Pilot Contamination Attack Detection in 5G Massive MIMO Systems Using Generative Adversarial Networks

Fatemeh Banaeizadeh*, Michel Barbeau*, Joaquin Garcia-Alfaro†, Evangelos Kranakis*, Tao Wan*,‡

* School of Computer Science, Carleton University, K1S 5B6, Ottawa, Ontario, Canada
Email: FatemehBanaeizadeh@cmail.carleton.ca, {barbeau,kranakis}@scs.carleton.ca

† Institut Polytechnique de Paris, Telecom SudParis, 91120 Palaiseau, France
Email: joaquin.garcia_alfaro@telecom-sudparis.eu

‡ CableLabs, 858 Coal Creek Circle Louisville, Colorado, USA
Email: t.wan@cablelabs.com

***Abstract*—Reliable and high throughput communication in Massive Multiple-Input Multiple-Output (MIMO) systems strongly depends on accurate channel estimation at the Base Station (BS). However, the channel estimation process in massive MIMO systems is vulnerable to pilot contamination attacks, which not only degrade the efficiency of channel estimation, but also increase the probability of information leakage. In this paper, we propose a defence mechanism against pilot contamination attacks using a deep-learning model, namely Generative Adversarial Networks (GAN), to detect invalid uplink connections at the BS. Training of the models is performed via legitimate data, which consists of received signals from valid users and real channel matrices. The simulation results show that the proposed method is able to detect the pilot contamination attack with 98% accuracy in the best scenario.**

Keywords: Massive MIMO, Pilot Contamination Attack, Generative Adversarial Network, Network Security.

I. Introduction

Fifth generation cellular networks (5G) offer high data rates, low latency, reliability and security through key technologies such as massive MIMO, millimeter wave, and software-defined wireless networks. Among all-aforementioned technologies, massive MIMO is one of the most promising innovations that has attracted the attention of the research community in recent years. This technology is an evolved version of conventional MIMO used in 4G. It is able to address the shortcomings of conventional MIMO such as high computational power, high cost, and lack of scalability. It can also achieve greater spectral efficiency than 4G [1], [2].

In massive MIMO systems, the Base Station (BS) in each cell is equipped with a few hundreds of antennae serving single-antenna users at the same time and frequency domains. The communication channel between a BS and a user comprises an uplink channel, from user to BS, and a downlink channel, from BS to user. The uplink channel is the focus of this research. The uplink channel is used for sending training sequences (pilots) and data. Users first send orthogonal pilots to the BS on the uplink. Then, the BS utilizes the received orthogonal pilot sequences to estimate channels and get the Channel State Information (CSI). Accurate channel estimation has considerable impact not only on achieving reliable downlink communications, but also on improving user throughput. However, impairments such as interference, propagation channel effects, and pilot contamination reduce the accuracy of channel estimation, thus degrading the performance of massive MIMO systems [3]. In this paper, the main focus is on the pilot contamination issue, a major impairment to channel estimation. We propose a defence mechanism, using a deep-learning method called GAN, to detect pilot contamination originating from the impersonation of valid pilots by an adversary.

In general, the CSI plays a vital role in the channel estimation process. The BS can obtain this information through orthogonal and unique pilots, which are sent by legitimate users. CSI acquisition can be based on a reciprocity concept in Time-Division Duplex (TDD) based systems or feedback in Frequency-Division Duplex (FDD) based systems. Advantages, such as less overhead and one-way channel estimation, have made TDD systems more popular than FDD systems. However, the TDD systems suffer from limitation of coherence interval, especially in networks with high mobility. This limitation prevents the assignment of unique orthogonal pilots to users in a dense cell, or in a multi-cell network, which increases the reuse of orthogonal pilots. It also causes pilot contamination problems [3]. From a security standpoint, adversaries can impersonate valid orthogonal pilots to intentionally cause the pilot contamination phenomenon, e.g., to compromise channel estimation in the BS and change the beamforming direction to facilitate some other attacks [4].

Since pilot contamination attacks significantly reduce network throughput and threaten data confidentiality, we are motivated to find a mitigation strategy to this problem. In recent years, the use of neural network techniques in wireless communications has shown great progress in solving problems like channel estimation, signal detection (i.e., estimation of transmitted data symbols by users in the BS) and attack detection [5], [6]. In the security area, the use of neural networks allows the BS to learn the features of normal data during the training phase, hence, enabling the BS to identify and discard abnormal data during the remaining phases.

In this paper, we propose a method to detect pilot contamination attacks using a GAN-based approach. Our defence mechanism allows the BS to detect impersonation of legitimate users in the network. GAN is a deep-learning technique which combines two types of neural networks simultaneously [7]: a generative network, named *Generator* (G) and a discriminative network named *Discriminator* (D). Traditionally, G produces invalid (i.e., fake) data, which resembles legitimate data. D is responsible for detecting the invalid data. The two networks (both G and D) play a minimax game during the training phase, until they reach an optimal phase (e.g., Nash equilibrium). Therefore, G and D act as adversaries during the training phase (while improving their skills), until the equilibrium is reached between them [8].

The main contributions of this paper are as follows: (1) we propose a security framework in the BS to detect pilot contamination attacks in massive MIMO systems without the need for complex mathematical operations; (2) the BS does not need to have a priori knowledge of either the user's or the adversary's channel parameters, because it learns the distribution of the channels during the training phase; and (3) our simulation results show a high detection accuracy, hence validating the feasibility of our approach to detecting the malicious impersonation of pilot sequences.

The remaining sections are organized as follows. Section II presents the literature review. Sections III and IV provide the system and threat model, respectively. Section V describes the architecture of our proposed GAN. Section VI provides the experimental work. Section VII concludes the paper.

II. Related Work

In this section, we discuss existing techniques for mitigating pilot contamination attacks in massive MIMO systems. In [9], the authors present a low-complexity detector of pilot contamination attack for single-cell networks. The method assumes that the BS has a priori knowledge of the adversary's channel parameters, which may not be practical in real settings. The use of a Phase-Shift Keying (PSK) based detection method is proposed in [10]. The idea is to use a set of random PSK symbols as stochastic pilots, which are sent by the users during the channel training phase. The BS uses the stochastic pilots to calculate their phase difference and detect anomalies, i.e., to detect the presence of adversaries.

The effect of pilot contamination attacks on channel estimation and downlink transmission rates is evaluated in [11]. The authors present simulation results to show how an adversary can reduce the downlink throughput by more than 50%, especially when the adversary is located much closer to the BS than the legitimate users.

All the aforementioned methods are based on a statistical analysis of the received signals, whose computational complexity increases as the number of BS antennae grows. In contrast, the use of learning approaches like GAN can provide low-overhead defence strategies for attack detection. In [12], the authors present a method for detecting pilot contamination attacks in 5G grant-free IoT networks. A GAN approach is presented to generate the synthetic samples used during the training of a neural network. The resulting approach uses characteristics of Channel Virtual Representation (CVR) like Angle-of-Arrival (AoA) to detect the dissimilarity between signals from the adversary and the legitimate users.

In our work, we evaluate the use of a GAN-based methodology to construct a pilot contamination attack detector. The generator acts as an estimator of valid channels from noisy signals, while the discriminator acts as the pilot contamination attack detector. We validate our approach via simulation. The results show the feasibility and effectiveness of the approach.

III. System Model

We consider a single-cell massive MIMO system equipped with K single-antenna users and a M-antenna BS. The channel vector ($h \in \mathbb{C}^{M\times 1}$) between each user and the BS antennae is modeled by a correlated-Rayleigh fading channel [13], [14]:

$$h = R_r^{1/2} h_{iid} \tag{1}$$

where $h_{iid} \in \mathbb{C}^{M\times 1}$ is an independent realisation of a Rayleigh fading channel with Gaussian distribution, and $R \in \mathbb{C}^{M\times M}$ is the correlation matrix in the receiver side (BS) which shows correlation between antenna elements in the BS. In fact, R is a matrix that is complex conjugate symmetric. It has a Toeplitz structure. The correlation coefficient of each pair of antennae is expressed by the following equation:

$$\rho_{ij}^r = \frac{E[h_{ki}h_{kj}^*] - E[h_{ki}]E[h_{kj}^*]}{\sqrt{Var[h_{ki}]Var[h_{kj}]}} \tag{2}$$

where ρ_{ij}^r is an element of R_r which shows the correlation coefficient between antennae i and j in the BS, $h_{ki} \in \mathbb{C}^{M\times 1}$, $h_{kj} \in \mathbb{C}^{M\times 1}$ are the channel of user k with antennae i and j in the BS respectively, $E[h_{ki}h_{kj}^*]$, $Var[h_{ki}]$ and h_{kj}^* are the expected value, the variance and the conjugate transpose of channel of user K and antenna j in the BS, respectively.

In a normal situation, the received signal ($Y \in \mathbb{C}^{M\times\tau}$) at the BS from all the legitimate users is expressed as follows:

$$Y = \sum_{i=1}^{K} \sqrt{P_i} h_i \phi_i + n \tag{3}$$

where $\sqrt{P_i}$ is the uplink transmit power of the ith user, $h_i \in \mathbb{C}^{M\times 1}$ is the uplink channel of the ith user, $\phi_i \in \mathbb{R}^{1\times\tau}$ is the orthogonal pilot sequence of ith user with length τ. We can generate K orthogonal pilot sequences with length τ via a *Hadamard* matrix [15] in Matlab™, $n \in \mathbb{C}^{M\times\tau}$ is the additive white Gaussian noise matrix at the BS.

The BS uses the received signal and known orthogonal pilot sequences to estimate user channels and adopt precoding and downlink beamforming.

IV. Threat Model

The threat model used in our work comes from [16], [17]. We assume the presence of an active adversary. First, the adversary eavesdrops signals between the BS and a legitimate user. Later

on, the adversary impersonates the legitimate user with respect to the BS. We assume that the eavesdropping phase can affect both the uplink and downlink signals.

More precisely, and during the eavesdropping phase, the adversary identifies pilots being used by the legitimate user. Then, during the impersonation phase, the adversary synchronizes with the victim (i.e., the legitimate user) and impersonates the user with synthetic versions of the pilots, in order to compromise the orthogonality and accuracy properties of the channel estimation between the BS and the victim. If the attack is successful, the received signal at the BS is combined with the signal sent by the attacker and changed as follows:

$$Y = \sum_{i=1}^{K} \sqrt{P_i} h_i \phi_i + \sqrt{P_A} h_A \phi_A + n \tag{4}$$

where $\sqrt{P_A}$ is the uplink transmit power of adversary, ($P_A = P_i = 1$), and $h_A \in \mathbb{C}^{M \times 1}$ is the channel matrix of the adversary. Notice that if the BS is not able to detect the attack, the downlink data is beamformed towards the adversary and the victim (i.e., both the adversary and the victim will receive the downlink signals after the attack). Hence, a successful attack will put at risk the reliability and confidentiality of the communications between the BS and the victim. Notice as well that the pilot signal that the adversary sends to the BS is the same as the one used by the victim. Therefore, it violates the orthogonality rule and increases interference (due to the non-orthogonality of the two pilot sequences).

Next, we propose a defence method that allows the BS to detect pilot contamination attacks, after a period of training. The approach assumes the use of Generative Adversarial Networks (GAN), directly executed by the BS. The discriminator (D) used in our GAN architecture acts under the role of a BS. The goal is to learn the distribution and features of legitimate data during the training phase. Then, during the testing phase, D is able to detect invalid data, i.e., channels that present anomalies with respect to those legitimate channels used during the training phase, thus discarded as pilot contamination attacks.

V. GAN-BASED PILOT CONTAMINATION DETECTION

GAN is a deep learning technique. It consists of two independent neural networks, namely the Discriminator (D) and the Generator (G) [7]. They play a minimax game until reaching an optimal status, when D is not able to distinguish the real data from the synthetic data. In other words, G receives random noise (z) from a distribution P_z and generates a synthetic sample $G(z)$ with distribution P_g. G learns the distribution of the real data (P_x) during the training, in order to create synthetic data that can be validated by D as real data, i.e., G aims at minimizing the power of D in differentiating real data from synthetic (fake) data. The objective function of G is expressed as follows:

$$\min_G V(D,G) = \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z))] \tag{5}$$

where z is random noise from a normal distribution, $\mathbb{E}_{z \sim p_z}$ is the expected value on synthetic samples, $G(z)$ is the output of G after mapping noise to the real data, and $D(G(z))$ is the output of D for $G(z)$, which shows the probability whether it is synthetic or real data.

Likewise, D tries to maximize the probability of properly differentiating legitimate (real) data from invalid (fake) data, as follows:

$$\max_D V(D,G) = \mathbb{E}_{x \sim p_x}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z))] \tag{6}$$

where x is a sample from legitimate data, $\mathbb{E}_{x \sim p_x}$ is the expected value on legitimate samples, and $D(x)$ is the output of D for x, which comes from legitimate data.

The minimax game between the two networks (D and G) is modeled as follows:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x \sim p_x}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z))] \tag{7}$$

Our proposed GAN architecture is depicted in Fig. 1. The design of D contains four layers. The three first layers consist of *Convolutions*, followed by *Batch Normalization*, *Leaky ReLU*, and *Dropout* operations. The output layer is composed of a linear activation function that is followed by a Mean Square Error (MSE) layer, acting as a loss function. The use of the MSE layer as a loss function during the training phase makes the optimization process much easier [18]. Its aim is twofold. First, it helps at having a more stable training process, since D learns to reject the outputs that G stabilizes on, forcing G to try something new at each iteration. Second, it forces G to have a better channel estimation process (i.e., a more realistic $\hat{H}$) from the noisy received signals (Y), based on the difference from the decision boundary (which is derived from the linear activation function). In fact, G is forced to move near legitimate channel data. The difference to the decision boundary is calculated as a loss function by D and is back-propagated to G. The inputs for D are the legitimate channel (H) and the channel generated by G ($\hat{H}$).

Fig. 1 also depicts the design of G. It is constructed as a channel estimator, with encoding and decoding blocks. It generates $\hat{H}$ from Y. In the image processing area, similar designs are used by autoencoders to reconstruct real images from noisy (high quality) images. In our case, since the signals received by the BS are combined with noise, we process the received signals as if they were noisy images, hence using an autoencoder architecture for G. This way, G can estimate and generate channel matrices from noisy signals. Based on this idea, G is composed of four encoding blocks and four decoding blocks. Each encoding block consists of layers with *Convolutions*, followed by *Batch Normalization* and *ReLU* operations. The output of the last encoding block is sent to the first decoding block. Each decoding block is composed of *Transposed Convolutions*, followed by *Batch Normalization* and *ReLU* operations. The last layer of the last decoder uses a $\tanh$ activation function, to scale up the output values in the range -1 and 1.

Adam optimization with learning rate equal to 2×10^{-4} is used to update the weights of D and G during the training via *Back-propagation*. The initial weight is set to 0.02. The size of the filters in both D and G is equal to (4,4). The number of filters in the layers of D and G are (64,128,256,1) and (128,128,64,32,32,64,128,2), respectively.

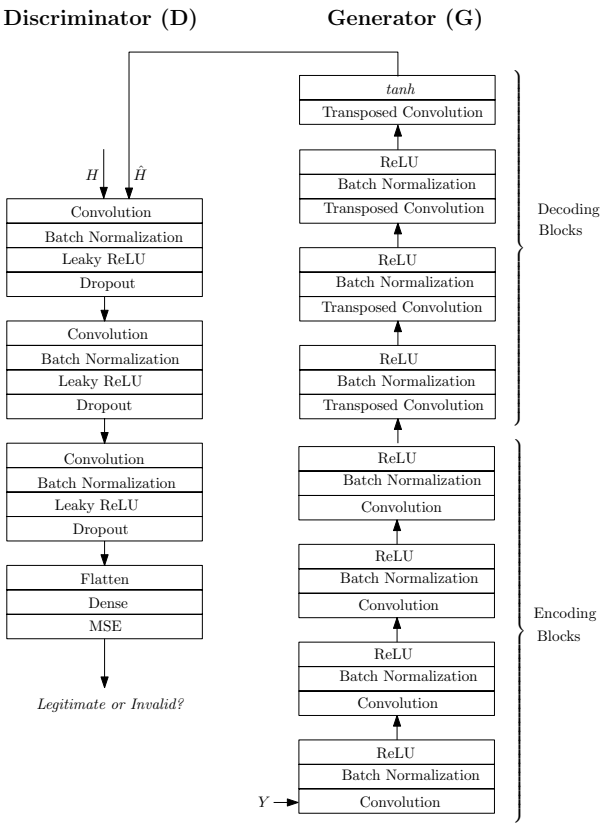


Fig. 1. Architecture of our proposed GAN

According to the defined inputs and outputs in our proposed GAN-based attack detection method, the minimax loss function of our GAN design is changed as follows:

$$\min_{G}\max_{D} V(D,G) = \mathbb{E}_{H\sim p_H}[\log D(H)] + \mathbb{E}_{Y\sim p_Y}[\log(1 - D(G(Y,\phi))] \quad (8)$$

where H is the legitimate channel matrix and $G(Y,\phi)$ is the channel generated by G (i.e., $\hat{H}$).

VI. Experimental Work

We evaluate our work via simulation. We use Matlab™, to simulate a single-cell network and produce the training data. Then, we experiment with the training and detection processes using Python, under the Google Colaboratory (Colab) environment [19]. The code and results of our simulations are available online, in a companion github repository [20]. Next, we provide some more details and results about our experimental work.

TABLE I
Simulation parameters in Matlab's 5G Toolbox™

Parameter	Value
Number of antenna BS	64
Number of users	8
Pilot length	8
Channel model	Correlated Rayleigh Fading
Training dataset size	4000
Testing dataset size	1000
Size of received signal (Y)	(64,8,2)
Size of channel matrix (H)	(64,8,2)

A. Training Phase

The training process is conducted in Python, using Google's Colab. Algorithm 1 summarizes the main steps of the process. To train the model, we simulate a single-cell network using Matlab™ with the characteristics indicated in Table I. The cell is equipped with a 64-antenna BS and 8 legitimate users. Orthogonal pilot sequences with length $\tau = 8$ are generated for each user, using a *Hadamard* matrix. The training data only contains legitimate data. Such a training dataset consists of 4000 received signals (Y), used as the input of G, as well as the corresponding dataset with 4000 legitimate channels (H) for D. Each received signal (Y) and each legitimate channel matrix (H) corresponds to a three dimensional matrix of size (64,8,2). The third dimension shows the values of real and imaginary parts of the received signal (channel) matrix.

B. Testing Phase

After the training phase, the weights of the resulting trained G and D networks are stored and evaluated in a secondary phase, with regard to a second, different, dataset (cf. Table I). The testing phase is also conducted in Python, under the same computational environment (i.e., Google's Colab).

The testing phase validates the accuracy of the trained GAN in detecting the pilot contamination attacks. It selects a given legitimate user (i.e., User 1) as the victim. Then, the adversary impersonates the victim to communicate with the BS. During

Algorithm 1 *Training Process*

Collect the *training dataset* and add one size
$N = BatchSize$
$NumBatchSize = TrainingDataSize/N$
for $i \leftarrow 1$ to $Epoch$ **do**
  for $j \leftarrow 1$ to $NumBatchSize$ **do**
    Select N invalid samples ($\hat{H}$) from G
    Select M legitimate samples (H) from the dataset
    Label legitimate samples as 1 and invalid samples as 0
    Train D to differentiate legitimate from invalid samples
    Calculate loss and accuracy of D and update its weights

$$\nabla_{P_H} \frac{1}{M} \sum_{k=1}^{M} [\log D(H_K)] + [\log(1 - D(G(Y_K, \phi_K))]$$

    Select N samples from the received signals (Y)
    Estimate the channel ($\hat{H}$)
    Calculate loss of G and update its weights

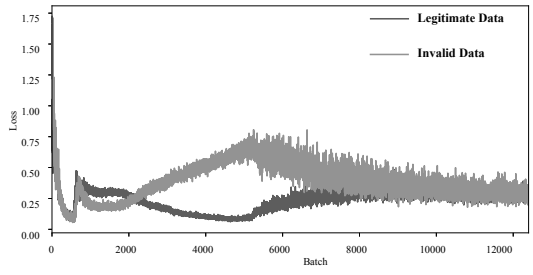
$$\nabla_{P_Y} \frac{1}{N} \sum_{k=1}^{N} \log(1 - D(G(Y_K, \phi_K))$$

the testing phase, the output of G, together with legitimate data, is fed to D. The output of D is distributed in the range $(-1, 1)$. By using the linear activation function in the last layer of D, we can determine a decision boundary (i.e., threshold) to distinguish legitimate data from invalid data.

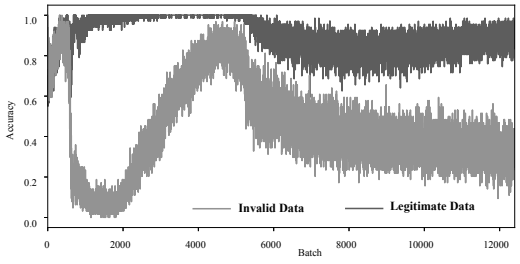
The outputs of D for legitimate and invalid data shows that, approximately, most invalid data is distributed in the range $(-0.001, 0.001)$, while the legitimate data are distributed out of this range. In order to select the appropriate threshold, we calculate the absolute value of the outputs of D, to differentiate legitimate data from invalid data, based on their magnitude (regardless of the sign). The outputs of D are tested with different thresholds. Values higher than the threshold are counted as legitimate data. Values lower than the threshold are counted as invalid data.

C. Results

Fig. 2 shows the results of simulations composed of 63 batches per epoch, up to 200 epochs (i.e., 12600 batches). Fig. 2(a) shows D vs. G loss during the training process. Similarly, Fig. 2(b) shows the accuracy of D during the same process. On average, a training based on 12600 batches takes about 15 minutes when using GPU hardware acceleration (with RAM consumption of about 1 GB and storage consumption of about 30 GB). Table II shows the runtime execution performance for other three representative batch sizes (1024, 4096, and 8192). From Fig. 2, we see that both D and G reach an equilibrium after 100 iterations (i.e., more than 6000 batches), in which G has already approximated the distribution of legitimate channels, i.e., G is able to estimate the channel matrix between the BS and users from the received signal with less error, while D is not able to distinguish it from the legitimate data with high accuracy. From Table II we see that, with GPU hardware acceleration, the time consumption to reach the aforementioned equilibrium is about 10 minutes, hence validating the feasibility of our proposal in terms of time constraints for a single adversary targeting a single pilot contamination attack. Notice that collusion of adversaries, in order to target several pilots per BS, is also feasible via learning parallelization strategies [21].



(a)



(b)

Fig. 2. Training. Light gray represents invalid data. Dark gray represents legitimate data. (a) D vs. G loss during the training process. (b) Accuracy of D during the training process.

TABLE II
RUNTIME PERFORMANCE

Number of Epochs	Max. number of Batches	Hardware Acceleration		
		GPU	*TPU*	*None*
16	1024	120 s	1000 s	1200 s
64	4096	300 s	5000 s	5100 s
128	8192	600 s	9600 s	9700 s
200	12800	900 s	15372 s	16500 s

Fig. 3 shows the accuracy of our attack detection method, through Receiver Operating Characteristic (ROC) curves for five different threshold values. Additional information is provided in Table III, with some other metrics, all reporting the proportion of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) in metrics such as Accuracy $((\text{TP} + \text{TN})/(\text{TP} + \text{TN} + \text{FP} + \text{FN}))$, Precision $(\text{TP}/(\text{TP} + \text{FP}))$, Recall $(\text{TP}/(\text{TP} + \text{FN}))$, and F_1 Scoring $\frac{\text{TP}}{\text{TP}+\frac{1}{2}\cdot(\text{FP}+\text{FN})}$. The highest accuracy is obtained with a threshold of 0.001, reaching about 98% (i.e., TP= 1000, FP= 0, FN= 47, and TN= 953).

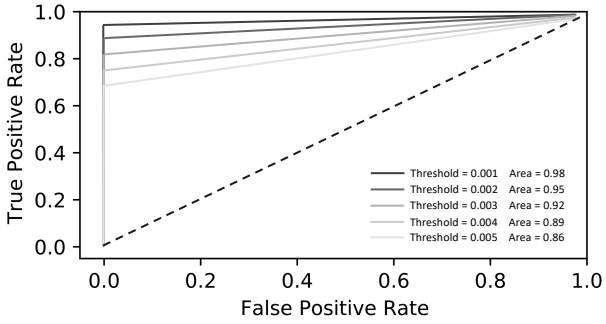


Fig. 3. Receiver Operating Characteristic (ROC) curves

TABLE III
DETECTION PERFORMANCE W.R.T. THRESHOLD VALUES

Threshold	Accuracy	Precision	Recall	F_1 Scoring
0.001	98%	100%	95.5%	97.5%
0.002	95%	100%	89.8%	94.6%
0.003	92%	100%	84.0%	91.0%
0.004	89%	100%	77.0%	87.5%
0.005	86%	100%	72.4%	83.9%

D. Discussion and Future work

Our work shows the initial steps and results as well as the role of adversarial models in the detection of pilot contamination attacks in massive MIMO systems. Although the proposed GAN-based security framework shows a high accuracy in detecting pilot contamination attacks, it might also encounter major challenges in dense single-cell networks (i.e., those that do not use unique orthogonal pilot sequences for all users or in multi-cell networks when the same set of pilots is used in all cells). In such networks, the adversary is not the only underlying cause of pilot contamination. Legitimate users in neighbor cells may also reuse orthogonal pilots, resulting in signal interference to the BS.

The aforementioned issues give rise to some research questions for future work that are as follows: (1) How to distinguish the received interference from a legitimate user and the received interference from an adversary against the BS? (2) In the proposed method, we consider the worst situation in which both a user and an adversary send pilot sequences simultaneously; but, if the adversary is located close to the location of a legitimate user and sends the pilot while the legitimate user does not send the pilot, how can this be detected by the BS? (3) After detecting the attack, how can we avoid that the downlink signals sent to the legitimate users are intercepted by the adversary?

Future experimental work shall also enhance the datasets, e.g., by including as well the Angle-of-Arrival (AoA), the Angle-of-Departure (AoD) and the received signal strength, w.r.t. the location of legitimate users and the BS. This will help the BS to take into account further characteristics of legitimate signals vs. invalid signals. It will also enable the BS to correlate the received signals of mobile users at different time intervals, e.g., base on their speed and location.

VII. Conclusion

Pilot contamination attacks against massive MIMO systems can increase the probability of information leakage. Utilization of learning approaches, such as neural networks, converts the BS to a powerful classifier capable of detecting the presence of adversaries in the network. In this paper, we proposed a GAN-based approach that enables the BS to detect the non-legitimate use of pilot sequences in single-cell networks. We empirically validated, via simulation, the efficiency of the proposed defence strategy in distinguishing legitimate signals from invalid ones. The method increases the robustness of the BS against pilot contamination attacks. We hope our work, albeit with limitations, can encourage more research into the use of machine leaning in 5G and future networks to improve network performance.

Acknowledgements — Research supported in part by NSERC (Natural Sciences and Engineering Research Council) of Canada and MITACS (Mathematics of Information Technology and Complex Systems) grants.

References

[1] T. L. Marzetta, "Massive MIMO: An Introduction," *Bell Labs Technical Journal*, vol. 20, pp. 11 – 22, 2015.

[2] L. Lu, G. Y. Li, A. L. Swindlehurst, A. Ashikhmin, and R. Zhang, "An Overview of Massive MIMO: Benefits and Challenges," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 5, pp. 742 – 758, 2014.

[3] O. Elijah, C. Y. Leow, T. A. Rahman, S. Nunoo, and S. Z. Iliya, "A Comprehensive Survey of Pilot Contamination in Massive MIMO—5G System," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 2, pp. 905 – 923, 2016.

[4] X. Zhou, B. Maham, and A. Hjorungnes, "Pilot contamination for active eavesdropping," *IEEE Transactions on Wireless Communications*, vol. 11, no. 3, pp. 903–907, 2012.

[5] T. Erpek, T. J. O'Shea, Y. E. Sagduyu, Y. Shi, and T. C. Clancy, "Deep Learning for Wireless Communications," *arXiv:2005.06068*, pp. 1–33, 2020.

[6] M. B. Mashhadi and D. Gunduz, "Deep Learning for Massive MIMO Channel State Acquisition and Feedback," *arXiv preprint arXiv:2002.06945*, 2020.

[7] Z. Wang, Q. She, and T. E. Ward, "Generative adversarial networks in computer vision: A survey and taxonomy," *arXiv preprint arXiv:1906.01529*, pp. 1–16, 2019.

[8] C. C. Aggarwal *et al.*, "Neural networks and deep learning," *Springer*, vol. 10, pp. 978–3, 2018.

[9] M. Hassan, A. Ahmed, and M. Zia, "Detection of Pilot Contamination Attack in Massive MIMO System," *52nd Asilomar Conference on Signals, Systems, and Computers*, pp. 1674–1678, 2018.

[10] D. Kapetanović, G. Zheng, K.-K. Wong, and B. Ottersten, "Detection of Pilot Contamination Attack Using Random Training and Massive MIMO," *24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 13–18, 2013.

[11] B. Akgun, M. Krunz, and O. O. Koyluoglu, "Pilot Contamination Attacks in Massive MIMO Systems," *IEEE Conference on Communications and Network Security (CNS)*, 2017.

[12] B. Akgun, M. Krunz, and O. O. Koyluoglu, "Pilot Contamination Attack Detection for 5G MmWave Grant-Free IoT Networks," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 658 – 670, 2021.

[13] T. Brown, E. D. Carvalho, and P. Kyritsi, *Practical Guide to the MIMO Radio Channel with Matlab Examples*. John Wiley and Sons Ltd, 2012.

[14] E. Björnson, J. Hoydis, and L. Sanguinetti, *Massive MIMO Networks: Spectral, Energy, and Hardware Efficiency*, vol. 11(3-4). Now Publishers Inc. Hanover, MA, USA, 2017.

[15] K. J. Horadam, *Hadamard matrices and their applications*. Princeton university press, 2012.

[16] L. Sun and Q. Du, "Physical Layer Security with Its Applications in 5G Networks: A Review," *China Communications*, vol. 14, no. 12, pp. 1–14, 2017.

[17] D. Kapetanovic, G. Zheng, and F. Rusek, "Physical layer security for massive MIMO: An overview on passive eavesdropping and active attacks," *IEEE Communications Magazine*, vol. 53, no. 6, pp. 21 – 27, 2014.

[18] J. Brownlee, *Generative Adversarial Networks with Python, Deep Learning Generative Models for Image Synthesis and Image Translation*. Machine Learning Mastery, 2019.

[19] E. Bisong, "Google colaboratory (colab)," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pp. 59–64, Apress, 2019.

[20] F. Banaeizadeh, M. Barbeau, J. Garcia-Alfaro, E. Kranakis, and T. Wan, "Pilot contamination attack detection in massive mimo using generative adversarial networks [github repository]." http://j.mp/mimoGAN, 2021.

[21] S. Pal, E. Ebrahimi, A. Zulfiqar, Y. Fu, V. Zhang, S. Migacz, D. Nellans, and P. Gupta, "Optimizing multi-GPU parallelization strategies for deep learning training," *IEEE Micro*, vol. 39, no. 5, pp. 91–101, 2019.