Transfer Learning Based Intrusion Detection System using Gramian Angular Field for Connected Vehicles

Muhammad Anwar Shahid*, Arunita Jaekel*, Ning Zhang*, Tim Allsopp †

*University of Windsor, Windsor, ON, Canada
†Telus Communications Inc., Canada

***Abstract*—Intelligent Transportation System (ITS) uses advanced technologies such as Vehicular ad hoc Network (VANET) to improve the safety and congestion on the roads. The architecture of VANETs is inherently decentralized, characterized by direct vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications. VANETs are exposed to various known threats and attacks such as bogus information, Denial of Service (DoS), Sybil, Data Replay attacks and many more. In this paper, we propose a deep transfer learning based Intrusion Detection System (IDS) using VGG16 and GAF algorithm to detect position and speed falsification attacks in VANET. This innovative approach utilizes the power of deep transfer learning and visualization of temporal sequences in BSM data to improve the efficiency of IDS in V2X communications. Preliminary results, using the VeReMiExt dataset, indicate that the proposed approach outperforms existing techniques for most attack types.**

***Index Terms*—VANET, ITS, Transfer Learning, VGG16, VeReMi Dataset, BSM, GAF**

I. Introduction

The emergence of Vehicular Ad Hoc Networks (VANETs) has opened a new chapter for the Intelligent Transportation Systems (ITS) [1]. VANETs enable vehicles to communicate with each other and with roadside infrastructure, forming a real-time network that supports safety applications such as collision avoidance alerts, traffic management, and infotainment services. The architecture of VANETs is inherently decentralized, characterized by direct vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications [2].

Vehicles in VANET share sensitive information about their current status, including position, speed, heading etc., through Basic Safety Messages (BSMs). Attackers can harm the network by manipulating the legitimate data (in BSMs or other packets), creating sybil nodes and/or disrupting the network [3]. Timely detection of security attacks is of utmost importance due to the serious consequences of compromised networks, such as traffic disruptions, accidents, and the manipulation of traffic data [4]. In this paper, we focus primarily on BSM related attacks, where a malicious vehicle inserts incorrect position/speed information in its BSMs before broadcasting them to other nodes. These are *insider* attacks, as they are carried out by authenticated nodes that have valid credentials to access the network. Such attacks can compromise the integrity and reliability of the system, posing serious risks to passengers and drivers. Developing a robust intrusion detection system (IDS) for detecting and mitigating these security threats has become indispensable.

Traditional machine learning and deep learning based IDS has emerged as a highly effective tool for detecting security attacks in VANET [5] [6] [7]. Machine learning algorithms can learn useful patterns in the data and classify messages as malicious or legitimate. However, there remain numerous challenges with these techniques, such as high variability in data and real-time processing requirements that can impact the performance of the IDS [8]. Deep learning (DL) based IDS, in particular, can be effective in detecting attacks, but requires extensive training, on large datasets, and significant time and computational resources. One potential solution to these issues is the use of the Transfer Learning (TL) approach to develop IDS for connected vehicles. TL leverages the use of a pre-trained model from the source domain, which can then be "fine-tuned" through some additional training with datasets related to the target domain. This helps to reduce the training time, improve performance, and work with a smaller amount of data in the target domain [9].

Transfer learning based approaches have yielded promising results in different areas including text and image classification and as well as network intrusion detection. Transfer learning based IDS for general network traffic have been proposed in [10] [11] [8]. In a previous work [12], we presented a CNN-LSTM model for detecting DoS attacks in VANET, using the pre-trained model in [13]. In this paper, we propose a novel approach, where the initial pre-trained model was trained for image classification rather than intrusion detection. We first convert the information in the BSMs into images, by using Grammian Angular Fields (GAF) technique. GAF allows the data points to adhere to their temporal dependencies while creating an image format for capturing visual patterns. These images are then given as input to the proposed TL based model, which uses VGG16 as the pre-trained component. VGG16 [14] is a publicly available pre-trained model, trained on millions of images, and has been extensively used for various image classification tasks. We demonstrate that VGG16's hierarchical feature extraction

capabilities are able to recognize anomalies in the different BSM derived images and effectively identify anomalous BSMs. This paper focuses specifically on the *detection* of suspicious BSMs that are likely to be malicious. A comprehensive security mechanisms requires additional steps, including reporting suspected misbehavior to designated authorities, taking steps to mitigate any adverse effects and taking appropriate actions against malicious actors, which are out of the scope of this paper.

The remainder of this paper is organized as follows: Section II first gives an overview of TL based intrusion detection and then discusses some recent image based classification techniques for detecting attacks. Section III explains our proposed framework and Section IV discusses our results. Finally, we present our conclusions and some directions for future work in Section V.

II. Related Work

Numerous ML/DL based IDS for both in-vehicle and inter-vehicle communication have been proposed to ensure the security of vehicular communication [4] [15] [16] [17]. In recent years, transfer learning based IDS has been successfully implemented for general network traffic. In this section, we will first review existing TL based approaches for network attack detection, and then focus on image based classification techniques, which can be used for network attack detection.

A. Transfer Learning Based IDS

In [9], the authors use a pre-trained CNN model to detect various types of IoT attacks. The BOT-IoT dataset was used for training the network and UNSSW-NB15 for testing, achieving an accuracy of above 97%. A deep neural network trained on the NSL-KDD dataset was used to implement a transductive transfer learning framework in [8]. The model was tested on the CIDD dataset and achieve an accuracy of 80%. In [11] the authors developed a TL based framework based on the pre-trained ResNet model [18] to detect different types of attacks in cyber-physcial systems. UNSW-NB15 was used as the source dataset and CICIDS2017 as the target dataset for performance evaluation. This model reported a very high accuracy of 99.9% and outperformed other deep learning based approaches, such as CNN, DNN and LSTM. The work in [12] presents a transfer learning approach to detect DoS attacks using BSMs. A CNN-LSTM model pre-trained on NSL-KDD dataset was used on VeReMiExt dataset to detect DoS attacks. The proposed model achieved accuracies ranging from 95% to 99% for different types of DoS attacks.

In [19], the authors propose a deep transfer learning based framework for detecting IoT attacks. They introduce an improved Deep Transfer Learning (DTL) model adjusted with two autoencoders (AEs). The first distinct autoencoder (AE1) is trained on labeled data from the source domains. In contrast, the second distinct autoencoder (AE2) is trained on unlabeled data from the target domain. To validate the effectiveness of the proposed model, 9 real-world IoT datasets are employed in the experimental study. The results show that the model has better AUC scores compared with the other traditional DL approaches.

B. Image Based Classification for Attack Detection

Image based misbehavior detection requires the information in network packets to be converted into images before processing. The work in [20] proposes an image-based network intrusion detection system for internet traffic dataset using deep neural networks such as VGG19. It uses the weights transported from a pre-trained VGG-16 model and the ImageNet data to predict intrusion features, which are then passed to a deep neural network to classify intrusions. The min-max normalization process is used to scale the dataset and then convert it into RGB images. The results demonstrate the effectiveness of the proposed model, compared to the existing machine learning algorithms such as SVM, Decision Trees, and Logistic Regression, for both binary and multi-class classification.

A TL based intrusion detection system for controller area networks (CAN) that aims to improve the detection rate and efficiency in terms of training and testing time is presented in [21]. This model uses a pre-trained CNN-LSTM hybrid model to extract the characteristics of CAN traffic data and uses the labeled datasets of CAN intrusions from a smaller sample data to fine-tune the CNN model in order to address the problem of lack of labeled data in the automotive field. They use a simple normalization method in the range of 0-255 to convert the CAN dataset into images. The IDS demonstrates a detection accuracy of 100% for fuzzy attacks and 99.9% for various other attack types. According to the obtained results, the proposed CNN-LSTM model outperforms CNN, LSTM, and other proposed models in terms of accuracy, precision, recall, and F1 score, approaching nearly 100% scores.

In [22], authors highlight the growing susceptibility of interconnected vehicular networks and propose a transfer learning solution based on pre-trained CNN architectures for detecting abnormalities and unauthorized intrusions in the CAN bus dataset. First, they convert selected features into images because the pre-trained CNN model is designed to accept 3D images. They use seven convolutional layers along with 3 max-pooling layers from the pre-trained CNN model for knowledge transfer and a dense layer for the classification. They use the car hacking dataset as a source and the OTIDS dataset as a target dataset. The proposed model achieved accuracy, precision, recall and F1 scores exceeding 99% for binary classification, showcasing the potential of transfer learning in enhancing vehicular network security.

In light of the current approaches to performing intrusion detection focusing on network traffic data, [23] introduces a new technique that employs Gramian Angular Field (GAF) [24], to map the temporal sequences to

images. The work is based on the CIC-IDS 2017 dataset, and achieves better results compared to other ML models such as KNN, LR and SVM. Similarly, in [25], the authors convert ECG recordings into Gramian Angular Field (GAF) images and then classify these images using VGG19 models. This conversion to the image space allows the use of CNN architectures that have achieved high performance on images across numerous computer vision tasks.

III. Proposed Attack Detection Framework

In this section, we present our transfer learning based framework for detecting position and speed falsification attacks using the VGG16 model. A GAF based transformation is used to convert BSM data into images while preserving its temporal dependencies. We use the VGG16 model, a type of neural network trained on millions of images with 1000 classes, to enhance the detection accuracy of attacks in VANET.

Fig. 1 shows the architecture of the proposed framework for detecting position and speed falsification attacks. We consider 8 different attack types (4 position falsification and 4 speed falsification attacks), as shown in Table I, which are defined in the VeRemiExt dataset [3].

TABLE I: Description of Attack Types

Sr	Attack Types	Description
1	Constant Position	Position coordinates (X, Y) are fixed throughout the simulation from the start till the end of the trip for a vehicle
2	Constant Position Offset	A fixed offset (X, Y) is added to the actual position (X, Y) of the vehicle.
3	Random Position	Position coordinates (X, Y) are random at every time step. Random values are limited within the simulation area.
4	Random Position Offset	A random offset is added to the actual position (X, Y) of the vehicle
5	Constant Speed	Speed coordinates (VX, VY) are fixed throughout the simulation
6	Constant Speed Offset	A fixed offset (X, Y) is added to the actual speed (VX, VY) of the vehicle.
7	Random Speed	Speed coordinates (VX, VY) are random at every time step. Random values are limited within the simulation area.
8	Random Speed Offset	A random offset is added to the actual speed (VX, VY) of the vehicle.

The preprocessing module performs feature selection and feature extraction on the initial data to create the final dataset consisting of 18 features plus a label (having one of 9 possible values). Next, the image conversion module converts the preprocessed data to images using our image decoding algorithm B2img, which is shown in Algorithm 1. The images generated by the image conversion module are separated into the training and test sets, and the training data is used to train the VGG16 based transfer learning model. Finally, the test dataset is used to evaluate the performance of the proposed model.

A. Preprocessing Block

The VeReMi Extension dataset consists of individual JSON files containing BSM logs received by each vehicle, as well as ground truth files containing unaltered data with the true position and speed of each vehicle. For

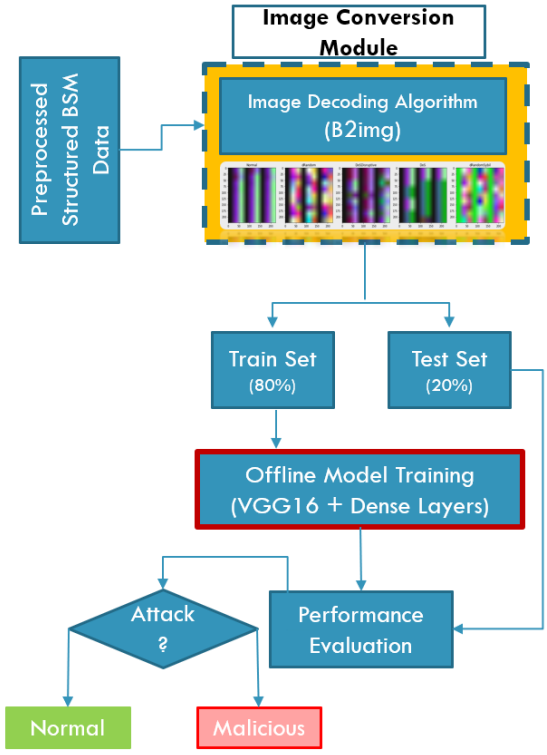

Fig. 1: Proposed IDS Framework

each sender, the i^{th} BSM from the sender, received at time t_i, contains important information about its current position $P_i = (p_{i,x}, p_{i,y}, p_{i,z})$, speed $V_i = (v_{i,x}, v_{i,y}, v_{i,z})$, acceleration, heading etc. The BSM data from the individual JSON files were combined and processed to remove duplicates and add labels. An initial feature selection step was then performed to remove non-contributing features, such as file names, z-coordinates of position, speed etc. (which are always set to 0), and noise. After this preliminary feature selection, the dataset contained 12 features, plus the label. Models based on these 12 features from individual BSMs did not meet performance expectations, particularly for constant and random speed offset attacks. So, we explored 6 new features that combine information from 2 consecutive BSMs from the same sender to provide additional insight into the vehicle's behavior. These features were first reported in our previous work [12] and summarized in Table II. After including the 6 new features, the total number of features we have used for classification is: $m = 12 + 6 = 18$.

TABLE II: Additional Features

Feature Name	Formula	Explanation
DiffPosX	$\Delta P_{x,i} = \lvert P_{x,i} - P_{x,i-1} \rvert$	Absolute difference of x-coordinates of position of 2 BSMs
DiffPosY	$\Delta P_{y,i} = \lvert P_{y,i} - P_{y,i-1} \rvert$	Absolute difference of y-coordinates of position of 2 BSMs
DiffSpdX	$\Delta V_{x,i} = \lvert V_{x,i} - V_{x,i-1} \rvert$	Absolute difference of x-coordinates of speed of 2 BSMs
DiffSpdY	$\Delta V_{y,i} = \lvert V_{y,i} - V_{y,i-1} \rvert$	Absolute difference of y-coordinates of speed of 2 BSMs
DistCbsm	$d = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$	Distance traveled by a sender between two consecutive BSMs
DiffRcvTime	$\Delta t_i = \lvert t_i - t_{i-1} \rvert$	Difference in receive time between two consecutive BSMs

B. Image Conversion Module

The generation of images that can preserve the temporal dependencies among data elements is one of the critical aspects of the proposed method. The image conversion module converts the data segments that have gone through pre-processing into images by using Gramian Angular Fields (GAF) techniques [24], which encodes time series data into a series of images that can capture the underlying temporal relationships to reveal potential anomalies or patterns. This allows the structured BSM data to be used with deep learning models for image classification, such as VGG16, ResNet, Inception.

Algorithm 1: Converting BSM Data into Images (B2img)

Require: Structured data $\{X_1, X_2, \ldots, X_n\}$ with m features

Ensure: Image representation of the structured data

1: **Initialize:** Gramian matrix G of size $m \times m$

2: **Compute Gramian Matrix:**

$$G_{ij} = \cos^{-1}\left(\frac{\langle X_i, X_j \rangle}{\|X_i\| \cdot \|X_j\|}\right), \quad 1 \leq i, j \leq m$$

where $\langle X_i, X_j \rangle$ is the dot product of features X_i and X_j, and $\|X_i\|$ is the Euclidean norm of X_i

3: **Normalize Gramian Matrix:**

$$H_{ij} = \frac{G_{ij}}{\sqrt{\sum_{k=1}^{m} G_{ik}^2 \sum_{k=1}^{m} G_{kj}^2}}, \quad 1 \leq i, j \leq m$$

4: **Create Image from Normalized Gramian Matrix:**

$$\text{Image} = \text{convert\_to\_image}(H)$$

A sequence of m BSMs are needed to generate a single channel of a $m \times m$ RGB image, where m is the number of features in the dataset (we have used $m = 18$); and 3 such images are combined to create the final RGB image. Therefore, the total number of BSMs required for a complete RGB image is $3m$ If the total number of BSMs (n_v) from a vehicle v is not a multiple of $3m$, then the last $n_v \ mod \ 3m$ BSMs are discarded so that the generated images represent complete data sequences. This is done to ensure consistency, as partial images may introduce inaccuracies and impact the performance of the proposed approach during processing. For image conversion, the BSMs are first divided into groups of m consecutive BSMs from the same sender. Algorithm 1 shows the steps required for processing each group of m BSMs and combining them to create a single image of size $m \times m$. We note that the prediction time depends on the number of BSMs needed to generate an image, which in turn depends on the number of features (m). So, one important objective is to reduce m as much as possible (while still maintaining adequate performance), so that prediction times remain within acceptable limits. In this paper, we focus on scenarios where detected misbehavior is reported to a central authority for further processing, and does not require immediate physical/kinematic actions.

In Step 1, the time series data in each BSM is first normalized. Next, in Step 2, the Gramian matrix G is calculated and the values are normalized in Step 3, using the equations given in [24] to generate the normalized Gramian matrix H. Finally, in Step 4, the $m \times m$ image is created by mapping the values of the normalized Gramian matrix to pixel intensities. Three consecutive normalized $m \times m$ matrices from the same sender H_R, H_G, H_B are then combined to create a single RGB image. The next three matrices are combined to form the next image and so on. This process continues until all Gramian matrices for a sender have been included as part of an image. The generated images can be used as a visual representation of the structured BSM data for further analysis and classification in our proposed approach.

C. Offline Model Training

Fig. 2 shows the structure of our VGG16 based offline training module. The model training starts with convolutional ReLU based non-linear activation layers. These layers are divided into 5 different blocks, each containing multiple convolutional layers. The weights of these layers are frozen and each block has a max-pooling layer at the end to help mitigate the risk of overfitting. The sizes of filters in the convolutional base layers are augmented, from 64 in the initial block to 512 in the last block, to help the network learn more complicated features at each stage by maintaining the spatial resolution of the image.

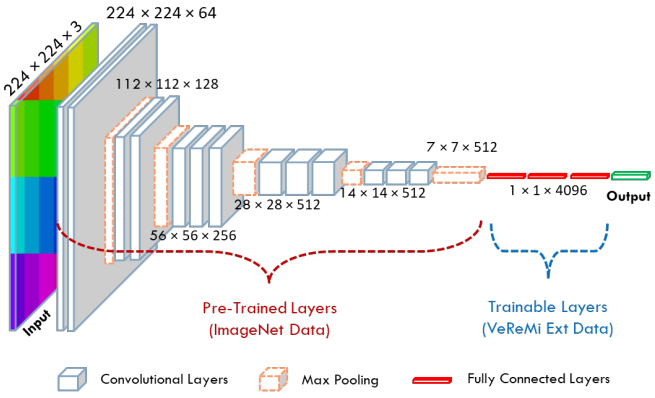

Fig. 2: Layers wise architecture of VGG16 model

The convolutional layers in VGG16 are followed by 3 fully connected (FC) layers, which are trained on VeReMi extension dataset with 9 classes. These FC layers aim to extract non-linear combinations and to learn complex relationships between features. The first FC layer (FC1) consists of 4096 neurons and ReLu activation function and receives its input from the last max-pooling layer of the fifth convolutional block in the VGG16 model.

The second FC layer (FC2) receives the output of FC1 layer and applies dense combination with 4096 neurons under the ReLu activation function. Both FC1 and FC2 layers extract high level features, perform dimensionality reduction and combine non-linear features to classify the objects more accurately [26]. The last fully connected layer (FC3) has as many neurons as the number of classes in the dataset. It uses the softmax activation function to output class probabilities for 9 classes (8 attack types plus legitimate BSMs) that we have considered.

IV. Performance Evaluation

We have used the VeremiExt dataset [3] to train and test our proposed model. The VeReMi extension dataset includes 19 different attack types, including 8 different position and speed falsification attacks, which are the primary focus of this paper. The attackers' saturation is kept at 30% as compared to legitimate vehicles to avoid class imbalance issues, which can lead to overfitting in most classification tasks. The total size of the dataset is 11.92 GB encoded in JSON file format. We evaluate the performance of our proposed framework using well-known metrics [19] for classification as given below:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

The F1 score is the harmonic mean of precision and recall, and is a good metric for overall performance of a model, particularly for an unbalanced dataset. For evaluation of the proposed model, we partition the vehicles so that there is no overlap among vehicles in the training and test sets. This ensures that all vehicles sending BSMs in the test set will be completely new to the models. The models will not encounter any BSMs with the same *senderID* or *senderPseudo* fields as those used for training. This is important because in the VeremiExt dataset, all BSMs from an "attacker" vehicle are considered malicious. This means that if there is overlap between vehicles in the training and test sets, it can lead to overfitting, where the model simply learns the senderIDs of vehicles from the BSMs that are labelled as malicious.

A. Detection of Position Falsification Attacks

In this section, we discuss the results of our proposed IDS for position falsification attacks. Table III shows the comparative performance of the proposed approach in terms of key metrics such as accuracy, precision, recall, and F1 score, against existing approaches for various position falsification attacks. It can be seen that the proposed approach has the overall best performance, consistently achieving scores of 95% or higher across all metrics for all 4 attack types. It achieves the best overall performance for 3 of the 4 attacks and is very close to the highest score for the fourth (random position offset) attack. The work in [27], uses classical machine learning algorithms, and has lower performance compared to the other three. Finally, we note that the proposed approach is able to achieve high performance, even when the test data does not contain any vehicle overlap with the training data; while the other approaches do not ensure this, leading to possible overfitting and skewing of test scores.

TABLE III: Comparison of position falsification attacks

			Attack Types			
Metrics	Approaches	Train-Test Vehicle Overlap	Constant Position	Constant Position Offset	Random Position	Random Position Offset
Accuracy	**Proposed**	**No**	**0.999**	**0.981**	**0.999**	**0.967**
	Paper1 [3]	Yes	0.995	0.961	0.999	0.988
	Paper2 [27]	Yes	0.645	0.698	0.625	0.706
	Paper3 [15]	Yes	0.992	0.988	0.998	0.996
Percision	**Proposed**	**No**	**0.998**	**0.998**	**0.999**	**0.979**
	Paper1 [3]	Yes	0.998	0.998	0.998	0.998
	Paper2 [27]	Yes	0.62	0.62	0.55	0.61
	Paper3 [15]	Yes	0.938	0.927	0.974	0.967
Recall	**Proposed**	**No**	**0.998**	**0.945**	**0.997**	**0.974**
	Paper1 [3]	Yes	0.987	0.873	0.999	0.961
	Paper2 [27]	Yes	0.650	0.690	0.610	0.710
	Paper3 [15]	Yes	0.913	0.853	1.000	0.967
F1 Score	**Proposed**	**No**	**0.995**	**0.959**	**0.999**	**0.952**
	Paper1 [3]	Yes	0.992	0.931	0.999	0.979
	Paper2 [27]	Yes	0.630	0.650	0.650	0.580
	Paper3 [15]	Yes	0.926	0.889	0.987	0.967

B. Detection of Speed Falsification Attacks

While position falsification attacks have been discussed extensively in the literature, there has been relatively limited work on speed falsification attacks. Table IV shows the comparison of accuracy, precision, recall, and F1 score between the proposed and other approaches for various speed falsification attacks. Our proposed approach performs very well, with scores at or near the best scores for all attacks, except constant speed offset attack. For this attack, Paper3 [15] has the best performance, while the proposed approach is next in terms of overall performance as indicated by the F1 scores. The performance of the approach in Paper1 [3] decreases considerably compared to position falsification attacks and Paper2 [27], which uses classical machine learning algorithms, again has the lowest performance for speed falsification attacks. As mentioned earlier, unlike the other approaches, the proposed approach ensures disjoint vehicle sets for training and testing.

TABLE IV: Comparison of speed falsification attacks

			Attack Types			
Metrics	Approaches	Train-Test Vehicle Overlap	Constant Speed	Constant Speed Offset	Random Speed	Random Speed Offset
Accuracy	**Proposed**	**No**	**0.999**	**0.856**	**0.982**	**0.999**
	Paper1 [3]	Yes	0.944	0.816	0.981	0.893
	Paper2 [27]	Yes	0.547	0.627	0.606	0.640
	Paper3 [15]	Yes	0.996	0.997	0.998	0.999
Percision	**Proposed**	**No**	**0.999**	**0.988**	**0.991**	**0.991**
	Paper1 [3]	Yes	0.997	0.996	0.998	0.997
	Paper2 [27]	Yes	0.590	0.590	0.650	0.570
	Paper3 [15]	Yes	0.980	0.967	0.962	0.987
Recall	**Proposed**	**No**	**0.995**	**0.528**	**0.995**	**0.852**
	Paper1 [3]	Yes	0.819	0.397	0.939	0.650
	Paper2 [27]	Yes	0.560	0.620	0.620	0.630
	Paper3 [15]	Yes	0.953	0.987	1.000	1.000
F1 Score	**Proposed**	**No**	**0.997**	**0.754**	**0.976**	**0.998**
	Paper1 [3]	Yes	0.899	0.568	0.968	0.787
	Paper2 [27]	Yes	0.590	0.570	0.620	0.640
	Paper3 [15]	Yes	0.966	0.978	0.980	0.993

V. Conclusion

In this paper, we present a TL-based IDS capable of detecting different variants of position and speed falsification attacks. We employ the GAF technique to convert temporal sequences of BSM data into a series of RGB images. This method enables the utilization of publicly available pre-trained models, such as VGG16 for attack detection in the connected vehicles field. To evaluate the performance of our proposed framework, we utilize VeReMiExt, a state of the art publicly available dataset. The proposed model demonstrates consistent superior performance compared to existing approaches for most attack types. The performance for constant speed offset attacks is lower compared to [15], but better than the other 2 techniques. We note that for all these attacks, the results for our proposed approach uses *disjoint* vehicle sets for training and testing, while this is not the case for the other approaches. In future work, we plan to extend our framework to detect additional attacks in connected vehicles, including traffic congestion, Sybil, and data replay attacks.

References

[1] M. L. Bouchouia, H. Labiod *et al.*, "A survey on misbehavior detection for connected and autonomous vehicles," *Vehicular Communications*, p. 100586, 2023.

[2] M. A. Shahid, "Fixed cluster based cluster head selection algorithm in vehicular adhoc network," Master's thesis, University of Windsor (Canada), 2019.

[3] J. Kamel, M. Wolf *et al.*, "Veremi extension: A dataset for comparable evaluation of misbehavior detection in vanets," in *ICC 2020*. IEEE, 2020, pp. 1–6.

[4] S. Ercan, M. Ayaida, and N. Messai, "Misbehavior detection for position falsification attacks in vanets using machine learning," *IEEE Access*, vol. 10, pp. 1893–1904, 2021.

[5] S. Gyawali and Y. Qian, "Misbehavior detection using machine learning in vehicular communication networks," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[6] J. Nagarajan, P. Mansourian *et al.*, "Machine learning based intrusion detection systems for connected autonomous vehicles: A survey," *Peer-to-Peer Networking and Applications*, vol. 16, no. 5, pp. 2153–2185, 2023.

[7] M. A. Shahid and A. Jaekel, "Hybrid approach to detect position forgery attacks in connected vehicles," in *2023 14th International Conference on Network of the Future (NoF)*. IEEE, 2023, pp. 47–51.

[8] N. Sameera and M. Shashi, "Deep transductive transfer learning framework for zero-day attack detection," *ICT Express*, vol. 6, no. 4, pp. 361–367, 2020.

[9] E. Rodríguez, P. Valls *et al.*, "Transfer-learning-based intrusion detection framework in iot networks," *Sensors*, vol. 22, no. 15, p. 5621, 2022.

[10] E. Mahdavi, A. Fanian *et al.*, "Itl-ids: Incremental transfer learning for intrusion detection systems," *Knowledge-Based Systems*, vol. 253, p. 109542, 2022.

[11] H. Wang, H. Zhang *et al.*, "Resadm: A transfer-learning-based attack detection method for cyber–physical systems," *Applied Sciences*, vol. 13, no. 24, p. 13019, 2023.

[12] M. A. Shahid, A. Jaekel *et al.*, "Dos attack detection in vanet using transfer learning approach for bsm data," in *2024 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2024, pp. 748–753.

[13] WhiteHatCyberus, "Deep learning evaluation of ids datasets," https://github.com/WhiteCyberus/Deep-Learning-Evaluation-of-IDS-Datasets, Year of last update, if available.

[14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[15] T. Alladi, V. Kohli *et al.*, "A deep learning based misbehavior classification scheme for intrusion detection in cooperative intelligent transportation systems," *Digital Communications and Networks*, vol. 9, no. 5, pp. 1113–1122, 2023.

[16] A. Kumar, M. A. Shahid *et al.*, "Machine learning based detection of replay attacks in vanet," in *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2023, pp. 1–6.

[17] A. Sharma and A. Jaekel, "Machine learning approach for detecting location spoofing in vanet," in *2021 ICCCN*. IEEE, 2021, pp. 1–6.

[18] K. He, X. Zhang *et al.*, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[19] L. Vu, Q. U. Nguyen *et al.*, "Deep transfer learning for iot attack detection," *IEEE Access*, vol. 8, pp. 107 335–107 344, 2020.

[20] M. Masum and H. Shahriar, "A transfer learning with deep neural network approach for network intrusion detection," *International journal of intellligent computing research*, vol. 12, no. 1, 2021.

[21] N. Khatri, S. Lee, and S. Y. Nam, "Transfer learning-based intrusion detection system for a controller area network," *IEEE Access*, 2023.

[22] A. Haddaji, S. Ayed, and L. C. Fourati, "A transfer learning based intrusion detection system for internet of vehicles," in *2023 15th international conference on developments in esystems engineering (dese)*. IEEE, 2023, pp. 533–539.

[23] D. S. Terzi, "Gramian angular field transformation-based intrusion detection," *Computer Science*, vol. 23, 2022.

[24] Z. Wang and T. Oates, "Imaging time-series to improve classification and imputation," *arXiv preprint arXiv:1506.00327*, 2015.

[25] C. Camara, P. Peris-Lopez *et al.*, "Ecg identification based on the gramian angular field and tested with individuals in resting and activity states," *Sensors*, vol. 23, no. 2, p. 937, 2023.

[26] L. Chen, S. Li *et al.*, "Review of image classification algorithms based on convolutional neural networks," *Remote Sensing*, vol. 13, no. 22, p. 4712, 2021.

[27] O. Slama, M. Tarhouni *et al.*, "One versus all binary tree method to classify misbehaviors in imbalanced veremi dataset," *IEEE Access*, vol. 11, pp. 135 944–135 958, 2023.