*, San Francisco, CA, USA, 2019, pp. 50–56.



**Ahmed Shafee** received the BS and MS degrees in computer engineering from Helwan University, Cairo, Egypt, in 2011 and 2018, respectively, and the PhD degree from the Department of Electrical & Computer Engineering, Tennessee Technological University, USA, in 2022. He is currently an assistant professor with the Department of Computer Science, Adams State University, USA. His research interests include machine and deep learning, artificial intelligence, intrusion detection, cryptography and network security, edge intelligence, smart-grid and AMI networks, and electric vehicles.



**Mohamed Mahmoud** (Senior Member, IEEE) received the PhD degree from the University of Waterloo, in April 2011. Currently, he is a professor with the Department Electrical and Computer Engineering, Tennessee Tech University, USA. His research interests include security and privacy preserving schemes for smart grid, e-health, and intelligent transportation systems. He has received NSERC-PDF award. He won the Best Paper Award from IEEE International Conference on Communications (ICC'09), Dresden, Germany, 2009. He is the author for more than 100 papers published in IEEE conferences and journals. He serves as an associate editor of the *IEEE Internet of Things Journal* and Springer journal of *Peer-to-Peer Networking and Applications*. He served as a technical program committee member for several IEEE conferences.



computer Engineering, Mississippi State University.

**J. W. Bruce** received the BSE degree from the University of Alabama, Huntsville, in 1991, the MSEE degree from the Georgia Institute of Technology, in 1993, and the PhD degree from the University of Nevada Las Vegas, in 2000, all in electrical engineering. He was a graduate research fellow of the Audio Engineering Society from 1998–2000. In 2018, he joined the faculty with the Department of Electrical & Computer Engineering, Tennessee Technological University in Cookeville, TN. From 2000–2018, he served with the Department of Electrical and Com-



**Gautam Srivastava** received the BSc degree from Briar Cliff University, USA, in 2004, and the MSc and PhD degrees from the University of Victoria in Victoria, British Columbia, Canada, in 2006 and 2011, respectively. He then taught for three years with the Department of Computer Science, University of Victoria, where he was regarded as one of the top undergraduate professors in the Computer Science Course Instruction at the University. From there in the year 2014, he joined a tenure-track position with Brandon University in Brandon, Manitoba, Canada,

where he currently is active in various professional and scholarly activities. He was promoted to the rank associate professor in January 2018. He is popularly known, is active in research in the field of data mining and Big Data. In his 8-year academic career, he has published a total of 43 papers in high-impact conferences in many countries and in high-status journals (SCI, SCIE) and has also delivered invited guest lectures on Big Data, cloud computing, Internet of Things, and cryptography at many Taiwanese and Czech universities. He is an editor of several international scientific research journals. He currently has active research projects with other academics in Taiwan, Singapore, Canada, Czech Republic, Poland, and USA. He is constantly looking for collaboration opportunities with foreign professors and students. He received Best Oral Presenter Award in FSDM 2017 which was held at the National Dong Hwa University (NDHU) in Shoufeng (Hualien County) in Taiwan (Republic of China) on November 24–27, 2017.



**Abdullah Balamsh** received the PhD degree in electrical and computer engineering from the University of Arizona, Tucson, AZ, USA, in 2004. In August 1996, he received a scholarship from the Saudi government to pursue his MS and PhD degrees. He is an associate professor with King Abdulaziz University (KAU), Jeddah, Saudi Arabia. In 2015, he became the deputy director with the Center of Excellence in Intelligent Engineering Systems. In 2006, he became the head of the Scientific Equipment Maintenance Center and the IT Unit, KAU College of Engineering. In January 1995, he joined the Department of Electrical and Computer Engineering of KAU, Jeddah, as a teaching assistant. From 1991 to 1994, he worked for the Saudi Consolidated Electricity Company as a system engineer. His research interests include machine learning and soft computing.



**Abdulah J. Aljohani** (Senior Member, IEEE) received the BSc (Eng.) degree in electronics and communication engineering from King Abdulaziz University, Jeddah, Saudi Arabia, in 2006, and the MSc degree with distinction and PhD degree, awarded with no corrections, in wireless communication from the University of Southampton, Southampton, U.K., in 2010 and 2016, respectively. He is currently an assistance professor with the Department of Electrical and Computer Engineering, King Abdulaziz University, Jeddah, Saudi Arabia. His research interests include machine learning, and optimization, distributed source coding, free-space optical communication, channel coding, cooperative communications, and MIMO systems.