FindSpy: A Wireless Camera Detection System Based on Pre-trained Transformers

Zhixin Shi*†,Hao Wu*†‡,Jing Zhang*†,Meng Zhang*†,Weiqing Huang*†
*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, 100093, China
†School of Cyber Security, University of Chinese Academy of Sciences, Beijing, 100049, China
E-mail: {shizhixin,wuhao0010,zhangjing2812,zhangmeng,huangweiqing}@iie.ac.cn

***Abstract*—The wireless cameras have become a major concern in cybersecurity due to privacy breaches. Recent flow based methods for wireless cameras detection have achieved promising results. However, these methods require specialized equipment for deployment and massive labeled data for training, which makes them impractical in real-world scenarios. In this paper, we propose FindSpy, a lightweight wireless camera detection method based on Pre-trained Transformers to address the challenge. By utilizing the air interface technique, FindSpy can obtain data without connecting to the wireless network where the camera is located. Additionally, FindSpy learns air interface WiFi traffic representation by pre-training a traffic representation model from large-scale unlabeled data and fine-tuning it on few labeled data. FindSpy can accurately detect wireless cameras with CNN-LSTM classifier. Extensive experiments show that FindSpy outperforms the state-of-the-art methods on few data. Concretely, FindSpy achieves a detection accuracy of over 98% by analyzing just five data packets.**

***Index Terms*—Air Interface, Transformer, Camera Detection**

I. INTRODUCTION

Surveillance cameras are playing an increasingly vital role in the smart home. According to Grand View Research [1], the global smart home security camera market size is expected to expand at a compound annual growth rate of 18.7% from 2022 to 2030. Among the types of cameras available, wireless cameras are preferred for their versatile functionality, but they give rise to the problem of privacy leakage. For example, criminals use them to capture personal activities [14] and to steal passwords of debit cards [15] , which becomes a significant threat to individual privacy. Therefore, how to effectively detect wireless cameras is always an important problem in cybersecurity research.

Various methods have been attempted to detect such cameras. According to different data sources, the existing methods can be divided into three groups: 1) optical signal based detection, 2) electromagnetic signal based detection, and 3) flow based detection. The methods in group one typically use the flashlight of a mobile phone to detect wireless cameras, while those in group two leverage the electromagnetic signals to discover wireless cameras with professional equipment. However, these methods have poor detection accuracy or require specialized equipment for detection, which makes them difficult to deploy in real-world scenarios. Therefore, some methods in group three analyze the network traffic to detect wireless cameras with machine learning techniques. Methods like DeWiCam [5], DeepDeSpy [26] etc., have obtained promising results by utilizing flow based techniques without professional equipment.

‡ Corresponding author

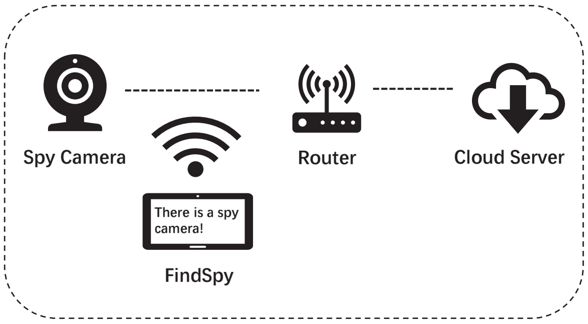

Fig. 1. The spy camera interacts with a remote server when it is working. FindSpy can grab the data for analysis through the air interface and perform camera identification quickly and efficiently.

Unfortunately, existing flow based detection methods suffer from the following problems. First, these methods need to connect to the network where the camera is located, which is difficult to achieve in a real environment. Specifically, an attacker may use various methods to hinder network discovery. Second, these methods require large amounts of labeled data to train their model, which is hard to obtain. Additionally, noises, such as packet jitter and loss, occur frequently in production environments, which makes it challenging to label traffic data.

To address these challenges, we propose FindSpy, a lightweight method capable of detecting hidden wireless cameras. Fig. 1 is a simple example of our system. FindSpy leverages the monitor mode of wireless network cards to collect data from the air interface, enabling traffic analysis without connecting to the camera network. Given that air interface traffic contains limited information, we pre-train a traffic representation model from large-scale unlabeled air interface traffic data, which is collected from various environmental conditions. After pre-training, we fine-tune the model to enhance its generalization ability for traffic data. Finally, we detect wireless cameras with deep learning on small dataset.

We implement the prototype of FindSpy and evaluated it using data from real environments. We collected data from 15 cameras of various brands in real-world environments including laboratory, home, and hotel settings.We used different sets

2023 IEEE Symposium on Computers and Communications (ISCC) | 979-8-3503-0048-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISCC58397.2023.10218088

of cameras for testing. Extensive experiments show that FindSpy outperforms the state-of-the-art methods on small training data. Concretely, FindSpy achieves a detection accuracy of over 98% by analyzing just five data packets.

In summary, we make the following contributions:

- We are the first to pre-train a traffic representation model from large-scale unlabeled air interface traffic data, which can get better air interface traffic representation.
- We propose FindSpy, a hidden wireless camera detection system based on pre-trained transformers, which performs better with only small training data.
- We implement a prototype of FindSpy and evaluate its performance in different environments using 15 cameras from various vendors. The results show that FindSpy achieves an average detection accuracy of over 98% with just five packets, which outperforms state-of-the-art methods.

The rest of this paper is organized as follows. In Section 2, we introduce some background knowledge and systematically outline related works. In Section 3, we propose our wireless camera detection system based on the pre-trained model with details. Then we provide our experimental methodology and results in Section 4. Finally, we summarize our primary jobs and discuss future work in Section 5.

II. RELATED WORKS

A. Wireless Camera Detection

The detection of hidden wireless cameras can be achieved using various methods, including optical signal-based, RF signal-based, and traffic-based detection. Optical signal-based methods typically use smartphone applications to detect camera reflections from their lenses, but their practical usability is limited due to their need for knowledge of the camera's approximate location and their low accuracy [6]. RF signal-based methods use specialized equipment to detect signals in the 2.4 GHz and 5 GHz bands, but are susceptible to interference from other devices and are more commonly used in engineering applications [7]. Traffic-based detection can be either active or passive, with active detection involving the capture of network traffic from the gateway or other locations, and passive detection involving the sniffing of empty traffic. Rahmadi Trimananda et al. [8] use active detection to identify the device by capturing relevant traffic from the routing side and automatically extracting an unknown signature consisting of a simple sequence of packet lengths and directions. This approach requires a connection to the network for active sniffing and may not be feasible in certain environments. Passive detection, on the other hand, can detect and identify the type of IoT devices, their state, and ongoing user activities without the need for network connectivity. Abbas Acar et al. [9] use a cascading method to study the switching status of connected devices, while Liu et al. use statistical learning methods to manipulate the video scene of a spy camera and investigate changes in the response of the packet stream to detect wireless cameras. However, these methods are limited by their detection accuracy and may not be effective in detecting unknown cameras.Cheng's traffic side-channel-based detection mechanism improves the practicality and simplicity of wireless camera detection by implementing special traffic characteristics based on learning-assisted detection with user intervention. However, this model only focuses on the working state of the camera, and detection may not be possible if the camera is in a silent state, i.e., networked but unmonitored.

B. Pre-training for Wireless Traffic Classification

WiFederated, proposed by Steven et al. [10], is a federated learning (FL) approach that trains machine learning models for WiFi sensing tasks. This approach offers a device-free and non-intrusive method for human activity monitoring. Mohammud et al. [11]proposed the use of self-supervised contrastive learning to improve activity recognition performance when using multiple views of the transmitted WiFi signal captured by synchronized receivers deployed in different positions. This approach can enhance the model's ability to recognize complex activities and improve its generalization performance. Gu et al. [12]proposed WiGRUNT, a WiFi-enabled gesture recognition system using a dual-attention network, which can mimic how a keen human being intercepts a gesture regardless of the environment variations. These studies demonstrate the effectiveness of pre-trained models in WiFi traffic classification and provide a promising avenue for further research in this area.

The majority of research in the field of WiFi traffic classification involves the acquisition of network traffic through routers or other access methods, which can be analyzed based on specific protocols to gain detailed insights. However, fewer studies focus on the analysis of WiFi traffic captured directly from the air interface. Promiscuous mode on wireless network cards can be adjusted to capture WiFi data in the surrounding area; however, this data is often encrypted based on the IEEE 802.11 protocol [13], making it difficult to extract useful information. As a result, WiFi traffic analysis based on the air interface remains a challenging task.

III. SYSTEM ARCHITECTURE

In this section, we present the architecture and training mechanism of our hidden wireless camera detection system. The system is composed of three primary components: a Flow2Token module, a Traffic Characterization module, and a Classification module, as illustrated in Figure 2. The Flow2Token module captures WiFi traffic based on the air interface and segments the data into flows. The data payload portion of the data frame is then extracted and encoded for pre-training the model. The Traffic Characterization module pre-trains a large amount of unlabeled air interface WiFi traffic data and fine-tunes the camera traffic to obtain a characterization model of air interface WiFi traffic, using two pre-training tasks. Finally, the Classification module utilizes a CNN-LSTM model for camera traffic classification.

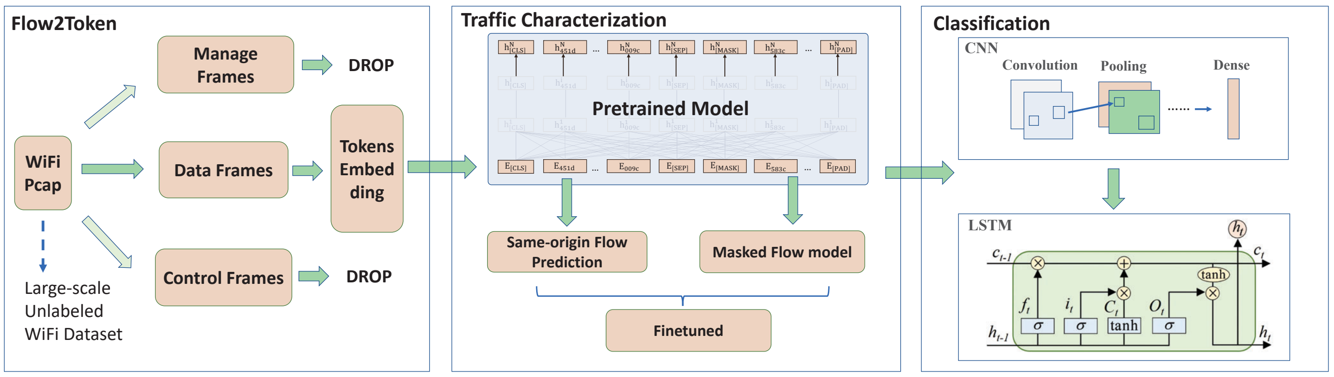

Fig. 2. Overview of the model Framework. It learns different traffic distributions from large-scale unlabeled air interface WiFi data to obtain the pre-trained model. The system comprises three primary components: a Flow2Token module, a Traffic Characterization module, and a Classification module.

A. Flow2Token Module

In this section, we describe the data requirements and acquisition process for FindSpy, a hidden wireless camera detection system. The data required by FindSpy is composed of two parts: pre-training data and data to be tested. The pre-training data is obtained from the air traffic generated by a large number of unlabeled devices in different environments, and this data is used for model pre-training. During the acquisition process, the pre-training data undergoes no labeling, and only the retransmitted frame part caused by network instability is removed. The payload portion of the data frame is then extracted for pre-training the model.

The data to be detected is all air interface traffic in a certain area, and the traffic data is extracted from the mixed flow based on the source MAC address and destination MAC address. FindSpy removes management and control frames and analyzes only the data frames. Typically, cameras send a large amount of data to the server during their operation, while people generate a large amount of downstream traffic during normal online behavior. Therefore, the direction of data transfer is an important property, and FindSpy filters out traffic generated by normal internet behavior based on the direction of traffic transmission. After removing the traffic with significantly more downlink data than uplink data, the remaining traffic is suspected camera traffic. This traffic is passed to the detection model as data to be tested for classification.

The core of the FindSpy traffic collection module is based on the wireless network card running in monitor mode, which accepts and saves all WiFi packets. All traffic data can be obtained without connecting to any network, and the range of data it can obtain is related to the power of the network card. Given the complexity of the wireless network environment, FindSpy can accept a certain degree of packet loss.

B. Traffic Characterization Module

Air interface WiFi traffic data differs significantly from natural language and cannot be interpreted like typical sentences. Therefore, it is transformed into datagram tokens for pre-training and classification. Our model encodes the traffic data by converting it into hexadecimal, with each cell consisting of two adjacent bytes. Byte-Pair Encoding is then used to represent the tokens, where each token cell ranges from 0 to 65535, and the maximum dictionary size V is represented as 65536. Additionally, we include special tokens such as [CLS], [SEP], [PAD], and [MASK] for the training task. The first token of each sequence is always [CLS], and the final hidden layer state associated with this token represents the complete sequence of the classification task. The token [PAD] acts as a padding symbol to satisfy the minimum length requirement. Sub-Flow pairs of a Flow are separated by [SEP], and the token [MASK] is used to understand the context of the flow during pre-training.

We propose two pre-training tasks (MFM and SFP) to capture the contextual relationship between traffic bytes by predicting the masked token and the correct transmission order by predicting the Same-origin Flow. In the MFM phase, each token in the input sequence is randomly masked with a 15% probability. If a token is selected, it has an 80% probability of being replaced with [MASK] and a 10% chance of choosing a random token to replace it or leave it unchanged. The model needs to predict the context of the masked token based on the surrounding tokens. Specifically, for the task in this paper, the model needs to learn the traffic distribution of this flow, consisting of several packets, and make predictions about this token. From the perspective of air interface WiFi traffic, the data is the traffic payload, which is a binary stream generated from the content recorded by the camera controlled by the upper layer protocol. The model can learn the traffic distribution of the flow in the task. We use the negative log-likelihood as our loss function, and for the input group P, we formally define it as:

$$L_{MFM} = -\sum_{i=1}^{m} log(P(MASK_i = token_i|\bar{P};\theta)) \quad (1)$$

This loss function ensures that the model is trained to predict the masked token in the sequence correctly, taking into account the surrounding context of the flow.

Natural Language Processing (NLP) involves a range of important tasks, such as Question Answering (QA) and Natural

Language Inference (NLI), which are centered around understanding the relationship between two sentences. Similarly, in the context of air interface traffic data, it is crucial to allow the model to comprehend the relationship between individual flows in order to learn the traffic distribution of the target application. This requires the model to capture the connection between individual packets to determine whether specific flows are generated by the same application, such as different flows generated by cameras. To this end, we utilized a pre-training strategy involving a binarized next flow prediction task. Specifically, each pre-training example involves selecting two flows A and B, where there is a 50% probability that B is the actual following flow behind A (labeled as IsNext), and a 50% chance that B is a random flow from the flow library (labeled as NotNext). We define the loss function for the input sub-flow F as:

$$L_{SFP} = -\sum_{j=1}^{n} \log(P(y_j|F_j;\theta)) \tag{2}$$

Here, a value of 0 for y implies that the two flows are homologous, while a value of 1 indicates the opposite case. In addition, we pre-trained the model on a masked flow modeling (MFM) task, in which each token in the input sequence has a 15% chance of being randomly masked and replaced with a special token [MASK]. The model then needs to predict the masked token based on the context of the surrounding tokens. We employ the negative log-likelihood as the loss function for the input group P, which is defined as:

$$L_{MFM} = -\sum_{i=1}^{m} \log(P(MASK_i = token_i|\bar{P};\theta)) \tag{3}$$

Finally, the overall loss for pre-training is defined as the sum of the MFM and SFP losses:

$$L_{sum} = L_{MFM} + L_{SFP} \tag{4}$$

To fine-tune our pre-trained model, we utilized air interface traffic data generated by a variety of wireless cameras. The cameras were operated continuously, and their downlink data frames were filtered and collected using network cards. This collected data was then used to fine-tune the pre-trained model and obtain a more optimized and efficient model for our purposes.

Our pre-training dataset consisted of approximately 2 terabytes of unlabeled air interface WiFi traffic data. This dataset was gathered from various network environments and contained traffic from both common application protocols such as Transport Layer Security and File Transfer Protocol, as well as private protocols like MMTLS. The inclusion of private protocols in the dataset allowed for a more diverse range of traffic patterns to be learned by the model, enhancing its ability to accurately classify and predict traffic flows in real-world scenarios.

C. *Classification Module*

After obtaining the fine-tuned pre-training model, flow characterization is performed on the air interface flow to be detected based on the model. The extracted features are then processed using a CNN-LSTM model, which combines CNN layers and LSTM layers to classify the input data as either camera or non-camera traffic. In the field of traffic classification, the CNN-LSTM model has proven to be an effective model structure. CNN is primarily used to extract both time domain and frequency domain information, while LSTM is used to process sequence information. This model structure can capture the spatio-temporal relationship of traffic data, thus improving the classification accuracy. Moreover, the CNN-LSTM model structure is flexible and can be adjusted to meet the specific requirements of the task at hand, such as increasing or reducing the number of convolutional layers, changing the number of LSTM units, and so on.

The CNN-LSTM model has several advantages that make it suitable for flow-based traffic classification. Firstly, the model can effectively extract both the spatial and temporal features of the input data, which are crucial for accurate classification. Secondly, it can capture the long-term dependencies of the input sequence, which is useful in modeling the complex patterns of traffic data. Additionally, the model structure is relatively simple and can be easily implemented, making it practical for real-world applications. Finally, the CNN-LSTM model is highly adaptable and can be adjusted to meet the specific requirements of different classification tasks, making it a versatile choice for flow-based traffic classification.

IV. PERFORMANCE EVALUATION

We evaluated FindSpy's performance based on real-world data to answer the following questions:

- Q1: How effective is the model in characterizing air interface WiFi traffic?
- Q2: Is FindSpy superior to the most advanced flow-based hidden wireless camera detection system?
- Q3: Does FindSpy's detection performance remain stable in complex real-world environments?

In this section, we first describe the experimental setup, including the dataset and indicators. Then We conducted several comparative experiments to demonstrate the effectiveness of the traffic characterization and detection performance of our model, as well as its performance compared to other existing schemes. Additionally, we demonstrate the detection performance of our model in real-world environments.

A. *Experiment Setup*

1) Datasets: To evaluate the proposed scheme, we conducted a series of experiments under different conditions. Due to the absence of a publicly available dataset for air interface traffic data of cameras, we collected data from 15 cameras of various brands in real-world environments including laboratory, home, and hotel settings. In the laboratory environment, we deployed 20 access points and 50 Wi-Fi devices, each running different applications such as web browsing, video calling, and file uploading, to simulate a bandwidth range covering all channels in the 2.4G wireless network. The network in the laboratory environment was complex, with an average packet loss rate of 32%. In contrast, we set up three access points and 10 Wi-Fi devices in the home environment to simulate daily use. The home environment had a simple network topology, with an average packet loss rate of 3 in 10,000. We also tested the FindSpy system in real hotel environments by deploying three cameras in each of the five hotels based on the hotel's own Wi-Fi network.

To pre-train the model, we split the collected unlabeled Wi-Fi air interface traffic data into two sets containing camera traffic and non-camera traffic. We used different sets of cameras for testing. Furthermore, we divided the camera dataset into flows containing 5, 10, 30, 50, and 100 packets for comparison experiments.

2) Performance Metrics: We calculate TPR (true positive rate) and TNR (true negative rate) to quantitatively evaluate the model performance, and we define

$$TPR = \frac{TP}{TP + FN} \tag{5}$$

which portrays the ratio of the number of positive instances predicted by the classifier that are indeed positive instances to all predicted positive instances (including false negative instances),

$$TNR = \frac{TN}{TN + FP} \tag{6}$$

which calculates the ratio of negative instances that the classifier incorrectly considers as a positive class to of all negative instances.

$$Accuracy = \frac{(TP + TN)}{TP + TN + FP + FN} \tag{7}$$

$$Precision = \frac{(TP)}{TP + FP} \tag{8}$$

$$Recall = \frac{(TP)}{TP + FN} \tag{9}$$

$$F1 = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 * TP}{2 * TP + FP + FN} \tag{10}$$

B. Visualization

To provide visual representation of the model's traffic characterization capability, we computed the cosine similarity of the flow data generated by five cameras and five non-camera devices, which were previously characterized by the proposed model. The resulting similarity scores were visualized as a heat map. As depicted in the Fig. 3, the similarity scores among the camera-generated flows are relatively high, while flows generated by non-camera devices exhibit distinct differences from camera flows. This indicates that the proposed model is capable of effectively distinguishing between camera and non-camera traffic.

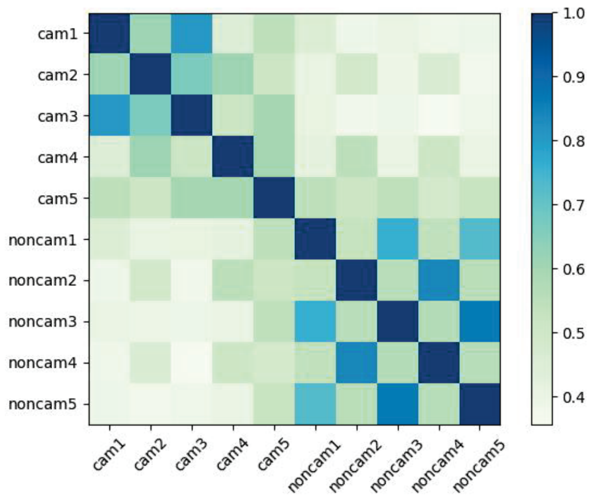

Fig. 3. After the traffic characterization of the camera stream and non camera stream, the cosine similarity is calculated and visualized using a thermal map.

C. Comparison Experiment

To further evaluate the performance of the proposed system, several experiments were conducted to compare it with other classification models and training approaches. Specifically, we compared the performance of our fine-tuned classifier to that of the same classifier without fine-tuning and several other classification models. Furthermore, the effect of training with varying amounts of data was analyzed. Finally, we compared our system with existing state-of-the-art schemes in the field. These experiments were conducted to assess the effectiveness of our proposed approach and to demonstrate its superiority over other available methods.

1) Comparison of different model structures: As shown in the TABLE I, we compare the results of using this scheme with only pre-training without fine-tuning, and the results of using different classifiers after fine-tuning. It can be seen that the detection performance of this scheme is better.

TABLE I
THE RESULTS OF USING THIS SCHEME WITH ONLY PRE-TRAINING WITHOUT FINE-TUNING, AND THE RESULTS OF USING DIFFERENT CLASSIFIERS AFTER FINE-TUNING.

Data	Acc(%)	Pre(%)	Rec(%)	F1(%)
SVM	66.05	54.07	51.77	53.90
LSTM	95.56	97.37	90.97	93.91
Pre-Nonfinetuned-SVM	97.31	97.92	96.52	97.21
Pre-Nonfinetuned-LSTM	97.51	97.93	96.93	97.43
Pre-Finetuned-LSTM	98.01	97.37	98.57	97.96
FindSpy	**99.49**	**99.56**	**99.33**	**99.45**

2) Comparison of different data quantities: The amount of data used for fine-tuning is a critical factor that affects the accuracy of the model's detection. The air interface traffic is composed of various flows, which, in turn, consist of several packets. Achieving better detection with a shorter data flow (i.e., fewer packets) is desirable. To demonstrate the effectiveness of our approach in this regard, we conducted comparative experiments using different amounts of data.

The results showed that even with a small amount of fine-tuning data, our model achieved excellent performance. Fig. 4 presents the results of the tests with flow sizes of 3, 5, 10, 50, and 100 packets, respectively. The model's accuracy was above 96% in all cases, and the detection performance with as few as 3 packets was outstanding. This finding is significant as it greatly improves the speed of detection, making it more efficient in real-world scenarios. The amount of training data

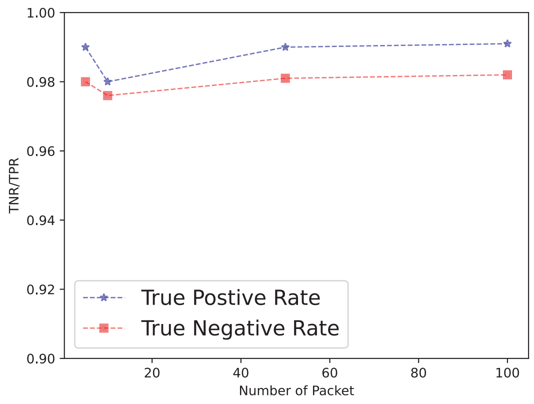

Fig. 4. Training with the different number of packets.

used during the fine-tuning process is a crucial factor that impacts the final detection accuracy. In this study, we used 100, 300, 500, and 1000 flows (with each flow containing only three packets, as per previous experiments) for training and evaluated their impact on accuracy, as shown in Fig. 5. Our results indicate that when 300 packets are used for training, the True Negative Rate (TNR) of the results is relatively low; however, increasing the number of training packets to 500 yields excellent results. Our findings suggest that training the model with just 500 flows of traffic data is sufficient to achieve a TNR and True Positive Rate (TPR) above 95%. Additionally, training the model with traffic containing over 1000 packets can yield an accuracy close to 100%, indicating that the model can achieve good results even with limited training data. Notably, we observed a partial performance degradation when the number of packets was increased from three to five. This can be attributed to the similarities in this part of the data, which is common in many applications.

The results in the TABLE II suggest that our proposed scheme can achieve relatively good detection performance with only a small amount of training and detection data. We compared our scheme with existing methods using the same amount of data. While all schemes achieved good detection performance with a large amount of data, our proposed scheme outperformed existing methods with fewer samples. Thus, our scheme requires less data processing during operation, which is beneficial for the lightweight design of detection equipment.

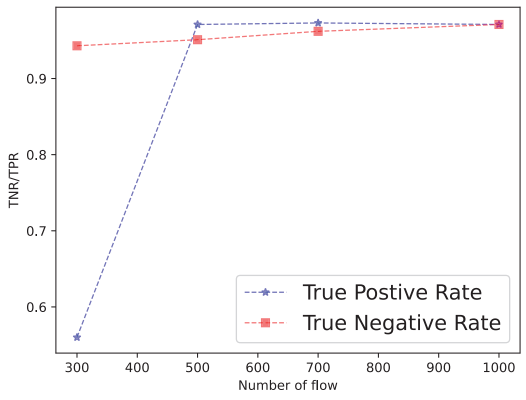

Fig. 5. Training with the different number of flows.

3) Comparison with state of the art: In this study, we performed a comparative analysis of FindSpy, DeWiCam [5], and DeepDeSpy [26] by varying the number of data packets used for training and detection, as depicted in Fig. 6. Our results indicate that the detection performance of these methods is reliant on the amount of data used. When a large amount of data is available, all methods can achieve good detection accuracy. However, in scenarios where only a small amount of data is available, FindSpy exhibits superior detection accuracy compared to the other methods, indicating better stability and reliability in detecting hidden wireless cameras.

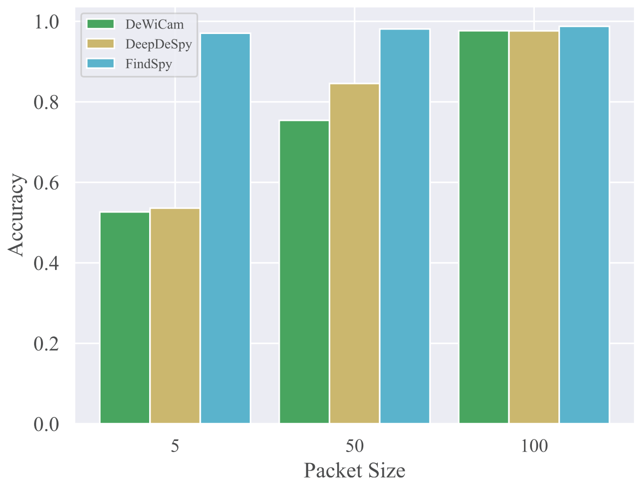

Fig. 6. Detection effect of DeWiCam, DeepDeSpy and FindSpy.

4) Real environment: Moreover, we conducted a real-world evaluation of the performance of FindSpy. To this end, we deployed three cameras in each of the five hotels, based on the hotel's WiFi network. We collected 100 sets of data using FindSpy in each area and evaluated the detection accuracy, as presented in Table III. This real-world evaluation helps to validate the practical usefulness of our proposed approach and provides evidence of its effectiveness in detecting hidden cameras in a real-world setting.

TABLE II
DETECTION PERFORMANCE IN REAL ENVIRONMENTS.

Data	Acc(%)	Pre(%)	Rec(%)	F1(%)
Hotel1_Camera_1	97.00	97.73	95.56	96.63
Hotel1_Camera_2	98.00	97.78	97.78	97.78
Hotel1_Camera_3	97.00	1.00	91.11	95.35
Hotel2_Camera_1	96.00	97.67	93.33	95.45
Hotel2_Camera_2	98.00	97.78	97.78	97.78
Hotel2_Camera_3	96.00	1.00	97.78	95.65
Hotel3_Camera_1	97.00	95.65	97.78	96.70
Hotel3_Camera_2	96.00	95.56	95.56	95.56
Hotel3_Camera_3	97.00	1.00	93.33	96.55
Hotel4_Camera_1	94.00	91.49	95.56	93.48
Hotel4_Camera_2	95.00	91.67	97.78	94.62
Hotel4_Camera_3	93.00	97.50	86.67	91.76
Hotel5_Camera_1	87.00	84.78	86.67	85.71
Hotel5_Camera_2	91.00	90.91	88.89	89.89
Hotel5_Camera_3	91.00	95.00	84.44	89.41

The evaluation results demonstrate that FindSpy achieves high detection performance in various hotel environments, except for the fifth hotel where the detection accuracy of the three cameras shows a slight degradation. This may be attributed to the complex internal network environment in this area, with numerous smart wireless devices causing a high packet loss rate due to the poor performance of the access point (AP). Despite this, the overall performance of FindSpy is excellent, confirming its suitability for detecting unauthorized cameras in hotel environments.

V. CONCLUSION

We present a novel approach for wireless camera detection, called FindSpy, which utilizes pre-trained transformers to achieve high accuracy and low computational complexity. The system is composed of three primary components, namely Flow2Token, Traffic Characterization, and Classification. The Flow2Token module captures WiFi traffic on the air interface, performs data cleaning by means of deduplication and data payload extraction, and converts it into flow units. The Traffic Characterization module employs self-supervised learning to pre-train and fine-tune a large amount of unlabeled air interface WiFi traffic data with camera traffic to obtain an accurate characterization model of air interface WiFi traffic. The Classification module uses the CNN-LSTM model for camera traffic classification. The proposed method is a novel contribution to the field, as it is the first to explore the use of self-supervised learning for representation learning based on air interface traffic in wireless camera detection. Our experimental results demonstrate that the proposed system achieves an accuracy of over 98% with only five packets of downlink traffic, and its performance remains stable in real environments.

ACKNOWLEDGMENT

This work was supported by the Youth Innovation Promotion Association CAS.

REFERENCES

[1] The market analysis report on the global smart home security camera market http://9l8.cn/rS72n

[2] Fing Network Tools http://8e9.cn/FljhH

[3] WiFiman http://8e9.cn/cf76A

[4] Liu T , Liu Z , Huang J , et al. Detecting Wireless Spy Cameras Via Stimulating and Probing[C]// 2018:243-255.

[5] Cheng Y , Ji X , Lu T , et al. DeWiCam: Detecting Hidden Wireless Cameras via Smartphones[C]// the 2018. 2018.

[6] Hidden Camera Detector. https://apps.apple.com/us/app/hidden-camera-detector/id532882360.

[7] Dawson C , Inkpen S , Nolan C , et al. A New Approach to Pipeline Leak Detection Using Electromagnetic Sensing[C]// International Pipeline Conference. 2016:V003T04A008.

[8] Trimananda R, Varmarken J, Markopoulou A, et al. Packet-level signatures for smart home devices[C]//Network and Distributed Systems Security (NDSS) Symposium. 2020, 2020.

[9] Acar A, Fereidooni H, Abera T, et al. Peek-a-boo: I see your smart home activities, even encrypted![C]//Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks. 2020: 207-218.

[10] Hernandez S M, Bulut E. WiFederated: Scalable WiFi Sensing Using Edge-Based Federated Learning[J]. IEEE Internet of Things Journal, 2021, 9(14): 12628-12640.

[11] Bocus M J, Lau H S, McConville R, et al. Self-Supervised WiFi-Based Activity Recognition[C]//2022 IEEE Globecom Workshops (GC Wkshps). IEEE, 2022: 552-557.

[12] Gu Y, Zhang X, Wang Y, et al. WiGRUNT: WiFi-enabled gesture recognition using dual-attention network[J]. IEEE Transactions on Human-Machine Systems, 2022, 52(4): 736-746.

[13] Crow B P, Widjaja I, Kim J G, et al. IEEE 802.11 wireless local area networks[J]. IEEE Communications magazine, 1997, 35(9): 116-126.

[14] CBSNewYork Team. 2021. Brooklyn Man Thomas Tamborski Accused Of Spying On Female Roommates With Hidden Cameras. https://tinyurl.com/2d4n2bsp..

[15] Brian Krebs. 2019. Hidden Cam Above Bluetooth Pump Skimmer. https://tinyurl.com/d5sm9p9v.

[16] Abbas Acar, et al. Peek-a-Boo: I see your smart home activities, even encrypted!, WiSec 2020.

[17] Richardson I . H.264 and MPEG-4 video compression : video coding for next-generation multimedia[M]. Wiley, 2004.

[18] Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In Advances in neural information processing systems, pages 3079–3087.

[19] Peters M E, Neumann M, Iyyer M, et al. Deep contextualized word representations. arXiv 2018[J]. arXiv preprint arXiv:1802.05365, 1802, 12.

[20] Radford A, Narasimhan K, Salimans T, et al. Improving language understanding with unsupervised learning[J]. 2018.

[21] Bowman S R, Angeli G, Potts C, et al. A large annotated corpus for learning natural language inference[J]. arXiv preprint arXiv:1508.05326, 2015.

[22] Williams A, Nangia N, Bowman S R. A broad-coverage challenge corpus for sentence understanding through inference[J]. arXiv preprint arXiv:1704.05426, 2017.

[23] Dolan B, Brockett C. Automatically constructing a corpus of sentential paraphrases[C]//Third International Workshop on Paraphrasing (IWP2005). 2005.

[24] Sang E F, De Meulder F. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition[J]. arXiv preprint cs/0306050, 2003.

[25] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2383–2392.

[26] Dao D, Salman M, Noh Y. DeepDeSpy: A Deep Learning-Based Wireless Spy Camera Detection System[J]. IEEE Access, 2021, 9: 145486-145497.

[27] Hong Ye He, Zhi Guo Yang, and Xiang Ning Chen. 2020. PERT: Payload Encoding Representation from Transformer for Encrypted Traffic Classification. In 2020 ITU Kaleidoscope: Industry-Driven Digital Transformation, Kaleidoscope, Ha Noi, Vietnam, December 7-11, 2020. IEEE, 1–8.