Towards Neural Network-based Communication System: Attack and Defense

Zuobin Xiong, Zhipeng Cai, *Senior Member, IEEE,* Chunqiang Hu, *Member, IEEE,* Daniel Takabi, and Wei Li, *Member, IEEE*

Abstract—Recent progress has witnessed the excellent success of neural networks in many emerging applications, such as image recognition, text classification, and speech analysis. In order to achieve secure communication, the utilization of neural networks has been realized yet has not raised sufficient research attention. Additionally, the existing neural network-based communication system falls short due to its critical security flaws. In this paper, we investigate the security vulnerabilities of the existing neural communication system. Based on our analysis, we design two kinds of attack models, including *target man-in-the-middle attack* and *target fraud attack*. After that, to improve the security performance of neural communication systems, we develop a new defense mechanism to facilitate two-way secure communication by separating secret key from plaintext and incorporating defensive loss into the training process. Moreover, we show the effectiveness of our proposed neural communication system via theoretical proof. Finally, we implement comprehensive real data experiments to evaluate the performance of our attack and defense methods from the aspects of classification accuracy, communication efficiency and communication qualify, which confirms the advantages of our proposed neural communication system compared with the state-of-the-art.

Index Terms—Secure Communication, Deep Learning, Adversarial Examples, Generative Adversarial Networks

1 Introduction

Neural networks are undergoing an accelerated advancement and have been successfully exploited to accomplish a variety of complex tasks, such as image classification, natural language processing, speech analysis, and realistic data generation [1]. Recently, along with the theory improvement and technique breakthrough, neural networks are endowed power to learn how to prevent confidential data from being inferred, which can be employed for secure communication [2], [3], [4]. Advancing this line of works, in [5], it has been shown that neural networks can learn to perform lossy communications and protect communication security between two sides of communication at the presence of a powerful eavesdropping adversary.

Traditionally, in the study of secure communication, adversaries are usually assumed to have limited capacities, including computation power (*e.g.*, limited to polynomial time), possibility of successful attack (*e.g.*, limited to a negligible probability), *etc*. As a matter of fact, however, the ability of adversaries could be more than these. For instance, in generative adversarial networks (GANs) [1], the adversaries are neural networks that attempt to determine whether a sample is generated by a model or drawn from a given data distribution. Moreover, different from traditional secure communication mechanisms, practical approaches to train GANs consider one or a small number of adversaries that can be optimized to be the strongest.

- *Zuobin Xiong, Zhipeng Cai, Daniel Takabi, and Wei Li are with the Department of Computer Science, Georgia State University, Atlanta, GA, USA, 30303.*
  E-mail: zxiong2@student.gsu.edu, {zcai, takabi, wli28}@gsu.edu.
- *Chunqiang Hu is with School of Big Data and Software Engineering, Chongqing University, Chongqing, China, 400044.*
  Email: chu@cqu.edu.cn
- *Corresponding authors: Chunqiang Hu and Wei Li*

Leveraging the advantages of neural networks inspires the emergence of a promising solution, neural network-based communication, with eye-catching features to secure communication. In neural network-based communication, an end-to-end system can be learnt automatically without using any specific encryption/decryption algorithms nor indicating the way of applying these algorithms. Due to the non-linearity and computational complexity of neural networks, they have the intrinsic properties to be used in the protection of confidential data. In addition, a merit that the traditional secure communication methods do not have is the selective protection, while a neural communication system not only can learn how to protect secret information but also can learn what to protect by itself. For example, given a plaintext containing multiple attributes (*e.g.*, multiple objects in an image), if we want to prevent adversaries from seeing one attribute (a special class of object) while making the remaining attributes public, there may have some limitations for the traditional methods unless complex attribute-based encryption is designed. Furthermore, in many real applications, protecting secure information while preserving data usability is a challenging issue. To solve this challenge in the traditional communications systems, the system designers may need to separate the selected secret attributes from others and then compute the secret attributes to be hidden using different algorithms. In contrary to that, neural communication systems can use the end-to-end automatic methods to achieve protection easily, in which the only effort is just to slightly change the objective functions to train the neural networks. The application of neural networks in the selective protection has been investigated for image, text, and speech data in literature [6], [7], [8].

Facing the bright prospects of developing neural network-based communication systems, there are still shortcomings in the existing works. First of all, this research direction has not received enough attention, and [5] is the only existing work that studies neural network-based communication to our best knowledge. In [5], an adversarial neural cryptography (ANC) system has been proposed, which displays us nice properties of using neural network-based communication systems. Nevertheless, the ANC system has several critical security flaws: (i) the method of calculating data distance may be utilized by attackers to infer secure information; (ii) the assumption of attackers is not practical without considering attackers' possible prior knowledge; and (iii) the failure of ensuring data integrity can be explored by attackers to breakdown the system.

In this paper, we first show that the above security flaws can cause system breakdown, for which two types of novel attack models (including target man-in-the-middle attack and target fraud attack) are designed. The goal of attackers is to cause incorrect classification of the receiver's received data by manipulating data at the sender's side or during the transmission. For this purpose, the target man-in-the-middle (TMIM) attacker intends to manipulate the ciphertext so that the receiver receives incorrect data, while the target fraud (TF) attacker aims at tampering plaintext before data is encoded and then sends it to deceive the receiver. In addition, to defend these attacks, a new neural network-based communication system is proposed, in which the architecture of neural networks is improved, plaintext and secret key are separated in the training process, and a defensive loss is adopted in system design. Through theoretical analysis and experiments, we show that our proposed neural communication system can mitigate attack performance and achieve secure communication. Finally, we conduct extensive real data experiments to evaluate the performance of TMIM attack, TF attack, and our defense mechanism in terms of classification accuracy, convergence, throughput, and communication quality. The multi-fold contributions of this paper are summarized as follows.

- Towards the existing neural network-based communication system, ANC, we analyze its security flaws and design two methods for adversarial attack.
- To defend our proposed attacks, we develop a new neural communication system that can achieve secure communication by bounding the attack performance.
- We theoretically prove the effectiveness of our proposed neural communication system.
- By implementing experiments on different datasets, we validate that our proposed attacks on ANC are efficient and our proposed neural communication system has superiority in attack defense, communication efficiency, and communication quality.

The rest of this paper is organized as follows. The related works and preliminaries are introduced in Section 2 and Section 3, respectively. In Section 4, the security vulnerabilities of the existing neural communication system are analyzed. Our attack methods are detailed in Section 5 and validated in Section 6. Then, our defense mechanism is presented in Section 7 and evaluated in Section 8. Finally, this paper is concluded in Section 9.

2 Related Works

We first present related works on neural networks for security and then summarize the existing methods of adversarial example attack.

2.1 Neural Networks for Security

In this paper, neural networks are employed to encode and decode data during communication, the idea of which is similar to encryption and decryption in cryptography algorithms. In the following, we mainly introduce the methods of combining neural networks and cryptography for security.

In [9], the authors proposed to train two neural networks on the output of each other, so that the weights of the two neural networks can be synchronized. Due to non-linearity of neural networks, these synchronized weights are hard to be discovered by adversaries and can be used as seeds of pseudo-random generation. The similar idea appeared in [3], where the secret key is generated over a public channel by two mutually trained neural networks. But, this neural network-based key exchange/generation scheme lacks mathematical proof and might be vulnerable to attacks. Geometric attack, majority attack, and genetic attack are designed to invade this scheme successfully by Andreas Ruttor [2] a few years later.

Chaotic neural networks are another popular application of neural networks in cryptography. Lian *et al.* proposed a cipher algorithm based on chaotic neural networks to encrypt JPEG2000 encoded images [10]. Their algorithm has high security with low cost and can keep the original file format and compression ratio unchanged. In [11], chaotic hopfield neural networks were proposed to encrypt data with time varying delay, where the chaotic neural networks are utilized to generate binary sequences for permuting plaintext. Based on the high-dimension Lorenz chaotic systems and perceptron model within a neural network, Wang *et al.* designed a chaotic image encryption system with a perceptron model [12], in which the Lorenz seed is used as additional input of perceptron at each iteration. However, it has been shown that learning process from these existing works probably can not generate a secure cryptography system [4], [13], [14].

To avoid uncertainty of security performance, the integration of neural networks and cryptography is also investigated in other ways. In [15], homomorphic encryption was incorporated into neural networks, called "Crypto-Nets". By using Crypto-Nets, a private data holder can share his encrypted data with a third party trainer and get accurate prediction without security issues. Also, an improved version of Crypto-Nets was presented for cloud service with high throughput and accuracy [16].

In the above works, neural networks are mainly applied to provide randomness for pseudo-random number and key generation in cryptography and secure communication. Differently, in our considered problem, neural networks are directly hired to secure end-to-end communications apart from the existing approaches.

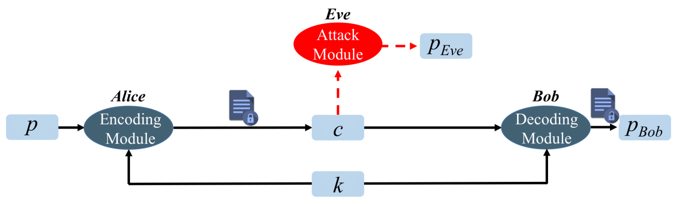


Fig. 1: The framework of Adversarial Neural Cryptography (ANC) system.

2.2 Adversarial Example Attack

Adversarial examples have attracted lots of research interests to invade machine learning models since they were found by Szegedy *et al.* [17]. According to the prior knowledge held by attackers, adversarial example attack can be categorized into two classes: white-box attack and black-box attack. Under white-box attack, attackers have full knowledge about the parameters of target models; while in black-box attack, they do not have such information [18], [19]. The methods of generating adversarial samples include optimization-based methods and generation-based methods [20]. In the optimization-based methods, the problem of finding perturbation is formulated as an optimization problem and can be solved by different solvers [21], [22], [23], [24], [25]. In the generation-based methods, the typical process is to train neural networks with specific loss functions such that the outputs of neural networks are adversarial samples [26], [27].

Although there are various existing attack methods, they only work on classification models. What's more, no adversarial example attack is proposed towards our considered adversarial neural network-based communication system that can be treated as an auto-encoder.

3 Preliminaries

In this section, the basic knowledge about Adversarial Neural Cryptography (ANC) and Generative Adversarial Networks (GANs) is introduced.

3.1 Adversarial Neural Cryptography

The framework of ANC system is shown in Fig. 1, where $Alice$ is an encoding module, Bob is a decoding module, and Eve is an attack module, which are configured with neural network parameters θ_A, θ_B, and θ_E, respectively. For the purpose of secure communication, a sender can use the encoding module $Alice$ to encode a plaintext p with a pre-shared key k and then transmit the corresponding ciphertext c to a receiver who can use the decoding module Bob to recover c with k and obtain the plaintext p_{Bob}. In the ANC system, the pre-shared key k is assumed to be generated and established by neural network synchronization [2], [28], with which a pair of neural networks are trained by exchanging the output of each other. This pair of two multi-layer networks start from randomly initial weights, learn from the output of each other, and obtain common identical weights for both networks at the end of training process. Synchronization of neural networks can be considered as the stage for key generation, and the common identical weights of the two networks can be used as a key. When the pair of neural networks are trained synchronized, the identical network weight parameters will be the shared key between the encoding and decoding modules. Meanwhile, the attack module Eve is used by an attacker who intends to learn p_{Eve} by eavesdropping c. In the presence of attack module Eve, the mainly desired security property of the ANC system is to achieve confidentiality for communications.

Technically, these encoding, decoding, and attack modules (*i.e.*, $Alice$, Bob, and Eve) are trained to optimize predefined loss functions through learning, *e.g.*, gradient descent. On the one hand, to secure communications between the encoding and decoding modules, the objective is to minimize the communication loss that is defined as $\mathcal{L}_{AB} = d(p, Bob(Alice(p,k),k))$ to measure the reconstruction error of decoding module. On the other hand, with the attack module, the attacker's objective is to recover the input of encoding module by minimizing $\mathcal{L}_E = d(p, Eve(Alice(p,k)))$, where $\mathcal{L}_E$ indicates the reconstruction error of the attack module. Thus, the complete loss function of ANC system is $\mathcal{L}_{ANC} = \mathcal{L}_{AB} - \alpha\mathcal{L}_E$ with α being a scale parameter to balance $\mathcal{L}_{AB}$ and $\mathcal{L}_E$, which is trained by updating θ_A, θ_B, and θ_E alternatively. When the training process stops, the encoding and decoding modules can be deployed to communicate securely with little information loss, and the attack module cannot extract any useful information from c even holding the strongest attack power. Especially, the encoding and decoding modules are readily used to take any given input for encoding and decoding with their pre-shared key, as long as the input data comes from the same distribution as their training dataset.

3.2 Generative Adversarial Networks

As the most creative idea of deep learning in recent years, Generative Adversarial Networks (GANs) have been widely applied in the field of adversarial learning since it was proposed in 2014 [1]. GANs consist of two "adversarial" models, including a generator G and a discriminator D. These two adversarial models play with each other to complete in a min-max game, where G intentionally generates samples from a real data distribution to fool D while D judges whether its input is the fake data generated by G or the real data. The generator G is built as a deep neural network that can be designed in various ways, and the discriminator D is also a neural network learned by the traditional supervised learning methods that classifies inputs into two sets (*i.e.*, fake or real). Mathematically speaking, G can be expressed in any form but a simple differentiable function, and $G(z)$ is an output sample drawn from p_g, where z is a low-dimensional vector sampled from a prior distribution p_z. The goal of D is to classify the data from $G(z)$ as fake and the data from training set p_{data} as real. Accordingly, GANs can be formulated as a structured probabilistic model to optimize the following loss function:

$$\begin{aligned}\min_G \max_D \mathcal{L}_{GAN}(G,D) =& \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \\ & \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))],\end{aligned} \tag{1}$$

where G aims to minimize $\mathcal{L}_{GAN}(G,D)$, while D aims to maximize it.

Furthermore, GANs can be extended to a conditional version with an additional input y that could be any kind of auxiliary information (*e.g.*, class labels or data from other domains). The objective function of conditional GANs [29] is expressed as:

$$\begin{aligned}\min_G \max_D \mathcal{L}_{cGAN}(G, D) =& \mathbb{E}_{x \sim p_{data}(x)}[\log D(x|y)] + \\ & \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z|y)))]. \end{aligned} \tag{2}$$

In our defense mechanism, GANs-based idea is exploited to configure the networks of encoding, decoding, and attack modules.

4 Vulnerabilities of ANC System

Through comprehensive and deep analysis, we find out essential security flaws of the ANC system.

First of all, during the training process of the ANC system, the distance between the input of the encoding module and the output of the decoding module, denoted by $d(p, p_{Bob})$, should be minimized for information reconstruction, while the distance between the input of the encoding module and the output of the attack module, denoted by $d(p, p_{Eve})$, should be maximized for information security. Usually, the distance, $d(\cdot, \cdot)$, is estimated by L_1 norm or L_2 norm distance, which may be utilized by attackers to infer actual information. Take L_1 distance for example. Suppose the loss function of attack module Eve is calculated based on a plaintext, *i.e.*, $d(p, p_{Eve}) = \sum_{i=1}^{N} |p_i - p_{Eve_i}|$, where p_i is the i-th bit of plaintext p, p_{Eve_i} is the i-th bit of the attack module Eve's reconstructed plaintext, and N is the length of plaintext. When the ANC system achieves its best performance at the end of training process, $d(p, p_{Eve})$ is equal to the length of plaintext p, *i.e.*, $d(p, p_{Eve}) = N$, which means every bit reconstructed by the attack module is incorrect. Then in the follow-up prediction stage, the attack module might be able to get correct plaintext with a higher probability by flipping p_{Eve} to make the reconstruction loss $d(p, p_{Eve}) = 0$.

Moreover, the ANC system fails to guarantee data integrity, which can be exploited by attackers to invade the system. It is well-known that to verify data integrity, a receiver should use the recovered message and the secret key as the inputs of verification function, *e.g.*, message authentication code. In the ANC system, due to the nature of neural networks, the recovered message could be very similar to the original message but can not be exactly the same. Therefore, the recovered message can not be used for verifying data integrity. Currently, these is no scheme to realize data integrity for the ANC system and other similar neural network-based communication systems.

By exploiting these security vulnerabilities, we first design two types of attack to beat the ANC system and then develop a new defense mechanisms to improve its security performance.

5 Attack Methodology

Inspired by aforementioned analysis on the security flaws, we develop two attack methods, named "target man-in-the-middle attack" (TMIM attack) and "target fraud attack" (TF attack) to break data integrity of the ANC system.

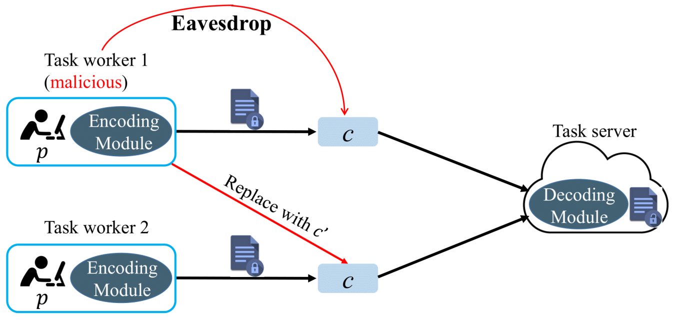


Fig. 2: An example of TMIM attack in crowdsensing scenario.

5.1 Target Man-in-the-Middle Attack

Once the training process is completed in ANC, the encoding module $Alice$ is deployed on the sender side, and the decoding module Bob is deployed on the receiver side for communication. Because the pre-shared key is inside the encoding and decoding modules, any legitimate sender can directly use the encoding module $Alice$ as a black-box to encode a plaintext and transmit to the receiver who can use the decoding module Bob as a black-box to decode the ciphertext. Notice that the ANC system only considers eavesdropping attack at the training stage. That is, the attack module Eve can be only used for eavesdropping attack, in which attackers only know the ciphertext c for reconstructing p_{Eve} without any extra prior knowledge about the communications.

Nevertheless, powerful attackers may exist and be able to obtain extra prior knowledge from various channels in real world. A practical scenario in crowdsensing is illustrated in Fig. 2, where the ANC system is adopted for data transmission. The task workers use the encoding module $Alice$ to encode their sensory data and send the encoded contents to the task server, while the task server can run the decoding module Bob to get the decoded data. Assume the task worker 1 is malicious and intends to attack on task worker 2's transmission. Under this situation, task worker 1 can perform chosen plaintext attack (CPA) by inputting multiple plaintexts p into the encoding module and then eavesdrop their corresponding ciphertexts c [30]. With these paired plaintexts and ciphertexts (p, c) as prior knowledge, task worker 1 can infer or extract important information about the encoding module to break the entire ANC system. By eavesdropping the communication channel of task worker 2, task worker 1 can replace the original encoded data c by an adversarial sample c' and send it to the task server to realize TMIM attack. The definition and implementation of our TMIM attack is addressed in the following part of this section.

Assume that in the communication system, a plaintext $p \in P$ is encoded by encoding module and decoded by decoding module, where P is the plaintext space agreed upon by both parties. Due to the property of neural networks, the input and network parameters are set to be continuous floating-point number for differentiability. Accordingly, every plaintext is a sequence of continuous floating-point number. In real applications, the plaintext can be different kinds of data, such as image/video data, or text data. Considering such heterogeneous data domains, a pre-processing is required on the value of each plaintext p, where the plaintext is normalized in range $[0, 1]$ for encoding. For example, image/video data is rescaled in pixel level and text data is processed by word embedding. Finally, the plaintext space is $P \in [0, 1]^l$, where l is the length of plaintext. Each plaintext p has a corresponding class label $y \in Y = \{y_1, y_2, \ldots, y_k\}$, where k is the number of class in the plaintext space P.

In TMIM attack, the attacker can be any type of machine learning algorithm. Here, we adopt a neural network as the representative of attack module. Suppose the attacker has the ability to implement chosen plaintext attack, in which he can select a plaintext p_{Eve}, send it to encoding module (*i.e.*, step (1) in Fig. 3(a)), and get its corresponding encoded ciphertext c_{Eve} (*i.e.*, step (2) in Fig. 3(a)). With the paired plaintext-ciphertext data (p_{Eve}, c_{Eve}), the attacker can train a classifier using neural networks and gain a class distribution $Y' = \{y'_1, y'_2, \ldots, y'_k\}$ of c_{Eve}. Notice that Y and Y' would be extremely similar if the attacker can obtain enough data samples through chosen plaintext attack. Next, the attack can choose either step (3) or step (4) to launch TMIM attack, which is detailed in Section 5.1.1 and Section 5.1.2. The goal of TMIM attacker is to find an adversarial example that misleads decoding module (Bob) and breaks data integrity of the ANC system.

Ideally, in the ANC system, the class label of encoding module's input and the class label of decoding module's recovered result from the ciphertext c should be exactly the same. Thus, as the man in the middle, if the attacker can replace the original ciphertext c with another different ciphertext c' such that the class label of decoding module's recovered result from c' still falls into the plaintext space P but is different from the label of encoding module's original plaintext, a successful TMIM attack is achieved.

As aforementioned in Section 2, previous works on adversarial example mainly target classification models. Differently, the ANC system is more like an auto-encoder based generative model, to which the existing attack methods may not be applied directly. For generative models, adversarial example attack could be target attack or untarget attack. Under target attack, an adversary intends to obtain $I(Bob(c', k)) = y_t \neq I(p)$, where $y_t \in Y'$ is the adversary's target class. Under untarget attack, it only requires $I(Bob(c', k)) \neq I(p)$. In this paper, we focus on target attack, because it has more severe consequences and can be transformed to untarget attack easily. Our proposed target man-in-the-middle attack is defined below.

Definition 1. *Target Man-in-the-Middle Attack (TMIM Attack). Given an original plaintext $p \in P$, an encoding module $Alice(p, k) = c$, and a decoding module $Bob(c, k) = p_{Bob}$, an attacker replaces c by c' with an aim of $Bob(c', k) \in P$ and $I(Bob(c', k)) = y_t \neq I(p)$, where I is an identification oracle that can distinguish different data, and y_t is attacker's target class.*

In practice, the oracle I could be human eyes judgment or a pre-trained classifier on plaintext domain P. Two novel approaches of implementing TMIM attack are developed in the following.

5.1.1 Simple TMIM Attack

Suppose the attacker can train a classification model f that can classify c_{Eve} to a class y_{Eve}, where y_{Eve} is the ground truth label of p_{Eve}. If the attacker repeats chosen plaintext attack enough times with different p_{Eve}, f would be accurate enough to tell what is the class of ciphertext c output by encoding module. In addition, by using model f, the attacker obtains each class's distribution and corresponding ciphertexts. For a target class y_t, the attacker extracts the ciphertexts that belong to y_t and use the mean of those extracted ciphertexts to create a new ciphertext c'. Then, he can directly replace the original ciphertext c by c' (*i.e.*, step (3) in Fig. 3(a)) and sends it to the decoding module of receiver. At receiver side, the reconstructed plaintext is $p_{Bob} = Bob(c', k)$ that becomes an adversarial example with a high probability.

In Section 6, we set up experiments to exam the performance of our simple TMIM attack and show the attack results in Fig. 5.

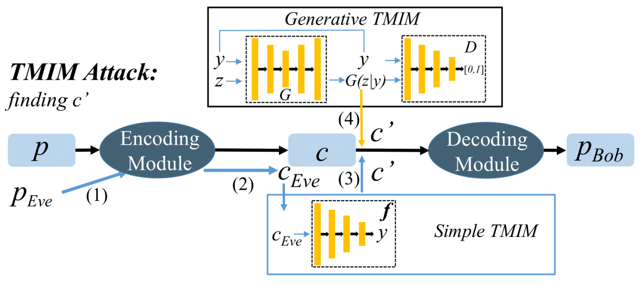
5.1.2 Generative TMIM Attack

In the simple TMIM attack, the adversarial ciphertext c' is always identical for a target class y_t as c' is calculated based on the mean of ciphertexts belonging to y_t. As a result, the decoding module's reconstructed plaintext $p_{Bob} = Bob(c', k)$ is always the same for the class y_t. In order to improve the diversity of reconstructed plaintext by decoding module, we adopt the cGANs as a generator to produce the adversarial ciphertext c' based on a given target label y_t.

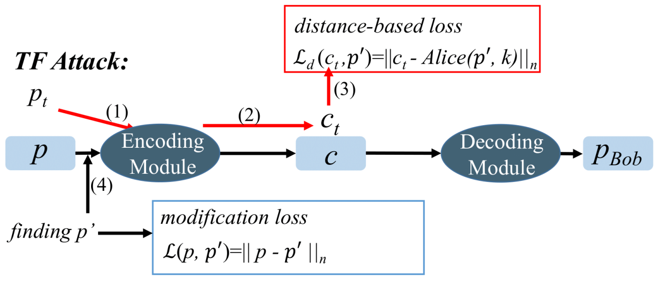
The workflow of generative TMIM attack is shown in black square of Fig. 3(a), where a conditional generator G and a conditional discriminator D are employed. The idea of generative TMIM attack is the same as that of simple TMIM attack – finding an adversarial ciphertext c' to fool the decoding module. Recall that the attacker has paired data (c_{Eve}, y_{Eve}), which is the real data distribution of cGANs. In our cGANs, the generator G takes a predefined random noise $z \sim \mathcal{N}(0, 1)$ and condition y to output the generated ciphertext $G(z|y)$; on the other hand, the discriminator D takes real ciphertext c or generated $G(z|y)$ with label y as the inputs to differentiate whether its input data is real or fake by assigning $D(\cdot|y)$ different values, where $\cdot$ can be either c or $G(z|y)$.

The loss function of generative TMIM attack is similar to that in Eq. (2). Once the training process of cGANs is finished, the attacker is able to generate countless and diverse adversarial ciphertext c' by using $c' = G^*(z|y_t)$, where the G^* is an optimized ciphertext generator. Then, the generated adversarial ciphertext c' is sent to decoding module (*i.e.*, step (4) in the Fig. 3(a)) to decode. In the experiments, the attack results of our generative TMIM attack is shown in Fig. 6.

Essentially speaking, TMIM attack is an unrestricted adversarial example attack because the crafted example c' is not restricted to be close to the original plaintext c, *i.e.*, the modification constraint $\|c - c'\|_n < \delta$ is not required where n is the L_n norm distance, and $\delta > 0$ is a system parameter. Since ciphertext data is not visible or checkable by any identifiers during communications, c' still seems imperceptible to the decoding module (Bob) as long as $p_{Bob} = Bob(c', k) \in P$, no matter how large $\|c - c'\|_n$ is.



(a) The framework of target man-in-the-middle attack


TF Attack:
distance-based loss
$\mathcal{L}_d(c_t, p') = \|c_t - Alice(p', k)\|_n$
Encoding Module
Decoding Module
finding p'
modification loss
$\mathcal{L}(p, p') = \|p - p'\|_n$

(b) The framework of target fraud attack

Fig. 3: The frameworks of proposed target man-in-the-middle (TMIM) attack and target fraud (TF) attack towards ANC system.

5.2 Target Fraud Attack

In this subsection, target fraud attack (TF attack) is proposed to attack a different part of the ANC system: the input plaintext p of encoding module.

As we illustrated, the ANC communication system can be applied in real applications such as data transmission or data collection, where the encoding module $Alice$ is deployed at the sender side, and the decoding module Bob is deployed at the receiver side. Ideally, the data p sent from the sender should be the same as the data p_{Bob} received by the receiver through communication. Assume that there is a stronger attacker who has the ability to use the encoding module $Alice$ to encode data and eavesdrop the encoded ciphertexts. In addition, the attacker can modify the original data to crafted-but-imperceptible adversarial data such that the adversarial data sent from sender is totally different from the recovered data at the receiver side. In this case, the integrity of ANC system is broken. For example, as shown in Fig. 4, in a crowdsensing system, the task worker could be malicious to perform such an attack by modifying his/her original data p to the adversarial data p' through our designed TF attack method. Then, the malicious task worker can encode the modified data p' through the encoding module $Alice$, and finally the task server decodes data with the decoding module Bob and obtain an adversarial data of target class, instead of the original data's ground truth class.

More examples of TF attack in image domain can be found in our experiments. A formal definition of target fraud attack is given by Definition 2.

Definition 2. *Target Fraud Attack (TF Attack). Given an original plaintext $p \in P$, an encoding module $Alice(p, k) = c$, and a decoding module $Bob(c, k) = p_{Bob}$, the attacker provides an adversarial example p' to the encoding module with an aim of $Bob(Alice(p', k), k) \in P$ and $I(Bob(Alice(p', k), k)) = y_t \neq I(p')$ under two requirements, including (i) distance requirement $\|p' - p\|_n < \delta$ and (ii) semantic requirement $I(p') = I(p)$, where I is an identification oracle that can distinguish different data, and y_t is the attacker's target class.*

Assume the attacker still owns the classification model f and has the capability of performing chosen plaintext attack and eavesdropping communications. The attacker can select a plaintext p_t from the target class y_t, sends p_t through encoding module, and eavesdrops the encoded ciphertext c_t (*i.e.*, step (1) and (2) in Fig. 3(b)). According to the target

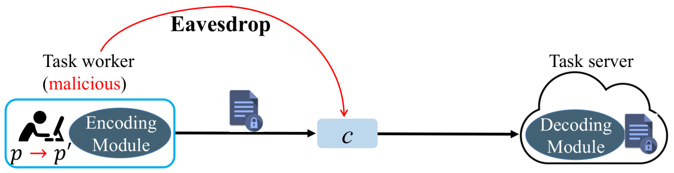


Fig. 4: An example of TF attack in crowdsensing scenario.

ciphertext c_t, the attacker chooses a loss function as the objective to generate adversarial examples through minimizing the selected loss function. After that, the optimized adversarial plaintext p' is sent to encoding module (*i.e.*, step (4) in Fig. 3(b)) and then transmitted to decoding module for target fraud attack.

According to Definition 2, the difference between the original plaintext p and adversarial plaintext p' should be restricted, *i.e.*, $\|p - p'\|_n < \delta$, where n represents L_n norm of two plaintexts. Accordingly, the modification loss is calculated as:

$$\mathcal{L}(p, p') = \|p - p'\|_n. \tag{3}$$

The objective of attacker is to classify the identification result $I(p_{Bob})$ into a target class y_t rather than the original y. For this purpose, the ciphertext c is utilized as a bridge, because both encoding module and decoding module are black-boxes for the attacker, and the only way to interact with p_{Bob} is to control ciphertext c. In this paper, a distance-based loss is considered, for which the ciphertext loss function can be formulated in Eq. (4) based on p' and c_t.

$$\mathcal{L}_d(c_t, p') = \|c_t - Alice(p', k)\|_n, \tag{4}$$

which implies that the encoding result of p' is pushed to the target ciphertext c_t.

By combining the plaintext loss and ciphertext loss together, we can obtain the complete loss function for target fraud attack in Eq. (5).

$$\mathcal{L}_{FA} = \lambda_1 \mathcal{L}(p, p') + \lambda_2 \mathcal{L}_d(c_t, p'), \tag{5}$$

The results of target main-in-the-middle attack and target fraud attack on the ANC system are analyzed in Section 6.

TABLE 1: The architecture of encoding/decoding module

Layer	Networks of encoding module
1	Fully Connected, [H×W×C+L] × [H×W×C], Leaky ReLU
2	Fully Connected, [H×W×C] × [H×W×C], Sigmoid
	Networks of decoding module
1	Fully Connected, [H×W×C+L] × [H×W×C], Leaky ReLU
2	Fully Connected, [H×W×C] × [H×W×C], Sigmoid

6 Validation of Attack Models

In this section, we conduct intensive experiments on three datasets, including MNIST, Fashion-MNIST and CIFAR100, to investigate the performance of our proposed target man-in-the-middle attack and target fraud attack. In the implementation, we adopt image data as plaintext for performance evaluation, which is more complex than using binary string data and can provide better visualization.

6.1 Experiment Settings

Image data is adopted in our experiments for better visualization of results, which means that the encoding and decoding operations are performed on the corresponding pixel level. For example, the pixel value of an image is in the range of $[0, 255]$ for gray scale and $[0, 255]^3$ for three-channel RGB image. Through normalization, it can be easily mapped to a small range of $[0, 1]$, on which our neural network modules are conducted. The architecture of our encoding module and decoding module are described in TABLE 1, where H, W, C are the height, width, channel of input images, respectively, and L is the length of key. The value of L (default by 256 bits) can be customized by different security requirements. As shown in the following experiment result, a shorter key has a better throughput and transmission quality. The device used for model training is a Linux server with Intel(R) Xeon CPU E5-1607, 16 GB memory, and the NVIDIA GeForce RTX 2080 GPU with 11 GB memory. For learning based models, the training stage may cost much time and resource to converge. But once the models are trained well, the only calculation is multiplication that can be executed on a simple chip. For example, when the key size is 256 on MNIST dataset, the memory usage of our proposed system is around 20MB that is 4MB smaller than the memory usage of ANC.

6.2 Evaluation of Target Man-in-the-Middle Attack

Corresponding to Section 5.1, we implement both simple TMIM attack and generative TMIM attack and display the results in Fig. 5 and Fig. 6.

In Fig. 5, the decoded results look very similar with each other for different plaintexts. This is in line with our analysis in Section 5: the result of simple TMIM attack always remains the same because the modified ciphertext c' for a batch of attack samples is the same. More diverse decoded results are generated by generative TMIM attack as shown in Fig. 6. By comparing Fig. 5 and Fig. 6, the original plaintext images in both simple TMIM attack and generative TMIM attack are the same, but the decoded images in Fig. 6(a) are more vivid and diverse. Similar observation can be found in Fig. 5(b) and Fig. 6(b) on Fashion-MNIST dataset. For the evaluation on CIFAR100 dataset, comparable observation can be found when simple TMIM attack is launched as shown in Fig. 5(c), where the decoded images are identical. While in Fig. 6(c), the results of generative TMIM attack on CIFAR100 dataset are a little bit worse than those of simple dataset MNIST and Fashion-MNIST. The reason is that there are much more classes in CIFAR100 dataset and each class has less training data, which decreases the performance of GAN-based generative TMIM attack.

For a comprehensive understanding of simple TMIM attack and generative TMIM attack, we also present the quantitative results in TABLE 2. Recall that in the definition of TMIM attack, we hope that the original plaintext p and the attacked plaintext p_{Bob} should have different class labels by using the predefined identifier I, *i.e.* $I(p_{Bob}) = y_t \neq I(p)$. In TABLE 2, the average classification accuracy of identifier I on decoded p_{Bob} to target class y_t is used to measure the effectiveness of our attack approaches. For both MNIST and Fashion-MNIST datasets, 10 different target class labels are adopted to measure the classification accuracy. More specifically, the classification accuracy of adversarial plaintext generated by simple TMIM attack reaches 99.50%, and the classification accuracy of adversarial plaintext generated by generative TMIM attack is a little bit lower. Considering the diversity of adversarial samples and the classification accuracy, generative TMIM attack would be a better choice in practice with less classes.

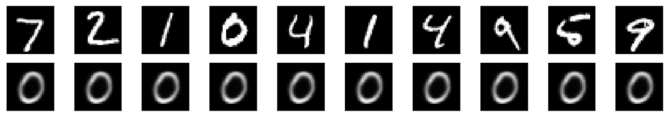
6.3 Evaluation of Target Fraud Attack

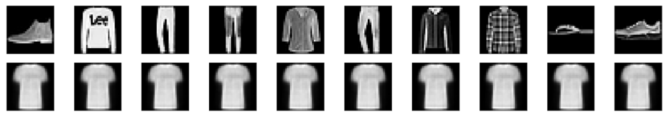
Target fraud attack is aiming to change the original plaintext p to the adversarial plaintext p' such that p' can be turned to target class y_t after encoding and decoding. In the experiments, the distance based loss function, Eq. (4), is adopted for target fraud attack.

Firstly, the attack results of 4 different L_n norms are shown in Fig. 7 for comparison. Fig. 7(a), Fig. 7(b), Fig. 7(c), and Fig. 7(d) present the attack results of L_0, L_1, L_2, and L_∞ norms, respectively, where the three images from left to the right of each norm group respectively correspond to the plaintext p, the adversarial plaintext p', and the decoded ciphertext p_{Bob}, and the notation above each image is the predicted label by identifier $I(\cdot)$. The results show that for each norm, the decoded ciphertext p_{Bob} is predicted as target class y_t="0", which means target fraud attack is successful. Particularly, the generated adversarial plaintext and the decoded ciphertext with L_2 and L_∞ norms have better image visual quality. Similarly, we can draw the same conclusions on Fashion-MNIST with target class y_t="T-shirt" and CIFAR100 datasets with target class y_t="Wolf".

In addition, the statistics of performance of target fraud attack are reported in TABLE 3. Different from TMIM attack, the goals of target fraud attack is to classify the decoded data into a target class while keeping the class labels of the adversarial image and the original image the same. To this end, we define a new metric to evaluate the performance of target fraud attack as follows:

$$TF_{accuracy} = \frac{Num_{(I(p')=y \wedge I(p_{Bob})=y_t)}}{Num_p},$$


(a) The results of simple TMIM attack on MNIST dataset with target class “digit=0”


(b) The results of simple TMIM attack on Fashion-MNIST dataset with target class “T-shirt”

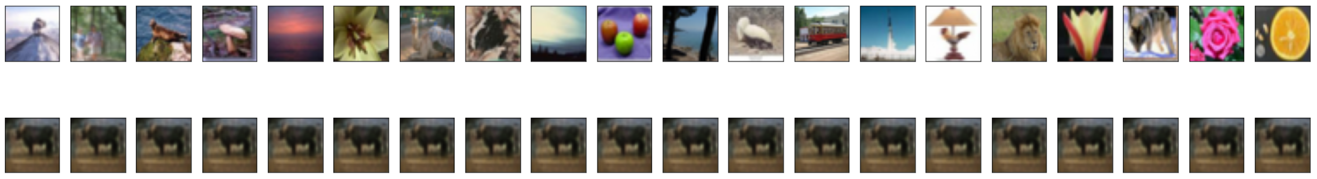
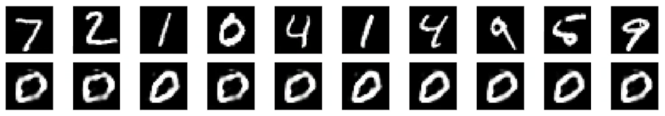
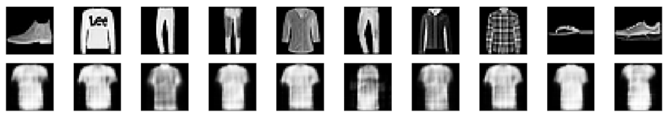

(c) The results of simple TMIM attack on CIFAR100 dataset with target class “cattle”

Fig. 5: The results of simple TMIM attack. In each subfigure, the first row shows encoding module’s original plaintext p, and the second row shows decoding module’s decoded result p_{Bob}.


(a) The results of generative TMIM attack on MNIST dataset with target class “digit=0”


(b) The results of generative TMIM attack on Fashion-MNIST dataset with target class “T-shirt”

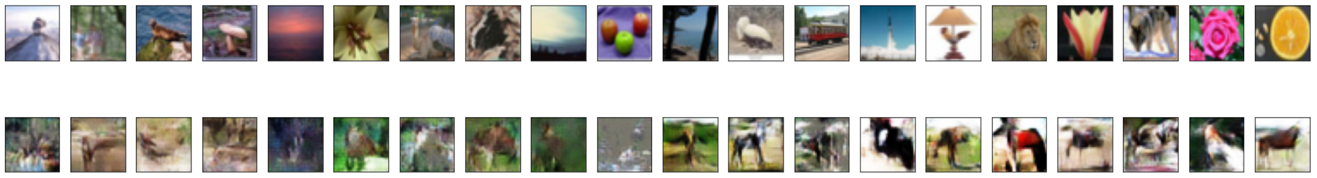

(c) The results of simple TMIM attack on CIFAR100 dataset with target class “cattle”

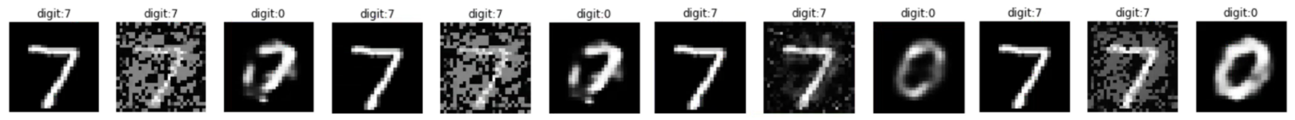
Fig. 6: The results of generative TMIM attack. In each subfigure, the first row shows encoding module’s original plaintext p, and the second row shows decoding module’s decoded result p_{Bob}.

TABLE 2: Quantitative performance statistics of simple TMIM attack and generative TMIM attack on MNIST and Fashion-MNIST datasets

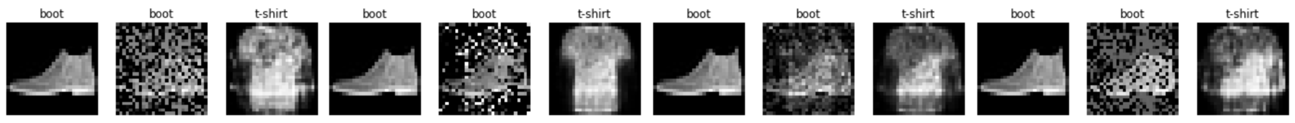
MNIST	digit=0	digit=1	digit=2	digit=3	digit=4	digit=5	digit=6	digit=7	digit=8	digit=9
Simple TMIM attack	99.50%	99.50%	99.50%	99.50%	99.50%	99.50%	99.50%	99.50%	99.50%	99.50%
Generative TMIM attack	98.35%	98.29%	98.84%	98.66%	97.39%	98.62%	99.22%	99.41%	98.60%	98.98%
Fashion-MNIST	T-shirt	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Boot
Simple TMIM attack	91.63%	90.30%	92.51%	92.22%	93.41%	92.66%	92.94%	92.92%	91.78%	92.50%
Generative TMIM attack	87.96%	89.54%	88.36%	89.97%	87.88%	87.05%	89.09%	88.96%	89.34%	88.79%

TABLE 3: Quantitative performance statistics of target fraud attack with different L_n norms on MNIST dataset (target class: “digit=0”) and Fashion-MNIST datasets (target class: “T-shirt”).

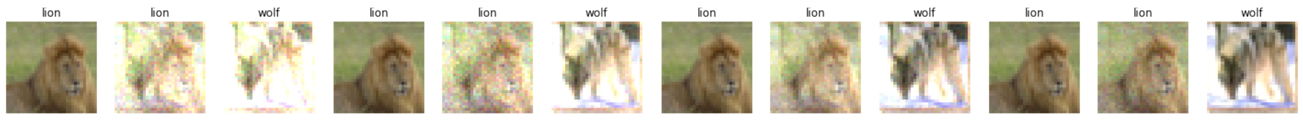
MNIST	digit=0	digit=1	digit=2	digit=3	digit=4	digit=5	digit=6	digit=7	digit=8	digit=9
L_0 norm	-	77.83%	80.09%	77.08%	79.90%	76.08%	76.31%	76.78%	76.14%	78.35%
L_1 norm	-	80.76%	79.78%	79.06%	81.17%	78.62%	79.12%	82.52%	82.27%	82.29%
L_2 norm	-	94.66%	94.05%	94.74%	93.31%	93.82%	94.09%	94.19%	93.12%	95.73%
L_∞ norm	-	98.56%	98.53%	96.07%	96.31%	97.33%	96.95%	98.58%	97.94%	96.40%
Fashion-MNIST	T-shirt	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Boot
L_0 norm	-	69.62%	69.33%	70.79%	69.73%	70.53%	69.05%	70.58%	70.59%	68.36%
L_1 norm	-	75.14%	75.48%	76.12%	76.05%	74.55%	75.15%	76.09%	75.71%	75.29%
L_2 norm	-	81.01%	80.09%	80.94%	80.07%	80.43%	80.68%	80.33%	80.74%	80.65%
L_∞ norm	-	83.41%	85.47%	82.98%	82.84%	82.38%	82.47%	82.71%	83.41%	82.49%



(a) TF attack with L_0 norm (b) TF attack with L_1 norm (c) TF attack with L_2 norm (d) TF attack with L_∞ norm



(e) TF attack with L_0 norm (f) TF attack with L_1 norm (g) TF attack with L_2 norm (h) TF attack with L_∞ norm



(i) TF attack with L_0 norm (j) TF attack with L_1 norm (k) TF attack with L_2 norm (l) TF attack with L_∞ norm

Fig. 7: The results of target fraud attack with different L_n norms on MNIST, Fashion-MNIST and CIFAR100 datasets. Each subfigure contains three images for comparison.

where y is the original label of plaintext p, y_t is the target label selected by the attacker, $Num_{(I(p')=y \wedge I(p_{Bob})=y_t)}$ denotes the number of adversarial plaintexts for successful TF attack, and Num_p is the number of total test plaintexts. Essentially, the value of $TF_{accuracy}$ indicates the success rate of TF attack. As shown in TABLE 3, with a given target class (*e.g.* digit=0 for MNIST dataset and T-shirt for Fashion-MNIST dataset), the attacker produces adversarial plaintexts from other 9 classes. Except the first column, each column of TABLE 3 represents the value of $TF_{accuracy}$ when the adversarial plaintext is generated from a specific class, *e.g.*, in the second column of TABLE 3, $TF_{accuracy} = 77.83\%$ means the attack accuracy when the adversarial plaintext is generated from the class "digit=1" to perform TF attack on MINIST dataset, and $TF_{accuracy} = 69.72\%$ means when the adversarial plaintext is generated from the class "Trouser" to perform TF attack on Fashion-MINIST dataset with L_0 norm. From these results, we can conclude that TF attack with L_∞ norm achieves the best attack performance.

7 Defense Mechanism

In the previous section, we propose target man-in-the-middle attack and target fraud attack toward the existing ANC system, which demonstrates that the ANC system is not secure enough to be used in practical applications. Moreover, in the neural communication system, the original plaintext p and the corresponding decoded ciphertext p_{Bob} may not be exactly the same in every bit, and thus traditional methods of digital signature and message authentication codes cannot be properly used in the ANC to defend target man-in-the-middle attack or target fraud attack. As a matter of fact, the failure of the ANC system comes from the inappropriate system design rather than the neural networks. In this section, we develop a new neural communication system based on neural networks that can mitigate the above two kinds of attack.

The prerequisite of launching target man-in-the-middle attack and target fraud attack is that the attacker can perform chosen plaintext attack to obtain paired data (p_{Eve}, c_{Eve}). Thus, the core idea of our defense strategy is to prevent or mitigate the information gain of chosen plaintext attack. As described in Section 5, the attacker is able to input a plaintext to encoding module directly and then get its corresponding ciphertext, because the ANC system integrates the pre-shared key into the encoding and decoding modules as shown in Fig. 1. What's worse, the same key in ANC is used multiple times, which increases the risk of being broken by cryptanalysis. As a result, to avoid potential security flaws, the secret key in our proposed system is separated from the encoding and decoding modules with less reuse possibility.

The framework of our proposed secure communication system is given in Fig. 8, where an encoding module $Alice$, a decoding module Bob, and an attack module Eve are taken into account. Compared with the ANC system, our proposed system has a similar structure but totally different designs. In our system, the encoding module and decoding module are not autoencoder-based symmetric networks any more. Instead, they are set to be identical networks. At the first layer of the encoding module, a plaintext p with length l and a secret key k are concatenated as the input, and the size of encoded result is also l. The network structure of the decoding module is exactly same as that of the decoding module. The advantages of such settings are two-fold: (i) configuring the same network structure and parameters at the two sides of communications can reduce the training time by half in practice; and (ii) once our communication system is trained, either module can perform encoding and decoding functions, which is convenient and efficient for practical implementation. For example, if p is the input of encoding module, $Alice(p, k)$ is the encoding module to compute the ciphertext c; whereas, the module $Alice$ becomes a decoding module with $Alice(c, k) = p$ if it is deployed on the receiver side. In this way, one module (either the encoding module or the decoding module) can be used by the sender or receiver to satisfy the need of the two-way communications.

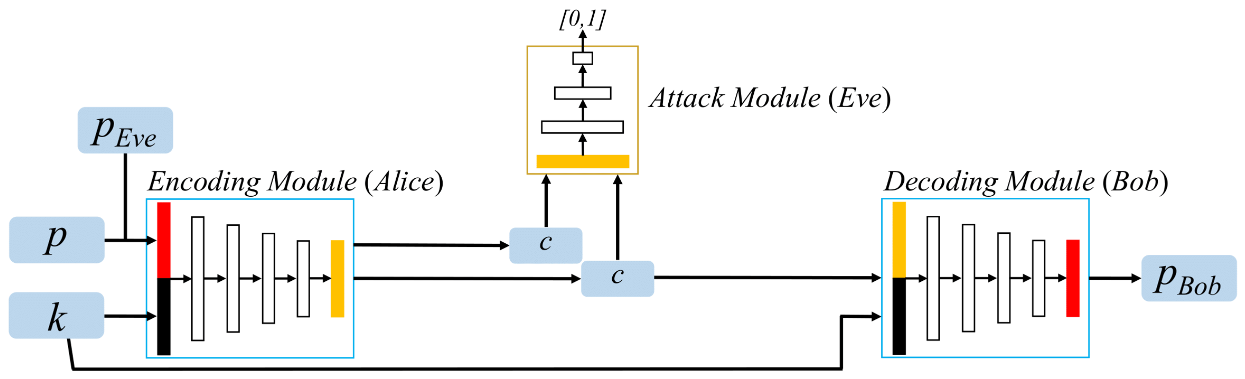


Fig. 8: The framework of our proposed new neural communication system to mitigate TMIM attack and TF attack.

Notably, at the sender side, both the plaintext p and the key k are needed to run the encoding module in the encoding processes. To query the encoding module and proceed chosen plaintext attack towards our proposed defense system, the attacker should input p_{Eve} and his key k' to the encoding module $Alice$ and get $c_{Eve} = Alice(p, k')$. In our system, the secret key is separated from the encoding/decoding module, and every user has his distinct key, so the key k' used by the attacker may not be equal to k. Consequently, the attacker's ciphertext c_{Eve} may not be the same as the correct ciphertext c unless the attacker tries the correct key k out. Under this situation, the attacker needs to use advanced approaches to retrieve more information from c_{Eve}. Thus, in our system, the attack module Eve is upgraded to be a binary classifier; that is, our system aims to defend a more powerful attacker. The input domain of the attack module Eve contains C and C_{Eve}, where C is the set of correct ciphertexts computed with the real key k, and C_{Eve} is the set of ciphertexts computed with the attacker's key k'. For the attacker, the objective is to discriminate C and C_{Eve} using the discriminator D so that he can gain more information from ciphertexts via the attack module Eve. If Eve can discriminate C and C_{Eve}, it is able to distinguish between k and k'; otherwise, both k and k' seem indistinguishable to the attacker. Based on the correct discrimination results, the attacker can get correct paired data (p_{Eve}, c_{Eve}) for future attacks. Accordingly, the loss function of attack module Eve can be formulated in Eq. (6).

$$\mathcal{L}_{Eve} = \mathbb{E}_{c\sim C}[\log Eve(c)] + \mathbb{E}_{c\sim C_{Eve}}[\log(1 - Eve(c))]. \quad (6)$$

To defend chosen plaintext attack from the attacker, hiding C_{Eve} in C is necessary during the training process of encoding module, which means the distributions of C and C_{Eve} should be trained indistinguishable no matter what key is used in encoding module. Apart from this, encoding module and decoding module are expected to work together to minimize the communication loss in the system, *i.e.*

$$\min \mathcal{L}(p, p_{Bob}) = \|p - Bob(Alice(p, k), k)\|_n, \quad (7)$$

where n is the L_n norm. We define the loss function for encoding module and decoding module together in Eq. (8) to integrate the two above goals: hiding C_{Eve} in C, and minimizing the communication loss.

$$\mathcal{L}_{(Alice,Bob)} = \mathbb{E}_{c\sim C_{Eve}}[\log(1 - Eve(c))] + \mathcal{L}(p, p_{Bob}). \quad (8)$$

The formal expression for training our proposed neural communication system is expressed as:

$$\begin{aligned}\min_{(Alice,Bob)} \max_{Eve} \mathcal{L}_{system} =& \mathbb{E}_{c\sim C}[\log Eve(c)]+ \\ & \mathbb{E}_{c\sim C_{Eve}}[\log(1 - Eve(c))]+ \\ & \|p - Bob(Alice(p, k), k)\|_n.\end{aligned} \quad (9)$$

Theorem 1. *When the training process of our system is converged with the optimal encoding module (Alice), the probability that the attacker can differentiate between any correct ciphertext c and any ciphertext c_{Eve} is at most 50%.*

Proof: By substituting Eq. (6), Eq. (7) and Eq. (8), Eq. (9) can be rewritten as:

$$\begin{aligned}\mathcal{L}_{system} =& \mathbb{E}_{c\sim C}[\log Eve(c)] + \mathbb{E}_{c\sim C_{Eve}}[\log(1 - Eve(c))] \\ & + \mathcal{L}(p, p_{Bob}) \\ =& \mathbb{E}_{p\sim P}[\log Eve(Alice(p, k))] \\ & + \mathbb{E}_{p\sim P_{Eve}}[\log(1 - Eve(Alice(p, k')))] \\ & + \|p - Bob(Alice(p, k), k)\|_n.\end{aligned}$$

During the training process, the plaintext p is sampled from the plaintext space $P \in [0, 1]^l$ to optimize the loss function, where l is the length of p. In our system, $p \in P$ is a sequence of continuous float-point number, and P and P_{Eve} are high-dimensional continuous space in $[0, 1]$ for each dimension. Thus, within P and P_{Eve}, the expectation calculation of $\mathcal{L}_{System}$ can be equivalently transformed to integration over all possible p, *i.e.*,

$$\begin{aligned}\mathcal{L}_{system'} =& \int_p P(p)[\log Eve(Alice(p, k))] \\ & + \int_p P_{Eve}(p)[\log(1 - Eve(Alice(p, k')))] \\ & + \|p - Bob(Alice(p, k), k)\|_n \\ =& \int_p \{P(p)[\log Eve(Alice(p, k))] \\ & + P_{Eve}(p)[\log(1 - Eve(Alice(p, k')))]\} \\ & + \|p - Bob(Alice(p, k), k)\|_n.\end{aligned}$$

In our system, the attack module Eve is defined as a discriminator with an output range [0, 1], and the objective of module Eve is to maximize his output value. Since $\|p - Bob(Alice(p, k), k)\|_n$ does not influence Eve's optimization, it can be treated as a constant value to attack module Eve. To get the maximum value for attack module Eve, we should take the partial derivative of $\mathcal{L}$ with respect to Eve inside the integration, where $\mathcal{L}$ is expressed as:

$$\begin{aligned}\mathcal{L} =& P(p)[\log Eve(Alice(p,k))] \\ &+ P_{Eve}(p)[\log(1 - Eve(Alice(p,k')))].\end{aligned}$$

The first order and the second order partial derivatives of $\mathcal{L}$ with respect to Eve are computed in the following.

$$\frac{\partial \mathcal{L}}{\partial Eve} = \frac{P(p)}{Eve(Alice(p,k))} - \frac{P_{Eve}(p)}{1 - Eve(Alice(p,k'))}.$$

$$\frac{\partial^2 \mathcal{L}}{\partial Eve^2} = -\frac{P(p)}{[Eve(Alice(p,k))]^2} - \frac{P_{Eve}(p)}{[1 - Eve(Alice(p,k'))]^2}.$$

It is obvious that $\frac{\partial^2 \mathcal{L}}{\partial Eve^2} < 0$ is always true. So, when $\frac{\partial \mathcal{L}}{\partial Eve} = 0$, the maximum output of attack module Eve can be achieved. From $\frac{\partial \mathcal{L}}{\partial Eve} = 0$, we have

$$P(p)(1 - Eve(Alice(p,k))) = P_{Eve}(p)Eve(Alice(p,k')).$$

On the other hand, when the training process is terminated after convergence, encoding module is supposed to have enough power to mimic the distribution of $Alice(p,k)$ by using $Alice(p,k')$. Thus, we can replace $Alice(p,k)$ and $Alice(p,k')$ with the same notation c and rewrite the above equation as:

$$P(p)(1 - Eve(c)) = P_{Eve}(p)Eve(c).$$

Accordingly, the value of attack module Eve is

$$Eve(c) = \frac{P(p)}{P(p) + P_{Eve}(p)}.$$

Theoretically speaking, when the training process is converged, attack module Eve owns the strongest power to manipulate her plaintext p_{Eve}; that is, the distribution of $P_{Eve}(p)$ will be the same as the distribution of $P(p)$ if convergence is reached. By setting $P_{Eve}(p) = P(p)$, the optimal output value of attack module Eve is $Eve^*(c) = 0.5$, which means the attack module Eve is not able to tell the source of ciphertext c with a probability more than 50% even she has the strongest capability. □

Theorem 1 shows that when our system is trained to be optimal, the success rate of attacker's chosen plaintext attack can be effectively bounded. Notice that in TMIM attack and TF attack, a common prerequisite is that the attacker is capable of performing chosen plaintext attack. Once the success rate of chosen plaintext attack is low, it will become harder for the attacker to implement TMIM attack and TF attack. In other words, our system can effectively mitigate the impacts of TMIM attack and TF attack

8 Evaluation of Defense Mechanism

In this section, we design experiments to evaluate the performance of our defense mechanism, where the same settings and datasets are adopted as previous experiments in Section 6.

We have proved that our proposed system can bound the performance of chosen plaintext attack and mitigate the impacts of TMIM attack and TF attack.

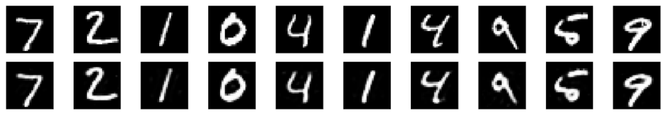
TABLE 4: Quantitative comparison between ANC and our system on the capability of attack defense

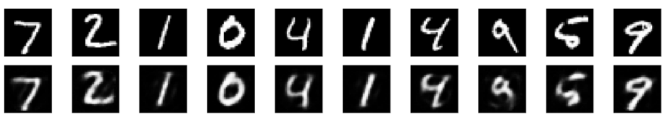
Dataset	MNIST		Fashion-MNIST		CIFAR100	
Model	ANC [5]	Ours	ANC [5]	Ours	ANC [5]	Ours
S. TMIM	99.50%	17.73%	92.28%	15.92%	52.20%	4.78%
G. TMIM	98.63%	17.14%	88.69%	14.28%	45.67%	4.49%
TF	97.40%	14.47%	83.13%	10.66%	39.17%	3.62%

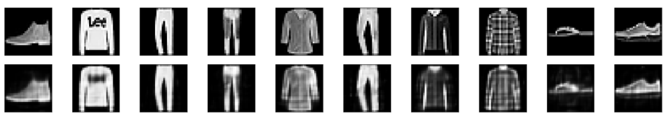
To evaluate the practical defense capability of our system, we implement TMIM attack and TF attack in the same way to attack our proposed system and make a comparison with the ANC system. The comparison results are shown in TABLE 4, where the values are corresponding classification accuracy at receiver side for TMIM attack and $TF_{Accuracy}$ for TF attack. Since there are 100 classes in CIFAR100 dataset, it is not possible for us to present the TMIM attack and TF attack results for all classes in this paper with limited page length. Instead, the average performance of on all classes is listed in TABLE 4. More specifically, a larger value in TABLE 4 indicates a stronger attack power and a worse defense capability. To get a clear observation, we calculate the average index for all classes in each dataset on the two systems; especially, for TF attack, we use L_∞ norm because it achieves the best attack performance. From the results, one can see that our system decreases the attack performance drastically, which is only slightly higher than random guess (*i.e.,* 10% for MNIST and Fashion-MNIST and 1% for CIFAR100). This is because in our proposed system, the encoding module outputs indistinguishable ciphertexts, which may not be usable for the attacker to launch a successful attack.

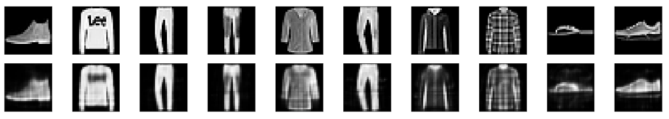
Besides, we compare the ANC system and our proposed system in terms of convergence speed, throughput, and MSE in TABLE 5 and Fig. 9.

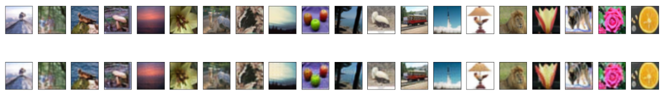
In TABLE 5, the value of convergence represents how many epochs are used to train the system to be converged, *i.e.,* the number of epochs for the loss of encoding module, decoding module, and attack module to become stable, which implies the training efficiency. Our system converges faster than ANC as we analyze in Section 7, because our system treats the encoding module and decoding module identical to reduce the training cost of a neural network. The throughput reflects how many images can be transmitted through the communication system after the training process is completed, indicating the communication efficiency. As shown in TABLE 5, our system's throughput can reach 104.489 images/s, but the ANC system only has 87.074 images/s. These results are resulted from the fact that the networks in our system only use the 3 fully-connected layers, which is simpler than the network setting of the ANC system. The communication quality of system is measured by mean square error (MSE), in which a smaller MSE between the input image at the sender and the decoded image at the receiver represent a better communication quality. From TABLE 5, one can find that our system has a little bit larger MSE compared with ANC. But, this difference is not quite obvious from the aspect of human eyes visualization as shown in Fig. 9, after all our system provides a stronger security protection to defend attacks. Moreover,


(a) The communication results of ANC on MNIST dataset


(b) The communication results of our system on MNIST dataset


(c) The communication results of ANC on Fashion-MNIST dataset


(d) The communication results of our system on Fashion-MNIST dataset


(e) The communication results of ANC on CIFAR100 dataset

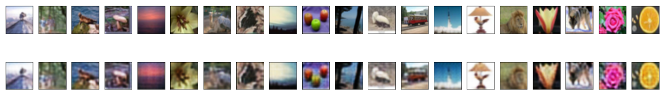

(f) The communication results of our system on CIFAR100 dataset

Fig. 9: Visualized comparison between ANC and our system. In each subfigure, the first row shows encoding module's original plaintext p, and the second row shows decoding module's decoded result p_{Bob}.

TABLE 5: Performance comparison between ANC and our system in terms of convergence, throughput, and MSE

	Convergence	Throughput	MSE
ANC [5]	1.701×10^4	87.074	0.491
Our System (key-256)	1.274×10^4	104.489	0.627
Our System (key-128)	1.227×10^4	109.568	0.586

security and data quality could be a customizable trade-off by adjusting the key length. TABLE 5 demonstrates that adopting a shorter key size gains higher convergence speed, larger throughput, and smaller error.

In each subfigure of Fig. 9, the first low shows the original input images, and the second row shows the decoded images. In both ANC and our proposed defense system, the decoded ciphertext is almost same as the original plaintext though the images output by our system are slightly blurred. Different system frameworks between ANC and our system can account for this difference. The optimization of the encoding and decoding modules in our system (see Eq. (8)) considers not only the communication loss for but also the defensive loss to defend chosen plaintext attack. This extra defensive loss yielded by the attack module Eve may bring uncertain influence on the training process of the encoding and decoding modules, resulting in lower quality. Whereas, in ANC, the loss function of communication quality only considers the difference between original and decoded plaintext, which enforces the training focus on communication quality. Nonetheless, such little quality loss is trivial and can be ignored in our system.

From the above comprehensive evaluation, we can conclude that our proposed defense system provides secure, efficient, and high-quality communication between sender and receiver.

9 Conclusion

In this work, we investigate the weakness of existing neural communication system through deep analysis and real data experiment; especially, we design two adversarial attacks, including target man-in-the-middle attack and target fraud attack to invade the system. Experimental results illustrate that our proposed attack methods successfully perform desired attack purpose. Then, to improve the neural network-based communication systems, we develop a new neural communication system that can mitigate the impacts of the proposed attacks and enhance communication efficiency and communication quality. The experimental results can well validate the superiority of our proposed neural communication system over the state-of-the-art.

Acknowledgments

This work was partially supported by the National Science Foundation of U.S. (No. 1912753, No. 2011845, No. 2054968, and No. 2118083).

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[2] A. Ruttor, "Neural Synchronization and Cryptography," Ph.D. dissertation, -, Nov. 2007.

[3] W. Kinzel and I. Kanter, "Neural cryptography," in *Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP'02.*, vol. 3. IEEE, 2002, pp. 1351–1354.

[4] A. Klimov, A. Mityagin, and A. Shamir, "Analysis of neural cryptography," in *Proceedings of the 8th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*, ser. ASIACRYPT '02. Berlin, Heidelberg: Springer-Verlag, 2002, p. 288–298.

[5] M. Abadi and D. G. Andersen, "Learning to protect communications with adversarial neural cryptography," *arXiv preprint arXiv:1610.06918*, 2016.

[6] Z. Xiong, W. Li, Q. Han, and Z. Cai, "Privacy-preserving auto-driving: a gan-based approach to protect vehicular camera data," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 668–677.

[7] Y. Li, T. Baldwin, and T. Cohn, "Towards robust and privacy-preserving text representations," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 25–30. [Online]. Available: https://www.aclweb.org/anthology/P18-2005

[8] N. Raval, A. Machanavajjhala, and L. P. Cox, "Protecting visual secrets using adversarial nets," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 1329–1332.

[9] I. Kanter, W. Kinzel, and E. Kanter, "Secure exchange of information by synchronization of neural networks," *EPL (Europhysics Letters)*, vol. 57, no. 1, p. 141, 2002.

[10] S. Lian, G. Chen, A. Cheung, and Z. Wang, "A chaotic-neural-network-based encryption algorithm for jpeg2000 encoded images," in *International Symposium on Neural Networks*. Springer, 2004, pp. 627–632.

[11] W. Yu and J. Cao, "Cryptography based on delayed chaotic neural networks," *Physics Letters A*, vol. 356, no. 4-5, pp. 333–338, 2006.

[12] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[13] K. Qin and B. J. Oommen, "On the cryptanalysis of two cryptographic algorithms that utilize chaotic neural networks," *Mathematical Problems in Engineering*, vol. 2015, pp. 1–9, 2015.

[14] Y. Zhang, C. Li, Q. Li, D. Zhang, and S. Shu, "Breaking a chaotic image encryption algorithm based on perceptron model," *Nonlinear Dynamics*, vol. 69, no. 3, pp. 1091–1096, 2012.

[15] P. Xie, M. Bilenko, T. Finley, R. Gilad-Bachrach, K. Lauter, and M. Naehrig, "Crypto-nets: Neural networks over encrypted data," *arXiv preprint arXiv:1412.6181*, 2014.

[16] N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, p. 201–210.

[17] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[18] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rkZvSe-RZ

[19] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*. ACM, 2017, pp. 506–519.

[20] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "A general framework for adversarial examples with objectives," *ACM Transactions on Privacy and Security (TOPS)*, vol. 22, no. 3, p. 16, 2019.

[21] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[22] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.

[23] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2574–2582.

[24] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2016, pp. 372–387.

[25] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 39–57.

[26] Z. Zhao, D. Dua, and S. Singh, "Generating natural adversarial examples," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=H1BLjgZCb

[27] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," *arXiv preprint arXiv:1801.02610*, 2018.

[28] E. Klein, R. Mislovaty, I. Kanter, A. Ruttor, and W. Kinzel, "Synchronization of neural networks by mutual learning and its application to cryptography." in *NIPS*, 2004, pp. 689–696.

[29] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[30] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Chapman Hall/CRC, 2007.

Zuobin Xiong received the B.S. degree from the Department of Mathematics, Northeast Forestry University, Harbin, Heilongjiang, China in 2016, and the M.S. degree from College of Computer Science and Technology, Harbin Engineering University, Harbin, Heilongjiang, China in 2019. Currently, he is pursuing the Ph.D. degree at the Department of Computer Science, Georgia State University, Atlanta, GA, USA. He won the Best Paper Award in the 8th IEEE International Conference on Smart Data in 2022. His research interests lie in a broad area of cybersecurity and privacy, including Privacy-preserving Machine Learning, Privacy-preserving Data Mining, Internet of Things (IoT), Differential Privacy and other privacy and/or security issues of related topics.

Zhipeng Cai is currently a Professor at Department of Computer Science, Georgia State University, USA. He received his PhD and M.S. degrees in the Department of Computing Science at University of Alberta, and B.S. degree from Beijing Institute of Technology. Prior to joining GSU, Dr. Cai was a research faculty in the School of Electrical and Computer Engineering at Georgia Institute of Technology. Dr. Cai's research areas focus on Internet of Things, Machine Learning, Cyber-Security, Privacy, Networking and Big data. Dr. Cai is the recipient of an NSF CAREER Award. He served as a Steering Committee Co-Chair and a Steering Committee Member for WASA and IPCCC. Dr. Cai also served as a Technical Program Committee Member for more than 20 conferences, including INFOCOM, ICDE, ICDCS. Dr. Cai has been serving as an Associate Editor-in-Chief for Elsevier High-Confidence Computing Journal (HCC), and an Associate Editor for more than 10 international journals, including IEEE Internet of Things Journal (IoT-J), IEEE Transactions on Knowledge and Data Engineering (TKDE), IEEE Transactions on Vehicular Technology (TVT).

Chunqiang Hu received the B.S. degree in computer science and technology from Southwest University, Chongqing, China, in 2006, the M.S. and Ph.D. degrees in computer science and technology from Chongqing University, Chongqing, China, in 2009 and 2013, respectively, and the second Ph.D. degree in computer science from The George Washington University, Washington, DC, USA, in 2016. He was a Visiting Scholar with The George Washington University from 2011 to 2012. He is currently a Faculty Member with the School of Big Data & Software Engineering, Chongqing University. His research interests include Blockchain, privacy-aware computing, big data security and privacy, and algorithm design and analysis. He was honored with the Hundred-Talent Program by Chongqing University. He is a senior member of CCF (China Computer Federation) and a member of the ACM.

Daniel Takabi is currently an Associate Professor of computer science and a Next Generation Scholar with Georgia State University, Atlanta, GA, USA. He is also a Founding Director of the Information Security and Privacy: Interdisciplinary Research and Education (INSPIRE) Center, designated as the National Center of Academic Excellence in Cyber Defense Research (CAE-R). His research interests include various aspects of cybersecurity and privacy, including trustworthy AI, privacy-preserving machine learning, adversarial learning, advanced access control models, insider threats, and usable security and privacy.

Wei Li is currently an Associate Professor in the Department of Computer Science at Georgia State University. Dr. Li received her Ph.D. degree in computer science, from The George Washington University, Washington DC, in 2016 and M.S. degree in Computer Science from Beijing University of Posts and Telecommunications, in 2011. She won the Best Paper Awards in ACM MobiCom Workshop CRAB 2013 and international conference WASA 2011, respectively. Her current research spans the areas of blockchain technology, security and privacy for the Internet of Things and Cyber-Physical Systems, secure and privacy-aware computing, Big Data, game theory, and algorithm design and analysis. She is a member of IEEE and a member of ACM.