Cooperating attackers in neural cryptography

Lanir N. Shacham,[1] Einat Klein,[1] Rachel Mislovaty,[1] Ido Kanter,[1] and Wolfgang Kinzel[2]
[1]*Minerva Center and Department of Physics, Bar-Ilan University, Ramat-Gan 52900, Israel*
[2]*Institut für Theoretische Physik, Universität Würzburg, Am Hubland 97074 Würzburg, Germany*

(Received 1 December 2003; published 23 June 2004)

A successful attack strategy in neural cryptography is presented. The neural cryptosystem, based on synchronization of neural networks by mutual learning, has been recently shown to be secure under different attack strategies. The success of the advanced attacker presented here, called the "majority-flipping attacker," does not decay with the parameters of the model. This attacker's outstanding success is due to its using a group of attackers which cooperate throughout the synchronization process, unlike any other attack strategy known. An analytical description of this attack is also presented, and fits the results of simulations.

DOI: 10.1103/PhysRevE.69.066137 PACS number(s): 87.18.Sn, 89.70.+c

The use of neural networks in the field of cryptography has recently been suggested [1] and has since been a source of interest for researchers from different fields [2]. The neural cryptosystem is based on the ability of two neural networks to synchronize. The two networks undergo an online learning procedure called *mutual learning*, in which they learn from each other simultaneously, i.e., every network acts both as a teacher and as a student. At every time step the networks receive a common input vector, calculate their outputs, and update their weight vectors according to the match between their mutual outputs [3]. The input/output relations are exchanged through a public channel until their weight vectors are identical and can be used as a secret key for encryption and decryption of secret messages. Thus we have a public key-exchange protocol which is not based on number theory nor does it involve long numbers and irreversible functions, and is essentially different from any other cryptographic method known before.

The question is whether this system is secure, and to what degree? Since the data are transferred through a public channel, any attacker who eavesdrops might manage to synchronize with the two parties, and reveal their key. Yet the attacker is in a position of disadvantage: while the parties perform *mutual* learning and approach one another, the attacker performs dynamic learning and "chases" them, therefore they have an advantage over him. The system's security depends on whether they manage to exploit this advantage so that the attacker will forever stay behind.

The synchronization is based on a competition between attractive and repulsive stochastic forces between the parties. Attractive forces bring them closer to each other, and repulsive forces drive them apart and delay the synchronization. Synchronization is possible only if the attractive forces are stronger than the repulsive forces $(A>R)$. On the one hand, if the attractive forces are too strong, synchronization is relatively fast and easy, so that an attacker eavesdropping on the line and trying to synchronize will manage to do so easily. On the other hand, if the repulsive forces are too strong, synchronization will be hard for the attacker, but also for the two parties. A secure system is one which manages to balance these forces so that the net force between the parties is positive and stronger than for the attacker $[(A-R)_{\text{parties}}>(A-R)_{\text{attacker}}]$.

The following is the model we use: The networks are tree parity machines (TPM) with K hidden units $\sigma_i=\pm 1, \quad i=1,\ldots,K$ feeding a binary output, $\tau=\Pi_{i=1}^{K}\sigma_i$, as shown in Fig. 1. We used $K=3$. The networks consist of a discrete coupling vector $\mathbf{w}_i=W_{i1},\ldots,W_{iN}$ and disjointed sets of inputs $\mathbf{x}_i=X_{i1},\ldots,X_{iN}$ containing N elements each. The input elements are random variables $x_{ij}=\pm 1$. Each component of the weight vector can take certain discrete values $W_{ij}=\pm L,\pm(L-1),\ldots,\pm 1,0$, and is initiated randomly from a flat distribution.

The local field in the *i*th hidden unit is defined as

$$h_i=\mathbf{w}_i\cdot\mathbf{x}_{i,} \tag{1}$$

and the output in the *i*th hidden unit is the sign of the local field. The output of the tree parity machine is therefore given by

$$\tau=\prod_{i=1}^{K}\text{sgn}(h_i)=\prod_{i=1}^{K}\sigma_i.$$

During the mutual learning process, the two machines A and B exchange their output values $\tau^{A/B}$. They update their weights using the Hebbian learning rule only in cases in which their outputs agree and only in hidden units which agree with the output,

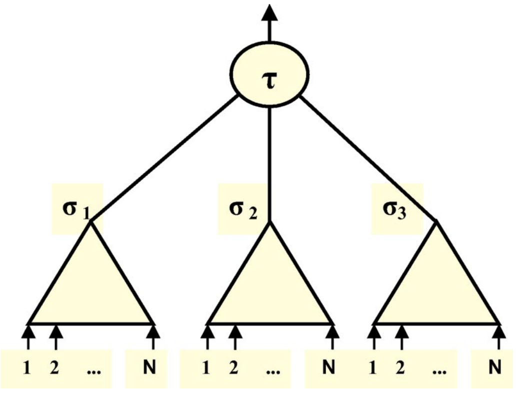

FIG. 1. A tree parity machine with $K=3$.

$$\mathbf{w}_i^A(t+1) = \mathbf{w}_i^A(t) + \mathbf{x}_i \tau^A \theta(\tau^A \sigma_i^A)\,\theta(\tau^A \tau^B),$$

$$\mathbf{w}_i^B(t+1) = \mathbf{w}_i^B(t) + \mathbf{x}_i \tau^B \theta(\tau^B \sigma_i^B)\,\theta(\tau^A \tau^B). \qquad (2)$$

This leads them to a parallel state in which $W^A = W^B$. The attacker C tries to learn the weight vector of one of the two machines, say A, yet unlike the simple teacher-student scenario [4,5], the teacher's weights in this case are time-dependent, therefore the attacker must use some attack strategy in order to follow the teacher's steps.

The following are possible attack strategies, which were suggested by Shamir *et al.* [2]. The *genetic attack*, in which a large population of attackers is trained, and at every new time step each attacker is multiplied to cover the 2^{K-1} possible internal representations of $\{\sigma_i\}$ for the current output τ. As the dynamics proceed, successful attackers stay while the unsuccessful ones are removed. *The probabilistic attack*, in which the attacker tries to follow the probability of every weight element by calculating the distribution of the local field of every input and using the output, which is publicly known. *The naive attacker*, in which the attacker imitates one of the parties. The most successful attacker suggested so far is the *flipping attack* (geometric attack), in which the attacker imitates one of the parties, but in steps in which his output disagrees with the imitated party's output, he negates ("flips") the sign of one of his hidden units. The unit most likely to be wrong is the one with the minimal absolute value of the local field, therefore that is the unit which is flipped.

While the synchronization time increases with L^2 [6], the probability of finding a successful flipping attacker decreases exponentially with L,

$$P \propto e^{-yL}$$

as seen in Fig. 2. Therefore, for large L values the system is secure [6]. This can be supported also by the fact that close to synchronization, the probability for a repulsive step in the mutual learning between A and B scales like $(\epsilon)^2$, while in the dynamic learning between the naive attacker C and A it scales like ϵ, where we define $\epsilon = \text{prob}(\sigma_i^C \neq \sigma_i^A)$ [9].

The attackers mentioned above try to imitate the parties, each using different heuristics. They use an ensemble of independent attackers. These attackers all develop an overlap with the parties during the synchronization process and also an overlap between themselves, *yet each attacker evolves independently, and is not influenced by the state of the other attackers*.

It has been shown that among a group of Ising vector students which perform learning, and have an overlap R with the teacher, the best student is the center-of-mass vector (which was shown to be an Ising vector as well), which has an overlap $R_{\text{c.m.}} \propto \sqrt{R}$ for $R \in [0:1]$ [10]. Therefore, letting the attackers cooperate throughout the process may be to their advantage.

The new "majority flipping attacker" presents a general strategy which can be applied to some of the heuristic attackers mentioned, and can improve their results, and it uses the attackers as a *cooperating group rather than as individuals*, an approach which has not been done before. The majority strategy is the following: we start with a group of M random attackers. Instead of letting them work independently and hope for one to be successful, we let them cooperate—when updating the weights, instead of each machine being updated according to its own result, all are updated according to the majority's result. This "team-work" approach improves the attacker's performance. Naturally, we chose to apply it to the most successful attacker, the "flipping attacker," thus creating the "majority-flipping attacker."

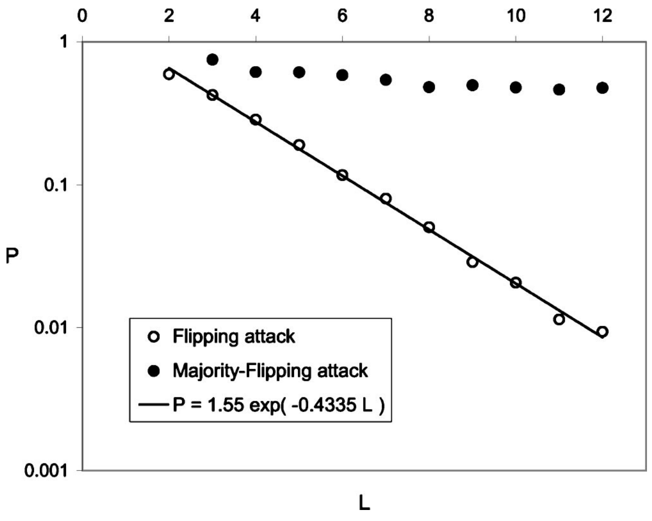

FIG. 2. The attacker's success probability P as a function of L, for the flipping attack and the majority-flipping attack, with $N = 1000$, $M = 100$, averaged over 1000 samples. To avoid fluctuations, we define the attacker as successful if he found out 98% of the weights.

The main result of this paper is the improvement of the success rate of the flipping attacker when using the majority scheme: The regular flipping attacker, although relatively successful, is weakened by increasing L, and the probability for a successful attacker, P, drops exponentially with L [6]. When using the majority scheme, this probability seems to approach a constant value ~0.5 independent of L [7].

When applying the majority strategy to the flipping attack, we create M flipping attackers. In the beginning of the process, during a certain time, the regular flipping attack is performed; those among the M machines that disagree with party A have one of their hidden unit's signs negated, and then their weights' vectors updated according to their new internal representations.

After a certain time, we start to perform the majority procedure: In every odd time step we perform the regular flipping attack, and in every even time step we perform a majority-flipping procedure, which consists of the following two steps.

(i) All attackers who disagree with party A flip one of their hidden units, according to the regular flipping attack procedure.

(ii) Now all the M attackers have the same output but different internal representations of $\{\sigma_i\}$. We check which of the four possible internal representations appears the most.

Then, instead of updating every attacker according to its own internal representation, all are updated according to the same internal representation—the majority's representation. It is as if we let the machines "vote," and all must use the internal representation that was "elected."

When the attackers perform the majority step, they all perform the same step, therefore an overlap is developed between them. The larger the overlap between them, the less effective they are, because effectively there are fewer attackers. In the limit when all the attackers are identical, there is effectively only one attacker. There is no way to avoid this similarity between them. We rather prevent it from developing too quickly, and we do so by performing the majority step only on even time steps, and not from the beginning of the process but after a waiting time of about $\frac{1}{3}$ of the entire synchronization time [11].

The result of using this scheme is shown in Fig. 2. When comparing the success of the flipping attacker with and without the majority strategy, we see that for the latter the success probability drops exponentially with L, while for the former it remains around 0.5 even when L is increased. Similar results of the majority-flipping attack success were obtained in the case of the chaotic neural network model [8].

Why is the majority-flipping attack so successful? Every update of the weights can either bring every attacker closer to party A (an "attractive step") or farther away (a "repulsive step"). A repulsive step between the attacker and A occurs when there is a difference in their internal representations (in steps where A and B perform an update). A good attack strategy is one that manages to reduce the probability for a repulsive step, and the majority-flipping attacker does this by using the majority vote. Once an overlap is developed between an attacker and machine A, the probability for a correct (attractive) internal representation P_a is larger than the probability for a repulsive one. For a group of $M \gg 1$ uncorrelated attackers, which all have an overlap ρ_{AC} with A, the probability that their majority is correct is 1. However, if the attackers are correlated, which is the case here, $P_a < 1$, yet it is larger than P_a of just one flipping attacker, as can be seen in Fig. 3 (in our simulations we obtained similar results for all $M > 50$). The majority's advantage over a random choice is the essence of this attack, as shown also in the Bayes optimal classification algorithm versus the Gibbs learning algorithm, where choosing the majority proves to be better than a random choice [10].

The semianalytical description of this process confirms these results and gives us further insight into the majority attacker's success. In the semianalytical description, we describe the system using $(2L+1)\times(2L+1)$ order parameters, and we manage to simulate the system in the thermodynamic limit. We represent the state of the TPMs using a matrix $\mathbf{F}$ of size $(2L+1)\times(2L+1)$, as described in [9]. The elements of $\mathbf{F}$ are f_{qr}, where $q,r=-L,\ldots,-1,0,1,\ldots,L$. The element f_{qr} represents the fraction of components in a weight vector in which the A's components are equal to q and the matching components of B are equal to r. Hence, the overlap between the two units and the norm of party A, for instance, are given by

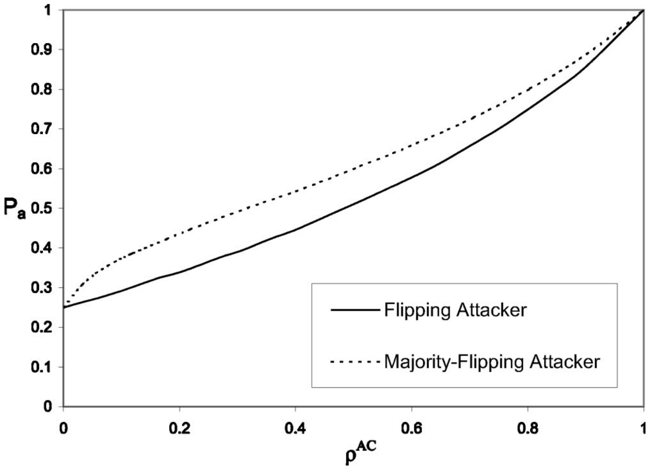

FIG. 3. The probability of attacker C to have a correct internal representation as a function of the average overlap between the attackers and one of the parties, for flipping and majority-flipping attacks, measured in simulations with $N=1000$, $M=300$, averaged over 10^5 samples.

$$R = \sum_{q,r=-L}^{L} qrf_{qr}, \quad Q_A = \sum_{q=-L}^{L} q^2 f_{qr} \tag{3}$$

and the overlap $\rho_{AB}=R_{AB}/\sqrt{Q_A Q_B}$. There are three matrices representing the mutual overlap between a pair of hidden units among A, B, and C (we omitted the hidden unit's index for the sake of simplicity). We do not create M attackers but rather one that represents one of the M attackers in the simulations.

The procedure at every time step is as follows.

(i) We randomly choose K local fields for the K hidden units of machine A, from a Gaussian distribution with the mean 0 and the standard deviation $\sqrt{Q_A}$.

(ii) We then randomly choose K local fields for the K hidden units of machine B, from a Gaussian distribution with the mean $R_{AB}h_A/Q_A$ and the standard deviation $\sqrt{Q_B - R_{AB}^2/Q_A}$ (taking into account B's overlap with A).

(iii) If the outputs of A and B disagree, they are not updated and we continue to the next time step. If they agree, we update the matrices representing A and B and then update the attacker as described in the next step.

(iv) We set the internal representation of the attacker. For $K=3$, there are eight possible internal representations. We calculate their probabilities $P_1,\ldots,P_8$, according to the attacker's overlap with A and B and the local fields of A and B. For example, the internal representation $+++$ has the probability

$$P(+++) = \prod_{m=1}^{3} P(h_m^C > 0 | h_m^A, h_m^B, \{R,Q\}).$$

For simplicity, we assume that there is no significant difference between the attacker's overlap with A and its overlap with B and therefore we use only one of them so that

$$P(h_i^C > 0|h_i^A,\{R,Q\}) = H\left(\frac{-R_{AC}h_A}{\sqrt{Q_A^2 Q_C - R_{AC}^2 Q_A}}\right),$$

where $H(x)=\int_x^\infty e^{-t^2/2}dt/\sqrt{2\pi}$.

Next we simulate the flipping, when the eight possible states are reduced to four: either states 1–4 (states with positive output) flip to 5–8 (states with negative output) or vice versa, depending on A's output. We calculate the probabilities of the states' flipping. For example, the probability that state $+++$ flipped to state $-++$ is $P(+++)P(h_1^C < h_2^C, h_1^C < h_3^C)$, where

$$P(h_1^C < h_2^C, h_1^C < h_3^C) = \int_0^\infty P(h_1^C|h_1^A,h_1^B,)dh_1^C \cdot$$

$$\int_{h_1^C}^\infty P(h_2^C|h_2^A,h_2^B,)dh_2^C \int_{h_1^C}^\infty P(h_3^C|h_3^A,h_3^B,)dh_3^C.$$

We now remain with probabilities for four possible internal representations. In the case of a regular flipping step, we randomly choose one of these four states according to their probabilities, but in the case of a majority step, the probability of choosing the correct internal presentation is higher. We do not calculate it, but rather measure it in the simulations, and use the measured probability (presented by the dashed line in Fig. 3) in the analytical procedure. Figure 4 shows the success probability of one of the M attackers as a function of L. It shows a fairly good agreement between the analytical and the simulation results (see [12]).

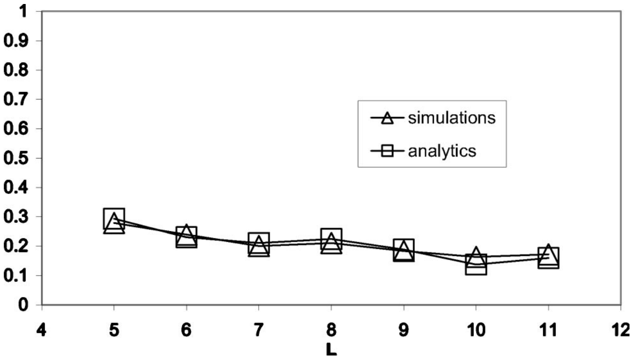

FIG. 4. The probability for one of the M attackers to be successful as a function of L, obtained from the analytical calculations and simulations with $N=1000$, $M=100$. Here we define synchronization when the average mutual overlap of the three hidden units reaches 0.99. Results were averaged over 1000 samples.

To conclude, an important step in the field of neural cryptography has been made, presenting an attacking approach under which the TPM cryptosystem is insecure. The question is, can we create a more sophisticated system that will be secure under the majority attack? A secure system will be one for which the probability for a correct step of the majority flipping attacker will be near the flipping attacker's curve in Fig. 3, yet the synchronization time of the parties will still remain polynomial with L. There can be many ideas for such a system, for example a system in which $K > 3$, so that repulsive forces are stronger. Yet keeping the synchronization time polynomial with L is not easy when repulsive forces are too strong, so these models are still under consideration, and the challenge still remains.

[1] I. Kanter, W. Kinzel, and E. Kanter, Europhys. Lett. **57**, 141 (2002).

[2] A. Klimov, A. Mityagin, and A. Shamir, ASIACRYPT 288–298 (2002).

[3] R. Metzler, W. Kinzel, and I. Kanter, Phys. Rev. E **62**, 2555 (2000); W. Kinzel, R. Metzler, and I. Kanter, J. Phys. A **33**, L141 (2000).

[4] A. Engel and C. Van den Broeck, *Statistical Mechanics of Learning* (Cambridge University Press, Cambridge, 2001).

[5] W. Kinzel, *Handbook of Graphs and Networks*, edited by S. Bornholdt and G. Schuster (Wiley, VCH, 2003).

[6] R Mislovaty, Y. Perchenok, I. Kanter, and W. Kinzel, Phys. Rev. E **66**, 066102 (2002).

[7] The anti-Hebbian learning case [1] presents strong finite size effects and lower P_{majority} even for $N=10\,000$. However, results seem to converge to the Hebbian case in the thermodynamic limit.

[8] R. Mislovaty, E. Klein, I. Kanter, and W. Kinzel, Phys. Rev. Lett. **91**, 118701 (2003).

[9] M. Rosen-Zvi, E. Klein, I. Kanter, and W. Kinzel, Phys. Rev. E **66**, 066135 (2002).

[10] M. Copelli, M. Boutin, C. Van Der Broeck, and B. Van Rompaey, Europhys. Lett. **46**, 139 (1999).

[11] A waiting time of $\frac{1}{3}$ the synchronization time causes the attackers to reach the single attacker limit at the end of the synchronization process, so that their efficiency is maximal.

[12] The difference between the majority-flipping attack's success presented in Fig. 2 (~0.5) and that in Fig. 4 (~0.25) is due to a different halting condition. In Fig. 2, synchronization is defined for the parties as 100% identical weights and 98% identical weights for the attacker. In Fig. 4, in order to be able to compare between the analytical and the simulations results, we defined synchronization as overlap equal to 0.99.