A Review of Machine Learning and Cryptography Applications

Korn Sooksatra* and Pablo Rivas†, *Senior, IEEE*
School of Engineering and Computer Science
Department of Computer Science, Baylor University
*Email: Korn_Sooksatra1@Baylor.edu, †Email: Pablo_Rivas@Baylor.edu

***Abstract*—Adversarially robust neural cryptography deals with the training of a neural-based model using an adversary to leverage the learning process in favor of reliability and trustworthiness. The adversary can be a neural network or a strategy guided by a neural network. These mechanisms are proving successful in finding secure means of data protection. Similarly, machine learning benefits significantly from the cryptography area by protecting models from being accessible to malicious users. This paper is a literature review on the symbiotic relationship between machine learning and cryptography. We explain cryptographic algorithms that have been successfully applied in machine learning problems and, also, deep learning algorithms that have been used in cryptography. We pay special attention to the exciting and relatively new area of adversarial robustness.**

***Index Terms*—neural cryptography, deep learning, block ciphers, generative adversarial networks, adversarial robustness**

I. Introduction

Cryptography is concerned with studying how we can protect data, offering security against intruders [1]. Much of the security depends on sophisticated means to obtain pseudo-random mappings that are invertible, allowing both encryption and decryption tasks [2]. The strength of cryptographic algorithms often relies on publicizing the algorithms for the community to study for weaknesses and possible attacks so that when a vulnerability is detected, the algorithm can be updated or patched [3]. This public nature of algorithms also has positive effects concerning mass production, implementation, and deployment; however, one important additional aspect that must not be overlooked is trustworthiness: knowing exactly how the algorithm works, or at least knowing that one can have access to that information creates trust [4]. The issue of trust has hindered the progress of machine learning algorithms in the field of cryptography until recently.

The machine learning community has brought a new wave of interest in the field with exciting and innovative research. The problem of trustworthiness is no longer tabu and is the object of much study today [5]. Much progress has been made in the area of explainable AI, which also promotes trust in AI [6]–[8]. This has opened the door for other disciplines to consider machine learning, particularly deep learning models, to address highly complex problems from a different perspective.

In 2016, Abadi *et al.* [9], connected the idea of adversarial learning and neural networks to train a neural model to learn its own encryption mechanism. This research captured the attention of many researchers in both areas, cryptography, and machine learning [10]–[13]. However, while the authors of [9] gathered much attention, they are other lesser-known successful models that need to be put in the proper context to display the problems that have been solved and the challenges that still exist. This paper aims to provide context on recent research in the intersection between cryptography and machine learning.

The paper is organized as follows: Section II describes preliminaries on both deep learning and adversarial neural networks; Section III presents a general introduction to state of the art machine learning algorithms that benefit from cryptography; Section IV cryptography research that benefits directly from machine learning; we present a discussion and future work in Section V and conclusions in Section VI.

II. Preliminaries

A. Deep Learning

A deep learning model is a machine learning model that has a long structure and numerous parameters [14]. Most deep learning models are modeled as neural networks. Layers in this kind of neural network can be homogeneous or not homogeneous. Hence, each layer in the model can be either convolutional, recurrent, or fully-connected. Fig. 1 shows an example of deep learning model consisting of two convolutional layers and one fully-connected layer. Convolutional neural networks (CNNs) have been of particular interest in the deep learning community in different applications [15].

B. Adversarial Neural Network

In 2014, Goodfellow et.al [16], [17] proposed a state-of-art scheme, called Generative Adversarial Networks (GANs) which motivated the idea of adversarial neural network. GANs composed of two neural networks: a generative model (called generator) and a discriminative model (called discriminator). The generator receives a noisy input and generates an image to send to the discriminator. The discriminator receives an image to calculate the probability p_d that the image is real and the probability $p_{d'}$ that the image is generated by the generator. The discriminator's goal is to maximize p_d when the input is real and $p_{d'}$ when the input is not real, and the generator's goal is to minimize the maximization of the discriminator. As a result, after the learning process and competition between both

978-1-7281-7624-6/20/$31.00 ©2020 IEEE
DOI 10.1109/CSCI51800.2020.00105

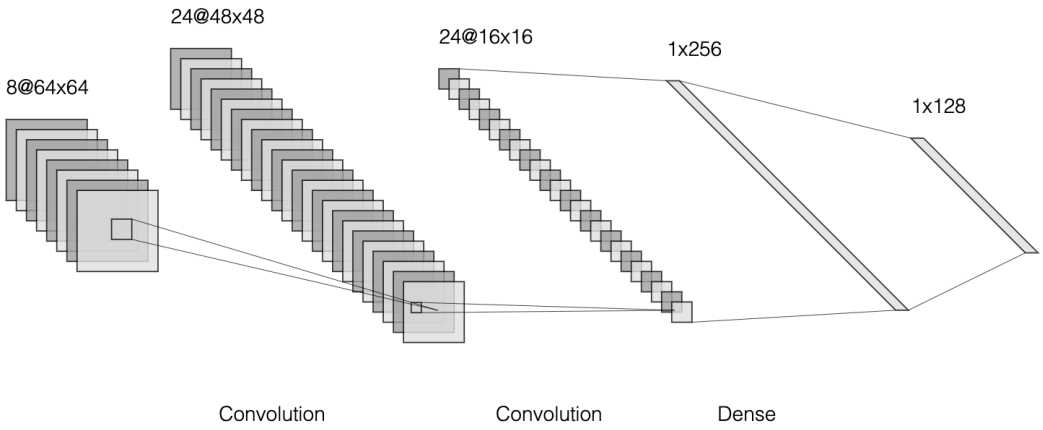

Fig. 1. Example of deep learning model represented as a convolutional neural network.

of them, the generator can generate an image which is almost identical with the real image. Fig. 2 describes the architecture of this scheme.

Existing works applied this idea to let neural network models learn by themselves where there are competitors in the schemes. This is called adversarial neural network and categorized in unsupervised learning.

III. Cryptography for Machine Learning

In some applications, obtaining training data only from one source in a deep learning model is not enough. Thus, many data owners would like to let the model learn their own data and also data from other sources. To successfully achieve the goal, they can have one cloud and let the model in the cloud learn from their data. In addition, they do not want leak their sensitive information, and this scenario is called collaborative deep learning. In this paper, collaborative deep learning can be categorized into two types: **sharing encrypted training data** and **sharing encrypted gradients**. To preserve the data owners' privacy, all the data during the learning process (including operands for operations on the cloud) need to be encrypted, called fully homomorphic encryption [18].

A. Sharing encrypted training data

Li et.al [19] designed two schemes (i.e. *basic scheme* and *advanced scheme*) to preserve privacy for users who desired to collaboratively train their data with deep learning on the cloud, and the users as well as the cloud are honest-but-curious. This means that all the things in the models are honest to perform their duties, but try to reveal information in the learning process. First, *basic scheme* is composed of data owners and the cloud and based on multi-key fully homomorphic encryption (MK-FHE) [20]. The mechanism is listed as below:

1) Data owners generate public keys, secret keys and evaluation keys, encrypt their own data with the public keys (i.e. training data, partial weight and desired target) with his/her own public key and send them to the cloud.
2) The cloud receives encrypted data from all the data owner and let the model learn with the encrypted data using public keys and evaluation keys of the data owners.
3) After updating all the encrypted weights, the cloud sends the encrypted and updated weights back to the data owners.
4) The data owners jointly decrypt it to gain the individual updated weights by using secure multi-party computation (SMC) with their secret keys [21].

Second, *advanced scheme* is more complicated to ensure that the data owners do not need to communicate to each other for decrypting the encrypted results. This scheme consists of data owners, the cloud and authorized center (AU). This scheme applies double encryption, called BCP scheme [22], and MK-FHE, and its mechanism works as follows:

1) Data owners generate their own public keys and secret keys (no evaluation keys), encrypt their own data (i.e. training data, partial weight and desired target) with their own public key and send them to the cloud. Note that AU also has the data owners' secret keys.
2) The cloud receives encrypted data from all the data owner, but it cannot perform addition and multiplication operations since it does not have evaluation keys. Hence, it adds noise to them and sends them to AU.
3) AU decrypts them with the data owners' secret keys and encrypts all the data with one public key. Then, it sends them back to the cloud.
4) Now, the cloud can calculate encrypted and updated weights from all the data encrypted with the same public key. After obtaining the results, the cloud sends the results to AU to transform each result to the data encrypted with the particular data owner's public key.
5) The cloud sends each result back to its data owner, and each data owner can decrypt the result with his/her secret key.

The system was theoretically proved that it is semantically secure [23] if the public key scheme is semantically secure. Also, the system can preserve privacy for the parameters of

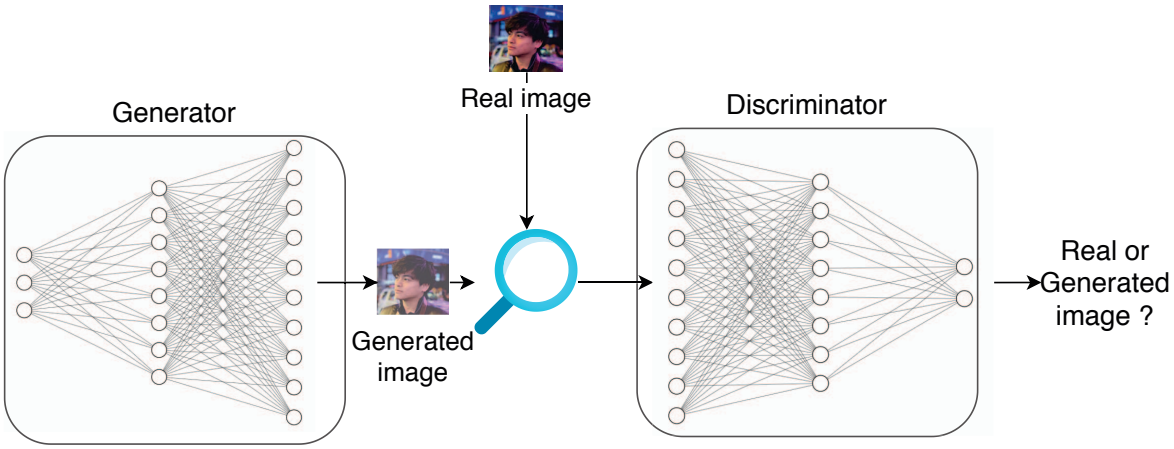

Fig. 2. Generative Adversarial Network's architecture.

deep learning (i.e. the weights) if the cloud does not collude with AU.

In 2019, Kwabena et.al [24] improved *basic scheme* in [19] by applying multi-scheme fully homomorphic encryption (MS-FHE) [25] which is that the data owners can use different encryption schemes to join the collaborative deep learning. Also, the estimated activation function for *Sigmoid* function is more accurate and *Relu* function is estimated for alternative (since a homomorphic encryption cannot perform division operation). Furthermore, this work's scheme has a slightly more accuracy of classification tasks than the ones of previous works [26]–[28] and runs faster than [28].

B. Sharing encrypted gradients

Phong et.al [29] proposed additively homomorphic encryption scheme for collaborative deep learning by improving the work in [30] which applied asynchronous stochastic gradient descent (ASGD) [31], [32] for the learning method and was called gradients-selective ASGD because each data owner chooses which gradients to globally share to preserve his/her privacy. Also, [30] showed additional scheme which leveraged differential privacy by adding Laplace noise [33] to all the gradients. However, Phong and the others had a proof to show that gradients-selective ASGD and differential privacy still leaked some sensitive information of the data owners although these methods slightly altered the values of gradients. In [29], ASGD is also applied for the learning method and the mechanism is summarized as follows:

1) A data owner downloads the encrypted weight stored in the cloud using the secret key.
2) The data owner computes the gradient after using the global weight and the training data to learn in his/her deep learning model.
3) The data owner encrypts the gradient multiplied with the learning rate with his/her secret key and sends back to the cloud.
4) The cloud updates the global weight with the data owners' encrypted message by performing only addition operation.

This work proves that there is no gradient's leakage to the honest-but-curious cloud in this mechanism, and when the data owner decrypts the encrypted result, the decrypted result is the same as if the operations on the cloud were performed with an unencrypted gradient.

IV. Machine Learning for Cryptography

Not only encryption schemes have been used in machine learning models, but also in this decade, machine learning has certainly existed in encryption or cryptography field. The machine learning-based cryptography can be categorized into two types: **non-adversarial-machine-learning cryptography** and **adversarial-machine-learning cryptography**.

A. Non-adversarial-machine-learning cryptography

The prior works [34], [35] leveraged neural network models to create encryption schemes. [34] in 2012 and [35] in 2015 which enhanced the former utilized the mixing method and non linearity property of neural network models to build encryption and decryption schemes. The secret keys of the schemes are the parameters (weights and biases) of the network or mixed with some random numbers. The drawbacks of these kinds of schemes are that if the architectures of the networks are exposed, the schemes will not be secure.

In 2017, Shruti et.al [36] applied a deep learning model to create a symmetric encryption and decryption scheme based on a genetic algorithm [37] and DNA computing. This work includes the key generation part and encryption and decryption part. First, the generic algorithm is applied to perform the key generation, and this algorithm consists of mutation, crossover and selection. Key candidates which are binary strings are initialized, and then mutation is performed to expand the number of candidates by randomly flipping some bits on each string. After that, crossover is performed to also increase the number of candidates by randomly swapping some parts of a

pair of key candidates. Then, the key candidates which results the best fitness functions are obtained. This process keeps going on until the random values reach the threshold.

Next, the encryption scheme can be accomplished by the DNA computing. A bit string of plaintext is converted to a string of DNA by mapping 00 to A (Adenine), 01 to C (Cytosine), 10 to G (Guanine) and 11 to T (Thymine). This DNA string is processed by transcription and translation process [38], and then it is converted back to a binary string. After that, this string is performed by Exclusive OR (XOR) operation with one of the generated key. The decryption process is just the reverse of the encryption process.

B. Adversarial-machine-learning cryptography

In the present, researchers focus on adversarial neural cryptography in which the encryption and decryption algorithms can learn to improve its level of secrecy by themselves where there is an adversary existing in the scheme. In 2016, Abadi et.al [9], [39] utilized Generative Adversarial Networks (GANs) [16] and proposed a deep-learning based approach to provide secrecy for communications with adversarial neural cryptography. A simple scenario consists of Alice, Bob and Eve. Alice communicates with Bob with a symmetric encryption, they desire to have secrecy property, and Eve intercepts the communication and desires to obtain the plaintext from the encrypted message [40]. For instance, as can be seen in Fig. 3, Alice produces ciphertext C from plaintext P with symmetric key K and sends to Bob. At the same time, Eve also receives the ciphertext C. When Bob receives ciphertext C, it tries to decode ciphertext C with key K to recover plaintext P and outputs its recovered plaintext P_{Bob}. Similarly, Eve attempts to recover plaintext P and outputs its recovered plaintext P_{Eve}. Note that Bob has an advantage over Eve since it has key K, but Eve does not.

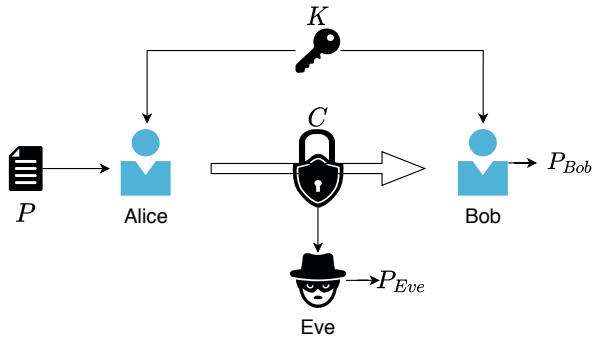

Fig. 3. Alice, Bob and Eve with a symmetric encryption.

The proposed scenario in this work assumes that Alice, Bob and Eve are neural networks where Alice's network, which has θ_A as a set of parameters, attempts to successfully protect C and communicate with Bob, Bob's network, which has θ_B as a set of parameters, tries to fully recover P from C. and Eve's network's goal, which has θ_E as a set of parameters, is to decode C in order to gain P. Therefore, Bob's loss function is defined as

$$L_B = d(P, P_{Bob}),$$

where

$$d(X, Y) = \Sigma_{i=0}^{N-1} |X_i - Y_i|, \tag{1}$$

and X_i is bit i of X. Eve's loss function is defined as

$$L_E = d(P, P_{Eve}),$$

and Alice-Bob's loss function is defined as

$$L_{AB} = L_B + \frac{(N/2 - L_E)^2}{(N/2)^2},$$

where N is the size of plaintext, and the latter component is to ensure that Eve's loss is not different to a random guess's loss.

Note that values in C, P_{Eve} and P_{Bob} can be real values. The goal of Alice and Bob is to minimize the average of L_{AB} of all plaintexts, and the goal of Eve is to minimize the average of L_E of all plaintexts. The input of Alice's network is $P||K$ where $||$ is a concatenation operator, the input of Bob's network is $C||K$, and the input of Eve's network is simply C. Alice's network's architecture starts with a fully-connected (FC) layer, and this is followed by a sequence of one-dimension convolutional layers. Then, it is ended with another FC layer, and the activation function of each layer is $tanh$. Bob's network's architecture is simply the reverse of Alice's, but the activation function of the last layer is $Sigmoid$, and Eve's network's one adds another FC layer at the end to make it stronger than those two networks. Also, the activation function of the last layer is $Sigmoid$. The mechanism is

1) Train Eve's network two rounds with static Alice and Bob to update θ_E.
2) Train Alice and Bob together to update θ_A and θ_B once.
3) Go back to step 1 if it is not enough with respect to a hyper-parameter.

As a result, with 16-bit key, plaintext and ciphertext, Bob's loss and Alice-Bob'loss can be significantly low. On the other hand, Eve's loss is close to the loss of random guess.

Coutinho et.al [41] argued that the work in [9], on the one hand, had some weaknesses in term of strong encryption and decryption's accuracy. On the other hand, they considered two key goals to improve the previous work. First, in practice, the decyprtion algorithm needs to be able to fully recover all the plaintext where the work in [9] could not achieve it; hence, this work tries to achieve this goal. Second, since [9] did not show any detail related to an encryption algorithm of Alice and Bob, there was no information to show how strong the encryption algorithm was. Additionally, Eve in [9] was very weak which lead to a weak encryption algorithm of Alice and Bob. Therefore, this work attempts to lead the neural networks of those to form unbreakable encryption and decryption, called One-Time Pad (OTP) [42] by making Eve stronger. Finally, with the improvement, they proposed Chosen-Plaintext Attack

Adversarial Neural Cryptography (CPA-ANC). To successfully achieve OTP which consists of XOR operations and some permutations, CryptoNet is created. To illustrate CyptoNet, according to [9], let P_i denote bit i of P, K_i denote bit i of K and C_i denote bit i of C. The input of Alice's network is still $P||K$, but each bit i of input is converted to an angle with function

$$f(x) = arccos(1 - 2x),$$

and the network's architecture has only one fully-connected layer. The output of this layer is converted back into bits with function

$$f^{-1}(x) = \frac{1 - cos(x)}{2}.$$

Bob's network is identical with Alice's, but it has $C||K$ as input instead. The reason for all these changes is for achieving OTP. Fig. 4 shows Alice's network's architecture for a better understanding where h_i is neuron i in the output layer.

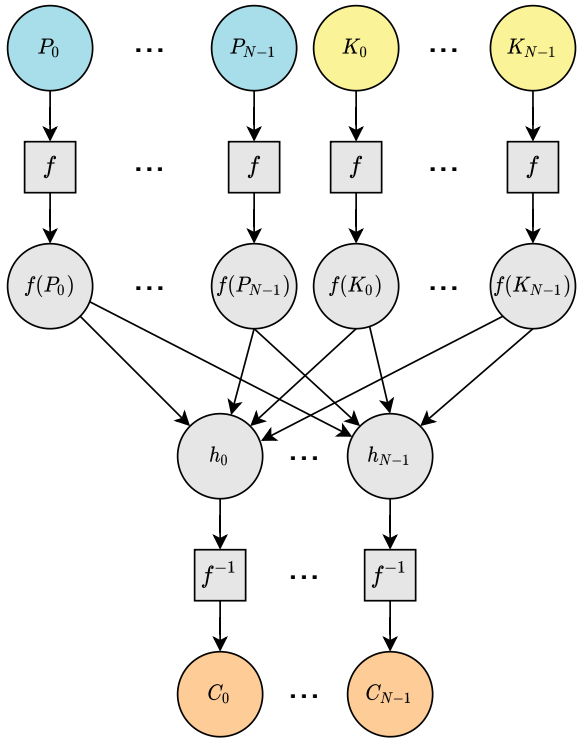

Fig. 4. Alice's CryptoNet.

In this work, there are three stages in this work. In the first stage, there are only Alice's and Bob's networks, and the result shows that Bob and Alice can learn to communicate where Bob is able to fully recover the plaintexts for all 4-bit, 8-bit and 16-bit plaintexts. However, the encryption algorithms that they learn are not secrecy, and, certainly, they are not able to achieve OTP since there is no adversarial neural cryptography or Eve in the scenario. In the second stage, an adversarial neural cryptography or Eve is added into the scenario; hence, the scheme can be identified by Fig. 3. The result of this stage is still not able to achieve OTP, and Alice and Bob cannot learn any strong encryption algorithm because Eve has only C as its input; thus, it is too weak for learning any strong encryption of Alice and Bob. Finally, in the third stage, the scenario is still the same as Fig. 3, but there was two differences. First, Alice's input is from one of a set of two plaintexts, denoted by $S_P = \{P^1, P^2\}$ where P^1 denotes the first plaintext and P^2 denotes the second plaintext, and , second, S_P is Eve's another input; hence, in term of neural network, Eve's input is $P^1||P^2||C$, and has two outputs which are the probability p_0 that C is from P^1 and the probability p_1 that C is from P^2. Therefore, this scheme is called CPA-ANC, and Eve's network is similar to Alice's. It has two hidden layers, and two neuron in the output layers. The first hidden layer has R neurons, and their outputs are converted to angles by function f^{-1}. Then, these values are the inputs of the second hidden layer and later, their outputs are the inputs of two neurons of the output. After that, these two outputs are converted to p_0 and p_1 with $Softmax$ functions. Fig. 5 illustrates the scenario of the third stage. As a result, Alice and Bob can achieve OTP for 58 trials out of 60 trials. Also, Bob can fully recover the plaintexts for 59 trials out of 60 trials. In conclusion, in this work, Alice and Bob can produce strong encryption algorithms, and Bob successfully gains all the bits of plaintexts in almost all cases.

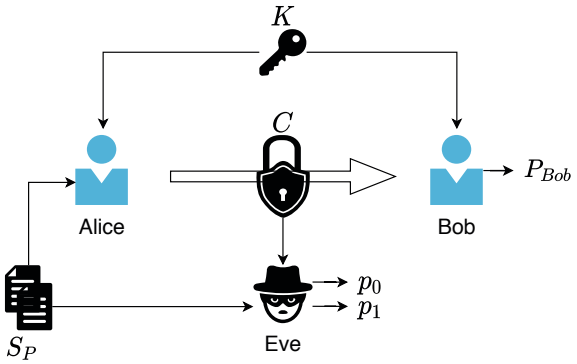

Fig. 5. CPA-ANC scenario.

In 2017, Hayes and Danezis [13] also leveraged GANs and adversarial training idea to generate steganographic images, which were images that embedded messages. Their key idea, which is depicted in Fig. 6, is that it is better when an adversary cannot notice that an encrypted message contains a message. The scenario in this work also has Alice, Bob and Eve. To illustrate it, Alice's input is a message P concatenated with image K and provides the output which is a steganographic image C to Bob and Eve. Then, Bob who also has image K as an input with C, and its output is a decrypted message P_{Bob}. Eve's input is only C, and its output is the confidence score c_{Eve} of how likely the input is a normal image or steganographic image, as illustrated in Fig 6. Analogous to GANs, Alice's network is a generator and Eve's network is a discriminator, but Alice also has another task which is to allow Bob to recover a message in the particular

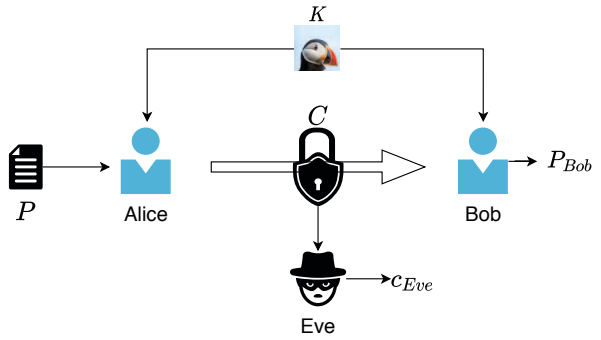

Fig. 6. Alice Bob and Eve with a steganographic image.

steganographic image. Therefore, Bob's loss function is

$$L_{Bob} = d(P, P_{Bob}), \tag{2}$$

where $d(X, Y)$ is as defined in (1) and X_i is bit i of X. Eve's loss function is

$$L_{Eve} = -y \cdot log(c_{Eve}) - (1 - y) \cdot log(1 - c_{Eve}), \tag{3}$$

where y is 1 when the input is K and 0 otherwise. This equation is identical with the discriminator in GANs. Alice's loss function is just a combination of $d(K, C)$, (2) and (3) as

$$L_{Alice} = \lambda_A \cdot d(K, C) + \lambda_B \cdot L_{Bob} + \lambda_E \cdot L_{Eve}, \tag{4}$$

where λ_A, λ_B and λ_E are Alice's weight, Bob's weight and Eve's weight respectively. The first part of (4) indicates that C is not supposed to be much different to K.

This work uses CelebA dataset [43] and BOSS dataset [44] for evaluation. After training Alice, Bob and Eve, Eve's loss is close to a random guess. In the other word, Eve's confidence score is close to 0.5. Also, Bob is able to successfully decode the messages, and the steganographic images still look similar to the original images. However, Alice's result is a little worse than conventional steganographic algorithms, and Eve's result is also slightly worse than a conventional steganalyzer.

V. Discussion and Future Works

Although an encryption scheme (i.e. a homomorphic encryption) has been applied to multiple collaborative deep learning models to preserve data owners' privacy in many existing works, there are still some other aspects that need to be considered for robustness such as an adversarial attack (i.e. an adversarial example). An adversarial attack was explained and constructed in [45]–[48], and [47]–[54] proposed approaches for deep learning model to be robust to adversarial attacks. However, some further works are still required for adversarially robust deep learning models.

On the other hand, whether or not deep learning models and adversarial neural cryptography models can be utilized in practice in encryption schemes is still an open question because non-adversarial deep learning models are not secure enough to be practically used, and adversarial neural cryptography models lack of the mathematical proof showing that encrypted messages can be successfully recovered into the original messages by the receiver. Additionally, the keys and plaintexts in those works are small (i.e. 16-bit); in practice, a key's and plaintext's size can be 256 bits or more.

According to the aforementioned remaining issues, the further works can be summarized as follows:

- Discovering a fully robust deep learning model for adversarial attacks.
- Designing an adversarial neural cryptography model with a mathematical proof for fully recovering encrypted messages.
- Developing an adversarial neural cryptography model which supports larger sizes of key and plaintext.

VI. Conclusion

We have discussed the recent research algorithms in the intersecting areas of cryptography and machine learning. We described cryptography algorithms from which machine learning can benefit, by means of providing privacy or otherwise. Similarly, we summarized approaches where cryptography can benefit from machine learning by modeling pseudo-random invertible functions, complex mappings, and others. Whenever cryptography has leveraged machine learning, new possibilities have been introduced into the field.

The area of adversarial robustness by which one is able to certify that a particular block-cipher deep learning-based model is of much interest within the machine learning community and will not be susceptible to adversarial attacks, either chosen plaintext or ciphertext attacks. This can further the issues associated with explainability and trustworthiness that are crucial in the adoption of deep learning technology in other fields of science.

Neural cryptography can potentially lead to wide-spread data privacy protection, if resources are invested in the areas of rapid development and testing of cryptography protocols and standards. Further, the dramatic decrease in the costs associated with storage make these type of neural models a important alternative to secure data communications.

Acknowledgment

The authors would like to thank the support of the Department of Computer Science at Baylor University during this research study.

References

[1] A. Stanoyevitch, *Introduction to Cryptography with mathematical foundations and computer implementations*. CRC Press, 2010.
[2] N. Ferguson, B. Schneier, and T. Kohno, "Cryptography engineering: Design principles and practical applications," 2010.
[3] T. Ristenpart and S. Yilek, "When good randomness goes bad: Virtual machine reset vulnerabilities and hedging deployed cryptography." in *NDSS*, 2010.
[4] R. Walton, "Cryptography and trust," *information security technical report*, vol. 11, no. 2, pp. 68–71, 2006.
[5] L. Floridi, "Establishing the rules for building trustworthy ai," *Nature Machine Intelligence*, vol. 1, no. 6, pp. 261–262, 2019.
[6] W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller, *Explainable AI: interpreting, explaining and visualizing deep learning*. Springer Nature, 2019, vol. 11700.

[7] R. Goebel, A. Chander, K. Holzinger, F. Lecue, Z. Akata, S. Stumpf, P. Kieseberg, and A. Holzinger, "Explainable ai: the new 42?" in *International cross-domain conference for machine learning and knowledge extraction*. Springer, 2018, pp. 295–303.

[8] A. Holzinger, "From machine learning to explainable ai," in *2018 World Symposium on Digital Intelligence for Systems and Machines (DISA)*. IEEE, 2018, pp. 55–66.

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

[10] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[11] S. Dörner, S. Cammerer, J. Hoydis, and S. Ten Brink, "Deep learning based communication over the air," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 132–143, 2017.

[12] C. Huang, P. Kairouz, X. Chen, L. Sankar, and R. Rajagopal, "Context-aware generative adversarial privacy," *Entropy*, vol. 19, no. 12, p. 656, 2017.

[13] J. Hayes and G. Danezis, "Generating steganographic images via adversarial training," in *Advances in Neural Information Processing Systems*, 2017, pp. 1954–1963.

[14] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1, no. 2.

[15] K. Mulligan and P. Rivas, "Dog breed identification with a neural network over learned representations from the xception cnn architecture," in *21st International Conference on Artificial Intelligence (ICAI 2019)*, 2019.

[16] 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*, 2014, pp. 2672–2680.

[17] ——, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[18] C. Gentry and D. Boneh, *A fully homomorphic encryption scheme*. Stanford university Stanford, 2009, vol. 20, no. 9.

[19] P. Li, J. Li, Z. Huang, T. Li, C.-Z. Gao, S.-M. Yiu, and K. Chen, "Multi-key privacy-preserving deep learning in cloud computing," *Future Generation Computer Systems*, vol. 74, pp. 76–85, 2017.

[20] A. López-Alt, E. Tromer, and V. Vaikuntanathan, "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption," in *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, 2012, pp. 1219–1234.

[21] O. Goldreich, "Secure multi-party computation," *Manuscript. Preliminary version*, vol. 78, 1998.

[22] E. Bresson, D. Catalano, and D. Pointcheval, "A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2003, pp. 37–54.

[23] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of computer and system sciences*, vol. 28, no. 2, pp. 270–299, 1984.

[24] O.-A. Kwabena, Z. Qin, T. Zhuang, and Z. Qin, "Mscryptonet: Multi-scheme privacy-preserving deep learning in cloud computing," *IEEE Access*, vol. 7, pp. 29 344–29 354, 2019.

[25] Z. Li and T.-H. Lai, "On evaluating circuits with inputs encrypted by different fully homomorphic encryption schemes." *IACR Cryptol. ePrint Arch.*, vol. 2013, p. 198, 2013.

[26] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 19–38.

[27] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff, "Privacy-preserving classification on deep neural network." *IACR Cryptol. ePrint Arch.*, vol. 2017, p. 35, 2017.

[28] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *International Conference on Machine Learning*, 2016, pp. 201–210.

[29] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1333–1345, 2018.

[30] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015, pp. 1310–1321.

[31] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang *et al.*, "Large scale distributed deep networks," in *Advances in neural information processing systems*, 2012, pp. 1223–1231.

[32] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild: A lock-free approach to parallelizing stochastic gradient descent," in *Advances in neural information processing systems*, 2011, pp. 693–701.

[33] R. Sarathy and K. Muralidhar, "Evaluating laplace noise addition to satisfy differential privacy for numeric data." *Trans. Data Priv.*, vol. 4, no. 1, pp. 1–17, 2011.

[34] E. Volna, M. Kotyrba, V. Kocian, and M. Janosek, "Cryptography based on neural network." in *ECMS*, 2012, pp. 386–391.

[35] H. Noura, A. E. Samhat, Y. Harkouss, and T. A. Yahiya, "Design and realization of a new neural block cipher," in *2015 International Conference on Applied Research in Computer Science and Engineering (ICAR)*. IEEE, 2015, pp. 1–6.

[36] S. Kalsi, H. Kaur, and V. Chang, "Dna cryptography and deep learning using genetic algorithm with nw algorithm for key generation," *Journal of medical systems*, vol. 42, no. 1, p. 17, 2018.

[37] D. Whitley, "A genetic algorithm tutorial," *Statistics and computing*, vol. 4, no. 2, pp. 65–85, 1994.

[38] Q. Limin, "The study of dna-based encryption method [d]," *Zheng Zhou: Zheng Zhou University of Light Industry*, 2008.

[39] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 308–318.

[40] M. Abadi, U. Erlingsson, I. Goodfellow, H. B. McMahan, I. Mironov, N. Papernot, K. Talwar, and L. Zhang, "On the protection of private information in machine learning systems: Two recent approches," in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. IEEE, 2017, pp. 1–6.

[41] M. Coutinho, R. de Oliveira Albuquerque, F. Borges, L. J. Garcia Villalba, and T.-H. Kim, "Learning perfectly secure cryptography to protect communications with adversarial neural cryptography," *Sensors*, vol. 18, no. 5, p. 1306, 2018.

[42] C. E. Shannon, "Communication theory of secrecy systems," *The Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.

[43] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[44] [Online]. Available: http://agents.fel.cvut.cz/boss/index.php?mode=VIEW&tmpl=materials

[45] 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.

[46] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," in *Advances in Neural Information Processing Systems*, 2019, pp. 125–136.

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

[48] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 9, pp. 2805–2824, 2019.

[49] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," *arXiv preprint arXiv:1412.5068*, 2014.

[50] A. Raghunathan, J. Steinhardt, and P. Liang, "Certified defenses against adversarial examples," *arXiv preprint arXiv:1801.09344*, 2018.

[51] S. Gowal, K. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, R. Arandjelovic, T. Mann, and P. Kohli, "On the effectiveness of interval bound propagation for training verifiably robust models," *arXiv preprint arXiv:1810.12715*, 2018.

[52] E. Wong and Z. Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5286–5295.

[53] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[54] M. Mirman, M. Fischer, and M. Vechev, "Distilled agent dqn for provable adversarial robustness," *International Conference on Learning Representations (ICLR) Conference*, 2019.