Informatics Research Review

School of Informatics, University of Edinburgh

Literature Review

The Modern Relationship Between Cryptography And Machine Learning

Gregory D. Hill - s1771232

School of Informatics **January 2018**

The Modern Relationship Between Cryptography And Machine Learning

1 Introduction

Rivest [22] summarises the entire relationship between the two "sister fields" of cryptography and machine learning by explaining how the two are somewhat paradoxical. Cryptographers want to define 'security' beyond reasonable doubt, typically such that a protocol cannot be broken by any polynomial-time algorithm. Contrarily, much of learning theory relates to the representation of an unknown function in a way that allows an agent to make accurate predictions. Therefore, if a cryptographic algorithm is learnable, we may question whether it was ever truly secure.

Cryptanalysis is an effervescent field which assumes a great deal of knowledge on the target scheme. It typically entails a course mathematical treatment of a subject algorithm to discover vulnerabilities. For example, given the encrypted ciphertext an ideal objective would be to uncover the decrypted plaintext. Rivest [22] studied it in some depth, and provided an example of where learning theory proved helpful in the cryptanalysis of a non-linear feedback shift-register. So elements of learning theory undoubtedly play an important role in manual investigation, but the automated analysis of such insecurity is rare. Without the aid of a cryptanalyst, an intelligent agent would likely only have access to the known-plaintext, not the specific design of the subject black-box cipher. In addition, many (if not all) publicly utilised cryptosystems will have been previously scrutinised by professionals to ensure that the output is indistinguishable from that of a stochastic process. Nevertheless, distinguished success has been shown in using genetic algorithms [8] to assess the randomness of ciphertext - though they are arguably a metaheuristic, not a learning algorithm. These systems are also tailored to suit, so presently, full cryptanalytic autonomy does not seem feasible. A lot of success has been shown in applying machine learning to side channel analysis [16] however. Though these attacks typically exploit information leakage from the physical implementation of the cryptographic algorithm rather than the design itself. For example, a learning algorithm may study the power usage to aid in the extraction of a key. This field of study has grown in recent years, most significantly due to the rise in popularity of deep learning - which can effectively model high-level abstractions.

The aim of this study is to build upon the original discussion by Rivest [22] and explore the prevailing research areas which have been founded on the intersection between cryptography and machine learning. It will focus on literature with a more mathematical inference, which unfortunately means that topics like side channel analysis [16] will be overlooked. Section 2 dissects further background material linked to the study by Rivest [22], specifically in relation to computational complexity. Section 3 evaluates literature where the causation of learning theory on cryptography has had striking consequences in terms of improving protocol design and increasing security. Section 4 discusses recent advancements in distributed computation and efforts to protect sensitive data. The review will be concluded with a short passage (Section 5) on the current state of the reviewed fields, in which several problems will be identified for future analysis.

2 Background

One of the first formalizations on the relationship between cryptography and machine learning came in the form of a doctoral dissertation from Harvard in 1989. Kearns [13] extensively defines the mathematical foundations needed to describe the ‘computational efficiency of learning algorithms’ based on the Distribution-Free Model (DNF), also known as the Probably Approximately Correct (PAC) model - which is used to track a model’s accuracy given the size of training data and diagnose the complexity for any given accuracy requirement. This was first introduced by Valiant [27] who presented the plausibility of learning machines - applications that acquire computational abilities without explicit programming. He describes several conceptual boolean expressions which were found to be learnable in polynomial time. Valiant [27] foreshadows the opinions later highlighted by Rivest [22] on the implied intractability of learning due to easily computable cryptographic algorithms, though he does prove the feasibility in some restricted non-trivial teaching scenarios. Kearns [13] explains these cryptographic limitations in more detail. Given a malicious adversary Eve, who can be viewed as a learning algorithm, the other parties, Alice and Bob, would like to communicate over an unprotected channel. A naïve implementation is a one time pad, where the two honest parties meet in person to exchange keys. Though, it is actually commonplace to use a public-key system for ease which use openly available *trapdoor functions* $f(x)$ that map an input x to an output that is computationally infeasible to invert (f^{-1}) without the aid of some key. Although learning this function should not be possible, in comparison to the one time pad it is ‘fair’ in that it has a small circuit which Eve could *possibly* learn. Kearns [13] dissects this challenge in an effort to refine several simplistic functions based on common number theoretic principals widely used in cryptography to prove their associated learning difficulty. He poises three elementary problems based on RSA, Quadratic Residues (QRs) and factoring Blum integers to prove their intractability. He formulates an interesting idea: cryptography from non-learnability. This summarises the idea of building a secure protocol from any ‘parameterized Boolean representation class’ that cannot be learnt in polynomial time - an example of this is given in Section 3.

There are two main results to assess in the computational difficulty of learning. The first is known as “representation-dependent” hardness which seeks to verify the learning algorithm’s inability to learn C given its chosen hypothesis class H. Conversely, “representation-independent” hardness results consider the intrinsic difficulty of prediction without a set hypothesis. Kearns and Valiant [12] analyse the latter set of results for the distribution-free learning of various arbitrary representational classes to show that if a polynomial time algorithm were to exist which solved even these elementary problems then it would introduce disastrous consequences in cryptography. This is due to the intractable number theoretic principles which underpin the independent hardness results, including those poised above [13], such as inverting the RSA function. Under standard assumptions, they show that their hardness results are not weakly learnable. In particular, they experiment with a candidate trapdoor function for each of the learning problems to show how the complexity of inverting the respective function given the key is difficult.

3 Cryptographic Improvements

Learning theory is often used as a tool to re-design cryptographic schemes. For instance, Arvandi et al. [3] presented a symmetric cipher design based on a Real-Time Recurrent Neural Network (RRNN). This had the advantage of negating key size limitations with adaptive security versus performance to suitably scale for different architectures. Arvandi and Sadeghian [2] previously described the security of such schemes, and proved that they were resistant to brute-force, differential and linear attacks. However, the author also outlined a possible chosen-plaintext attack due to the ciphertext’s input dependence. The architecture illustrated by Arvandi et al. [3] denoted an output layer half of the input layer’s size, filtered through a hidden layer with one unit ξ. The ‘secret key’ is formed from the input and output vectors with a critical value α - used to denote the learning error limit for the training routine to reach. Final network weights form the extended key to be shared by all participants, so a complex model with more layers and/or hidden units than initially described by Arvandi et al. [3] would vastly increase security, but at great cost to performance. It builds on earlier research by Wu [28] who constructed and analysed a block cipher using Recurrent Neural Networks (RNNs). The scheme was inherently capable of high-performance parallelization and provided both data integrity and authentication but suffered from a sensitive learning rate that could deviate the model's learning trajectory if set too small.

The field of research which pertains to the creation of learned cryptosystems is typically referred to as "Neural Cryptography". Kinzel and Kanter [14] [24] formally define this as the act of exchanging secret keys over a public channel by mutual learning. These methods typically use Tree Parity Machines (TPMs) which are a type of Multilayer Perceptron (MLP) - a feed-forward subset of Artificial Neural Network (ANN). They comprise a fixed size input and output with a variable number of hidden layers containing "units" (or neurons) which typically employ some non-linear activation function. Kinzel and Kanter [14] use K=3 hidden units in their experimental network as illustrated in Figure 1.

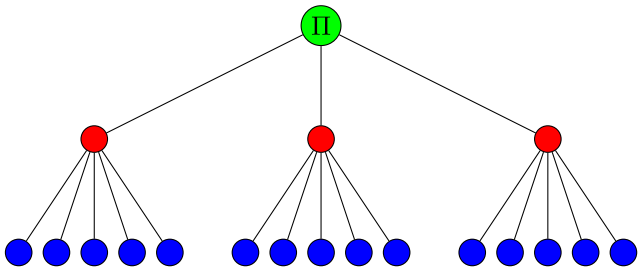

Figure 1: Tree Parity Machine

The inputs (blue) take $x_{ij} \in \{-1, 0, +1\}$ which feed to the hidden layer through the weights. Each hidden neuron's activation function (red) takes the sum of all associated inputs and weights, multiplied, and forwards the signum function - which basically returns $-1, 0, 1$ on condition. The binary output (green) $\tau = \prod_{i=1}^{K} \sigma_i$ is then synchronised according to the training routine, where at each step, the two parties will both receive three input vectors to guide the weight distribution. This system has many advantages over conventional schemes based on number theoretic principles in that it is arithmetically far simpler and has a distinctly lower computational complexity $O(N)$ [14] compared to that of other algorithms, for example key generation in RSA which is $O(N^2)$. The authors briefly discuss possible attacks, but conclude that their scheme is resilient without a more complete analysis. Fortunately, Ruttor et al. [25] reviewed the attack surface four years later. An initial assessment of older attacks, namely *geometric* and *majority*, justified their negation in networks of greater depth. They focused the remainder of their analysis however on *genetic attacks* and formulated a adversarial method to quickly generate the optimal neural network.

Not every learning problem is easily computable, Learning With Errors (LWE) is known to many for its intractability. The problem stems from the difficulty in which vectors generated by some 'noisy' set of linear equations can be distinguished between uniformly random vectors [21]. It has recently been incorporated as a hardness assumption into public-key cryptosystems [20] and is thought to enable post-quantum cryptography - which is widely concerned issue [9]. The classical systems based on older hardness assumptions, like factoring, are unlikely to disappear any time soon however. For instance, Potter et al. [18] investigated machine learning driven solutions for QR detection. A QR is an integer q that satisfies the following condition: $x^2 \equiv q \ (mod\ n)$. Factoring the square root is computationally difficult, so it has played a significant role in the past - most notably, the oblivious transfer [19]. No polynomial-time algorithm currently exists for determining the existence of x for any given q, so Potter et al. [18] attempted to infer a closed-form algorithm through a black box methodology. Unfortunately their solution was not able to scale for larger key sizes, with accurate performance up to 20 bits. Training time was super-polynomial and imitated the Jacobi approximation algorithm of QRs from composite numbers. So it is plausible that at some point in the future machine learning solutions may break these classical systems, but an easily computable solution does not yet exist.

4 Distributed Learning

Practitioners of contemporary machine learning systems often require a substantial quantity of data to fit their problem [10, 16], but data collection is a challenging and expensive process. To avoid the intrinsic cost, companies may want to buy and share data without detriment to their own clientèle. Netflix cancelled their open competition on film rating prediction in 2010 when researchers were able to de-anonymize their perturbed dataset to identify individuals [17]. Without explicit legal substance, trading in data is a challenging and possibly dangerous process. Yet in today's market, it has never been more essential - particularly due to the high consumer demand for tailored and efficient services [17]. Various subversion resilient learning schemes have been trialled in recent years, the most popular of which will be discussed here.

Agrawal and Srikant [1] proposed a data reconstruction procedure to allow external parties to estimate the original distribution of perturbed data without loss of integrity. They garnered the distorted data through client-side modification of sensitive values using *value-class membership* (restricting input to a categorical format) and *value distortion* (with random Uniform / Gaussian noise). They identified two efficient decision tree classifiers which utilised Bayesian methodologies for correction. This approach was later extended to logistic regression [5], motivated by the concept of *differential privacy* described by Dwork [6] on the formulation of risk an individual participant incurs when participating in a database. Agrawal and Srikant [1] guarantee a high level of privacy with their approach, but it is unlikely that simple perturbation can fully prevent the de-anonymization problem [17], especially as a potential adversary would likely have complete access to the dataset. Even with some loss of precision, in many cases it could still effectually leak sensitive data [6]. Therefore, to fully afford distributed machine learning, such that an external (honest) party will ultimately benefit from the private data without risk of adverse exposure, modern literature has focused on more extreme measures.

4.1 Encryption

The explicit encryption of any data in question can enable a learning algorithm to study the ciphertext in a unique way. For example, Graepel et al. [11] prove that the computational effort of a learning task can be outsourced to an external service while retaining client confidentiality through a Homomorphic Encryption (HE) scheme. These methods allow arithmetical operations to be performed on ciphertext such that the decrypted result will equal the relative operation from the plaintext. It was proposed by Rivest et al. [23] in 1978 to ensure the privacy of transformed data, but the process was vastly inefficient. Over the last decade however, significant progress has been made. Gentry [9] extensively defined the first Fully Homomorphic Encryption (FHE) scheme in 2009 for computing an arbitrary number of operations. This construction has many use cases, including private search engine queries, but it is distinctly suited to the distributed learning task. ML Confidential [11] assesses Somewhat Homomorphic Encryption (SHE) and Leveled Homomorphic Encryption (LHE) schemes (for their practicality in low-complexity scenarios) to augment their linear classifiers. The authors then show that their cloud service, provided with encrypted and labelled training vectors, is able to learn an encrypted model. Unfortunately the encryption scheme's cost of computation was relatively high which prohibited scale. Hence experimentation was trialled with fewer training and test examples than would be ideal in a typical learning task. They show that the time cost rapidly increases with more data and features. Bost et al. [4] also proposed a FHE scheme and devised a number of privacy-preserving classification tools explicitly for use on encrypted data. They experimented with a number of protocols and implemented a full 'building blocks library' for constructing additional modular classifiers. The authors also addressed several concerns with ML Confidential [11] regarding the simplistic nature of their chosen algorithm and the inefficiency of their final comparison.

Deep learning has become hugely popular in recent years due to the representational power of ANNs. Contemporary benchmarks on the widely tested Modified National Institute of Standards and Technology (MNIST) database rarely deviate too far from a classification accuracy of 100%. This is also true of the model described by Gilad-Bachrach et al. [10], which was shown to achieve an accuracy rate of 99% on the encrypted MNIST data. Their methodology and resulting implementation, named *CryptoNets*, utilised LHE to store and train an unconventional neural network comprising a series of approximated and scaled layers. The innovative approach disproved the stereotypical inefficiency of homomorphic schemes with a high throughput of around 59000 predictions per hour, far better then previous schemes [11, 4].

4.2 Multi-Party Computation (MPC)

Distributed learning can also be accomplished with secure MPC - sharing a computational task between two or more parties, in this case data mining. It has garnered significant popularity in recent years due to it's proposed usages in electronic voting [29] and cryptocurrencies. The general problem was introduced by Yao [29] in 1982. Suppose m users wish to compute the integer-valued function $f(x_1, \dots x_m)$, how does one calculate the result without knowing all inputs? This is slightly different to the mutual learning problem described by Kinzel and Kanter [14] in that neither party would have access to the other's data.

The most common distributed algorithm is the decision tree. It is a classification tool that comprises a set of leaves which stem from a root node. Data is sorted from the root to a particular leaf which classifies the instance. Hence, each node is a test of some attribute, and its branches specify the values it can take on. For example, a learned decision tree is often represented by if-then statements. The tree itself is first generated by some algorithm which performs a greedy search down through the space where each node is trained by some statistical test that is evaluated based on its discriminative performance. One popular choice [15, 7] for this task is known as the Iterative Dichotomiser 3 (ID3) algorithm which selects an attribute with the smallest entropy and then iterates through those that remain. Lindell and Pinkas [15] discuss how two users may compute the union of their databases through a particular mining algorithm without risking adverse data leakage using an efficient protocol which boosts a distributed ID3 algorithm. The authors observed that each 'leaf' could be computed separately before making the output public and moving onto the following node which is a special case in such privacy protocols, as intermediate values are not typically revealed. Lindell and Pinkas [15] further develop their protocol, but their assessment of complexity indicates congestion in a few places - such as when executing the oblivious transfer. Their overall design however was hugely efficient, requiring mere minutes to compute a million transactions on a standard PC.

A recently proposed scheme by Fong and Weber-Jahnke [7] also utilised ID3, but their approach was vastly more inefficient compared to that of Lindell and Pinkas [15]. One interesting component however, was the use of 'unrealized data sets' to balance the trade-off between local and shared privacy. By perturbing the data in a similar way to Agrawal and Srikant [1] they were able to convert the two sets into a third 'unrealized' set that could later be reconstructed. Each party was only required to store their encoded table, which meant that the scheme was found to be resilient against many smaller data leakages, but could be easily broken on mass due to the generic state of their reconstruction.

5 Conclusion

Back in 1991 when Rivest [22] first conducted his study on the interplay between machine learning and cryptography, the focus was primarily on the complexity of learning algorithms and using cryptographic assumptions to disprove their utility. He also described a few cryptanalytic problems in which he reasoned about the use of neural networks and concluded with a brief prelude to data compression. Over the following two decades there has been a dramatic shift in the particular fields of interest. Underlying cryptographic assumptions remain for the most part, unchanged, [8] and vendors rarely employ undocumented primitives. Therefore, the vast majority of cryptanalytic research in the way described by Rivest [22] on the level of cipher design, since this study was first published, remains unchanged.

This report has summarised the dominant research fields that currently exist on the compliment of machine learning and cryptography. Broadly speaking, there are two main contemporary implications of their relationship: learning theory often serves to improve or create new cryptographic schemes, including recently proposed quantum algorithms [20], but cryptography is also used to 'harden' distributed learning systems. There are many additional areas of intrigue that were not explored in this text, for example, side channel analysis was briefly discussed in Section 1 but there are many other uses of ANNs. Schmidt et al. [26] provide an in-depth review of additional areas in Pseudo-Random Number Generation (PRNG) and Steganography where these techniques have proven useful.

5.1 Cryptography

Learned cryptosystems are still in their infancy but, if the research is to be believed, could prove immeasurably more secure. One of the main problems in this area however is that most of the work discussed in Section 3 is over a decade old. It is unclear as to exactly why there are no recent advancements, maybe Kinzel and Kanter [14] realised it was too impractical or perhaps the impact caused by the discovery of *genetic attacks* [25] discouraged interest. This is obviously pure conjecture but further analysis would clearly prove beneficial - especially as computational resources have grown dramatically in the last decade, which could enable far deeper networks in the case of mutual learning.

5.2 Data Mining

Data leakage is of great concern to any corporation, whether it be from a malicious insider or external threat. Secure multi-party computation can alleviate many of the security issues inherent to simple perturbation and enable mutual learning without directly sharing sensitive data, but if an adversary is able to access the original distribution directly then privacy is still a concern. Research on HE [9] has found it to be more efficient overall, especially as it eliminates networking issues [15], but it is not without its problems. Homomorphic methods, despite recent work done by Gilad-Bachrach et al. [10], are still expensive.

References

[1] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. In *ACM Sigmod Record*, volume 29, pages 439–450. ACM, 2000.

[2] Maryam Arvandi and Alireza Sadeghian. Analysis of neural network based ciphers. Master of applied science - electrical and computer engineering, Ryerson University, jun 2003.

[3] Maryam Arvandi, Shuwei Wu, Alireza Sadeghian, William W Melek, and Isaac Woungang. Symmetric cipher design using recurrent neural networks. In *Neural Networks, 2006. IJCNN'06. International Joint Conference on*, pages 2039–2046. IEEE, 2006.

[4] Raphael Bost, Raluca Ada Popa, Stephen Tu, and Shafi Goldwasser. Machine learning classification over encrypted data. In *NDSS*, 2015.

[5] Kamalika Chaudhuri and Claire Monteleoni. Privacy-preserving logistic regression. In *Advances in Neural Information Processing Systems*, pages 289–296, 2009.

[6] Cynthia Dwork. Differential privacy. In *Encyclopedia of Cryptography and Security*, pages 338–340. Springer, 2011.

[7] P. K. Fong and J. H. Weber-Jahnke. Privacy preserving decision tree learning using unrealized data sets. *IEEE Transactions on Knowledge and Data Engineering*, 24(2):353–364, Feb 2012. ISSN 1041-4347. doi: 10.1109/TKDE.2010.226.

[8] Aaron Garrett, John Hamilton, and Gerry Dozier. A comparison of genetic algorithm techniques for the cryptanalysis of tea. 12, 01 2007.

[9] Craig Gentry. *A fully homomorphic encryption scheme*. Stanford University, 2009.

[10] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International Conference on Machine Learning*, pages 201–210, 2016.

[11] Thore Graepel, Kristin Lauter, and Michael Naehrig. Ml confidential: Machine learning on encrypted data. In *International Conference on Information Security and Cryptology*, pages 1–21. Springer, 2012.

[12] Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the ACM (JACM)*, 41(1):67–95, 1994.

[13] Michael J Kearns. *The computational complexity of machine learning*. MIT press, 1990.

[14] Wolfgang Kinzel and Ido Kanter. Neural cryptography. In *Neural Information Processing, 2002. ICONIP'02. Proceedings of the 9th International Conference on*, volume 3, pages 1351–1354. IEEE, 2002.

[15] Yehuda Lindell and Benny Pinkas. Privacy preserving data mining. In *Advances in Cryptology-CRYPTO 2000*, pages 36–54. Springer, 2000.

[16] Houssem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff. Breaking cryptographic implementations using deep learning techniques. In *International Conference on Security, Privacy, and Applied Cryptography Engineering*, pages 3–26. Springer, 2016.

[17] Arvind Narayanan and Vitaly Shmatikov. How to break anonymity of the netflix prize dataset. *arXiv preprint cs/0610105*, 2006.

[18] M. Potter, L. Reznik, and S. Radziszowski. Neural networks and the search for a quadratic residue detector. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1887–1894, May 2017. doi: 10.1109/IJCNN.2017.7966080.

[19] Michael O Rabin. How to exchange secrets with oblivious transfer. *IACR Cryptology ePrint Archive*, 2005:187, 2005.

[20] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.

[21] Oded Regev. The learning with errors problem. *Invited survey in CCC*, page 15, 2010.

[22] Ronald L Rivest. Cryptography and machine learning. In *International Conference on the Theory and Application of Cryptology*, pages 427–439. Springer, 1991.

[23] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

[24] Michal Rosen-Zvi, Einat Klein, Ido Kanter, and Wolfgang Kinzel. Mutual learning in a tree parity machine and its application to cryptography. *Physical Review E*, 66(6):066135, 2002.

[25] Andreas Ruttor, Wolfgang Kinzel, Rivka Naeh, and Ido Kanter. Genetic attack on neural cryptography. *Physical Review E*, 73(3):036121, 2006.

[26] T Schmidt, H Rahnama, and A Sadeghian. A review of applications of artificial neural networks in cryptosystems. In *Automation Congress, 2008. WAC 2008. World*, pages 1–6. IEEE, 2008.

[27] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[28] Shuwei Wu. A block cipher design using recurrent neural networks. Master of engineering - electrical and computer engineering, Ryerson University, 2003.

[29] Andrew C Yao. Protocols for secure computations. In *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on*, pages 160–164. IEEE, 1982.