IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 11, 2021, accepted October 1, 2021, date of publication October 20, 2021, date of current version November 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3121254

DeepDeSpy: A Deep Learning-Based Wireless Spy Camera Detection System

DINHNGUYEN DAO, (Member, IEEE), MUHAMMAD SALMAN, (Member, IEEE), AND YOUNGTAE NOH, (Member, IEEE)

Department of Electrical and Computer Engineering, Inha University, Incheon 22212, South Korea

Corresponding author: Youngtae Noh (ytnoh@inha.ac.kr)

This work was supported by the Research Grant from Inha University.

ABSTRACT Spy cameras planted in various private places, such as motels, hotels, homestays (*i.e.,* Airbnb), and restrooms, have raised immense privacy concerns. Wi-Fi spy cameras are used extensively by various adversaries because of easy installability, followed by size reduction. To prevent invasions of privacy, most studies have detected wireless cameras based on video traffic analysis and require additional synchronous data from external sensors or stimulus hardware to confirm the user's motion. Such supplements make the users uncomfortable, requiring extra effort and time for setting. This paper proposes an effective spy camera detection system called *DeepDeSpy* to detect the recording of a spy camera with no effort from the user. The core idea is using the channel state information (CSI) and the network traffic from the camera to detect whether the wireless camera records the movements of the user. The CSI signal is prone to motion, and detecting motion from an enormous amount of CSI data in real-time is challenging. This was handled by leveraging the convolutional neural network (CNN) and bidirectional long short-term memory (BiLSTM) deep learning methods. Such synergistic CNN and BiLSTM deep learning models enable instant and accurate detection by automatically extracting meaningful features from the sequential raw CSI data. The feasibility of *DeepDeSpy* was verified by implementing it on both a PC and a smartphone and evaluating it in real-life scenarios (e.g., various room sizes and user physical activities). The average accuracy achieved in different real-life settings was approximately 96%, reaching 98.9% with intensive physical activity in the large-size room. Moreover, the ability to achieve instant detection on a smartphone within only a one-second response time makes it workable for real-time applications.

INDEX TERMS Channel state information (CSI), network traffic, Wi-Fi monitoring mode, deep learning, convolutional neural network (CNN), bidirectional long short-term memory (BiLSTM).

I. INTRODUCTION

The use of surveillance cameras to consistently monitor surveillance-required places, such as banks, offices, roads, subways, and shopping malls, has become common everywhere to prevent unfortunate incidents, such as robbery, assault, and homicide. There is a huge proliferation in the development of cameras in terms of different sizes, shapes, functionalities, and applications. Modern technology has made possible cameras as small as the size of a human nail while embedding all the features required for the normal operation of a camera, such as Wi-Fi connectivity, audio support, and high definition visual quality. Moreover, they are available in different packages, such as power outlets, USB chargers, and smoke detectors [1]. Such innovations have permitted its illegal use in many private zones, such as hotel rooms [2], Airbnb [3], [4], restroom [5], and changing room [6], which can expose and record the private moments of the victims. Figure 1 gives a typical example, where the camera is placed in a hidden place (dress stand), and the victim is unaware of its placement. This camera (hereafter called spy camera) surreptitiously records the monitored area and streams the data to the remote storage (cloud storage) or local storage through a Wi-Fi connection. At the other

The associate editor coordinating the review of this manuscript and approving it for publication was Olutayo O. Oyerinde.

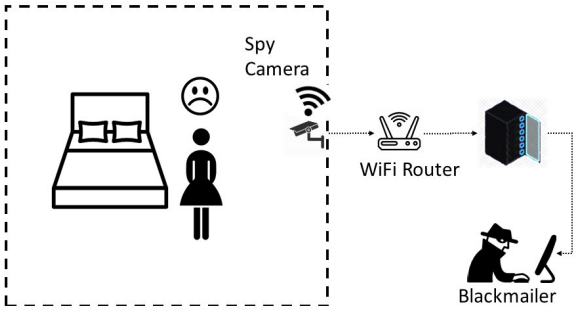

FIGURE 1. Spy camera installed in a hidden place in a private zone, which continuously sends the footage of the monitored area to a local/remote storage via a Wi-Fi link.

end, the offender can view and download the streaming, use it for blackmail or other illegitimate purposes. In the last few years, this type of privacy invasion has massively raised public privacy concerns [2]–[6].

To cope with this, people have sought reliable and affordable solutions that can detect nearby hidden cameras. Several solutions are available both in the form of hardware and smartphone applications. (1) The hardware device called spy camera hunter [7] is available on the market. On the other hand, this device is bulky and difficult to carry everywhere by the user. Furthermore, it only detects analog streaming, such as PAL, NTSC, and cannot identify encoded traffic, which is common in Wi-Fi IP cameras. (2) There are many smartphone applications available [11]–[14], and [15] for both iOS and Android. Most [11]– [14] require prior knowledge of the hidden camera placement, which is unrealistic because of its small size and unsuspected packagings, such as a fire alarm or wall clock. Another application stimulated by smartphone flashing is used for spy camera detection [11] by coercing the camera into generating a video signal. The emitted RF signal can be identified by scanning the RF bands for signals that match the flashing pattern. Unfortunately, this application is affected by the lighting conditions of the environment. Applications [12] and [13] are tracking the camera based on the radiated electromagnetic waves using the magnetometer sensor of the smartphone. On the other hand, the electromagnetic radiation from the camera is very small, and it requires close proximity to detect it. In summary, any application that requires proximity with the hidden camera is impractical.

On the other hand, *DeWiCam* [15] application does not require proximity with the spy camera. It forces the user to carry the smartphone while moving for approximately 15 to 30 seconds so that the motion is sensed via the accelerometer for camera localization in a room. Nevertheless, the *DeWiCam* has certain drawbacks, such as the requirement for producing an effort-intensive motion like jumping, waving hands, and walking around the room, as well as a long delay for detection.

The above drawbacks were addressed by exploiting the variations of two main parameters from a Wi-Fi camera: channel states information (CSI) and video traffic. These parameters were stimulated by the movements produced by everyday activities [16]. Finally, based on their correlation, camera detection is anticipated. The main question is how to implement a sophisticated detecting system that can tackle the substantial amount of raw CSI data to detect the spy camera efficiently in real-time. Herein, this paper proposes an effective and effortless solution called *DeepDeSpy*[1] that leverages the CSI and the camera traffic to detect a spy camera. The camera detection process involves the following steps: (1) The existence of a wireless camera[2] is identified based on its packet's length distribution [15]. (2) The traffic and CSI data are captured from the target camera. (3) A deep learning technique combining the CNN and LSTM networks can be used to classify human motion from CSI data and obtain reliable performance. (4) Camera recording is detected based on the correlation between the fluctuations of the camera traffic (*i.e.,* video traffic) and human motion (*i.e.,* CSI variation).

The major contributions of this study are as follows:

- This paper proposes a system for detecting a spy camera without making labor-intensive movements (e.g., jumping, and waving) or carrying a smartphone by leveraging the CSI signals of the Wi-Fi camera.
- A CNN-BiLSTM deep learning model that exploits a synergistic CNN and Bidirectional LSTM (BiLSTM) for CSI motion detection is proposed. The CNN can learn the features efficiently and automatically while BiLSTM processes the CSI data sequentially in the backward and forward directions to improve the effectiveness of motion recognition.
- Various effects of *DeepDeSpy* in diverse real-life were evaluated thoroughly: (1) the effect of multipath by assessing it in different-sized rooms, (2) passive motion sensing by placing the smartphone at different locations relative to the camera, and (3) sensitivity gain by assessing the physical activity of different intensities. The CNN-BiLSTM model was also installed in an android app to verify its feasibility in real-time.

The remainder of this paper is structured as follows. Section II summarized some related work on spy camera detection and related deep learning schemes. Section III presents the problem statement and explains the system model, attacker model, and design requirements. Section IV outlines the design of *DeepDeSpy* and explains the CSI-based human movement detection. The experimental settings and results are thoroughly discussed in Section V. Section VI provides a discussion and future work. Section VII concludes this paper. Table 1 lists the acronyms and their definitions used in this paper.

[1] *DeepDeSpy* stands for Deep learning-based Detecting a Spy-camera.
[2] Solely detecting a wireless camera is not enough, *i.e.,* it can be a general camera installed outside somewhere for security reasons.

TABLE 1. Abbreviations and acronyms.

Acronym	Definition
BiLSTM	Bidirectional long short-term memory
CNN	Convolutional neural network
CSI	Channel state information
DNN	Deep neural networks
I, P, B	Intra-coded, Predicted, and Bidirectional predicted pictures
LSTM	Long short-term memory
MAC	Media access control
PCA	Principal component analysis
PLD	Packet length distribution
ReLu	Rectified Linear Unit
RNN	Recurrent neural networks

II. RELATED WORK

Recently, the leaking of private videos captured by hidden cameras has raised serious concerns. Several victims (including celebrities) have committed suicide because of lost self-esteem in society [9], [10]. Such a high privacy concern has spiked endeavors to find a suitable, affordable, and reliable solution for detecting spy cameras. *DeWiCam* [15] uses a smartphone in monitoring mode and captures the wireless traffic from the wireless medium. They then identify the wireless camera traffic from mixed traffic (e.g., laptop, TV, and gaming traffic) based on the packet length distribution. Another approach in [12] used a smartphone with an additional Wi-Fi (which was set on promiscuous mode) to detect the traffic from the surrounding environment and a web camera (used as a spy camera) generating video traffic. The camera was stimulated by flashing lights. Based on such stimulation, the camera traffic was identified from non-camera traffic. Wu *et al.* addressed the camera detection by the similarity of simultaneous observation [38], where the timing pattern of data transmission of a random camera was correlated with the timing pattern of a reference camera whose timing pattern was previously known for that specific scene. On the other hand, this work requires pre-training the known camera to obtain the timing pattern for every stochastic scene. A commercial hardware-based solution device known as a spy camera hunter [7], can detect and display video streaming by a wireless camera. This device works on a range of frequencies, such as 1.2GHz, 2.4GHz, and 5.8GHz, but it identifies streaming from an analog video source, such as PAL, NTSC, and SECAM. In contrast, the Wi-Fi cameras currently available compress and encrypt the video footage before transmitting it. Hence, this device could not detect such streaming.

On the other hand, many studies related to Wi-Fi-based human activity recognition have been developed. Human activity recognition based on a traditional machine learning approach called CARM was presented [17]. The authors processed the CSI by filtering its noise using the principal component analysis (PCA) de-noising technique, and the features were extracted. These features were then classified for motion detection by different machine learning techniques, such as logistic regression, support vector machines (SVMs), hidden Markov model (HMM), and deep learning. For a sequential or temporal series of data samples, such as Wi-Fi CSI signal, recurrent neural networks (RNNs) showed effective performance in various applications, such as stock market prediction [18], machine translation [19], text generation [20]. On the contrary, RNNs suffer from the vanishing gradients caused by a long series of multiplications of small values, which makes the learning process degenerate. Long short-term memory (LSTM) solves these problems using a unique additive gradient structure, which boosts performance for time series with temporal dependencies, such as Wi-Fi CSI signals [21]. Chen *et al.* [22] used attention-based bidirectional long short-term memory (ABLSTM) to learn the representative features in two directions from raw sequential CSI measurements. Hence, it preserves both past and future information for extracting the feature, further enhancing the learning process. For these traditional learning algorithms, it is difficult to identify and extract the correct features effectively. Deep neural networks (DNN) address this problem by learning features automatically. The convolution neural network (CNN) was used because of its effective learning from local features. In addition, bidirectional feature learning was leveraged using BiLSTM to exploit the dependencies in time series CSI data efficiently.

III. THE USER AND ATTACKER MODELS AND DESIGN REQUIREMENTS

This section defines the issue resolved in this study. *Given a victim in a private place, it is possible to detect whether he/she is being recorded by a hidden camera in real-time without prior knowledge of the environment.* The assumption is explained from both the user's and attacker's sides and the design requirement of *DeepDeSpy*.

A. USER MODEL

A user staying in a private place, such as a hotel, motel, or Airbnb, is assumed to be interested in detecting the recording of a hidden camera available in that particular place. It is further assumed that abundant mixed wireless traffic is available from various devices, such as ordinary wireless cameras for security purposes, laptops, IP TVs, and smartphones. This makes the scenarios more realistic because, in practice, there may be many network devices within the range of the user, and its consideration is vital. Furthermore, the user is assumed to have a smartphone that supports the monitoring mode, *i.e.,* the wireless network interfacing card of the smartphone can see the network traffic in the surrounding.

The user is not assumed to have prior knowledge of the location and position of the installed camera, even though it has to be in the communication range of the smartphone (*i.e.,* the smartphone should be able to detect its packets). It is not assumed that the user should carry the smartphone and walk in the room for camera detection, as reported elsewhere [15]. Instead, the smartphone should detect the camera wherever it has been placed (within the communication range). Another assumption is that the user should

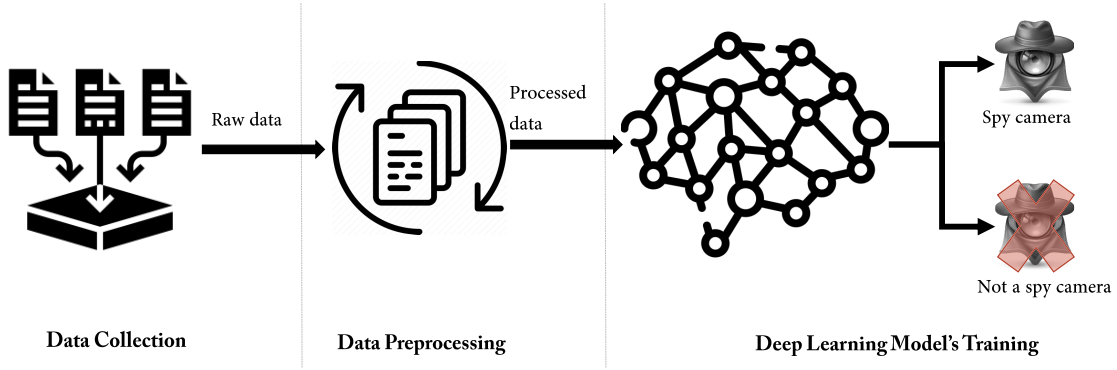

FIGURE 2. Deep learning-based camera detection method.

be associated with a network where the Wi-Fi camera is connected.

B. ATTACKER MODEL

It is assumed that the attacker can install the camera anywhere in a room. Moreover, the camera has a Wi-Fi device and access to local storage (inside the same building) or remote storage (e.g., cloud storage) to store its streaming data because there is limited internal memory inside the camera to store the captured footage for the long term. The attacker can access the live footage of the camera at any time online. The attacker also has full control over the network where the camera is connected. In addition, the attacker has full access to the local or cloud storage at which the streaming is stored.

The attacker keeps the camera for a considerably long period to record the target of interest. Hence, the attacker can place the camera in an unsuspected package, given that the target area is fully covered. The attacker can use more than one camera if the area is not fully covered with a single camera. Moreover, the attacker has full control over the camera configuration, such as changing the streaming quality and turning on and off its audio.

C. DESIGN REQUIREMENT

The goal of *DeepDeSpy* is to enable the users to detect a spy camera in a private place that is streaming their video without their consent. Hence, *DeepDeSpy* should meet the following requirements.

- Regardless of the size of the room, it should work in every indoor environment.
- It should work without joining the network.
- Regardless of video encryption, the spy camera should still be detected.
- It should work in any smartphone whose Wi-Fi card supports the monitoring mode.
- It should be a lightweight and affordable solution.

The fine-grained physical layer information referred to as CSI alongside video traffic analytic is used to detect whether a user is being recorded. To the best of the authors' knowledge, no known system so far has used these attributes concurrently to passively, painlessly, and instantly achieve impressive detection results.

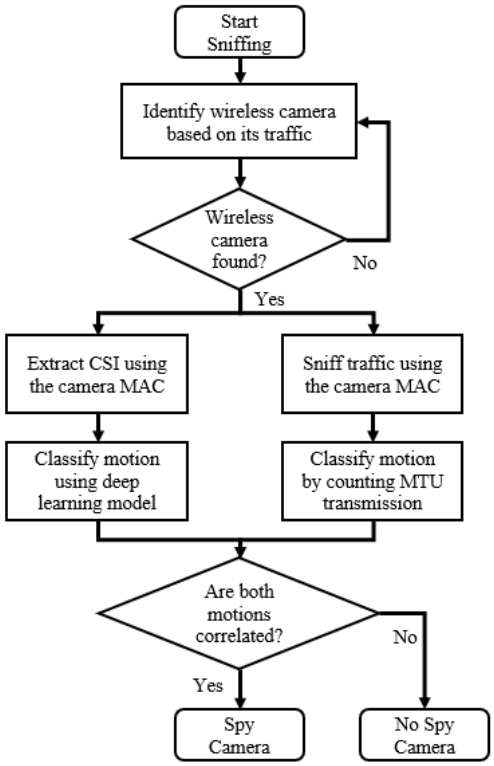

FIGURE 3. *DeepDeSpy* detection flowchart.

IV. DeepDeSpy METHODOLOGY

A. OVERVIEW

This section first introduces the flow of the *DeepDeSpy* algorithm illustrated in Figure 3. First, the smartphone Wi-Fi is placed in monitoring mode to inspect packets of all flows. Four features of each flow, including the transient Packet length distributions (PLD) feature, the hardware-related features, the bandwidth stability feature, and the PLD stability features, are calculated. The ExtraTrees classification algorithm is then used to identify the flow as wireless camera traffic [15]. The camera MAC address was extracted and added to

the list for further inspection. Second, two motion detection are performed simultaneously with each camera in the list to verify whether the camera is a spy camera (*i.e.,* recording the user in the target area): (1) detect human motion using the deep learning model after extracting CSI using Nexmon API [8] (presented in Section IV-B). (2) detect the motion in the recording area of the camera due to the amount of scene variation of camera traffic (presented in Section IV-C). Both parameters were extracted using a Nexus-5 smartphone with monitoring mode enabled. Finally, the motion detected by the CSI and camera traffic was correlated.

Furthermore, *DeepDeSpy* works with multiple cameras and multiple people. Table 2 lists the spy camera decision. Specifically, the most crucial case is case 1, where *DeepDeSpy* detects the motion from both the traffic and CSI to anticipate the spy camera. There could be a CCTV camera installed for security purposes, which may be listed in the camera inspection list. Case 2 anticipates a none-spy camera as it records another motion outside owing to no motion evidence inside from the CSI. This camera is therefore removed from the camera inspection list. Case 2 effectively mitigates the likelihood of false detection (*i.e.,* detecting a non-spy camera as a spy camera). Sometimes the none-line-of-sight motion (*i.e.,* a person moving outside the target area) causes some fluctuations in the CSI, but the camera bitrate could be stable because of no motion in the target area. Hence, such a camera is kept on the list for further inspection, as shown in case 3. Similarly, case 4 is also unclassified because of a lack of evidence of motion from both the bitrate and CSI.

TABLE 2. Probability of camera recording.

Case	Motion detection from traffic	Motion detection from CSI	Spy Camera Decision
1	✓	✓	Spy Camera
2	✓	×	None-spy Camera
3	×	✓	N/A
4	×	×	N/A

B. MOTION DETECTION FROM CSI DATA

DeepDeSpy requires two types of data to anticipate spy camera detection: (1) the CSI data to examine human motion inside the target place, and (2) the traffic of the camera, which is recording the target area. This data is detected concurrently, processed, and inspected for camera availability. The details are explained as follows.

1) CSI DATA COLLECTION

The CSI signal is affected vigorously by the multipath effect in indoor environments. This effect varies with the choice of room (room size), scatter inside the room (e.g., furniture), and human motion in that room. To fully evaluate the performance of *DeepDeSpy*, it was tested in rooms of different sizes, with different furniture arrangements. In all cases, the hidden camera was placed in a concealed place inside the room to realize a more realistic scenario. Table 3 provides details

TABLE 3. Environments for the experiment.

Room	Size	Data samples
Large room	$6m \times 8m$	120
Medium room	$4m \times 4m$	120
Small room	$1.5m \times 2.8m$	120

of the room and the number of samples collected. For each room, three physical activities were conducted: light physical activity, such as changing a jacket, medium physical activity (*i.e.,* moderate walking at a speed of 1.4m/s), and intensive physical activity (*i.e.,* skipping at a rate of 3 skips/sec). Forty samples were collected for each activity performed in each room, and each sample was recorded in 30 seconds, which was a combination of both sedentary behavior (initially staying still for 15s) and then physical activity. For the sake of unobtrusiveness sensing, the smartphone was placed in different corners of the room during data collection to assess the strength of the system to work effectively without requiring the user to carry the smartphone for camera detection.

2) CSI DATA PREPROCESSING

The raw CSI signal contains significant distortion that must be sanitized before the deep learning training phase. The data sanitizing step involves the removal of pilots and null subcarriers and extracting the magnitude from the sanitized CSI data. Sixty-four CSI subcarriers (52 data subcarriers, four pilots, and eight null subcarriers) in the 20 MHz bandwidth and 2.4 GHz frequency spectrum were considered. The information related to motion was preserved in the data subcarriers. They were classified into three categories: 1) the data subcarriers that carry the modulated data; 2) the pilot subcarriers that track the amplitude, frequency, and phase fluctuations; 3) the null subcarriers used as a guard carrier to avoid interference from the adjacent channels. These outliers (pilots and null subcarriers) for motion detection were removed to make the CSI information feasible for motion detection [23].

The CSI data subcarriers values can be represented as

$$H = [H_1, H_2, \ldots H_i, \ldots, H_N]^T, \quad i \in [1, 52], \tag{1}$$

where N is the number of subcarriers. The H_i subcarrier can be defined as

$$H_i = |H_i| e^{j \sin\{\angle H_i\}} \tag{2}$$

where $|H_i|$ and $\angle H_i$ denote the amplitude and phase responses for the i_{th} subcarrier, respectively. There are different approaches to detect motion using the amplitude [24], [25], phase [26], [27] or both terms [28], [29]. The amplitude term for training the deep learning model was exploited in the experiment.

3) CONVOLUTIONAL NEURAL NETWORKS

Because the CNN can extract spatial features to represent objects, it has been used with CSI to localize the indoor

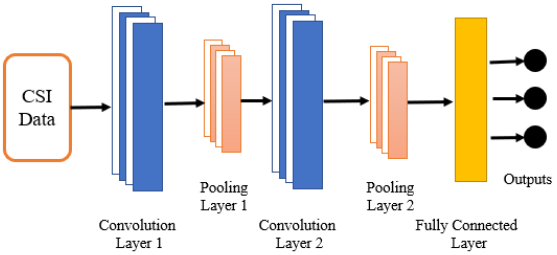

FIGURE 4. Illustration of a sample 1D-CNN with two convolution layers and one fully connected layer.

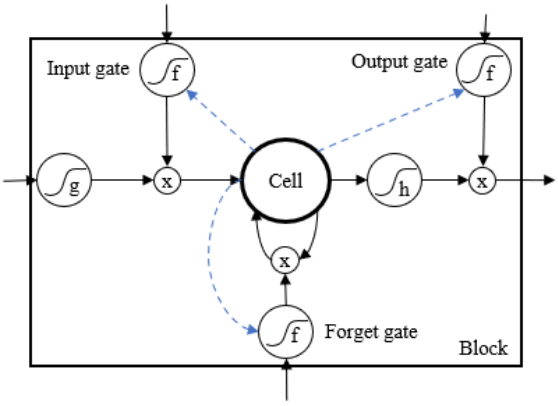

FIGURE 5. Illustration of the LSTM with input, output, forget gates, and activation functions.

IoT devices [30], [31], indoor people counting [32], and sign language gestures recognition [33]. 1D-CNN was leveraged in this model for automatically extracting the CSI features because of its effectiveness in extracting the features without manual engineering. Figure 4 shows the architecture of 1D-CNN, consisting of convolutional layers, pooling layers, and fully connected layers. The convolution operation uses a single filter to calculate one feature map. The ReLu function is used widely for activation purposes to keep the values of the features in a non-linear range (*i.e.,* from 0 to 1). On the other hand, it often yields enormous features, where some features are not required. For this purpose, the pooling blocks reduce the spatial size, keeping only the optimal features in the feature map. This makes the system more robust to noise and interference. These basic blocks can be increased or reduced based on the application requirement.

4) LONG SHORT-TERM MEMORY NETWORKS

In this paper, the Long Short-Term Memory network was applied to handle the Wi-Fi CSI data. LSTM networks were the same as RNNs, except that memory blocks replace the summation units in the hidden layer. The multiplicative gates allow LSTM memory cells to store and access information over long periods. As a result, they are better at finding and exploiting long-range dependencies in the data. The key component of LSTM is the cell state, which stores and passes information between the LSTMs. Each LSTM block has three gates: (1) forget gate, which is used to remove information from the cell; (2) input gate, which adds information to the cell state; (3) output gate. Figure 5 presents a single LSTM memory block. The three gates are non-linear summation units that are induced from the inside and outside blocks. The activation of the cell is controlled by multiplications (small circles with X inside). The input and output gates are used to multiply the input and output of the cell, while the forget gate is used to multiply the previous state of the cell. The activation function f of the gate is usually the logistic sigmoid, whereas the cell input and output activation functions (g and h) can be tanh, logistic sigmoid, or identity functions. The dashed lines denote the weighted ''peephole connections'' from the cell to the gates. The block yields a single output that is a multiplication from the cell followed by the h activation function and the output gate followed by f activation function.

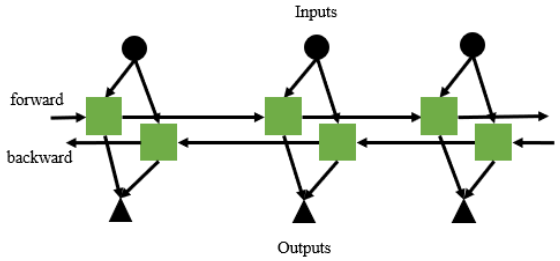

FIGURE 6. Illustration of the bidirectional LSTM.

5) BIDIRECTIONAL LONG SHORT-TERM MEMORY NETWORKS

A bidirectional LSTM network, which connects two LSTM layers of opposite directions to the same output, was used to access both past and future features in the current time. In doing so, the past and future features can be used efficiently via backward and forward states, improving accuracy. Figure 6 shows a BiLSTM sequence utilizing the aforementioned LSTM memory cells (green boxes) connected in the forward and backward directions.

6) DEEP LEARNING MODEL

In this application, the CNN and BiLSTM structures were merged to identify human motion in CSI data. This combination has three benefits. First, it extracts the features automatically without requiring detailed knowledge and generalization ability. Second, it holds temporal state information of the CSI over arbitrary time intervals, which further improves human motion detection. Third, the decoding process is less computationally expensive. Thus it can be run on the hardware of almost any smartphone.

a: CNN-BiLSTM FRAMEWORK

Figure 7 presents the proposed deep learning model for CSI-based human motion detection. The input data is the amplitude of the sanitized CSI (*i.e.,* 52 data sub-carriers) recorded in a sliding window of one second. First, for feature learning, three layers stacked 1D-CNN [35] are used to learn the features from the CSI signals. The BiLSTM,

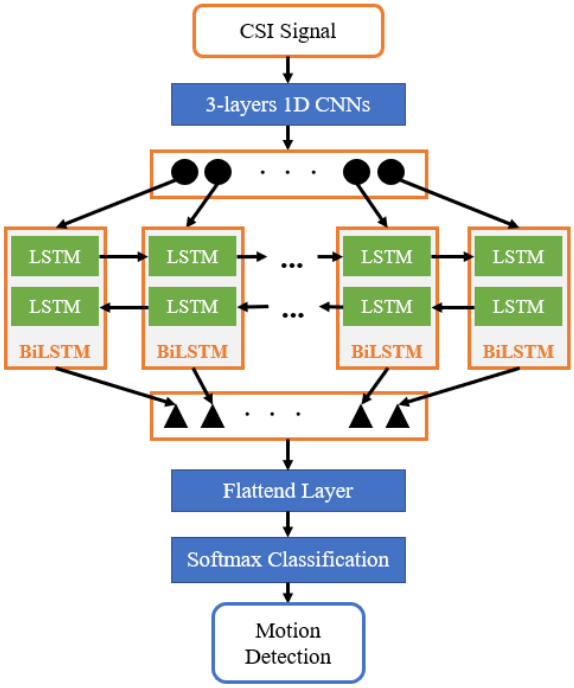

FIGURE 7. Model overview.

which incorporates backward and forward layers, is then used to learn the current state from the past and future states. The feature matrix formed is fed to the flattened layer that transforms it into a feature vector. Finally, the softmax classification takes the feature vector and classifies the user's state (*i.e.,* whether or not the user is moving). The overall model is illustrated in Figure 7.

b: MODELS PARAMETERS

Three 1D-convolution layers with 64 filters, which are the optimal parameters obtained from Section V-A, kernel size = 3, were chosen to extract the features with information from the neighboring data, and a ReLU activation function was used. The multipath effect affects the CSI data, which depends on different factors, such as room size and scatters inside the room. A 1D max-pooling layer with a pool size set to 3 was added to highlight the features most detected in diverse samples of CSI (e.g., different intensities of physical activities performed and different furniture placements in a room). The output from the 1D max-pooling layer was passed to the LSTM layer. The proposed model must preserve the past state (*i.e.,* static or dynamic) from the CSI data. Hence, 100 units (*i.e.,* hidden state or the output) were used in the LSTM layer to remember the complex pattern of the CSI. Moreover, ReLU activation, which enables the CNN to learn non-linear dependencies, was also allowed. After the LSTM layer(s) extracts the temporal dependencies of the sequential CSI features, the output is sent to the fully connected output layer with a Softmax activation binary classification output, *i.e.,* static and dynamic.

To compare the performance of this approach with the existing architectures, such as LSTM [21] and BiLSTM [22], firstly, the models were constructed using only LSTM and BiLSTM networks and added to the 1D convolution layers. These synergistic networks are referred to as the CNN-LSTM and CNN-BiLSTM networks, respectively. For training models, the learning rate was set to 0.0001 for the Adam optimizer, the mini-batch size was 32, and the number of training epochs was 100. These parameters are given from running an optimization framework called *Optuna*. These hyperparameters had little impact on the respective models, so they were chosen to be consistent across all models.

C. MOTION DETECTION FROM CAMERA TRAFFIC

The raw packets from the camera flow are detected using a Wi-Fi device that supports monitoring mode. The commodity cameras use the H.264 compression mechanism to the encoded video to save bandwidth [15]. The H.264 compression technique includes three types of pictures: Intra-coded (I), Predicted (P), and Bidirectional predicted (B) [36], [37]. The video data is compressed based on the Group of Pictures (GoP). The I-picture serves as a reference point for the other pictures in the GoP because it encodes the complete scenes independently. In contrast, the B and P pictures encode the inter-scene variations [1]. Thus, the GoP leads to lower video data transmission when there is no change in the scene (*i.e.,* still scene). By contrast, higher video data is transmitted when there is variation in the scene (*i.e.,* dynamic scene).

When there is a motion in the scene, multiple maximum-sized packets[3] are transmitted. Even in a static state, maximum-sized packets are transmitted because of the recurrent picture transmission. The maximum-sized packets are counted to confirm the motion in the target area. The motion recorded by the camera is expected if the amount of maximum-sized packets per unit time exceeds a certain threshold. In this experiment, the threshold was set to 50%, where half of the packets are MTU.

D. IMPLEMENTATION

This study used a wireless Wi-Fi camera working at 2.4GHz, named DEATTI, with several advanced features, such as support for 1080p resolution, night vision, and pan/tilt. First, a rooted Nexus-5 was used to collect data from wireless traffic. The monitoring mode was enabled with the CSI extraction feature by implementing the open-sourced GitHub repository called Nexmon CSI [8]. In monitored mode, the wireless traffic of the camera could be detected based on the camera's MAC address. Moreover, after the CSI extractor was set up, CSI data was obtained by listening to the frames on port 5500, which had a source address as 10.10.10.10 and destination address as 255.255.255.255. The CSI data, including the amplitude values and phase values in sub-carriers, are stored in the payload with the format

[3]The maximum transmission unit (MTU) of wireless cameras is 2,304 bytes.

TABLE 4. Parameter settings for the experiment.

Parameter settings	Parameter Tested
Camera brand	Deatti Wi-Fi IP camera
Compression algorithm	H.264
Smartphone	Google Nexus 5 (Android 8)
Window of recording	20-30 sec
Motion duration	5, 10, and 15 sec

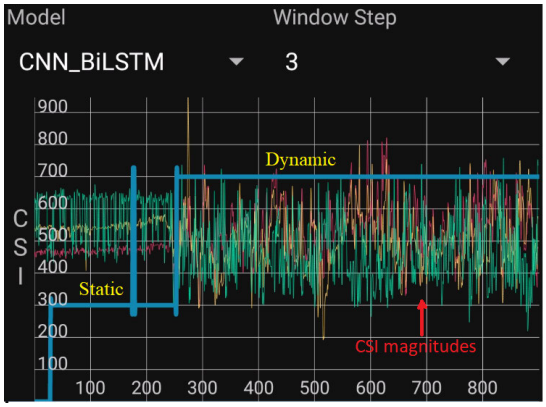

FIGURE 8. Snapshot of *DeepDeSpy* app. The blue line shows the motion detection result, while the slim lines show the magnitudes of raw CSI data of different subcarriers.

defined in [8]. Nexus-5, which has a single antenna, was used in the present experiment, and 64 sub-carriers of raw CSI data were collected at a 20 MHz wide channel.

To evaluate the effectiveness of the proposed approach, a Keras sequential model was constructed with a TensorFlow backend and compared with other deep learning-based variations, *i.e.,* conventional LSTM, bidirectional LSTM, which can learn the features automatically. All models used a one-second window containing 30 frames, where each frame contained 52 data subcarriers. A one-second sliding window was used to divide the activity recorded samples into 40,000 one-second data samples used for training and testing. The parameters of all the approaches were tuned carefully using a validation set from the training data. A 10 fold cross-validation was performed for evaluation: one-fold for testing, and the remaining is for training. The accuracy was calculated from the average of all 10 runs.

The model is converted to a TensorFlowLite model and implemented over the Nexus-5 smartphone. Our smartphone uses the modified driver of Nexmon [8], which enables it to detect the packets from the wireless traffic. It reads the packets from the camera flow and the corresponding CSI raw data through the UDP socket. When there are sufficient packets in the buffer from both sources, they are passed to the model and examined for recording detection. Figure 8 shows the classification result running on Nexus-5.

V. EVALUATION

This section assesses the performance of *DeepDeSpy*. The accuracy and robustness of the recording detection of the hidden camera were evaluated in the diverse and practical scenarios.

A. PARAMETERS TUNING

In this part, the optimal combination of parameter values for the model, such as the window size required for this application and the number of layers, which are needed to detect a spy camera accurately and responsively, were identified.

1) THE WINDOW SIZE

The impact of the input data size (or window size) on the final decision was also investigated to study the performance of *DeepDeSpy*. Figure 9 shows that the accuracy improves with increasing window size from one to five seconds across all models. On the other hand, the improvements are insignificant after a two-second window. Furthermore, the CNN with LSTM-based models (*i.e.,* CNN-LSTM and CNN-BiLSTM) achieved an approximately 2% gain in accuracy as the window size was increased from one to three seconds. A one-second window size, which sacrifices slight accuracy (*i.e.,* 96%) but achieves faster detection overall, was used because the intention was to make the spy camera detection system responsive and effective.

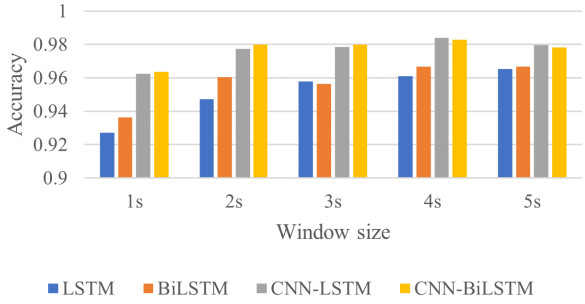

FIGURE 9. Impact of the window size on accuracy.

2) NUMBER OF LAYERS AND THE FILTER SIZE OF CNN STRUCTURE

The effects of the number of layers and the number of filter sizes on accuracy were investigated to determine the efficiency of the 1D-CNN structure. The overall accuracy improved with the increasing number of layers (from 1 to 5), as shown in Figure 10. On the other hand, the filter size should be chosen wisely. For example, if the smaller output filter sizes are used, such as 8 and 16, the accuracy still deteriorates regardless of increasing the number of layers; this occurs when less information is available for the next layer. Two or three layers can serve best for this application. In addition, by applying various output filter sizes, a 64 size was found to be the best fit for this application. In all subsequent evaluations, we use three layers with 64 filter size for the CNN structure.

B. MODELS COMPARISON

This section compares the performance of *DeepDeSpy* with various deep learning models to identify the best model in terms of accuracy and computational complexity.

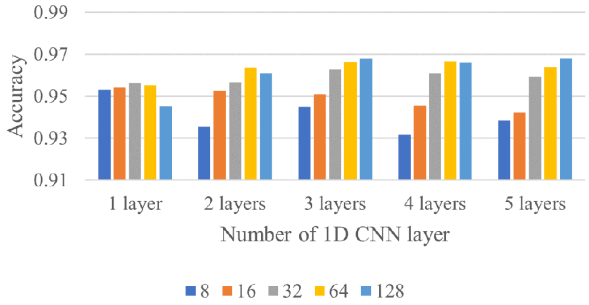

FIGURE 10. Impact of the number of layers on the accuracy.

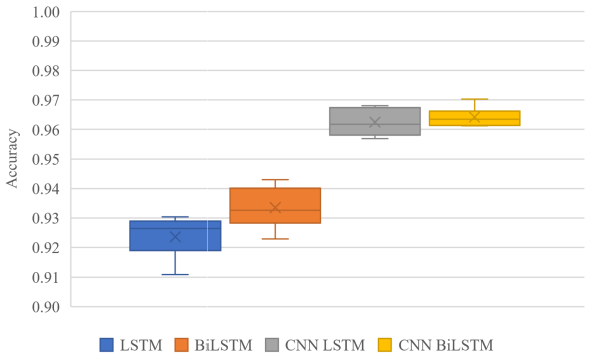

FIGURE 11. Accuracy of different models performed with 10 fold cross-validation.

Specifically, *DeepDeSpy* is intended for smartphones. Therefore, it should be a lightweight and efficient application.

1) OVERALL ACCURACY

Figure 11 presents box plots showing the average and the variance of accuracy attained by various models. To compare them fairly, they were trained on the same dataset, with the same parameters settings. The LSTM model showed the worst performance in terms of accuracy (92.36%) compared to BiLSTM (93.35%). The performance gain of BiLSTM was attributed to the use of bidirectional information (considering backward and forward data) for training its model. Leveraging the CNN layer for extracting the features from the CSI further improves the accuracy of these models. CNN with LSTM (CNN-LSTM) achieved an accuracy of 96.25%, whereas CNN with BiLSTM (CNN-BiLSTM) slightly outperformed the CNN-LSTM and achieved an accuracy of 96.32%. The slight improvement of CNN-BiLSTM was attributed to its consistency in all folds. Note that CNN-LSTM can achieve a better accuracy only in a folded dataset. The CSI data, often being unfolded, was treated effectively by the CNN-BiLSTM model in this experimental evaluation.

The related approach to detecting a spy camera was *DeWiCam* [15]. It uses the accelerometer of a smartphone to obtain the reading of human movement for in-room camera localization. On the other hand, there are two critical drawbacks of *DeWiCam*. First, it needs the user to carry the smartphone and perform labor-intensive motions, such as walking or jumping, because it uses an accelerometer sensor to sense

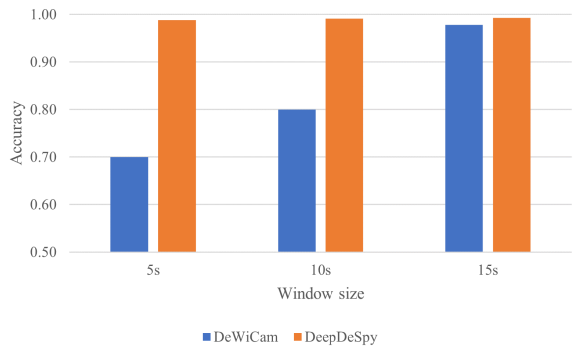

FIGURE 12. Detection accuracy of DeepDeSpy and DeWiCam.

TABLE 5. Testing time of the different models.

Time (milliseconds)	LSTM	BiLSTM	CNN-LSTM	CNN-BiLSTM
PC testing	0.287	0.459	0.285	0.412
Nexus-5 testing	9.028	13.136	5.614	6.352

the user's motion. The values from the sensor were compared with a predefined threshold to determine the motion period. Second, it requires users to stay still for a certain period and then move to detect human motion by perceiving both the rising and falling edges of the bitrate sequence using the CUSUM (cumulative sum control chart). As a result, the *DeWiCam* accuracy depends mostly on the amount of physical activity performed. For example, with a physical activity performed for 15 seconds, it can achieve 97.8% accuracy, but the performance decreases significantly by reducing the physical activity (e.g., 80% accuracy with 10 seconds of physical activity and 70% of accuracy with five seconds of physical activity). *DeepDeSpy*, on the other hand, is an effortless solution that does not require labor-intensive efforts and a long duration of physical activity because of its sensitivity. Under the scenario used in *DeWiCam*, *DeepDeSpy* achieved a detection rate of 96% for one second of physical activity and approximately 99% with five seconds of data or more. Figure 12 compares the performance of *DeepDeSpy* and *DeWiCam*.

2) TESTING TIME COMPARISON

Deep learning-based approaches take a longer time to train a model. On the other hand, once the model is trained, it does not require data pre-processing and feature extraction steps. Moreover, the online testing for deep learning-based approaches takes little time for classification and deciding for real-time applications.

A 7th generation core-i9 desktop computer with a processing speed of 2.80 GHz was used for training. The PC also contained GTX1080Ti NVIDIA GeForce GPU. For the testing, the computer and Nexus-5 smartphone were used. Table 5 lists the testing time for both the computer and smartphone.

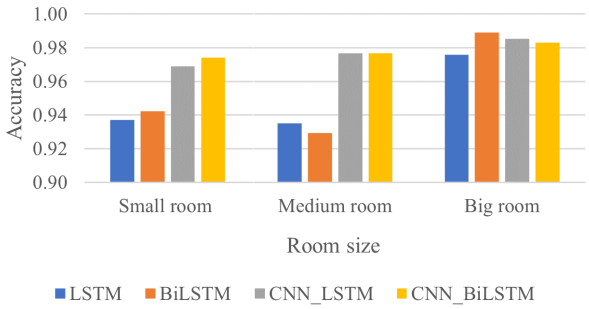

FIGURE 13. Accuracy of different models with different room sizes.

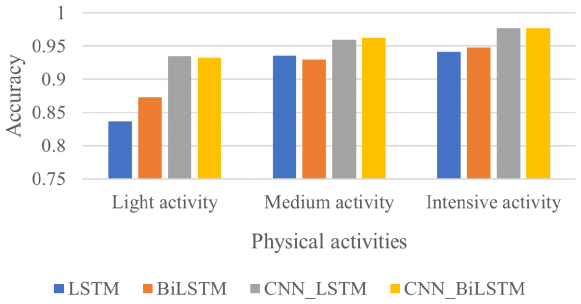

FIGURE 14. Impact of physical activity on accuracy.

Even with the smartphone, the testing time was very reasonable. For example, the testing time of CNN-BiLSTM on Nexus 5 was approximately six milliseconds per sample, whereas that on PC was around 0.4 milliseconds. Therefore, this model works effectively for CSI-based real-time motion detection.

C. EXPERIMENTAL RESULTS

The reliability of *DeepDeSpy* was tested by measuring its sensitivity regarding the intensity of physical activity (PA) performed and the diversity of indoor settings because the CSI fluctuations change with the intensity of the physical activity performed. For example, The CSI fluctuations caused by intensive physical activity, such as jumping, might be higher than that produced by a light physical activity, such as changing the jacket. Furthermore, The choice of chosen environment also plays a vital role in motion detection. For example, physical activity in a small room causes more multipath effects than in a large room. Finally, to show the unobtrusiveness of this system, it was placed in different locations in the room to assess camera detection without carrying a smartphone.

1) IMPACT OF THE ROOM SIZES

This study measured the effect of multipath reflection in the CSI signal caused by different indoor settings. Therefore, the room size was exploited as the variation in indoor settings. The experiment was conducted in three rooms with different sizes, *i.e.,* small, medium, and large sizes, as shown in Table 3. In each room, 40 samples were recorded for each physical activity, and the duration of each sample was 30 seconds. The initial 10 seconds was recorded deliberately in the static condition, and physical activity of diverse intensities (such as light, medium, and intensive) was performed after 10∼30 seconds. These mixed (static and then some physical activity) activities were performed to realize both sedentary and physical activity that has to be classified effectively by this model. Figure 13 presents the accuracy results.

The CNN-BiLSTM model outperformed the other models in all environmental settings. The merged structure of CNN and bidirectional LSTM automatically extracted the features while the classification was done based on the forward and backward information; this provides an effective generalization performance. The accuracy of the larger room was higher than the others because a smaller room has a smaller spacing between the walls that results in a severe multipath effect. Hence, the CSI experiences unstable behavior. The increased spacing between the walls (*i.e.,* larger rooms) leads to a more stable behavior of CSI (*i.e.,* lower multipath effect). For this reason, the CSI is prone to fluctuations in a small room (even in the static case), leading to more false positives, and causes more errors in motion detection. By contrast, the CSI in the larger room shows a more stable behavior in the static state.

2) EFFECT OF THE INTENSITY OF PHYSICAL ACTIVITY

Three types of activities were considered to assess the performance of *DeepDeSpy* with regards to physical activities: light physical activity, where the individual is changing the jacket; medium physical activity, where the user is walking around the room at a moderate pace; intensive physical activity in which the user is skipping in the target space. This experiment aimed to measure the sensitivity of camera detection in response to the intensity of physical activity. As shown in Figure 14, the accuracy of the model was proportional to the intensity of physical activity. CNN-LSTM and CNN-BiLSTM achieved approximately 98% accuracy with intense activity, whereas the accuracy was approximately 93.5% and 93.0%, respectively, with light physical activity.

3) SMARTPHONE PLACEMENT RELATIVE TO THE CAMERA POSITION

To passively detect the spy cameras (*i.e.,* the user not carrying a smartphone), the smartphone was placed at different positions relative to the camera position in the room, as shown in Figure 15. Figure 16 shows the results of the DeepDeSpy-enabled smartphone placed at different locations of the room. For each smartphone location in the room, 40 data samples were recorded, and the duration of each data sample was 30 seconds. Surprisingly, the smartphone had the strongest signal strength with the camera at Location 1, but it showed poor accuracy with all the models, *i.e.,* 83.1% average accuracy. In contrast, the other locations (Locations 2–5) achieved a significantly higher average accuracy of approximately 95.2%, 94.2%, 94.0%, and 95.0%, respectively, than Location 1. This is because the user motion in Location 1 does not obstruct the link between the smartphone and camera

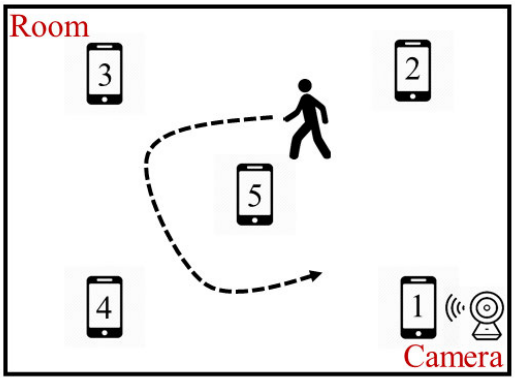

FIGURE 15. Smartphone placement at different locations relative to the camera.

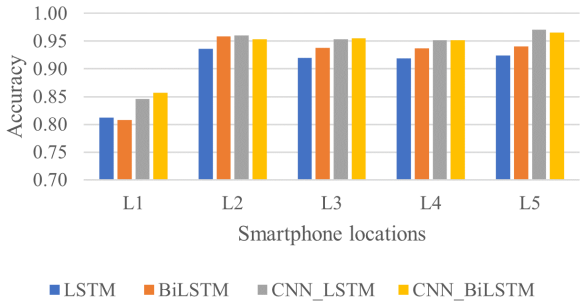

FIGURE 16. Accuracy of smartphone placements at five different locations.

directly, resulting in a lower variation in CSI due to user motion.

VI. DISCUSSION

A. COMBINING SEQUENTIAL DETECTION RESULTS

The accuracy can be increased by combining the decision performed by successive window's decision. And the final decision should be made based on the majority decision (detection or no detection). For example, the final decision based on the majority will be detection if there are three results from three successive windows with two anticipated detections with 90% accuracy. Decision-based on such successive window results can improve the accuracy by 99%.

B. Wi-Fi IN MONITORING MODE

To eavesdrop on the available network traffic, a smartphone needs to switch the Wi-Fi card into monitoring mode after being rooted and installed with modified Wi-Fi card firmware. In monitoring mode, data transmission and reception cannot occur [41] because the wireless adopter is busy listening to the packets in the wireless network. On the other hand, the connectivity can be resumed again once the monitoring application is suspended. A solution for this problem is to use an additional device to detect a Wi-Fi CSI signal, such as ESP32 Wi-Fi micro-controller in [42], then sending the data to a custom-built application in the smartphones.

C. EFFECT OF THE MULTIPATH

The multipath reflection in wireless signals is contributed by the number of scatters available in the room and the size of the room. A small room causes more reflection, leading to more multipaths that ultimately produce more false-positive results. To filter out the unwanted reflections, the key idea is to nullify the strong and irrelevant multipath signal and allow only the signals related to the motion [43], [44]. A lightweight solution for the efficient nullification of irrelevant reflected signals can be envisaged in future work.

D. BATTERY CONSUMPTION TEST

The packet detection process requires a monitoring mode, even though it is more power consuming than any other sensor or operation, such as cell scanning, accelerometer, or Wi-Fi scanning, as addressed in [45]. On the other hand, they require a phone to remain in monitoring mode for a long time, while *DeepDeSpy* requires monitoring mode for very little time. The power consumption is expected to be negligible. For further assessment, a battery consumption test for the *DeepDeSpy* should be performed in future work.

VII. CONCLUSION

This paper proposed *DeepDeSpy*, as an effective and lightweight approach for spy camera recording detection. It analyzes the fluctuation of CSI and network traffic of the camera at matched timestamp and anticipates the recording detection. A synergistic network of CNN and BiLSTM was proposed, which benefits automatic features extraction and spectral dependencies extraction. *DeepDeSpy* has been implemented on a PC and smartphone. The extensive evaluation showed that *DeepDeSpy* could achieve an average of 96% accuracy with only a one-second response time.

REFERENCES

[1] T. Liu, Z. Liu, J. Huang, R. Tan, and Z. Tan, "Detecting wireless spy cameras via stimulating and probing," in *Proc. 16th Annu. Int. Conf. Mobile Syst., Appl., Services*, Jun. 2018, pp. 243–255.
[2] A. Carter. *Study Says Guests are Finding Hidden Cameras Inside Rental Properties, Hotel Rooms*. ktnv news. Jul. 29, 2019. [Online]. Available: https://tinyurl.com/y7ukmt9r
[3] A. Castelan and J. Treanor. *Hidden Cameras Found Inside a Las Vegas Airbnb Rental Recording Naked People*. KSNV News3LV. Accessed: Oct. 20, 2016. [Online]. Available: https://tinyurl.com/zrtysgx
[4] J. S. IV. (Jan. 2015). *Couple Wakes up in Airbnb to Find Hidden Camera Watching Them*. [Online]. Available: https://tinyurl.com/y7ukmt9r
[5] R. Kenner. *Man Sues After Finding Hidden Cam in Hotel Bathroom*. Accessed: Sep. 9, 2012. [Online]. Available: https://tinyurl.com/ya8pwza9
[6] C. Mengxiao. *Hidden Camera Found in H&M Store's Changing Room*. Accessed: Jul. 8, 2017. [Online]. Available: https://tinyurl.com/y7psdcxr
[7] *The Signal Jammer*. Accessed: Jul. 17, 2020. [Online]. Available: https://www.thesignaljammer.com/
[8] S. Matthias, W. Daniel, and H. Matthias. (2017). *Nexmon: The C-Based Firmware Patching Framework*. Nexmon:Project. [Online]. Available: https://nexmon.org
[9] N. Smith. *South Korean Woman Commits Suicide After Doctor Filmed Her Using Spycam, Reports Say*. Accessed: Oct. 2, 2019. [Online]. Available: https://tinyurl.com/y52×98pk
[10] *Goo Hara and the Trauma of South Korea's Spy Cam Victims*. BBC News, Accessed: Nov. 28, 2019. [Online]. Available: https://tinyurl.com/y9jwkayl
[11] R. Marc, "How to find hidden cameras," Tech. Rep., Mar. 2002, pp. 1–36. [Online]. Available: http://www.tentacle.franken.de/papers/hiddencams.pdf
[12] B. Lagesse, K. Wu, J. Shorb, and Z. Zhu, "Detecting spies in IoT systems using cyber-physical correlation," in *Proc. IEEE Int. Conf. Pervas. Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2018, pp. 185–190.

[13] S. Zhu, C. Zhang, and X. Zhang, "Automating visual privacy protection using a smart LED," in *Proc. 23rd Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2017, pp. 329–342.
[14] T. Liu, Z. Liu, J. Huang, R. Tan, and Z. Tan, "Detecting wireless spy cameras via stimulating and probing," in *Proc. 16th Annu. Int. Conf. Mobile Syst., Appl., Services*, Jun. 2018, pp. 243–255.
[15] Y. Cheng, X. Ji, T. Lu, and W. Xu, "On detecting hidden wireless cameras: A traffic pattern-based approach," *IEEE Trans. Mobile Comput.*, vol. 19, no. 4, pp. 907–921, Apr. 2020.
[16] Y. Ma, G. Zhou, and S. Wang, "WiFi sensing with channel state information: A survey," *ACM Comput. Surv.*, vol. 52, vol. 3, no. 3, p. 46, 2019.
[17] W. Wang, A. X. Liu, M. Shahzad, K. Ling, and S. Lu, "Device-free human activity recognition using commercial WiFi devices," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1118–1131, May 2017.
[18] M. Hiransha, E. A. Gopalakrishnan, V. K. Menon, and K. P. Soman, "NSE stock market prediction using deep-learning models," *Proc. Comput. Sci.*, vol. 132, pp. 1351–1362, Feb. 2018.
[19] M. Sundermeyer, T. Alkhouli, J. Wuebker, and H. Ney, "Translation modeling with bidirectional recurrent neural networks," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1–12.
[20] D. Pawade, "Story scrambler-automatic text generation using word level RNN-LSTM," *Int. J. Inf. Technol. Comput. Sci.*, vol. 10, no. 6, pp. 44–53, 2018.
[21] S. Yousefi, H. Narui, S. Dayal, S. Ermon, and S. Valaee, "A survey on behavior recognition using WiFi channel state information," *IEEE Commun. Mag.*, vol. 55, no. 10, pp. 98–104, Oct. 2017.
[22] Z. Chen, L. Zhang, C. Jiang, Z. Cao, and W. Cui, "WiFi CSI based passive human activity recognition using attention based BLSTM," *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2714–2724, Nov. 2019.
[23] G. S. Matthew, *802.11 AC: A Survival Guide: Wi-Fi at Gigabit and Beyond*. Sebastopol, CA, USA: O'Reilly Media, 2013, pp. 1–154.
[24] J. Xiao, K. Wu, Y. Yi, L. Wang, and L. M. Ni, "FIMD: Fine-grained device-free motion detection," in *Proc. IEEE 18th Int. Conf. Parallel Distrib. Syst.*, Dec. 2012, pp. 229–235.
[25] L. Gong, W. Yang, D. Man, G. Dong, M. Yu, and J. Lv, "WiFi-based real-time calibration-free passive human motion detection," *Sensors*, vol. 15, no. 12, pp. 32213–32229, Dec. 2015.
[26] A. A. Uddin, R. Arablouei, F. D. Hoog, B. Kusy, R. Jurdak, and N. Bergmann, "Estimating angle-of-arrival and time-of-flight for multi-path components using wifi channel state information," *Sensors*, vol. 18, no. 6, p. 1753, 2018.
[27] H.-X. Chen, B.-J. Hu, L.-L. Zheng, and Z.-H. Wei, "An accurate AoA estimation approach for indoor localization using commodity Wi-Fi devices," in *Proc. IEEE Int. Conf. Signal Process., Commun. Comput. (ICSPCC)*, Sep. 2018, pp. 1–5.
[28] K. Qian, C. Wu, Z. Yang, Y. Liu, F. He, and T. Xing, "Enabling contactless detection of moving humans with dynamic speeds using CSI," *ACM Trans. Embed. Comput. Syst.*, vol. 17, no. 2, pp. 1–18, Jan. 2018.
[29] Z. Yang, Z. Zhou, and Y. Liu, "From RSSI to CSI: Indoor localization via channel response," *ACM Comput. Surv.*, vol. 46, no. 2, pp. 1–32, 2013.
[30] B. Berruet, O. Baala, A. Caminada, and V. Guillet, "DelFin: A deep learning based CSI fingerprinting indoor localization in IoT context," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2018, pp. 1–8.
[31] C.-H. Hsieh, J.-Y. Chen, and B.-H. Nien, "Deep learning-based indoor localization using received signal strength and channel state information," *IEEE Access*, vol. 7, pp. 33256–33267, 2019.
[32] Y.-K. Cheng and R. Y. Chang, "Device-free indoor people counting using Wi-Fi channel state information for Internet of Things," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.
[33] Y. Ma, G. Zhou, S. Wang, H. Zhao, and W. Jung, "SignFi: Sign language recognition using WiFi," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 2, no. 1, p. 23, 2018.
[34] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM networks," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul./Aug. 2005, pp. 2047–2052.
[35] M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, J. Zhu, P. Wu, and J. Zhang, "Convolutional neural networks for human activity recognition using mobile sensors," in *Proc. 6th Int. Conf. Mobile Comput., Appl. Services*, 2014, pp. 197–205.
[36] J. Minqiang, X. Yi, and N. Ling, "Improved frame-layer rate control for H.264 using MAD ratio," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 3, May 2004, pp. III–813.
[37] Y. Xin-Wei, W. Wang, S. Yang, Y. Cen, X. Yao, and T. Pan, "IPB-frame adaptive mapping mechanism for video transmission over IEEE 802.11e WLANs," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 2, pp. 5–12, 2014.
[38] W. Kevin and B. Lagesse, "Do you see what I see? Detecting hidden streaming cameras through similarity of simultaneous observation," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom*, Mar. 2019, pp. 1–10.
[39] S. N. Patel, J. W. Summet, and K. N. Truong, "Blindspot: Creating capture-resistant spaces," in *Protecting Privacy in Video Surveillance*. London, U.K.: Springer, pp. 185–201, 2009.
[40] S. Y. Cheng and C. Yang, "Portable remote control device with anti-hidden camera and anti-wiretapper functions," U.S. Patent Appl. 13 240 388, Mar. 29, 2012.
[41] Wikipedia Contributors. *Monitor Mode*. Wikipedia. Accessed: Aug. 5, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Monitor_mode
[42] S. M. Hernandez and E. Bulut, "Performing WiFi sensing with off-the-shelf smartphones," in *Proc. IEEE Int. Conf. Pervas. Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2020, pp. 1–3.
[43] Q. Kun, C. Wu, Y. Zhang, G. Zhang, Z. Yang, and Y. Liu, "Widar2.0: Passive human tracking with a single Wi-Fi link," in *Proc. 16th Annu. Int. Conf. Mobile Syst., Appl., Services*, 2018, pp. 350–361.
[44] X. Yaxiong, J. Xiong, M. Li, and K. Jamieson, "mD-Track: Leveraging multi-dimensionality for passive indoor Wi-Fi tracking," in *Proc. 25th Annu. Int. Conf. Mobile Comput. Netw.*, 2019, pp. 1–16.
[45] Y. Chon, S. Kim, S. Lee, D. Kim, Y. Kim, and H. Cha, "Sensing WiFi packets in the air: Practicality and implications in urban mobility monitoring," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2014, pp. 189–200.

DINHNGUYEN DAO (Member, IEEE) received the B.S. degree in mechanical engineering from the Hanoi University of Science and Technology, Vietnam, in 2013, and the M.S. degree from the Department of Electrical and Computer Engineering, Inha University, Incheon, South Korea, in 2016, where he is currently pursuing the Ph.D. degree. His research interests include wireless networks, traffic monitoring, programmable switch, deep learning, and intrusion detection.

MUHAMMAD SALMAN (Member, IEEE) received the B.S. degree in electronic engineering from the Balochistan University of Information Technology, Engineering and Management Sciences (BUITEMS), Quetta, Pakistan, in 2010, and the M.S. degree in electronic engineering from the Politecnico di Torino, Turin, Italy, in 2014. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Inha University, Incheon, South Korea.

From October 2014 to June 2019, he was a Lecturer with the Electrical and Computer Engineering Department, Effat University, Saudi Arabia. His research interests include bufferbloat mitigation, wireless networks, and software-defined networks.

YOUNGTAE NOH (Member, IEEE) received the B.S. degree in computer science from Chosun University, in 2005, the M.S. degree in information and communication from the Gwangju Institute of Science and Technology (GIST), in 2007, and the Ph.D. degree in computer science from the University of California, Los Angeles (UCLA), in 2012. He is currently an Associate Professor with the Department of Computer Science and Information Engineering, Inha University. Before joining Inha University, he was a Staff Member at Cisco Systems, until 2014. His research interests include data center networking, wireless networking, future internet, and mobile/pervasive computing.