A Deep Learning Approach to IoT Authentication

Rajshekhar Das, Akshay Gadre, Shanghang Zhang, Swarun Kumar, and José M. F. Moura
Department of Electrical and Computer Engineering
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh PA 15213 USA

***Abstract*—At its peak, the Internet-of-Things will largely be composed of low-power devices with wireless radios attached. Yet, secure authentication of these devices amidst adversaries with much higher power and computational capability remains a challenge, even for advanced cryptographic and wireless security protocols. For instance, a high-power software radio could simply replay chunks of signals from a low-power device to emulate it.**

This paper presents a deep-learning based classifier that learns hardware imperfections of low-power radios that are challenging to emulate, even for high-power adversaries. We build an LSTM framework, specifically sensitive to signal imperfections that persist over long durations. Experimental results from a testbed of 30 low-power nodes demonstrates high resilience to advanced software radio adversaries.

I. Introduction

Recent years have witnessed a variety of wireless technologies that allow extremely low-power devices to communicate with each other in the cloud, forming the building blocks for the Internet-of-Things (IoT). Low-power Wide-Area Networking (LP-WAN) solutions, such as 3GPP's NB-IoT(NarrowBand IoT) , LoRaWAN , and SIGFOX allow devices powered by 10-year batteries to connect to cell towers several miles away. Yet, ubiquitous communication at extremely low-power comes at a price: security vulnerabilities. While most devices in the Internet-of-Things are likely to be low-power [1], adversaries who intend to break the system need not be. In particular, this paper focuses on authentication, where a low-power device seeks to establish its identity to a wireless base station, in the presence of attacks from high-power adversaries. Perhaps the simplest illustration of such an attack is a high-power software-radio adversary that could simply replay segments of signals from a low-power device and therefore emulate it.

Despite the rich literature on security-frameworks in the context of IoT [2], past work from both the security and wireless communities struggle in the face of asymmetric adversaries that are vastly superior in power and computation, compared to low-power nodes. Cryptographic protocols from the security community can at best be implemented with short key lengths and limited computation by low-power devices, making them vulnerable to computationally more powerful adversaries. Recent work from the wireless community that proposes *fingerprinting* wireless devices based on their location [3], [4] or the transmitting device-specific hardware properties [5], [6], struggle to generalize across transmitters and for adversaries that may be co-located with the nodes.

[rajshekd,agadre,shanghaz,swarunk,moura]@andrew.cmu.edu

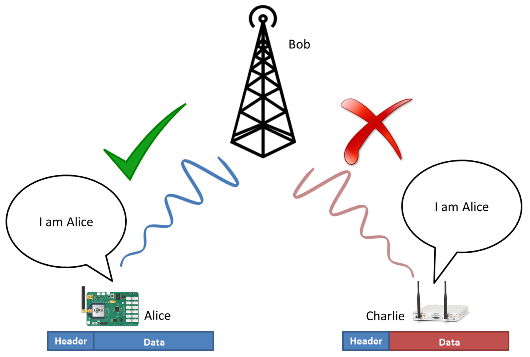

Fig. 1: Telling-Apart Legitimate Nodes from Malicious Nodes Emulating Them

This paper presents a system that securely identifies low-power devices (Fig. 1), amidst high-power adversaries, in a manner that is transmitter-agnostic. It presents, to our knowledge, the first deep learning model that *learns* the hardware imperfections of any given low-power radio device, and dynamically discovers which of these features make it unique. It demonstrates how it can tell apart legitimate nodes from high-power adversaries that transmit identical modulation, coding and even data, given that these adversaries inadvertently introduce their own distinct imperfections. Experimental validation from a 30-node testbed of LP-WAN radios and software-radio adversaries deployed in the CMU campus, reveals promising median identification accuracy of 99.58 % at high signal-to-noise ratios (SNR > 31 dB) that goes down to ≈ 90 % at low SNR.

At the core of our solution is a machine learning architecture that dynamically learns hardware imperfections of transmitters using the wireless channels of signals observed from them. Most of the machine learning techniques require cleverly crafted features that allow them to classify signals as belonging to different users accurately. However, given the large variety of hardware imperfections radios experience, it is a challenging to manually construct features that are truly unique to each radio. Hence, we employ deep neural networks that "learn" hierarchical non-linear features that can discover the hidden structures of the data better than hand-crafted features.

Developing a deep learning model particularly suited to wireless signals, while remaining computationally inexpensive is challenging. Traditional deep neural networks from image classification tend to exploit locality of features, which is often not the case in the wireless context, with hardware imperfections such as carrier frequency offsets affecting the entire signal uniformly [7]. We therefore use a special class of deep networks called Long Short Term Memory (LSTM) [8], [9] commonly used in automatic speech recognition and financial data prediction. These networks are designed to account for long term dependencies in sequential data and are therefore much more suited to the wireless setting. We further ensure that the learning architecture can identify users in real-time, by considering available computational capabilities at the base station in its design parameters. We present results from a testbed of low-power devices and software radios and evaluate system performance in various adversarial settings.

The remainder of the paper is organized as follows: section II details relevant background and our problem statement, sections III and IV introduce the LSTM framework, section V demonstrates the performance of LSTM for authentication, section VI concludes the paper with directions for future work.

II. Related Work

Related work falls under three main categories: (i) Cryptography for low-power networks; (ii) Device fingerprinting in wireless networks; and (iii) LSTM and neural networks.

Cryptography: There has been much research on securing low-power sensor networks using cryptography. While some approach this problem through implementation through cooperative ciphertext [10], [11], other approaches use ASIC and FPGA implementations [12], [13] of state-of-the-art cryptographic algorithms to improve security guarantees under power constraints. However, the above approaches face two primary problems: (i) overhead in transmitting and distributing keys; and (ii) vulnerability to high-power adversaries. Indeed, for LP-WAN IoT devices that run for 10 years on battery, even the smallest increase in power ($\sim \mu$J) requirement reduces the lifetime of devices significantly. Our approach performs IoT authentication purely at the base stations, with no modifications to low-power devices whatsoever, and functions even in the presence of high-power adversaries. Indeed, the training of the deep neural network can be offloaded to the cloud to further reduce computational complexity at the network edge.

Wireless Fingerprinting: Past work has exploited technology-specific features of wireless radios to uniquely identify them, for instance: rate switching, active scanning, packet arrival times, response to cleverly crafted frames [6], clock skew, and other implementation based features [14]. Other implementations use location-specific features such as RSS/CSIR[15], and channel response [16], that assume that the adversaries are not co-located with legitimate devices. Unlike the above approaches, our approach toward IoT authentication is both protocol agnostic and allows the adversary to be co-located with legitimate nodes. It aims to learn and utilize minuscule hardware and firmware imperfections of individual devices to uniquely identify them, with different features learned across different technologies.

LSTM and Neural Networks:

In recent years, novel deep learning algorithms have achieved state-of-the-art performance in a wide range of machine learning tasks. This success can be attributed to its ability to learn hierarchical representations, unlike traditional methods that rely upon hand-engineered features. Recurrent Neural Networks (RNNs) [17] have been particularly successful for the learning tasks on sequential data. RNNs capture the dynamics of sequences via cycles in the network of nodes and retain a state that can represent information from an arbitrarily long context window. Due to the problems of vanishing and exploding gradients, systems based on the long-short-term memory (LSTM) architecture have been proposed [18], [8]. LSTM has been deployed for image classification, learning financial time-series, and speech recognition. This paper, in contrast, applies them to low-power wireless signals, that have correlation across long time duration, allowing us to overcome unique challenges from multipath, noise, and high-power adversaries.

III. Background on LSTM

The LSTM networks are recurrent neural networks that can be seen as feed-forward neural networks unrolled in time with parameter sharing amongst the layers. Each of these recurring hidden layers (LSTM units) consist of *memory blocks* that are connected via a cell state. The cell state allows LSTM to keep track of the history of the sequential input and updates it wherever necessary. Each LSTM unit consists of connections called gates that control the information flow. One of the crucial gates we employ is the *forget* gate (f_t) [8], that controls what information from the previous cell state to drop. This function is responsible for controlling what proportion of the information from all the previous nodes needs to be forwarded to the next LSTM unit. A sigmoid (σ) value of “1” would allow the corresponding cell state component to pass completely, whereas a “0” would prevent further propagation, essentially, forgetting it.

The other gates include the *input* and *output* gates that control the input activations into the LSTM unit and the output activations to the next block. These are responsible for generating the effect of the input of the current block on the output cell state. The input ($\tilde{s_t}$) is combined with a sigmoid layer that selects that values to update, to yield the value that actually modifies the cell state via the *adder* block. Similarly, we combine the output activation (o_t) with a sigmoid layer to yield the desired output used in the next block. This is responsible for combining the two informations, i.e., input activations and previous cell states in appropriate proportions to output the cell state to the next block. The set of recursive equations are graphically presented in Fig 2. Here, the input at time t is given by x_t, h_{t-1} denotes the output from the previous LSTM unit, and s_t represents the state vector at time t. The LSTM network is governed by the equations below ($\odot$ denotes the Hadamard product):

$$f_t = \sigma(W_f x_t + b_f) \tag{1}$$
$$i_t = \sigma(W_i x_t + b_i) \qquad \tilde{s_t} = \tanh(W_s x_t + b_s) \tag{2}$$
$$s_t = f_t \odot s_{t-1} + i_t \odot \tilde{s_t} \tag{3}$$
$$o_t = \sigma(W_o x_t + b_o) \qquad h_t = o_t \odot \tanh(s_t) \tag{4}$$

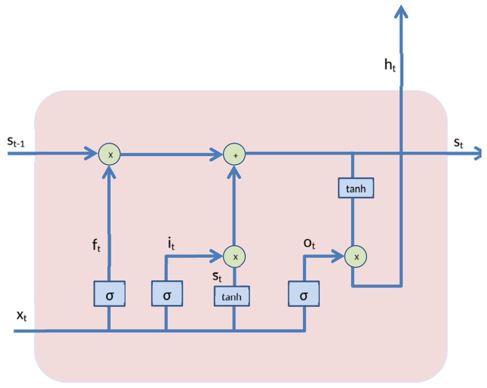

Fig. 2: Visualization of an LSTM Unit

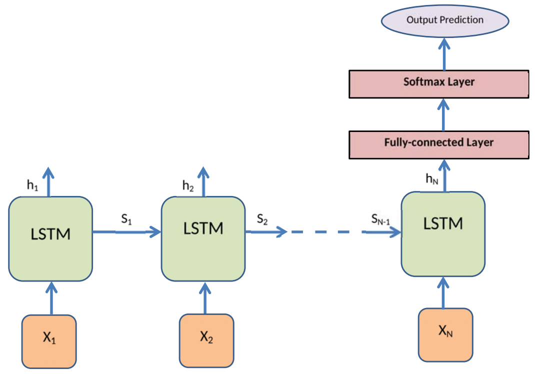

Fig. 3: LSTM Feature Learning and Device Identification

IV. Our Approach

In this section, we develop an LSTM deep learning architecture for our classifier, trade-offs among design parameters, resilience to noise and other propagation limitations, and impact of high power adversaries.

Threat model: We assume that transmissions from each legitimate radio follow state-of-the-art protocols and are pre-recorded in a training phase across a large sub-set of possible locations of these radios (to support resilience to multipath, see IV-D). The adversaries can be either low-power (milli-watts) or high power (up to 1 W) devices, including software radios. We assume the adversaries can listen to the radio signal waveform transmitted by the low-power device it seeks to emulate, and can replay these signals to mount an attack.

A. Classification Architecture and Approach

We model the IoT authentication problem as a multi-label classification task with the input being the physical I/Q samples of the data packets from the received signals, and the output being the ID of one of the devices. To design a suitable classifier, we have to understand the structure of the input that the machine learning algorithm can exploit. Here, we are addressing wireless signal inputs that are often corrupted by various wireless impairments. Generally, these impairments persist through time and sensing them requires correlation across large time samples. For example, a frequency offset causes the phase of the wireless signal to drift across time. Traditional wireless technologies try to identify the relation between phase drift and the frequency offset to correct it. Similarly, a timing offset propagates through the whole signal since the phase offset in the signal is dependent on it. The wireless channel is also dynamic in real-world situations and can lead to perturbations spanning across time that need to be detected and corrected by the base stations on-the-go.

The traditional approach of manually-engineering such features is extremely difficult. Thus, we require a robust approach to extract reliable representations of these impairments by exploiting temporal dependencies of the signal. We use deep neural networks that have the ability to learn these features themselves. LSTM is a special class of deep networks that serves our purpose. LSTM networks with their use of recurrence, combined with gated connections, encode the sequential history through an internal state. This allows them to learn higher order correlations between the signal samples as features that are rich in information and are highly discriminative.

The process of IoT authentication involves transmitting a preamble composed of multiple symbols. Each of these symbols spans a fixed known time length and a fixed set of frequencies. Our system detects these preamble symbols and passes them as inputs to our LSTM classifier. Every sample of the preamble has an in-phase ($\mathbf{I}$) component and a quadrature phase component ($\mathbf{Q}$). We concatenate the two components as input to each LSTM unit. If we send data from n_c time samples as input to an LSTM unit, then the input length of each unit becomes $2n_c$. We denote by N_p the number of LSTM units in a single layer of our deep network that is found as

$$N_p = \left\lceil \frac{\text{size of data}}{n_c} \right\rceil \tag{5}$$

We always take the output from the final LSTM unit since this contains the consolidated information of the whole network. We then feed this output to a fully-connected layer. This layer provides a good representation of the input signal for classification using a softmax layer at the end. It is important to note that, although we trained our model with a dataset having large variability, the variations are not exhaustive. To avoid over-fitting of the model onto the training set, we regularize the optimization by inserting dropout [19] layers post the fully connected layer. As a rule of thumb, we maintain a dropout rate of 50% throughout.

B. Performance vs Space-Time Complexity Trade-offs

Selecting the parameters for a suitable classifier network requires us to perform a complexity analysis to understand the trade-off between performance vs model complexity in terms of run-time and storage. In general, longer inputs contain richer information about the transmitter but affects the authentication complexity both in terms of time and memory.

Observe that the LSTM equations in Section III are matrix multiplications passed through a non-linear activation. The dimensions of the weight matrices have to be identical for all the gates as well as for all the units since each of them is just an unrolled version of the same entity. We denote the dimension as $M_w \times N_w$, where N_w and M_w are the dimensions of input and output activations. Since $N_w = 2n_c$ (section IV), the parameter space complexity per LSTM unit, given by $\mathcal{O}(2n_cM_w)$, varies linearly with the symbol length. Due to

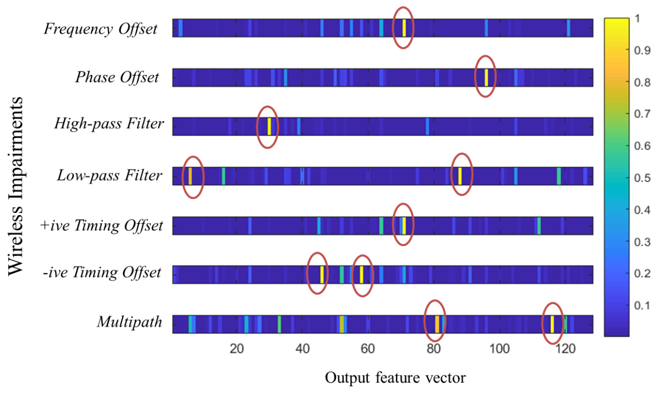

Fig. 4: Visualizing the effect of device imperfections on the feature vector

parameter sharing, this is also the space complexity of an entire layer. The time complexity, on the other hand, is also affected by the number of unrolled steps (N_p) due to recurrence, besides the input and output dimensions. Thus, the total time complexity is the order of $\mathcal{O}(N_p n_c M_w)$. For fixed length time series, the number of time steps and input dimension are related according to equation (5) as $n_c * N_p \propto L_D$, where L_D is the size of the data. Hence the total time complexity turns out to be a constant dependent only on the output dimension M_w and total signal length.

Achieving real-time performance requires us to use inputs of smaller length. However, this may be insufficient to reliably decode the device information thus affecting classification accuracy. Also, longer samples are generally more informative but at the cost of additional storage for extra "learnable" network parameters. To address this conflict, we carefully select the sample lengths to achieve the optimum trade-off between model complexity and classification quality.

C. *Hardware Features uncovered by LSTM*

In this section, we validate our claim that the LSTM networks indeed extract information about the hardware imperfections from the input signal. To do so, we analyze the response of the output features to device imperfections. In particular, we synthetically generate a sequence of chirp signals and distort the resulting signal with various hardware imperfections that include frequency offsets, phase offsets, filters, timing offsets and multipath. We observe the corresponding change in the output features. This leads to a visualization as shown in Fig. 4 where each row represents the change in the 128 length feature vector due to a wireless imperfection. We see that specific features, denoted by heightened response encircled in the figure, encode specific hardware characteristics. To confirm the consistency, we repeat these experiments with varying degrees of each imperfection. We observe that the peaks always occur at the same positions. This strongly corroborates our hypothesis that LSTM can leverage signal correlations to distinguish transmitters based on their hardware defects.

D. *Resilience to Multipath and High-Power Adversaries*

Resilience to Multipath: Deep networks learn a hierarchy of representations with increasing level of abstraction via an end-to-end single algorithm. Each stage can be considered to be a trainable feature transform. So, when the network is trained on data affected by multipath, it is able to leverage multipath as a feature rather than a distortion. This is justified by the peaks in Fig 4 corresponding to the seventh row. This suggests that certain features in the output indeed unearth the multipath affecting the transmitted signal. It is, however, important to note here that the results correspond to a static case when the entire system is spatially fixed. In such cases, new transmissions will be affected by similar multipath characteristics as those during training. However, if the system is geographically dynamic, one would expect the LSTM to confuse a valid device with an adversary due to unseen multipath effects. To mitigate this problem, we should train with data exhibiting greater variability by repeating the experiments for different relative spatial placements of the IoT devices to covering as many distinct scenarios as possible.

High-Power Adversaries: High-power adversaries are difficult to cope with in cryptographic authentication mechanisms. The main caveat for security of low-power IoT devices arises from the fact that most of their adversaries will have much larger compute resources and will be much more accurate. This means that any operation performed on the client device can be easily replicated by these devices. This motivates the need to offload the majority of the security overhead to much more powerful base stations that will be able to differentiate maleficent nodes from the actual client nodes.

Our approach overcomes these two problems in a unique way. First, although a much more powerful adversary can emulate a device, it will introduce its own imperfections that affect the signal it produces. These imperfections are unknown to the adversary since they are visible only at the receiver, e.g., frequency offsets are a function of both transmitter and receiver frequency offsets. This means that the transmitter will not be able to correct for these frequency offsets without access to the base stations. Although magnitude of these offsets might be small (half that of IoT devices), the temporal deviations are significant enough for LSTM to recognize an imposter node from the correct one.

Second, our approach offloads all the computation for security to the core of the network where the nodes are much more powerful than at the edge. Moreover, the training overhead can be lowered even further by performing it in the cloud and passing the trained weights to the edge for classification. This reduces overhead on client nodes and hence removes any advantage from the adversary to exploit its superior compute resources than those of the client node.

V. Experimental Results and Discussion

This section describes the experimental set-up and the experiments we conducted with real data collected from our testbed. These experiments demonstrate the robustness of our approach against signal perturbations that may arise from various physical factors such as random noise and fading in an unrestricted environment.

A. *Experimental Setup*

Our experiments use a testbed of LoRa low-power wireless chipsets as wireless nodes, all transmitting identical signals (i.e., we have one legitimate node and 29 adversaries). Our testbed is composed of 29 Semtech SX1276 chips as LoRa transmitters and a Semtech SX1257 chip as the receiver. In addition, we also employ USRP N210 software radios as adversaries. We transmit the data with spreading factor ten with a sampling rate of 125 KHz. We conduct experiments at various distances and under noisy and multipath-rich settings in a large university campus building. Note that each node is transmitting the same message and hence this is the most adversarial scenario where the receiver has to distinguish signals purely based on device hardware imperfections.

Our approach uses an LSTM unit of length 2048 and with $N_p = 21$ layers, unless specified otherwise. Each signal packet in the data is assigned a label corresponding to the device it is transmitted from. Each of the LoRa devices generate about 80 such instances for each scenario, so as to introduce enough variability in the data. We divide our data into two sets. The training set has 80% of the data and the validation set collects 20% of the data; the data is fed to the LSTM network. We build the model in Keras [20] – a python based library for neural network simulations. We run the model on Tesla P100 GPU accelerators by NVIDIA. Each of these powerful accelerators is powered by the NVIDIA Pascal architecture, designed to boost the throughput. Of the two variants available, we use the one with 16GB GPU memory.

B. Network Complexity Analysis

To understand the complexity of our machine learning algorithm, we test the validation accuracy on standard techniques such as multiclass-SVM basic and multilayer perceptrons (MLP) with varying number of hidden layers. The low performance of these techniques, as shown in table I, motivated us to select a more sophisticated architecture for sequence classification – the LSTM network. Furthermore, we experiment with our LSTM network configurations with varying levels of complexity. We start with a vanilla network, tapping the output at the final time step, and feeding it to the softmax layer for prediction. We denote this as *1-layer* w/o FC architecture. To learn higher order information, we augment the final LSTM unit with a fully-connected layer (denoted as *1-layer*). We further increase the complexity by adding additional LSTM layers with the same number of time steps as in the previous layer. These are represented by *2-layer* and *3-layer* architecture respectively. In these multi-layer LSTM instances, the inputs to each time step of a new layer are the output activations from the corresponding time step in the previous layer. The resulting classification accuracy on validation dataset is tabulated against the model complexity as shown in table I along with the baseline results.

Results: The performance of MLPs saturate at around 60% classification accuracy for *3 fully connected* layers, beyond which the models overfit. For LSTMs, we observe that the validation accuracy, at first, increases (performance improves) as we increase the complexity of the LSTM network. Then, a maxima is achieved, after which the accuracy decreases (performance worsens) as we add additional LSTM layers. This shows that overly complex models can overfit the data hence highlighting the need for careful calibration of hyper-parameters. According to our findings, we obtain the highest accuracy for *Type* 2 architecture with 2 LSTM layers with 21 time-steps each, followed by a fully-connected layer of length 128. The classification accuracy of $\mathbf{99.58}\%$ in the LOS scenario is the probability of a packet from a particular device being accurately classified as from that device. Moreover, all the misclassified packets correspond to likelihood values in the output of the softmax layer that are extremely small (10^{-5}); we can leverage these small values to drop all such packets in a real world scenario. Further, the use of high-power USRP N210 radios as adversaries replaying signals reduces accuracy only slightly to $\mathbf{98.6}\%$.

	Model	*Classification Accuracy*(%)
Previous Work	SVM[16]	28.69
	MLP[21]	59.51
LSTM Model	*1-layer* LSTM w/o FC	97.36
	1-layer LSTM	97.45
	***2-layer* LSTM**	**99.58**
	3-layer LSTM	96.60

TABLE I: Classification Accuracy (over 30 devices) for various models (2-layer LSTM is the best classifier)

C. Resilience to noise and channel impairments

In this section, we investigate the sensitivity of our proposed framework channel impairments such as noise, signal attenuation, and multipath.

Firstly, we introduce varying degrees of additive white Gaussian noise (AWGN) while placing the transmitter near the receiver within the line of sight. This ensures that we can reliably isolate the system from other impairments for analysis purposes.

Effect of Noise: Fig 5 demonstrates the decay of the classification accuracy on the validation set with increasing noise power. We observe that LSTM network is resilient to external noise. Even for PSNR values as low as -5dB, the accuracy falls only a little below 90%.

Next, we investigate the effect of signal attenuation and multipath. We place the receiver at varying relative positions with respect to the transmitters and observe its effect on the validation accuracy in a 1575 square meter area of a large university building. We test for both LOS and non-LOS scenarios. The results are summarized in table II where we report accuracy for varying complexity in different situations.
Effect of Multipath and Attenuation: We can see from the above table II that increasing the distance, increases the attenuations, making it difficult to distinguish between transmitters. The best accuracy that we obtain drops down to about 91%. As we move the transmitter out of the LOS of the receiver, the receiver has to even address inter-symbol interference due to multipath that renders the classification task much more challenging. However, as detailed in Section IV-D above, multipath introduces new features in the wireless channel unique to the transmitter's environment that assists the classifier. As a result, our system's accuracy remains

sufficiently high to recognize attacks from adversaries with high probability, though we require a more complex model to achieve it. In this case, we obtain a best classification accuracy of 88%. Due to lack of publicly available large datasets, we are restricted to certain degrees of complexity, beyond which the LSTM model tends to overfit.

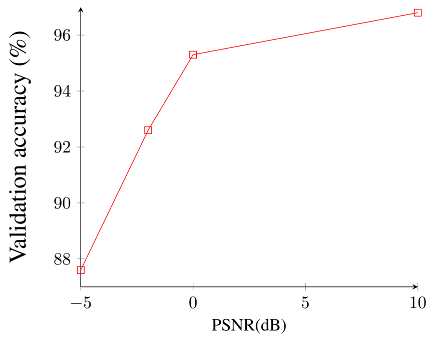

Fig. 5: Validation Accuracy vs Peak Signal-to-Noise Ratio

.	*Accuracy (%)*		
Placement	*1-layer* LSTM	*2-later* LSTM	*3-layer* LSTM
LOS (< 1 m)	97.45	**99.58**	96.60
LOS (>15 m)	89	**90.74**	88.42
NLOS	84.48	85.28	**88.10**

TABLE II: Classification accuracy of models across scenarios

D. Real-time analysis

Next, we test the run time for our most complex $Type$ 3 architecture with $L_D = 21,000$ samples and output dimension, $N_w = 1024$. The network was able to authenticate the device ID within 0.05 seconds that provides strong support for our claim about real time operations.

E. Diversity analysis

We performed our experiments with a sample length of 1024 I/Q samples that allowed us to optimize the length of each LSTM unit to fit exactly one preamble symbol. We find that the quality of the classification performance is not affected significantly (within 2% of the best performance) by our choice of n_c (in the range of 512 to 2048). This demonstrates that the LSTM network can find correlation across units to discern unique features from the input data without needing precise knowledge of the correct input size to each LSTM unit, as long as the whole data is input in the same order across a layer. This shows the ability of LSTM networks for PHY-protocol-independent identification of IoT devices.

VI. Conclusion and Future Work

We present a novel IoT authentication solution that leverages deep neural networks to learn wireless hardware imperfections. We present an LSTM classifier that exploits temporal correlation between the I/Q streams of wireless signals to uniquely identify low-power transmitters amidst high-power adversaries, regardless of the data they transmit or the underlying PHY-layer. We evaluate our system with state-of-the-art LoRa transmitters and much higher power software radio adversaries. Our classifier is resilient to challenges stemming from noise, multi-path, and signal attenuation. In the future, we will extend the approach to broader classes of cyber-attacks beyond device identification. We would also like to study how transmitters can calibrate their signals to amplify their unique features to improve classification accuracy.

References

[1] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645–1660, 2013.

[2] M Reza Rahimi, Jian Ren, Chi Harold Liu, Athanasios V Vasilakos, and Nalini Venkatasubramanian. Mobile cloud computing: A survey, state of art and future directions. *Mobile Networks and Applications*, 19(2):133–143, 2014.

[3] Jie Xiong and Kyle Jamieson. Securearray: Improving wifi security with fine-grained physical-layer information. In *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking*, pages 441–452. ACM, 2013.

[4] Stephanie Gil, Swarun Kumar, Mark Mazumder, Dina Katabi, and Daniela Rus. Guaranteeing spoof-resilient multi-robot networks. *Autonomous Robots*, 41(6):1383–1400, 2017.

[5] Adam C Polak, Sepideh Dolatshahi, and Dennis L Goeckel. Identifying wireless users via transmitter imperfections. *IEEE Journal on Selected Areas in Communications*, 29(7):1469–1479, 2011.

[6] Vladimir Brik, Suman Banerjee, Marco Gruteser, and Sangho Oh. Wireless device identification with radiometric signatures. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, pages 116–127. ACM, 2008.

[7] Andrea Goldsmith. *Wireless Communications*. Cambridge University Press, 2005.

[8] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 1999.

[9] Jürgen Schmidhuber. Deep learning in Neural Networks: An overview. *Neural Networks*, 61:85–117, 2015.

[10] Lyes Touati, Yacine Challal, and Abdelmadjid Bouabdallah. C-CP-ABE: Cooperative ciphertext policy attribute-based encryption for the internet of things. In *International Conference on Advanced Networking Distributed Systems and Applications (INDS)*, pages 64–69. IEEE, 2014.

[11] Fang Liu, Jose Manny Rivera, and Xiuzhen Cheng. Location-aware key establishment in wireless sensor networks. In *Proceedings of International Conference on Wireless Communications and Mobile Computing*, pages 21–26. ACM, 2006.

[12] Mustafa Nawari, Hazim Ahmed, Aisha Hamid, and Mohamed Elkhidir. FPGA-based implementation of elliptic curve cryptography. In *World Symposium on Computer Networks and Information Security (WSCNIS)*, pages 1–8. IEEE, 2015.

[13] Xuanxia Yao, Zhi Chen, and Ye Tian. A lightweight attribute-based encryption scheme for the internet of things. *Future Generation Computer Systems*, 49:104–112, 2015.

[14] Christoph Neumann, Olivier Heen, and Stéphane Onno. An empirical study of passive 802.11 device fingerprinting. In *32nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pages 593–602. IEEE, 2012.

[15] L Yu Paul and Brian M Sadler. MIMO authentication via deliberate fingerprinting at the physical layer. *IEEE Transactions on Information Forensics and Security*, 6(3):606–615, 2011.

[16] Liang Xiao, Larry Greenstein, Narayan Mandayam, and Wade Trappe. A physical-layer technique to enhance authentication for mobile terminals. In *IEEE International Conference on Communications*, pages 1520–1524. IEEE, 2008.

[17] LR Medsker and LC Jain. Recurrent neural networks. *Design and Applications*, 5, 2001.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[19] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[20] François Chollet et al. Keras, 2015.

[21] Abdul Ahad, Ahsan Fayyaz, and Tariq Mehmood. Speech recognition using multilayer perceptron. In *IEEE Students Conference*, volume 1, pages 103–109. IEEE, 2002.