Neural Cryptography Based on Complex-Valued Neural Network

Tao Dong and Tingwen Huang, *Fellow, IEEE*

***Abstract*— Neural cryptography is a public key exchange algorithm based on the principle of neural network synchronization. By using the learning algorithm of a neural network, the two neural networks update their own weight through exchanging output from each other. Once the synchronization is completed, the weights of the two neural networks are the same. The weights of the neural network can be used for the secret key. However, all the existing works are based on the real-valued neural network model. There are seldom works studying the neural cryptography based on a complex-valued neural network model. In this technical note, a neural cryptography based on the complex-valued tree parity machine network (CVTPM) is proposed. The input, output, and weights of CVTPM are a complex value, which can be considered as an extension of TPM. There are two advantages of the CVTPM: 1) the security of CVTPM is higher than that of TPM with the same hidden units, input neurons, and synaptic depths and 2) the two parties with the CVTPM can exchange two group keys in one neural synchronization process. A series of numerical simulation experiments is provided to verify our results.**

***Index Terms*— Complex valued, neural cryptography, neural synchronization, tree parity machine (TPM).**

I. Introduction

The public key exchange algorithm proposed by Diffie and Hellman [1] is a security protocol. It allows two parties A and B to create a secret key over an insecure channel through exchanging the information from each other. The secret key can be applied in many fields, such as identity authentication, data encryption, data privacy protection, and so on.

In modern cryptography, a number of cryptography algorithms are based on algebraic number theory [1]–[4]. It is a great challenge to propose a novel secret key framework, while the attacker E cannot infer the key even though it knows the details of algorithm framework and has the capacity to monitor the communication channel. The synchronization phenomenon of neural networks presents an opportunity to solve the key exchange issues [5]–[13]. Recently, with the in-depth research of an artificial neural network [14], [15], neural synchronization has been acknowledged to be capable of achieving this goal, which is called "neural cryptography" [16]. The principle of neural cryptography has a starting likeness to that of "secret key agreement through public discussion" [17], which allows two parties A and B to create a secret key over an insecure channel through exchanging the information from each other, while the attacker E cannot infer the key even though it knows the details of the algorithm framework and has the capacity to monitor the communication channel. Particularly, since the process of neural synchronization does not require a large number of computing resources, neural cryptography is appropriate for many applications, such as wireless sensor networks, UAV, clouding computing system, and so on [18], [19]. Consequently, neural cryptography has received a number of attentions.

There are a number of research efforts in neural cryptography [20]–[32]. Rosen *et al.* [20], Ruttor *et al.* [21], Metzler *et al.* [22], and Kinzel and Kanter [23] have proven that the synchronization of tree parity machines (TPMs) can be achieved by using the Hebbian learning rules. In [24], the synchronization process of neural networks based on TPM is firstly used in cryptography to deal with the problem of key exchange. Shacham *et al.* [25], Ruttor *et al.* [26], and Klimov *et al.* [27] have proposed four types of attack algorithms to attack the neural cryptography system. Ruttor [28] and Kinzel and Kanter [29] have found that TPM with one or two hidden units is insecurity against the attack. TPM with three hidden units can improve the security of neural synchronization to any level by increasing the synaptic depth of its network. Alternatively, there is another approach to enhance the security of neural synchronization. Mu and Liao [30] have proposed a novel network framework, which is called "tree state classification machine (TSCM)." Lei and Liao [31] have proposed a two-layer tree-connected feed-forward neural network (TTFNN) model. In [32], an algorithm called "Don't Trust My Partner (DTMP)" based on TPM is presented. In this algorithm, part A sends some error bits to part B, while part B has the ability to predict and correct the errors. Experiments show that this algorithm can significantly improve the security.

However, all the existing works are based on the real-valued neural network model. A few works consider the complex-valued neural network model. Comparing with the real-valued neural networks, the complex-valued neural network has a great advantage in signal processing since its state is complex. As a result, it is feasible to design the neural cryptography by using the complex neural networks. In this brief, we proposed a neural cryptography based on the complex-valued tree parity machine network (CVTPM). The input, output, and weights of the CVTPM are complex values, which can be considered as an extension of TPM. There are two advantages of the CVTPM that are as follows.

1) The security of the CVTPM is better than that of the TPM with the same K, N, and L.
2) The two parties with the CVTPM can exchange two group keys in one neural synchronization process.

This brief is organized as follows. In Section II, a novel model based on the complex-valued neural network is proposed. In Section III, the security of the CVTPM is investigated. In Section IV, the simulations illustrate that the CVTPM can improve the security.

II. Model Description

A. Network

The architecture of the CVTPM is shown in Fig. 1, which is a tree neural network. Similar to the TPM, there are three layers of

Manuscript received August 27, 2018; revised April 8, 2019, September 29, 2019, and November 5, 2019; accepted November 19, 2019. Date of publication December 23, 2019; date of current version October 29, 2020. This work was supported in part by the National Key Research and Development Project of China under Grant 2018AAA0100101, in part by the Chongqing Technological Innovation and Application Project under Grant cstc2018jszx-cyzdX0171, and in part by Chongqing Basic and Frontier Research Project under Grant cstc2019jcyj-msxm2105. *(Corresponding author: Tao Dong.)*

T. Dong is with the College of Electronics and Information Engineering, Southwest University, Chongqing 400715, China, with the Information Center, Chongqing Changan Automobile Company Ltd., Chongqing 401220, China, and also with the Chongqing Key Laboratory of Nonlinear Circuits and Intelligent Information Processing, Southwest University, Chongqing 400715, China (e-mail: david_312@126.com).

T. Huang is with the College of Electronics and Information Engineering, Texas A&M University at Qatar, Doha 23874, Qatar.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2019.2955165

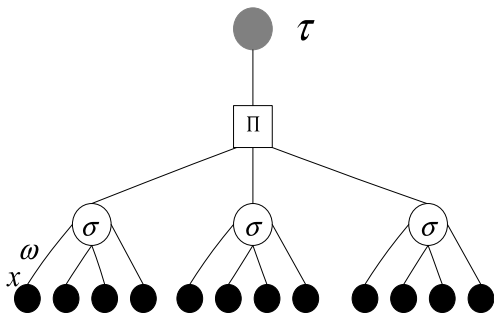

Fig. 1. CVTPM network with K = 3 and N = 4.

CVTPM: input layer, hidden layer, and output layer. The input layer has $K \times N$ input neurons. The hidden layer has K hidden neurons. Each neuron can be considered as an independent perceptron. The output layer has only one output neuron. The weight of CVTPM is a complex value, which can be defined as follows:

$$\omega_{k,j} = a_{k,j} + b_{k,j} i \tag{1}$$

where $a_{k,j} \in \{-L, -L+1, \ldots, L\}$, $b_{k,j} \in \{-L, -L+1, \ldots, L\}$, the index $k = 1, 2, \ldots M$ denotes the kth hidden unit of the network, $j = 1, 2, \ldots, N$ denotes the elements of vectors, and L is the synaptic depth of the CVTPM.

Let $x_{k,j}$ be the input vector, which is defined as follows:

$$x_{k,j} = a_x + b_x i \tag{2}$$

where $a_x \in \{-1, 1\}$, $b_x \in \{-1, 1\}$.

The kth hidden unit can be defined by

$$h_k = \frac{1}{\sqrt{N}} \left(\sum_{j=1}^{N} (a_{k,j} a_x) + i \sum_{j=1}^{N} (b_{k,j} b_x) \right) \tag{3}$$

and

$$\sigma_i = a_{\sigma_i} + b_{\sigma_i} i \tag{4}$$

where $a_{\sigma_i} = \mathrm{sgn}(\mathrm{Re}(h_i))$ and $a_{\sigma_i} = \mathrm{sgn}(\mathrm{Im}(h_i))$.

The output of the CVTPM τ can be defined as follows:

$$\tau = \prod_{I=1}^{k} \mathrm{Re}(\sigma_i) + \prod_{I=1}^{k} \mathrm{Im}(\sigma_i). \tag{5}$$

Remark 1: If $b_x = 0$ and $b_\omega = 0$, and $b_\sigma = 0$, then the CVTPM becomes the TPM, so the TPM can be regarded as a special case of our model.

Remark 2: When the two networks are said to be synchronized, they have the same weight, which can be used as a secret key for many subsequent cryptographic purposes. Since the weight of the CVTPM is a complex value, which is comprised of the real and imaginary parts, the CVTPM can exchange two group keys in one synchronization process.

Remark 3: The kth hidden unit can be defined by

$$h_k = \frac{1}{\sqrt{N}} \left(\sum_{j=1}^{N} (a_{k,j} a_x - b_{k,j} b_x) + i \sum_{j=1}^{N} (a_{k,j} b_x + b_{k,j} a_x) \right)$$

and

$$\sigma_i = a_{\sigma_i} + b_{\sigma_i} i$$

where $a_{\sigma_i} = \mathrm{sgn}(\mathrm{Re}(h_i))$ and $a_{\sigma_i} = \mathrm{sgn}(\mathrm{Im}(h_i))$. Comparing with (3) and (4), this hidden unit cannot improve the security but increases the time of synchronization process due to $b_\omega b_x$ and $a_\omega b_x$ can lead to the repulsive step increase in the process of synchronization.

B. Learning Rules

Suppose two partners A and B want to negotiate a secret key over a public channel. Both A and B are equipped with the CVTPM. Then, we give the mutual learning algorithm of the CVTPM, which is as follows.

1) The A's CVTPM and B's CVTPM initialize their weights ω_i^A and ω_i^B with the random complex value independently.
2) At each step, A and B receive a common input vector x_{ij} which is generated randomly and compute the output $\tau^{A/B}$ by (3)–(5). Then, A and B exchange τ_A and τ_B with each other on the public channel.
3) When A (B) receives $\tau_B(\tau_A)$, then A (B) updates the weight by the following rules.
   a) If $\tau_{re}^A \neq \tau_{re}^B$ and $\tau_{im}^A \neq \tau_{im}^B$, then weight $\omega_i^{A/B}$ cannot be updated.
   b) If $\tau_{re}^A = \tau_{re}^B = \mathrm{Re}(\sigma_i^{A/B})$ and $\tau_{im}^A = \tau_{im}^B = \mathrm{Im}(\sigma_i^{A/B})$, then update the weight $\omega_{i,j}$ of $\sigma_i^{A/B}$.
   c) If $\tau_{re}^A = \tau_{re}^B = \mathrm{Re}(\sigma_i^{A/B})$ and $\tau_{im}^A = \tau_{im}^B \neq \mathrm{Im}(\sigma_i^{A/B})$, then only update the real part of weight $\omega_{i,j}$ of $\sigma_i^{A/B}$.
   d) If $\tau_{re}^A = \tau_{re}^B \neq \mathrm{Re}(\sigma_i^{A/B})$ and $\tau_{im}^A = \tau_{im}^B = \mathrm{Im}(\sigma_i^{A/B})$, then only update the imaginary parts of weight $\omega_{i,j}$ of $\sigma_i^{A/B}$.
4) One of the following learning rules is used to update the weight.
   a) Hebbian learning rules

$$\begin{cases} \omega_{i,j}^{R+} = g\left(\omega_{i,j}^R + \left(\omega_{i,j}^R \tau_{re}\right)\Theta\left(\sigma_i^R \tau_{re}\right)\Theta\left(\tau_{re}^A \tau_{re}^B\right)\right) \\ \omega_{i,j}^{I+} = g\left(\omega_{i,j}^I + \left(\omega_{i,j}^I \tau_{im}\right)\Theta\left(\sigma_i^I \tau_{im}\right)\Theta\left(\tau_{im}^A \tau_{im}^B\right)\right). \end{cases}$$

   b) Anti-Hebbian learning rules

$$\begin{cases} \omega_{i,j}^{R+} = g\left(\omega_{i,j}^R - \left(\omega_{i,j}^R \tau_{re}\right)\Theta\left(\sigma_i^R \tau_{re}\right)\Theta\left(\tau_{re}^A \tau_{re}^B\right)\right) \\ \omega_{i,j}^{I+} = g\left(\omega_{i,j}^I - \left(\omega_{i,j}^I \tau_{im}\right)\Theta\left(\sigma_i^I \tau_{im}\right)\Theta\left(\tau_{im}^A \tau_{im}^B\right)\right). \end{cases}$$

   c) Random walk learning rules

$$\begin{cases} \omega_{i,j}^{R+} = g\left(\omega_{i,j}^R + \left(\omega_{i,j}^R\right)\Theta\left(\sigma_i^R \tau_{re}\right)\Theta\left(\tau_{re}^A \tau_{re}^B\right)\right) \\ \omega_{i,j}^{I+} = g\left(\omega_{i,j}^I + \left(\omega_{i,j}^I\right)\Theta\left(\sigma_i^I \tau_{im}\right)\Theta\left(\tau_{im}^A \tau_{im}^B\right)\right) \end{cases}$$

   where $\sigma_i^R = \mathrm{Re}(\sigma_i)$, $\sigma_i^I = \mathrm{Im}(\sigma_i)$, $\omega_{i,j}^{I+} = \mathrm{Im}(\omega_{i,j}^+)$, $\omega_{i,j}^{R+} = \mathrm{Re}(\omega_{i,j}^+)$, and function $g(\omega)$ is used to guarantee that each element of $\omega^{A/B}$ is in the range $[-L, +L]$, which is defined as follows:

$$g(\omega) = \begin{cases} \mathrm{sgn}(\omega) L, & \text{for } |\omega| > L \\ \omega, & \text{otherwise.} \end{cases}$$

5) Repeating procedures 2–4 until synchronization ($\omega^A = \omega^B$) is achieved. The final identical weighted vector $\omega^{A/B}$ can be used as the common secret key between A and B.

While in the whole synchronization process, the state vector $\sigma^{A/B}$ is consistently inaccessible to any others. So, four possible update behaviors can be defined as follows.

   a) The real part of the attractive step $\tau_{re}^A = \tau_{re}^B = \mathrm{Re}(\sigma_i^{A/B})$: the real parts of $\omega^{A/B}$ are updated in the same direction. The synchronization of the real part of weight can be achieved eventually through a series of such step.
   b) The imaginary part of the attractive step $\tau_{im}^A = \tau_{im}^B = \mathrm{Im}(\sigma_i^{A/B})$: the imaginary part of $\omega^{A/B}$ is updated in the same direction. The synchronization of the imaginary part

of weight can be achieved eventually through a series of such step.

c) The real part of the repulsive step $\tau_{re}^{A} = \tau_{re}^{B}$, $\mathrm{Re}(\sigma_i^{A}) \neq \mathrm{Re}(\sigma_i^{B})$: only one real part of $\omega^{A/B}$ is updated, while B or A is unchanged. The real part synchronization speed may reduce after a series of these repulsive steps.

d) The imaginary part of the repulsive step $\tau_{im}^{A} = \tau_{im}^{B}$, $\mathrm{Im}(\sigma_i^{A}) \neq \mathrm{Im}(\sigma_i^{B})$: only one imaginary part of $\omega^{A/B}$ is updated, while B or A is unchanged. The imaginary part synchronization speed may reduce after a series of these repulsive steps.

III. Study on Security for CVTPM

In this section, we investigate the security of the CVTPM. According to [28], a secure neural cryptography needs to satisfy the following conditions.

1) Increasing the synaptic depth L of the CVTPM, the average time of synchronization between two neural networks grows at a polynomial rate.
2) The average time of synchronization of E with the same network architecture and learning algorithm grows at an exponential rate.

According to [28], the neural cryptography security of the CVTPM is influenced by the dynamics of a synchronization process of CVTPM. The synchronization level of the CVTPM can be defined by the overlap between the two corresponding hidden units. By (3), one can see the both a_{σ_i} and b_{σ_i} are all comprised of the real part of weight and the imaginary part of weight, which means the overlap of σ_i is comprised of the overlap of the real part of weight and the overlap of the imaginary part of weight. Following [30] and [21], we define ρ is the overlap of σ_i, which are of the form:

$$\rho = \frac{\rho_{re} + \rho_{im}}{2} \tag{6}$$

where

$$\rho_{re} = \frac{\mathrm{Re}(\omega_i^{A})\mathrm{Re}(\omega_i^{B})}{\sqrt{\mathrm{Re}(\omega_i^{A})\mathrm{Re}(\omega_i^{A})}\sqrt{\mathrm{Re}(\omega_i^{B})\mathrm{Re}(\omega_i^{B})}}, \quad \rho_{re} \in [0,1]$$

$$\rho_{im} = \frac{\mathrm{Im}(\omega_i^{A})\mathrm{Im}(\omega_i^{B})}{\sqrt{\mathrm{Im}(\omega_i^{A})\mathrm{Im}(\omega_i^{A})}\sqrt{\mathrm{Im}(\omega_i^{B})\mathrm{Im}(\omega_i^{B})}}, \quad \rho_{im} \in [0,1]$$

$$\mathrm{Re}(\omega_i) = (\mathrm{Re}(\omega_{i,1}), \mathrm{Re}(\omega_{i,2}), \ldots, \mathrm{Re}(\omega_{i,n}))$$

$$\mathrm{Im}(\omega_i) = (\mathrm{Im}(\omega_{i,1}), \mathrm{Im}(\omega_{i,2}), \ldots, \mathrm{Im}(\omega_{i,n})).$$

At the beginning of synchronization, since the initial weighted vectors of A and B are random, one can obtain $\rho = 0$. The synchronization of A and B can be achieved after a series of learning steps, which means $\rho = 1$. The probability of the real part is defined by ε_{re}

$$\varepsilon_{re} = \frac{1}{\pi}\arccos(\rho_{re}) \tag{7}$$

and the probability of the imaginary part is defined by ε_{im}

$$\varepsilon_{im} = \frac{1}{\pi}\arccos(\rho_{im}). \tag{8}$$

In order to investigate the dynamics of the synchronization process, we need to define the average change of the real overlap and imaginary overlap in each learning step. Let $\langle \Delta\rho(\rho_{re})\rangle$ be the real change overlap, which is defined as follows:

$$\langle \Delta\rho(\rho_{re})\rangle = P_a(\rho_{re})\langle \Delta\rho_a(\rho_{re})\rangle + P_r(\rho_{re})\langle \Delta\rho_r(\rho_{re})\rangle. \tag{9}$$

Let $\langle \Delta\rho(\rho_{im})\rangle$ be the imaginary change overlap, which can be defined as follows:

$$\langle \Delta\rho(\rho_{im})\rangle = P_a(\rho_{im})\langle \Delta\rho_a(\rho_{im})\rangle + P_r(\rho_{im})\langle \Delta\rho_r(\rho_{im})\rangle \tag{10}$$

where $P_a(\cdot)(P_r(\cdot))$ is the event probability that an attractive (repulsive) step occurs; $\Delta\rho(\cdot)$ is the average change size in each attractive (repulsive) step.

Otherwise, the synchronization time grows at an exponential rate. Since the weight of CVTPM is a complex value, one can obtain that the security of neural cryptography is achieved, if and only if the following holds.

1) For A and B, in the synchronization process, one has

$$\Delta\rho(\rho) > 0. \tag{11}$$

2) For E, there exists $G \in (0, 1)$, such that

$$\Delta\rho(\rho) < 0. \tag{12}$$

In this brief, condition (2) will not be considered. We only focus on condition (1). First, we investigate the real part, following condition (1), and we can obtain:

$$P_a(\rho_{re})\langle \Delta\rho_a(\rho_{re})\rangle + P_r(\rho_{re})\langle \Delta\rho_r(\rho_{re})\rangle > 0. \tag{13}$$

According to [30], we know $\Delta\rho_a(\rho_{re})$ and $\Delta\rho_r(\rho_{re})$ are only related to the motion equations; therefore, as long as the motion equation is invariant, both $\Delta\rho_a(\rho_{re})$ and $\Delta\rho_r(\rho_{re})$ are fixed. Thus, (13) can be rewritten as follows:

$$-\frac{\Delta\rho_a(\rho_{re})}{\Delta\rho_r(\rho_{re})} > \frac{P_r(\rho_{re})}{P_a(\rho_{re})}, \quad \rho_{re} \in (0, 1). \tag{14}$$

Let

$$U(\rho_{re}) = -\frac{\Delta\rho_a(\rho_{re})}{\Delta\rho_r(\rho_{re})} \tag{15}$$

and

$$R(\rho_{re}) = \frac{P_r(\rho_{re})}{P_a(\rho_{re})}. \tag{16}$$

From [28] and [30], we can obtain $U(\rho_{re}) \sim (7/12)\pi^2\varepsilon^2$. It is easy to see that $U(\rho_{re})$ is constant. Thus, one can obtain that the security of neural cryptography is dependent on $R(\rho_{re})$. Substituting (7) into (16), we can obtain

$$R(\rho_{re}) = \frac{P_r(\varepsilon_{re})}{P_a(\varepsilon_{re})}. \tag{17}$$

$P_a(\varepsilon_{re})$ and $P_r(\varepsilon_{re})$ are as follows:

$$P_a(\varepsilon_{re}) = P(\theta|\mathrm{Re}(\tau^{A}) = \mathrm{Re}(\tau^{B})) \tag{18}$$

$$P_r(\varepsilon_{re}) = P\left(\mathrm{Re}(\sigma_i^{A}) \neq \mathrm{Re}(\sigma_i^{B})|\mathrm{Re}(\tau^{A}) = \mathrm{Re}(\tau^{B})\right) \tag{19}$$

$$P_u(\varepsilon_{re}) = P(\mathrm{Re}(\tau^{A}) = \mathrm{Re}(\tau^{B})) \tag{20}$$

where θ is $\mathrm{Re}(\tau^{A}) = \mathrm{Re}(\sigma_i^{A}) = \mathrm{Re}(\sigma_i^{B})$.

As the real and imaginary parts of hidden units are independent of each other, we can compute the probability that an attractive (repulsive) step occurs by using the method of [30]

$$P_a(\varepsilon_{re}) = \frac{1}{2P_u}\sum_{i=0}^{(K-1)/2}\binom{K-1}{2i}(1-\varepsilon_{re})^{K-2i}\varepsilon_{re}^{2i} \tag{21}$$

$$P_r(\varepsilon_{re}) = \frac{1}{P_u}\sum_{i=1}^{K/2}\binom{K-1}{2i-1}(1-\varepsilon_{re})^{K-2i}\varepsilon^{2i}. \tag{22}$$

Suppose K = 3, we can get

$$P_a(\varepsilon_{re}) = \frac{\frac{1}{2}(1-\varepsilon_{re})^3 + \frac{1}{2}(1-\varepsilon_{re})\varepsilon_{re}^2}{P_u(\varepsilon_{re})} \tag{23}$$

$$P_r(\varepsilon_{re}) = \frac{2(1-\varepsilon_{re})\varepsilon_{re}^2}{P_u(\varepsilon_{re})}. \tag{24}$$

Substituting (7) into (16), we can get

$$R(\varepsilon_{re}) = \frac{P_r(\varepsilon_{re})}{P_a(\varepsilon_{re})} = \frac{4\varepsilon_{re}^2}{(1-\varepsilon_{re})^2 + \varepsilon_{re}^2}. \tag{25}$$

As $\rho \to 1, \varepsilon \to 0$, we can get

$$\frac{R^{CVTPM}(\varepsilon_{re})}{U^{CVTPM}(\rho_{re})} = \frac{48}{7\pi^2} < 1. \tag{26}$$

Using the same method, one can obtain

$$\frac{R^{CVTPM}(\varepsilon_{im})}{U^{CVTPM}(\rho_{im})} < 1. \tag{27}$$

Following [28] and [30], we can get the real CVTPM that is secure if and only if both $U^{CVTPM}(\rho_{re}) > R^{CVTPM}(\varepsilon_{re})$ and $U^{CVTPM}(\rho_{im}) > R^{CVTPM}(\varepsilon_{im})$ are satisfied, which means $\Delta\rho(\rho_{re}) > 0$ and $\Delta\rho(\rho_{im}) > 0$.

Remark 4: From the above derivation, one can obtain the synchronization process of the real part (imaginary part) of CVTPM is the same as the synchronization process of TPM. So, the security of the real part (imaginary part) of the CVTPM is equal to the security of TPM. If the attackers want to know the group key, they must know both the real and imaginary parts of the weights of CVTPM. It is easy to see that the synchronization of the real and imaginary parts is independent of each other. Thus, we can get

$$P_E^{CVTPM} = P_E^{\mathrm{Im}} P_E^{\mathrm{Re}} < P_E^{TPM}$$

where P_E^{Re} is the probability that the attacker obtains the whole real part of connection weights of CVTPM at synchronization time and P_E^{Im} is the probability that the attacker obtains the whole imaginary part of connection weights of CVTPM at synchronization time. As both P_E^{Re} and P_E^{Im} are less than 1, the security of CVTPM is better than that of TPM with the same hidden units, input neurons, and synaptic depths.

Remark 5: As the synchronization process of the real part of CVTPM is independent of the synchronization process of the imaginary part of CVTPM, we can compute the synchronization time, respectively. First, we investigate the synchronization time of real part. Two random walks corresponding to $\mathrm{Re}(\omega_{i,j}^A)$ and $\mathrm{Re}(\omega_{i,j}^B)$ can move on a 1-D line with $m = 2L_{re} + 1$ sites. Let $T_{d,z}$ be the synchronization time for two random walks starting at position z and distance d. The synchronization time at position z and distance d is as follows:

$$T_{d,z} = S_{d,z} + \sum_{j=1}^{d-1} S_{j,1}$$

where $S_{d,z}$ denotes the average number of the possibility of the two random walkers that move to left and the possibility of the two random walkers that move to right at position z and distance d. Then, we can obtain that the synchronization time of CVTPM of the real part is as follows:

$$\langle t_{re}^{sync} \rangle = \frac{2}{m^2} \sum_{d=1}^{m-1} \sum_{z=1}^{m-d} \langle T_{d,z} \rangle.$$

By simple calculation, we can obtain

$$\langle t_{re}^{sync} \rangle \propto m^2 \propto L_{re}^2.$$

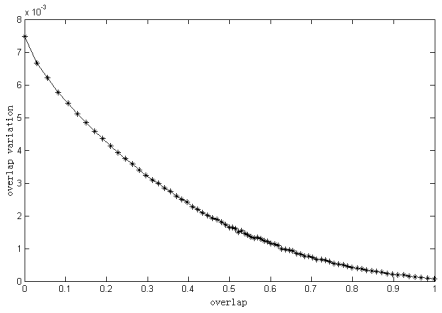

Fig. 2. Overlap of the CVTPM.

Using the same method, we can obtain

$$\langle t_{im}^{sync} \rangle \propto m^2 \propto L_{im}^2.$$

Then, we can obtain the synchronization time of CVTPM

$$\langle t_{CVTPM}^{sync} \rangle \propto \max\{m_{re}^2, m_{im}^2\} \propto \max\{L_{re}^2, L_{im}^2\}.$$

IV. Synchronization Time

In this section, we investigate the synchronization time of CVTPM. Remarkably, $\Delta\rho(\cdot) > 0$ enables the synchronization time grows at a polynomial rate with increasing L, and we can define the synchronization time of the real part and the synchronization time of the imaginary part as follows:

$$\langle t_{re}^{sync} \rangle \propto \frac{1}{\langle \Delta\rho_{re} \rangle} \propto L_{re}^2 \tag{28}$$

$$\langle t_{im}^{sync} \rangle \propto \frac{1}{\langle \Delta\rho_{im} \rangle} \propto L_{im}^2. \tag{29}$$

Then, we can obtain the synchronization time of CVTPM

$$\langle t_{CVTPM}^{sync} \rangle \propto \max\left\{\frac{1}{\langle \Delta\rho_{re} \rangle}, \frac{1}{\langle \Delta\rho_{im} \rangle}\right\} \propto \max\{L_{re}^2, L_{im}^2\}. \tag{30}$$

Remark 6: From (30), we can see that the synchronization time of CVTPM grows at a polynomial rate with increasing $\max\{L_{re}, L_{im}\}$. From [28]–[30], the synchronization time of CVTPM grows at a polynomial rate with increasing L_{TPM}. If the CVTPM and TPM have the same L, the synchronization time of CVTPM and TPM is the same order of magnitude.

V. Simulations

In this section, some experimental results are presented to demonstrate the performance of the CVTPM with $K = 3$ and $N = 1000$. The learning rule is the Hebbian learning rule.

Example 1: In this experiment, we consider the synaptic depth $L = 9$. Fig. 2 shows the overlap variation of CVTPM in the whole synchronization process. It can be seen that the overlap increases from 0 to 1, which means the synchronization is completed. We define the Euclidean distance to describe the degree of synchronization, which can be defined as follows:

$$ED = \sum_{i=1}^{k} \sum_{j=1}^{n} \left\| \omega_{i,j}^A - \omega_{i,j}^B \right\|$$

$$= \sum_{i=1}^{k} \sum_{j=1}^{n} \sqrt{\left(\mathrm{Re}(\omega_{i,j}^A) - \mathrm{Re}(\omega_{i,j}^B)\right)^2 + \left(\mathrm{Im}(\omega_{i,j}^A) - \mathrm{Im}(\omega_{i,j}^B)\right)^2}.$$

The *ED* should become zero when the two networks are synchronized. Without loss of generality, we choose seven groups of random weight parameters of CVTPM and TPM. Fig. 3 shows the number

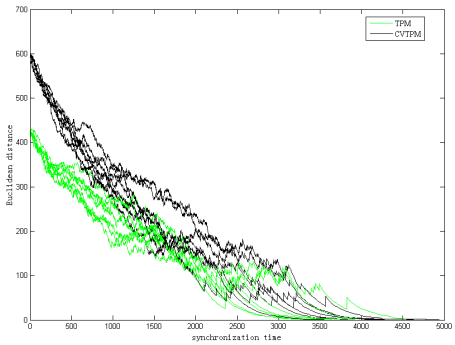

Fig. 3. Euclidean distance between weight vectors of the CVTPM and TPM.

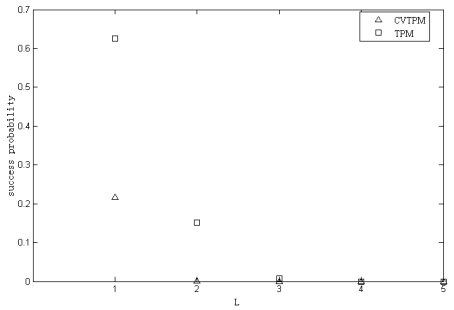

Fig. 4. Success probability of a simple attack as a function of L. Results are obtained in 10 000 simulations with learning rule using N = 1000.

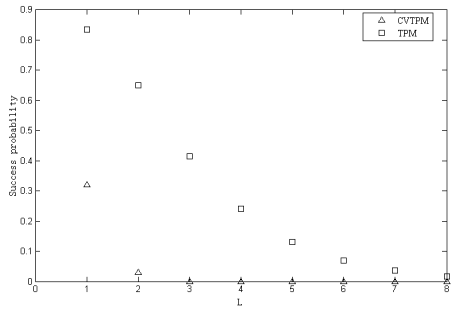

Fig. 5. Success probability of a geometric attack as a function of L. Results are obtained in 10 000 simulations with learning rule using N = 1000.

of iterations of CVTPM and TPM. The horizontal axis of Fig. 3 is synchronization time and the vertical axis is the Euclidean distance. It can be seen that the synchronization time of CVTPM and TPM is in the same order, which is consistent with remark 6.

Example 2: In this experiment, we study the security of CVTPM under simple attack [28]–[30]. In this case, the adversary only needs a CVTPM. We define P_E as the success probability that the attacker knows 90% of the weights at synchronization time. Fig. 4 shows the success probability of CVTPM and TPM under simple attack. The horizontal axis of Fig. 4 is the synaptic depth of CVTPM and the vertical axis is the success probability. It can be seen that the success probability of the CVTPM under simple attack is much lower than the success probability of TPM, which is consistent with remark 3.

Example 3: In this experiment, we study the security of CVTPM under geometric attack [28]–[30]. Similar to example 2, P_E denotes the probability that the attacker knows 90% of the weights at synchronization time. Fig. 5 shows the success probability of CVTPM and TPM under geometric attack. The horizontal axis of Fig. 4 is the synaptic depth of CVTPM and the vertical axis is the success probability. It can be seen that the success probability of CVTPM under geometric attack is much lower than the success probability of TPM, which is consistent with remark 3.

VI. Conclusion

In this brief, a neural cryptography based on the CVTPM is investigated. The input, output, and weights of CVTPM are all complex values, which can be considered as an extension of TPM. The security and synchronization time of CVTPM are also investigated. It is found that the security of CVTPM is higher than TPM with the same hidden units, input neurons, and synaptic depths. The two parties with CVTPM can exchange two group keys in one neural synchronization process. Finally, a series of numerical simulation experiments is provided to verify our results.

References

[1] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976.

[2] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman key distribution extended to group communication," in *Proc. 3rd ACM Conf. Comput. Commun. Secur.*, 1996, pp. 31–37.

[3] P. Balasubramaniam and P. Muthukumar, "Synchronization of chaotic systems using feedback controller: An application to Diffie–Hellman key exchange protocol and ElGamal public key cryptosystem," *J. Egyptian Math. Soc.*, vol. 22, no. 3, pp. 365–372, Oct. 2014.

[4] M. Eftekhari, "A Diffie–Hellman key exchange protocol using matrices over noncommutative rings," *Groups-Complex.-Cryptol.*, vol. 4, no. 1, pp. 167–176, 2012.

[5] H. Chen, P. Shi, and C.-C. Lim, "Cluster synchronization for neutral stochastic delay networks via intermittent adaptive control," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 11, pp. 3246–3259, Nov. 2019.

[6] P. Liu, Z. Zeng, and J. Wang, "Global synchronization of coupled fractional-order recurrent neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 8, pp. 2358–2368, Aug. 2019.

[7] H. Chen, P. Shi, and C.-C. Lim, "Exponential synchronization for Markovian stochastic coupled neural networks of neutral-type via adaptive feedback control," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 7, pp. 1618–1632, Jul. 2017.

[8] J. Wang, L.-M. Cheng, and T. Su, "Multivariate cryptography based on clipped hopfield neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 2, pp. 353–363, Feb. 2018.

[9] J.-L. Wang, Z. Qin, H.-N. Wu, and T. Huang, "Passivity and synchronization of coupled uncertain reaction-diffusion neural networks with multiple time delays," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 8, pp. 2434–2448, Aug. 2019.

[10] Q. Xiao, T. Huang, and Z. Zeng, "Global exponential stability and synchronization for discrete-time inertial neural networks with time delays: A timescale approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 6, pp. 1854–1866, Jun. 2019.

[11] Z. Zhang and J. Cao, "Novel finite-time synchronization criteria for inertial neural networks with time delays via integral inequality method," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 5, pp. 1476–1485, May 2019.

[12] A. Wang, T. Dong, and X. Liao, "Event-triggered synchronization strategy for complex dynamical networks with the Markovian switching topologies," *Neural Netw.*, vol. 74, pp. 52–57, Feb. 2016.

[13] T. Dong, A. Wang, H. Zhu, and X. Liao, "Event-triggered synchronization for reaction–diffusion complex networks via random sampling," *Phys. A, Stat. Mech. Appl.*, vol. 495, pp. 454–462, Apr. 2018.

[14] S. Lakshmanan, M. Prakash, C. P. Lim, R. Rakkiyappan, P. Balasubramaniam, and S. Nahavandi, "Synchronization of an inertial neural network with time-varying delays and its application to secure communication," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 195–207, Jan. 2018.

[15] Z. Ni and S. Paul, "A multistage game in smart grid security: A reinforcement learning solution," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 2684–2695, Sep. 2019.

[16] M. Rosen, I. Kanter, and W. Kinzel, "Cryptography based on neural networks—Analytical results," *J. Phys. A, Math. Gen.*, vol. 35, no. 47, pp. 707–713, 2002.

[17] U. M. Maurer, "Secret key agreement by public discussion from common information," *IEEE Trans. Inf. Theory*, vol. 39, no. 3, pp. 733–742, May 1993.

[18] M. Kobayashi, "Symmetric complex-valued Hopfield neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 4, pp. 1011–1015, Apr. 2017.
[19] T. Dong, X. Liao, and A. Wang, "Stability and Hopf bifurcation of a complex-valued neural network with two time delays," *Nonlinear Dyn.*, vol. 82, nos. 1–2, pp. 173–184, 2015.
[20] M. Rosen, I. Kanter, and W. Kinzel, "Mutual learning in a tree parity machine and its application to cryptography," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 66, no. 6, 2002, Art. no. 066135.
[21] A. Ruttor, G. Reents, and W. Kinzel, "Synchronization of random walks with reflecting boundaries," *J. Phys. A, Math. Gen.*, vol. 37, no. 36, p. 8609, 2004.
[22] R. Metzler, W. Kinzel, and I. Kanter, "Interacting neural networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 62, no. 2, p. 2555, Aug. 2000.
[23] W. Kinzel, R. Metzler, and I. Kanter, "Dynamics of interacting neural networks," *J. Phys. A, Math. Gen.*, vol. 33, no. 14, pp. 141–147, 2000.
[24] I. Kanter, W. Kinzel, and E. Kanter, "Secure exchange of information by synchronization of neural networks," *Europhys. Lett.*, vol. 57, no. 1, pp. 141–147, 2002.
[25] L. Shacham, E. Klein, R. Mislovaty, I. Kanter, and W. Kinzel, "Cooperating attackers in neural cryptography," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 69, no. 6, 2004, Art. no. 066137.
[26] A. Ruttor, W. Kinzel, R. Naeh, and I. Kanter, "Genetic attack on neural cryptography," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 73, no. 3, 2006, Art. no. 036121.
[27] A. Klimov, A. Mityagin, and A. Shamir, "Analysis of neural cryptography," in *Advances in Cryptology*. Berlin, Germany: Springer, 2002, pp. 288–298.
[28] A. Ruttor, "Neural synchronization and cryptography," Ph.D. dissertation, Fakultät für Physik und Astronomie, Inst. für Theoretische Physik und Astrophysik, Würzburg, Germany, 2006.
[29] W. Kinzel and I. Kanter, "Interacting neural networks and cryptography," in *Advances in Solid State Physics*. Berlin, Germany: Springer, 2002, pp. 383–391.
[30] N. Mu and X. Liao, "Approach to design neural cryptography: A generalized architecture and a heuristic rule," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 87, Jun. 2013, Art. no. 062804.
[31] X. Lei, X. Liao, F. Chen, and T. Huang, "Two-layer tree-connected feedforward neural network model for neural cryptography," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 87, Mar. 2013, Art. no. 032811.
[32] A. M. Allam and H. M. Abbas, "On the improvement of neural cryptography using erroneous transmitted information with error prediction," *IEEE Trans. Neural Netw.*, vol. 21, no. 12, pp. 1915–1924, Dec. 2010.