Cryptography and Machine Learning

Ryan Moriarty

Courant Insititute of Mathematical Sciences
New York University
rpm295@nyu.edu

Abstract

This paper provides a survey of the connection between the fields of cryptography and machine learning. The dual nature of the two subjects is explained and contributions made in both directions are summarized. Lastly, promising research areas at the intersection of the fields is explored.

I. Background on Cryptography

The minimal cryptographic object is the one-way function (OWF). A function $f : \{0,1\}^* \to \{0,1\}^*$ is one-way if it is efficiently computable and for all probabilistic polynomial time (PPT) algorithms $\mathcal{A}$,

$$\Pr_{x \in \{0,1\}^n}[\mathcal{A}(1^n, f(x)) \in f^{-1}(f(x))] \in \text{negl}(n). \tag{1}$$

These functions are assumed, but not proven to exist. Doing so requires proving $P \neq NP$, a famous unsolved problem in computer science. However, there are many candidate functions that are believed to be one-way. One familiar example is the multiplication function, which is easy to compute and, if we restrict our inputs to large primes of roughly the same size, difficult to invert (factor). Other candidate OWFs are the discrete logarithm problem and the subset sum problem.

Remarkably, it turns out that almost all other cryptographic tools can be derived from OWFs. Impagliazzo, Levin, and Luby [5] demonstrated how a pseudorandom generator (PRG), which is a function whose output appears indistinguishable from random to all PPT adversaries, can be constructed from any OWF. Furthermore, Goldreich, Goldwassser and Micali [4] proved a pseudorandom function family (PRF) can be constructed from any PRG. Formally, $\{f_k : \{0,1\}^{\ell_1(n)} \to \{0,1\}^{\ell_2(n)}\}_k$ is a PRF family if it is efficiently computable and for all PPT algorithms $\mathcal{A}$,

$$\left| \Pr_{f \leftarrow \{f_k\}}[\mathcal{A}^f = 1] - \Pr_{f \leftarrow U}[\mathcal{A}^f = 1] \right| \in \text{negl}(n). \tag{2}$$

These function families can be used directly for secret key encryption. For example, if Alice and Bob share secret key k, they can encrypt and decrypt any message m as follows:

- $Enc(m) = (r, f_k(r) \oplus m) = (r, c)$
- $Dec(r, c) = f_k(r) \oplus c = m$

where $r \leftarrow \{0,1\}^n$. Therefore, the simple assumption that OWFs exist implies the existence of secure encryption schemes.

Another family of functions used for encryption are trapdoor functions. These are similar to OWFs except they can be decrypted in polynomial time using a secret key. That is, there exists an efficient inversion algorithm $F^{-1}(k, x) = f_s^{-1}(x)$, where s is a function index and k is a secret key. One-time secure encryption and decryption can be considered as follows:

- $Enc(m) = f_s(m) = c$
- $Dec(k, c) = F^{-1}(k, c) = m$.

Traditionally, encryption schemes must be secure under a chosen plaintext attack (CPA).

In this setup, an adversary is allowed to request the encryptions of a polynomial number of messages of his choice. After-which, he submits two new messages, m_0 and m_1, and receives the encryption of one of them. He then must guess which of the two messages was encrypted. The scheme is deemed secure if for all PPT adversaries the probability of succeeding is at most negligibly greater than one-half. In other words, the two distributions are computationally indistinguishable:

$$(Enc_k(\cdot), Enc_k(m_0)) \overset{c}{\approx} (Enc_k(\cdot), Enc_k(m_1)). \tag{3}$$

II. Cryptography's impact on Learning Theory

Valiant [19] in his seminal paper "A Theory of the Learnable" looked at the above attack from a learner's perspective. He concluded that the existence of CPA secure schemes implies certain classes of functions are not learnable. If we replace an adversary with any machine learning algorithm and view the series of encryption message pairs as training data, we can view the problem under the probably approximately correct (PAC) framework. The inability of an algorithm to even approximately learn the decryption function, implies that is not weakly PAC learnable.

With this in mind, Kearns and Valiant [8] tried to find the simplest classes that cannot be learned under cryptographic assumptions. To do so, they looked at the RSA [17] trapdoor function family:

- Private Key: (n, d)
- Public Key: (n, e)
- Encryption: $Enc(m) = m^e \pmod{n}$ = c
- Decryption: $Dec(c) = c^d \pmod{n}$ = m

where n is a composite of two large primes, e is relatively prime with $\phi(n)$ and $e \cdot d = 1 \pmod{\phi(n)}$.

They presented a learning problem based on a modified version of RSA in which the learner receives examples of the following form: (binary(powers(c), n, e), $LSB(m)$) from some unknown distribution. Where powers(c) corresponds to the repeated squaring of the cipher-text for c to $c^{\lceil \log(n) \rceil}$ and $LSB(m)$ is the least significant bit of the message that was encrypted. The learner's goal is to output a hypothesis h which accurately predicts the LSB of the message on new cipher-texts, a task as hard as recovering the entire message. Formally, the scheme is deemed secure if $Pr_{c \sim D}[h(c) = LSB(m)] \leq \frac{1}{2} +$ negl(n).

The complexity of inverting the function *given* the trapdoor key corresponds to the complexity of the representations being learned. By presenting the learning task in this manner, the inversion function can be computed by multiplying together the appropriate powers of c (modulo n) according to which bits of d are set and outputting the least significant bit. Since these are NC^1 (the class of circuits consisting of AND, OR and NOT gates of fan-in having size polynomial in n and depth logarithmic in n) steps, Kearns and Valiant conclude NC^1 circuits are not weakly PAC learnable.

Citing the work of Pitt and Warmuth [13], they show reductions from NC^1 to other classes of interest in learning theory including:

- Boolean formulae
- Finite automata
- Constant depth threshold circuits.

Essentially, these classes subsume NC^1 circuits and thus, efficient algorithms for learning these classes would imply efficient algorithms for learning NC^1.

Of course, in cryptography we would like to have security regardless of what distribution examples are drawn from. However, Kharitonov [9] demonstrated these results hold for all non-trivial distributions as well.

III. Learning Theory's impact on Cryptography

Inversely, hard problems in learning theory can be used as the basis for cryptographic schemes. One well studied problem in learning theory is how to learn efficiently from noisy data. Work done by Angluin and Laird [2], proved that

PAC algorithms can be generalized to handle noisy classification error, as long as $err < \frac{1}{2}$. Further research was done by Blum, Kalai, and Wasserman [3] on the problem of learning parity functions in the presence of noise (LPN), who proved a sub-exponential algorithm for the problem exists. A generalization of this problem denoted *Learning with Errors* (LWE) was introduced by Regev [15] and has proved to be an amazingly versatile basis for cryptography.

The LWE problem asks to recover a secret $s \in Z_q^n$ (an n dimensional vector modulo a large prime q). The case where $q = 2$ corresponds to the LPN problem. Examples consists of random vectors $a \in Z_q^n$ and their inner product with s *plus* some gaussian noise term. That is, a learner sees training data of the form $(a, \sum_{i=1}^{n} a_i \cdot s_i + err \pmod q)$ with the goal of reconstructing s. One can view this as a series of random linear equations on s, for instance:

$$\begin{aligned} 3s_1 + 8s_2 + 3s_3 + 6s_4 &\approx 1 \pmod{11} \\ 7s_1 + 9s_2 + 10s_3 + 6s_4 &\approx 3 \pmod{11} \\ 2s_1 + 2s_2 + 3s_3 + 9s_4 &\approx 7 \pmod{11} \\ 3s_1 + 9s_2 + 3s_3 + 10s_4 &\approx 0 \pmod{11} \end{aligned}$$

where each equation is within ± 1 of the correct value (here the answer is $s = (5, 10, 4, 6)$).

Without the error, solving the equations could be done using elimination. With the error, the problem becomes intractable. The best known algorithm presented in [3] requires $2^{\mathcal{O}(n)}$ samples and time. Regev [14] showed that the problem reduces to worst-case lattice problems, for which there are no known quantum algorithms. Therefore, LWE is a natural candidate for a range cryptographic constructions.

A simple example of using LWE for public key encryption is shown below, where all additions are taken modulo q:

- Private Key: vector s chosen randomly from Z_q^n
- Public Key: m = samples (a_i, b_i) where $b_i = <a_i, s> + err$
- Encryption: chose a random set S from m, $Enc(0) = (\sum_{i \in S} a_i, \sum_{i \in S} b_i)$
  $Enc(1) = (\sum_{i \in S} a_i, \lfloor \frac{q}{2} \rfloor \sum_{i \in S} b_i)$
- Decryption: given (a, b), return 0 if $<a, s>$ is nearer to 0 than to $\lfloor \frac{q}{2} \rfloor$ modulo q and 1 otherwise

where q is chosen to be a large prime and m is on the order of $\log q$. Other cryptographic applications of LWE include secret key encryption, key exchange, homomorphic encryption and much more.

Because of the strong hardness guarantees that the LWE problem entails, we will likely see a migration to crypto-systems based on this problem as opposed to problems based on the hardness of factoring. Factoring based schemes like RSA can be easily broken by quantum algorithms, whereas schemes based on the hardness of lattice problems are thought to be resistant to quantum attacks. Of course, we would like messages encrypted today to also be secure if and when quantum computers are prevalent.

IV. Neural Cryptography

A recent area of research at the cross-section of machine learning and cryptography has been applying neural networks to derive crypto-systems. Physicists Kanter, Kinzel and Kanter [6] proposed using synchronizing neural networks as a means of key exchange (a common scenario in which two parties want to communicate securely, but first have to agree on a key). Their scheme uses two neural networks each trying to learn from the other's outputs when given common random inputs.

Both parties, Alice and Bob, use a tree parity machine with K perceptrons denoted σ_k, each with randomly initialized integer weight vectors w_k, as shown in figure 1. The protocol proceeds over a series of rounds, where on each round both machines receive identical input vectors x_k (where each coordinate takes a value $\in \{-1, +1\}$) and calculate the following for perceptrons k to K:

$$\begin{aligned} \sigma_k &= sgn(\vec{w}_k \cdot \vec{x}_k) \\ \tau &= \prod_{k=1}^{K} \sigma_k. \end{aligned} \tag{4}$$

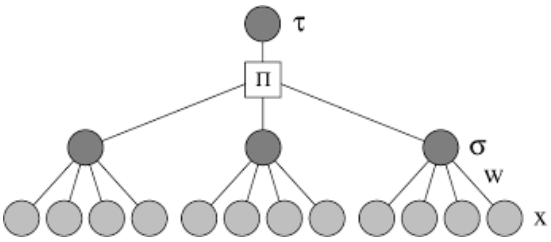

Figure 1

In words, each perceptron computes the sign of the sum of it's weights times it's inputs and the output of the network is simply the product of all the perceptrons.

If $\tau_A \neq \tau_B$ no update takes place and they generate another random input. If the outputs are equal, the perceptrons which agree with the output update their weights using the Hebbian rule: $\vec{w_k} -= \sigma_k \cdot \vec{x_k}$. The authors showed that after roughly 1000 iterations the two networks converge on identical weight vectors. These can then be hashed and used as a shared secret key.

Kilmov, Mityagin and Shamir [10] showed this protocol is vulnerable to several attacks. One such attack uses a genetic algorithm in which a large population of neural networks are initialized and those agreeing with the outputs of the Alice and Bob networks replicate. That is, a successful network is replaced by a number of new networks, one for each possible combination of perceptrons whose product would agree with the final output. This particular attack was successful in more than 50% of tests. It remains to be seen whether other neural architectures could confer the same functionality without compromising security.

More recently, work by Abadi and Anderson [1] has been done using generative adversarial networks (GANs) to develop novel crypto-systems. GANs are a class of algorithms in which a system of neural networks are initialized with conflicting goals. In this scenario, networks, Alice and Bob, attempt to communicate 16 bit messages using a shared secret while a third network, Eve, attempts to recover the original messages. The architecture of the networks consists of a fully-connected layer followed by a sequence of convolutional layers. The authors intended to create an architecture sufficient to learn mixing functions such as XOR, without encoding any particular algorithm.

The Eve network attempts to minimize her L_1 distance between her predicted message and the actual message. The Alice and Bob networks attempt to minimize Bob's L_1 distance minus a term dependent on Eve's L_1 distance. The training consists of rounds of mini-batch stochastic gradient descent, alternating between Alice and Bob training for one mini-batch and Eve training for two mini-batches. The training stops when Bob's reconstruction error falls below a threshold of .05 bits or after 150,000 steps, whichever comes first.

The results were promising, in 14 out of 20 trials Bob's error fell bellow the specified threshold while Eve's remained above 7.3 bits. Remarkably, these networks had no inbuilt knowledge of secure encryption, their constructions were entirely novel. Obviously this protocol is not yet robust enough for implementation, however the results demonstrate a promising area of future research.

V. Conclusion

This survey presented the myriad ways in which the fields of cryptography and learning are connected. Cryptography can be viewed as the opposite of learning. In learning, the goal is to uncover a rule underlying the given data whereas in cryptography, the goal is to ensure no such rule can be uncovered. Therefore, strong cryptographic protocols necessarily entail the unlearn-ability of certain classes of functions. Conversely, hard learning problems present a potential basis for crypto-systems. Finally, novel applications of neural networks provide promising new areas of research into how machine learning techniques can be used for secure communication.

VI. References

1. Abadi, M.; Andersen, D.G. Learning to protect communications with adversarial neural cryptography. arXiv 2016, arXiv:1610.06918.
2. D. Angluin and P. Laird. Learning from noisy examples. Machine Learning, 2(4):343– 370, 1988.
3. Avrim Blum, Adam Kalai, and Hal Wasserman. 2003. Noise-tolerant learning, the parity problem, and the statistical query model. J. ACM 50, 4 (July 2003), 506-519.
4. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. In Proceedings of the 25th IEEE Symposium on Foundations of Computer Science, pages 464-479, Singer Island, 1984. IEEE.
5. R. Impagliazzo, L. A. Levin, and M. Luby. 1989. Pseudo-random generation from one-way functions. In Proceedings of the twenty-first annual ACM symposium on Theory of computing (STOC '89), D. S. Johnson (Ed.). ACM, New York, NY, USA, 12-24.
6. I. Kanter, W. Kinzel and E. Kanter, Secure exchange of information by synchronization of neural networks, Europhys. Lett. 57, 141-147 (2002)
7. Michael Kearns. The Computational Complexity of Machine Learning. PhD thesis, Harvard University Center for Research in Computing Technology, May 1989. Technical Report TR-13-89. Also published by MIT Press as an ACM Distinguished Dissertation.
8. Michael Kearns and Leslie G. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. In Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, pages 433-444, Seattle, Washington, May 1989.
9. M. Kharitonov, Cryptographic hardness of distribution-specific learning, in: Proceedings of the 25th Symposium on Theory of Computing (STOC), 1993.
10. Klimov A., Mityagin A., Shamir A. (2002) Analysis of Neural Cryptography. In: Zheng Y. (eds) Advances in Cryptology — ASIACRYPT 2002. ASIACRYPT 2002. Lecture Notes in Computer Science, vol 2501. Springer, Berlin, Heidelberg
11. A. R. Klivans and A. A. Sherstov, "Cryptographic Hardness for Learning Intersections of Halfspaces," 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), Berkeley, CA, 2006, pp. 553-562.
12. Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. 2012. Foundations of Machine Learning. The MIT Press.
13. L. Pitt, M.K. Warmuth. Reductions among prediction problems: on the difficulty of predicting automata. Proceedings of the 3rd I.E.E.E. Conference on Structure in Complexity Theory, 1988, pp. 60-69.
14. O. Regev. On lattices, learning with errors, random linear codes, and cryptography.
15. O. Regev. The learning with errors problem (invited survey). In IEEE Conference on Computational Complexity, pages 191–204. 2010. Journal of the ACM, 56(6):34, 2009. Preliminary version in STOC'05.
16. Ronald L. Rivest. 1991. Cryptography and Machine Learning. In Proceedings of the International Conference on the Theory and Applications of Cryptology: Advances in Cryptology (ASIACRYPT '91), Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto (Eds.). Springer-Verlag, London, UK, UK, 427-439.
17. R. L. Rivest, A. Shamir, and L. Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21, 2 (February 1978), 120-126.
18. Shai Shalev-Shwartz and Shai Ben-David. 2014. Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press, New York, NY, USA.
19. Leslie G. Valiant. A theory of the learnable. Communications of the ACM, 27(11):1134-1142, November 1984.