Cryptography and Machine Learning

Ronald L. Rivest*
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

This paper gives a survey of the relationship between the fields of cryptography and machine learning, with an emphasis on how each field has contributed ideas and techniques to the other. Some suggested directions for future cross-fertilization are also proposed.

1 Introduction

The field of computer science blossomed in the 1940's and 50's, following some theoretical developments of the 1930's. From the beginning, both cryptography and machine learning were intimately associated with this new technology. Cryptography played a major role in the course of World War II, and some of the first working computers were dedicated to cryptanalytic tasks. And the possibility that computers could "learn" to perform tasks, such as playing checkers, that are challenging to humans was actively explored in the 50's by Turing [46], Samuel [39], and others. In this note we examine the relationship between the fields of cryptography and machine learning, emphasizing the cross-fertilization of ideas, both realized and potential.

The reader unfamiliar with either of these fields may wish to consult some of the excellent surveys and texts available for background reading. In the area of cryptography, there is the classic historical study of Kahn [20], the survey papers of Diffie and Hellman [11] and Rivest [37], and Simmons [44], as well as the texts by Brassard [8], Denning [10], and Davies and Price [9], among others. The CRYPTO and EUROCRYPT conference proceedings (published by Springer) are also extremely valuable sources. In the area of machine learning, there are standard collections of papers [29, 30, 23] for "AI" style machine learning, the seminal paper of Valiant [47] for the "computational learning theory" approach, the COLT conference proceedings (published by Morgan Kaufmann) for additional material of a theoretical nature, and the NIPS conference proceedings (also

*Supported by NSF grant CCR-8914428, ARO grant N00014-89-J-1988, and the Siemens Corporation.
email address: rivest@theory.lcs.mit.edu

published by Morgan Kaufmann) for many interesting papers. The ACM STOC and the IEEE FOCS conference proceedings also contain many key theoretical papers from both areas. The Ph.D. thesis of Kearns [21] is one of the first major works to explore the relationship between cryptography and machine learning, and is also an excellent introduction to many of the key concepts and results.

2 Initial Comparison

Machine learning and cryptanalysis can be viewed as "sister fields," since they share many of the same notions and concerns. In a typical cryptanalytic situation, the cryptanalyst wishes to "break" some cryptosystem. Typically this means he wishes to find the secret key used by the users of the cryptosystem, where the general system is already known. The decryption function thus comes from a known family of such functions (indexed by the key), and the goal of the cryptanalyst is to exactly identify which such function is being used. He may typically have available a large quantity of matching ciphertext and plaintext to use in his analysis. This problem can also be described as the problem of "learning an unknown function" (that is, the decryption function) from examples of its input/output behavior and prior knowledge about the class of possible functions.

Valiant [47] notes that good cryptography can therefore provide examples of classes of functions that are hard to learn. Specifically, he references the work of Goldreich, Goldwasser, and Micali [14], who demonstrate (under the assumption that one-way functions exist) how to construct a family of "pseudo-random" functions $F_k : \{0,1\}^k \rightarrow \{0,1\}^k$ for each $k \geq 0$ such that (i) each function $f_i \in F_k$ is described by a k-bit index i, (ii) there is a polynomial-time algorithm that, on input i and x, computes $f_i(x)$ (so that each function in F_k is computable by a polynomial-size boolean circuit), and (iii) no probabilistic polynomial-time algorithm can distinguish functions drawn at random from F_k from functions drawn at random from the set of all functions from $\{0,1\}^k$ to $\{0,1\}^k$, even if the algorithm can dynamically ask for and receive polynomially many evaluations of the unknown function at arguments of its choice. (It is interesting to note that Section 4 of Goldreich et al. [14] makes an explicit analogy with the problem of "learning physics" from experiments, and notes that their results imply that some such learning problems can be very hard.)

We now turn to a brief comparison of terminology and concepts, drawing some natural correspondences, some of which have already been illustrated in above example.

Secret Keys and Target Functions

The notion of "secret key" in cryptography corresponds to the notion of "target function" in machine learning theory, and more generally the notion of "key space" in cryptography corresponds to the notion of the "class of possible target functions." For cryptographic (encryption) purposes, these functions must also be efficiently invertible, while no such requirement is assumed in a typical machine learning context. There is another aspect of this correspondence in which the fields differ: while in cryptography it is common to assume that the size of the unknown key is known to the cryptanalyst (this usually falls under the general assumption that "the general system is known"), there is much interesting research in machine learning theory that assumes that the complexity (size) of

the target hypothesis is *not* known in advance. A simple example of this phenomenon is the problem of fitting a polynomial to a set of data points (in the presence of noise), where the degree of the true polynomial is not known in advance. Some method of trading off "complexity of hypothesis" for "fit to the data," such as Rissanen's Minimum Description Length Principle [36], must be employed in such circumstances. Further research in cryptographic schemes with variable-length keys (where the size of key is not known to the cryptanalyst) might benefit from examination of the machine learning literature in this area.

Attack Types and Learning Protocols

A critical aspect of any cryptanalytic or learning scenario is the specification of how the cryptanalyst (learner) may gather information about the unknown target function.

Cryptographic attacks come in a variety of flavors, such as *ciphertext only*, *known plaintext (and matching ciphertext)*, *chosen plaintext*, and *chosen ciphertext*. Cryptosystems secure against one type of attack may not be secure against another. A classic example of this is Rabin's signature algorithm [35], for which it is shown that a "passive" attack—forging a signature knowing only the public signature verification key—is provably as hard as factorization, whereas an "active attack"—querying the signer by asking for his signature on some specially constructed messages—is devastating and allows the attacker to determine the factorization of the signer's modulus—a total break.

The machine learning community has explored similar scenarios, following the pioneering work of Angluin [2, 3]. For example, the learner may be permitted "membership queries"—asking for the value of the unknown function on some specified input—or "equivalence queries"—asking if a specified conjectured hypothesis is indeed equivalent to the unknown target hypothesis. (If the conjecture is incorrect, the learner is given a "counterexample"—an input on which the conjectured and the target functions give different results.) For example, Angluin [2] has shown that a polynomial number of membership and equivalence queries are sufficient to exactly identify any regular set (finite automaton), whereas the problem of learning a regular set from random examples is NP-complete [1].

Even if information is gathered from random examples, cryptanalytic/learning scenarios may also vary in the prior knowledge available to the attacker/learner about the distribution of those examples. For example, some cryptosystems can be successfully attacked with only general knowledge of the system and knowledge of the language of the plaintext (which determines the distribution of the examples). While there is some work within the machine learning community relating to learning from known distributions (such as the uniform distribution, or product distributions [40]), and the field of "pattern recognition" has developed many techniques for this problem [12], most of the modern research in machine learning, based on Valiant's PAC-learning formalization of the problem, assumes that random examples are drawn according to an arbitrary but fixed probability distribution that is unknown to the learner. Such assumptions seem to have little relevance to cryptanalysis, although techniques such as those based on the "theory of coincidences" [25, Chapter VII] can sometimes apply to such situations. In addition, we have the difference between the two fields in that PAC-learning requires learning for all underlying probability distributions, while cryptographic security is typically defined as security no matter what the underlying distribution on messages is.

Exact versus Approximate Inference

In the practical cryptographic domain, an attacker typically aims for a "total break," in which he determines the unknown secret key. That is, he exactly identifies the unknown cryptographic function. Approximate identification of the unknown function is typically not a goal, because the set of possible cryptographic functions used normally does not admit good approximations. On the other hand, the theoretical development of cryptography has focussed on definitions of security that exclude even approximate inference by the cryptanalyst. (See, for example, Goldwasser and Micali's definitions in their paper on probabilistic encryption [15].) Such theoretical definitions and corresponding results are thus applicable to derive results on the difficulty of (even approximately) learning, as we shall see.

The machine learning literature deals with both exact inference and approximate inference. Because exact inference is often too difficult to perform efficiently, much of the more recent research in this area deals with approximate inference. (See, for example, the key paper on learnability and the Vapnik-Chervonenkis dimension by Blumer et al. [7].) Approximate learning is normally the goal when the input data consists of randomly chosen examples. On the other hand, when the learner may actively query or experiment with the unknown target function, exact identification is normally expected.

Computational Complexity

The computational complexity (sometimes called "work factor" in the cryptographic literature) of a cryptanalytic or learning task is of major interest in both fields.

In cryptography, the major goal is to "prove" security under the broadest possible definition of security, while making the weakest possible complexity-theoretic assumptions. Assuming the existence of one-way functions has been a common such weakest possible assumption. Given such an assumption, in the typical paradigm it is shown that there is no polynomial-time algorithm that can "break" the security of the proposed system. (Proving, say, exponential-time lower bounds could presumably be done, at the expense of making stronger initial assumptions about the difficulty of inverting a one-way function.)

In machine learning, polynomial-time learning algorithms are the goal, and there exist many clever and efficient learning algorithms for specific problems. Sometimes, as we shall see, polynomial-time algorithms can be proved not to exist, under suitable cryptographic assumptions. Sometimes, as noted above, a learning algorithm does not know in advance the size of the unknown target hypothesis, and to be fair, we allow it to run in time polynomial in this size as well. Often the critical problem to be solved is that of finding a hypothesis (from the known class of possibile hypotheses) that is consistent with the given set of examples; this is often true even if the learning algorithm is trying merely to approximate the unknown target function.

For both cryptanalysis and machine learning, there has been some interest in minimizing space complexity as well as time complexity. In the cryptanalytic domain, for example, Hellman [18] and Schroeppel and Shamir [42] have investigated space/time trade-offs for breaking certain cryptosystems. In the machine learning literature, Schapire has shown the surprising result [41, Theorem 6.1] that if there exists an efficient learning algorithm for a class of functions, then there is a learning algorithm whose space complexity grows only logarithmically in the size of the data sample needed (as ϵ, the approximation parameter, goes to 0).

Unicity Distance and Sample Complexity

In his classic paper on cryptography [43], Shannon defines the "unicity distance" of a cryptosystem to be $H(K)/D$, where $H(K)$ is the entropy of the key space K (the number of bits needed to describe a key, on the average), and where D is redundancy of the language (about 2.3 bits/letter in English). The unicity distance measures the amount of ciphertext that must be intercepted in order to make the solution unique; once that amount of ciphertext has been intercepted, one expects there to be a unique key that will decipher the ciphertext into acceptable English. The unicity distance is an "information-theoretic" measure of the amount of data that a cryptanalyst needs to succeed in exactly identifying the unknown secret key.

Similar information-theoretic notions play a role in machine learning theory, although there are differences arising from the fact that in the standard PAC-learning model there may be infinitely many possible target hypotheses, but on the other hand only an approximately correct answer is required. The Vapnik-Chervonenkis dimension [7] is a key concept in coping with this issue.

Other differences

The effect of *noise* in the data on the cryptanalytic/learning problem has been studied more carefully in the learning scenario than the cryptanalytic scenario, probably because a little noise in the cryptanalytic situation can render analysis (and legitimate decryption) effectively hopeless. However, there are cryptographic systems [48, 28, 45] that can make effective use of noise to improve security, and other (analog) schemes [49, 50] that attempt to work well in spite of possible noise.

Often the inference problems studied in machine learning theory are somewhat more general than those that occur naturally in cryptography. For example, work has been done on target concepts that "drift" over time [19, 24]; such variability is rare in cryptography (users may change their secret keys from time to time, but this is dramatic change, not gradual drift). In another direction, some work [38] has been done on learning "concept hierarchies"; such a framework is rare in cryptography (although when breaking a substitution cipher one may first learn what the vowels are, and then learn the individual substitutions for each vowel).

3 Cryptography's impact on Learning Theory

As noted earlier, Valiant [47] argued that the work of Goldreich, Goldwasser, and Micali [14] on random functions implies that even approximately learning the class of functions representable by polynomial-size boolean circuits is infeasible, assuming that one-way functions exist, even if the learner is allowed to query the unknown function. So researchers in machine learning have focussed on the question of identifying which simpler classes of functions are learnable (approximately, from random examples, or exactly, with queries). For example, a major open question in the field is whether the class of boolean functions representable as boolean formulas in disjunctive normal form (DNF) is efficiently learnable from random examples.

The primary impact of cryptography on machine learning theory is a natural (but negative) one: showing that certain learning problems are computationally intractable. Of

course, there are other ways in which a learning problem could be intractable; for example, learning the class of all boolean functions is intractable merely because the required number of examples is exponentially large in the number of boolean input variables.

A certain class of intractability results for learning theory are *representation dependent*: they show that given a set of examples, finding a consistent boolean function *represented in a certain way* is computationally intractable. For example, Pitt and Valiant [32] show that finding a 2-term DNF formula consistent with a set of input/output pairs for such a target formula is an NP-complete problem. This implies that learning 2-term DNF is intractable (assuming $P \neq NP$), but only if the learner is required to produce his answer in the form of a 2-term DNF formula. The corresponding problem for functions representable in 2-CNF (CNF with two literals per clause, which properly contains the set of functions representable in 2-term DNF) *is* tractable, and so PAC-learning the class of functions representable in 2-term DNF is possible, as long as the learner may output his answers in 2-CNF. Similarly, Angluin [1] has shown that it is NP-complete to find a minimum-size DFA that is consistent with a given set of input/output examples, and Pitt and Warmuth [33] have extended this result to show that finding an *approximately* minimum-size DFA consistent with such a set of examples is impossible to do efficiently. These representation-dependent results depend on the assumption that $P \neq NP$.

In order to obtain hardness results that are *representation-independent*, Kearns and Valiant [22, 21] turned to cryptographic assumptions (namely, the difficulty of inverting RSA, the difficulty of recognizing quadratic residues modulo a Blum integer, and the difficulty of factoring Blum integers). Of course, they also need to explain how learning could be hard in a representation-independent manner, which they do by requiring the learning algorithm not to output a hypothesis in some representation language, but rather to *predict* the classification of a new example with high accuracy.

Furthermore, in order to prove hardness results for learning classes of relatively simple functions, Kearns and Valiant needed to demonstrate that the relevant cryptographic operations could be specified within the desired function class. This they accomplished by the clever technique of using an "expanded input" format, wherein the input was arranged to contain suitable auxiliary results as well as the basic input values. They made use of the distribution-independence of PAC-learning to assume that the probability distribution used would not give any weight to inputs where these auxiliary results were incorrect.

By these means, Kearns and Valiant were able to show that PAC-learning the following classes of functions is intractable (here $p(n)$ denotes some fixed polynomial):

1. "Small" boolean formulae: the class of boolean formula on n boolean inputs whose size is less than $p(n)$.

2. The class of deterministic finite automata of size at most $p(n)$ that accept only strings of length n.

3. For some fixed constant natural number d, the class of threshold circuits over n variables with depth at most d and size at most $p(n)$.

They show that if a learning algorithm could efficiently learn any of these classes of functions, in the sense of being able to predict with probability significantly greater than

1/2 the classification of new examples, then the learning algorithm could be used to "break" one of the cryptographic problems assumed to be hard.

The results of Kearns and Valiant were also based on the work of Pitt and Warmuth [34], who develop the notion of a "prediction-preserving reducibility." The definition implies that if class A is so reducible to class B, then if class B is efficiently predictable, then so is class A. Using this notion of prediction-preserving reducibility, they show a number of classes of functions to be "prediction-complete" for various complexity classes. In particular, the problem of prediction the class of alternating DFAs is shown to be prediction-complete for P, and ordinary DFAs are as hard to predict as any function computable in log-space, and boolean formula are prediction-complete for NC^1. These results, and the notion of prediction-preserving reducibility, were central to the work of Kearns and Valiant.

The previous results assumed a learning scenario in which the learner was working from random examples of the input/output behavior of the target function. One can ask if cryptographic techniques can be employed to prove that certain classes of functions are unlearnable even if the learner may make use of queries. Angluin and Kharitonov [5] have done so, showing that (modulo the usual cryptographic assumptions regarding RSA, quadratic residues, or factoring Blum integers), that there is no polynomial-time prediction algorithm, *even if membership queries are allowed*, for the following classes of functions:

1. Boolean formulas,
2. Constant-depth threshold circuits,
3. Boolean formulas in which every variable occurs at most 3 times,
4. Finite unions or intersections of DFAs, 2-way DFAs, NFAs, or CFGs.

These results are based on the public-key cryptosystem of Naor and Yung [31], which is provably secure against a chosen-ciphertext attack. (Basically, the queries asked by the learner get translated into chosen-ciphertext requests against the Naor-Yung scheme.)

4 Learning Theory's impact on Cryptography

Since most of the negative results in learning theory already depend on cryptographic assumptions, there has been no impact of negative results on learning theory on the development of cryptographic schemes. Perhaps some of the results and concepts and Pitt and Warmuth [34] could be applied in this direction, but this has not been done.

On the other hand, the positive results in learning theory are normally independent of cryptographic assumptions, and could in principle be applied to the cryptanalysis of relatively simple cryptosystems. Much of this discussion will be speculative in nature, since there is little in the literature exploring these possibilities. We sketch some possible approaches, but leave their closer examination and validation (either theoretical or empirical) as open problems.

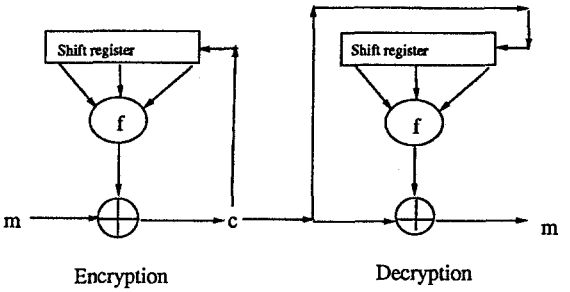

Figure 1: In cipher-feedback mode, each plaintext message bit m is encrypted by exclusive-oring it with the result of applying the function f to the last n bits of ciphertext, where n is the size of the shift register. The ciphertext bit c is transmitted over the channel; the corresponding decryption process is illustrated on the right.

Cryptanalysis of cipher-feedback systems

Perhaps the most straightforward application of learning results would be for the cryptanalysis of nonlinear feedback shift-registers operating in cipher-feedback mode. See Figure 1. The feedback function f is known only to the sender and the receiver; it embodies their shared "secret key."

If the cryptanalyst has a collection of matching plaintext/ciphertext bits, then he has a number of corresponding input/output pairs for the unknown function f. A learning algorithm that can infer f from such a collection of data could then be used as a cryptanalytic tool. Moreover, a chosen-ciphertext attack gives the cryptanalyst the ability to query the unknown function f at arbitrary points, so a learning algorithm that can infer f using queries could be an effective cryptanalytic tool.

We note that a definition of learnability that permits "approximate" learning is a good fit for this problem: if the cryptanalyst can learn an approximation to f that agrees with f 99% of the time, then he will be able to decrypt 99% of the plaintext.

Suppose first that we consider a known plaintext attack. Good cryptographic design principles require that f be about as likely to output 0 as it is to output 1. This would typically imply that the shift register contents can be reasonably viewed as a randomly drawn example of $\{0,1\}^n$, *drawn according to the uniform distribution.* (We emphasize that it is this assumption that makes are remarks here speculative in nature; detailed analysis or experimentation is required to verify that this assumption is indeed reasonable in each proposed direction.) There are a number of learning-theory results that assume that examples are drawn from $\{0,1\}^n$ according to the uniform distribution. While this assumption seems rather unrealistic and restrictive in most learning applications, it is a perfect match for such a cryptographic scenario. What cryptographic lessons can be

drawn from the learning-theory research?

Schapire [40] shows how to efficiently infer a class of formula he calls "probabilistic read-once formula" against product distributions. A special case of this result implies that a formula f constructed from AND, OR, and NOT gates can be exactly identified (with high probability) in polynomial time from examples drawn randomly from the uniform distribution. (It is an open problem to extend this result to formula involving XOR gates in a useful way; some modification would be needed since the obvious generalization isn't true.) Perhaps the major lesson to be drawn here is that in the simplest formula for f each shift register bit should be used several times.

Linial, Mansour, and Nisan [27] have shown how to use spectral (Fourier transform) techniques to learn functions f chosen from AC^0 (constant depth circuit of AND/OR/NOT gates having arbitrarily large fan-in at each gate) from examples drawn according to the uniform distribution. Kushelevitz and Mansour [26] have extended these results to the class of functions representable as decision trees. Furst, Jackson, and Smith [13] have elaborated and extended these results in a number of directions. We learn the lesson that the spectral characteristics of f need to be understood and controlled.

Hancock and Mansour [17] have similarly shown that monotone $k\mu$ DNF formulae (that is, monotone DNF formulae in which each variable appears at most k times) are learnable from examples drawn randomly according to the uniform distribution. Although monotone formula are not really useful in this shift-register application, the negation of a monotone formula might be. We thus have another class of functions f that should be avoided.

When chosen-ciphertext attacks are allowed, function classes that are learnable with membership queries should be avoided. For example, Angluin, Hellerstein, and Karpinski [4] have shown how to exactly learn any μ-formula using members and equivalence queries. Similarly, Hancock [16] shows how to learn 2μ DNF formula (not necessarily monotone), as well as $k\mu$ decision trees, using queries.

We thus see a potential "pay-back" from learning theory to cryptography: certain classes of inferrable functions should probably be avoided in the design of non-linear feedback shift registers used in cipher-feedback mode. Again, we emphasize that verifying that the proposed attacks are theoretically sound or empirically useful remains an open problem. Nonetheless, these suggestions should provide some useful new guidelines to those designing such cryptosystems.

Other possibilities

We have seen some successful applications of continuous optimization techniques (such as gradient descent) to discrete learning problems; here the neural net technique of "back propagation" comes to mind. Perhaps such techniques could also be employed successfully in cryptanalytic problems.

Another arena in which cryptography and machine learning relate is that of *data compression*. It has been shown by Blumer et al. [6] that pac-learning and data compression are essentially equivalent notions. Furthermore, the security of an encryption scheme is often enhanced by compressing the message before encrypting it. Learning theory may conceivably aid cryptographers by enabling ever more effective compression algorithms.

Acknowledgments

I would like to thank Rob Schapire for helpful comments.

References

[1] Dana Angluin. On the complexity of minimum inference of regular sets. *Information and Control*, 39:337–350, 1978.

[2] Dana Angluin. A note on the number of queries needed to identify regular languages. *Information and Control*, 51:76–87, 1981.

[3] Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, April 1988.

[4] Dana Angluin, Lisa Hellerstein, and Marek Karpinski. Learning read-once formulas with queries. Technical report, University of California, Report No. UCB/CSD 89/528, 1989.

[5] Dana Angluin and Michael Kharitonov. When won't membership queries help? In *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, pages 444–454, New Orleans, Louisiana, May 1991.

[6] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Occam's razor. *Information Processing Letters*, 24:377–380, April 1987.

[7] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

[8] Gilles Brassard. *Modern Cryptology.* Springer-Verlag, 1988. Lecture Notes in Computer Science Number 325.

[9] D. W. Davies and W. L. Price. *Security for Computer Networks: An Introduction to Data Security in Teleprocessing and Electronic Funds Transfer.* John Wiley and Sons, New York, 1984.

[10] D. E. Denning. *Cryptography and Data Security.* Addison-Wesley, Reading, Mass., 1982.

[11] W. Diffie and M. E. Hellman. Privacy and authentication: An introduction to cryptography. *Proceedings of the IEEE*, 67:397–427, March 1979.

[12] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis.* Wiley, 1973.

[13] Merrick Furst, Jeffrey Jackson, and Sean Smith. Improved learning of AC^0 functions. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 317–325, Santa Cruz, California, August 1991.

[14] 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.

[15] S. Goldwasser and S. Micali. Probabilistic encryption. *JCSS*, 28(2):270–299, April 1984.

[16] Thomas Hancock. Learning 2μ DNF formulas and $k\mu$ decision trees. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 199–209, Santa Cruz, California, August 1991.

[17] Thomas Hancock and Yishay Mansour. Learning monotone $k\mu$ DNF formulas on product distributions. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 179–183, Santa Cruz, California, August 1991.

[18] M. E. Hellman. A cryptanalytic time-memory trade off. *IEEE Trans. Inform. Theory*, IT-26:401–406, 1980.

[19] David P. Helmbold and Philip M. Long. Tracking drifting concepts using random examples. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 13–23, Santa Cruz, California, 1991. Morgan Kaufmann.

[20] D. Kahn. *The Codebreakers.* Macmillian, New York, 1967.

[21] 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.

[22] 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.

[23] Yves Kodratoff and Ryszard S. Michalski, editors. *Machine Learning: An Artificial Intelligence Approach*, volume III. Morgan Kaufmann, Los Altos, California, 1990.

[24] Anthony Kuh, Thomas Petsche, and Ronald L. Rivest. Learning time-varying concepts. In *Proceedings 1990 Conference on Computation Learning and Natural Learning (Princeton)*, 1990. To appear.

[25] Solomon Kullback. *Statistical Methods in Cryptanalysis.* Aegean Park Press, 1976.

[26] Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. In *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, pages 455–464, New Orleans, Louisiana, May 1991. ACM.

[27] Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. In *Proceedings of the Thirtieth Annual Symposium on Foundations of Computer Science*, pages 574–579, Research Triangle Park, North Carolina, October 1989.

[28] R. J. McEliece. *A Public-Key System Based on Algebraic Coding Theory*, pages 114–116. Jet Propulsion Lab, 1978. DSN Progress Report 44.

[29] Ryszard S. Michalski, Jaime G. Carbonell, and Tom M. Mitchell, editors. *Machine Learning: An Artificial Intelligence Approach*, volume I. Morgan Kaufmann, Los Altos, California, 1983.

[30] Ryszard S. Michalski, Jaime G. Carbonell, and Tom M. Mitchell, editors. *Machine Learning: An Artificial Intelligence Approach*, volume II. Morgan Kaufmann, Los Altos, California, 1986.

[31] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attack. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, pages 427–437, Baltimore, Maryland, 1990. ACM.

[32] Leonard Pitt and Leslie G. Valiant. Computational limitations on learning from examples. *Journal of the ACM*, 35(4):965–984, 1988.

[33] Leonard Pitt and Manfred K. Warmuth. The minimum DFA consistency problem cannot be approximated within any polynomial. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, Seattle, Washington, May 1989.

[34] Leonard Pitt and Manfred K. Warmuth. Prediction preserving reducibility. *Journal of Computer and System Sciences*, 41:430–467, 1990.

[35] M. Rabin. Digitalized signatures as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, January 1979.

[36] Jorma Rissanen. *Stochastic Complexity in Statistical Inquiry*, volume 15 of *Series in Computer Science*. World Scientific, 1989.

[37] Ronald L. Rivest. Cryptography. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science (Volume A: Algorithms and Complexity)*, chapter 13, pages 717–755. Elsevier and MIT Press, 1990.

[38] Ronald L. Rivest and Robert Sloan. Learning complicated concepts reliably and usefully. In *Proceedings AAAI-88*, pages 635–639, August 1988.

[39] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3:211–229, July 1959. (Reprinted in *Computers and Thought*, (eds. E. A. Feigenbaum and J. Feldman), McGraw-Hill, 1963, pages 39–70).

[40] Robert E. Schapire. Learning probabilistic read-once formulas on product distributions. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 184–198, Santa Cruz, California, August 1991.

[41] Robert Elias Schapire. *The Design and Analysis of Efficient Learning Algorithms.* PhD thesis, MIT EECS Department, February 1991. (MIT Laboratory for Computer Science Technical Report MIT/LCS/TR-493).

[42] R. Schroeppel and A. Shamir. A $TS^2 = O(2^n)$ time/space tradeoff for certain NP-complete problems. In *Proc. 20th Annual IEEE Symposium on Foundations of Computer Science*, pages 328–336, San Juan, Puerto Rico, 1979. IEEE.

[43] Claude Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28:656–715, October 1949.

[44] G. J. Simmons. Cryptology. In *The New Encyclopædia Brittanica*, pages 860–873. Encyclopædia Brittanica, 1989. (Volume 16).

[45] N. J. A. Sloane. Error-correcting codes and cryptography. In D. Klarner, editor, *The Mathematical Gardner*, pages 346–382. Wadsworth, Belmont, California, 1981.

[46] A. M. Turing. Computing machinery and intelligence. *Mind*, 59:433–460, October 1950. (Reprinted in *Computers and Thought*, (eds. E. A. Feigenbaum and J. Feldman), McGraw-Hill, 1963, pages 11–38).

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

[48] A. D. Wyner. The wire-tap channel. *Bell Sys. Tech. J.*, 54:1355–1387, 1975.

[49] A. D. Wyner. An analog scrambling scheme which does not expand bandwidth, part 1. *IEEE Trans. Inform. Theory*, IT-25(3):261–274, 1979.

[50] A. D. Wyner. An analog scrambling scheme which does not expand bandwidth, part 2. *IEEE Trans. Inform. Theory*, IT-25(4):415–425, 1979.