Distinguishing exceptions from noise in non-monotonic learning

Ashwin Srinivasan,
Stephen Muggleton,
Michael Bain
The Turing Institute,
36 North Hanover Street,
Glasgow G1 2AD,
UK.

Abstract

It is important for a learning program to have a reliable method of deciding whether to treat errors as noise or to include them as exceptions within a growing first-order theory. We explore the use of an information-theoretic measure to decide this problem within the non-monotonic learning framework defined by Closed-World-Specialisation. The approach adopted uses a model that consists of a reference Turing machine which accepts an encoding of a theory and proofs on its input tape and generates the observed data on the output tape. Within this model, the theory is said to "compress" data if the length of the input tape is shorter than that of the output tape. Data found to be incompressible are deemed to be "noise". We use this feature to implement a compression-guided specialisation procedure that searches for the best-fitting theory for the data (that is, the one with the shortest input tape length). The approach is empirically evaluated on the standard Inductive Logic Programming problem of learning classification rules for the KRK chess end-game.

1 Introduction

Induction is an uncertain process. Scientific theories are ascribed various degrees of belief depending on how well they agree with known facts. As new information becomes available certain hypotheses may seem more likely and others less so. For instance consider the Julian calendar in which leap years were held to be necessary once every 4 years. This can be represented in Prolog with negation by failure as

normal(Year) :- year(Year), not leap4(Year).
leap4(Year) :- modulo(Year,4,0).

This rule is correct up to around one part in a hundred and so up until 1582 errors could simply be treated as noise. However after 1500 years the mismatch with astronomical measurements forced a revision of the calendar under Pope Gregory XIII. In the Gregorian calendar the rules can be written as follows.

leap4(Year) :- modulo(Year,4,0), not leap100(Year).
leap100(Year) :- modulo(Year,100,0), not leap400(Year).
leap400(Year) :- modulo(Year,400,0).

The aim of Inductive Logic Programming (ILP) is to automate the construction and revision of logical theories by using example facts and background knowledge [17]. In the case above, examples are of the form *normal(1581)* or *not(normal(1582))* and background knowledge would contain a definition of *modulo*. ILP methods based on closed-world specialisation [3] would progressively specialise the overgeneral clause *normal(Year)* by inventing (and generalising) new abnormality predicates (corresponding to *leap4*, *leap100* and *leap400*). This process is capable of generating the Gregorian calendar theory and has recently been used to construct a complete and correct solution for the standard KRK illegality problem from the machine learning literature [2]. However, a key issue remains to be addressed: there is no mechanism by which a non-monotonic learning strategy can reliably distinguish true exceptions from noise. For example, a strategy based on closed-world-specialisation would continue specialising until a correct theory is obtained. In noisy domains, this will necessarily result in fitting the noise. In this paper we explore the possibility of using a general information-theoretic model developed in [18, 21] to help distinguish noise from true exceptions. An important consequence of adopting this model is that theories found to be "compressive" (described below) are, with very high probability, significant. A simple search procedure is developed to find as compressive an explanation as possible for the data. Its results are evaluated empirically for the standard ILP problem of learning classification rules in the KRK chess end-game.

2 Information-Theoretic Evaluation of Hypotheses

In the 1950's Carnap [7] and others suggested "confirmation theories" aimed at providing a statistical underpinning to the problem of the plausibility of inductive inferences. Various difficulties and paradoxes were encountered with these approaches which meant that they were never applied within machine learning programs [16]. Instead, confidence in alternative hypotheses has for the most part relied on either *ad hoc* notions of simplicity (the Occam's razor principle)

or on statistical tests of significance based on the coverage of a rule and prior probability estimates of the classes present in the data ([9, 10]).

The choice of the most compact hypothesis is the basis of Rissanen's "Minimal Description Length" (MDL) principle[25]. This states that the best theory for explaining a set of data is one which minimises the sum of:

1. the description length of the theory in bits and
2. the description length of the data when encoded using the theory.

Within machine learning the MDL principle has been applied by [11] to determine the the best sampling rate for character recognition and by [24] to the problem of learning decision trees. However, its application to first-order learning remains largely unexplored. It forms the motivation for the encoding measure used in [23]. However, the simplifications result in a number of problems (identified in [10]). Muggleton [18] addresses this issue using a model related to algorithmic information theory ([26, 13, 8]). In his approach, the significance of a hypothesis is evaluated by comparing the length of the input and output tapes of a reference Turing machine. The components to be minimised in the MDL approach are represented on the input tape as a Horn clause theory and a proof specification. The latter specifies how the examples on the output tape are to be derived using the theory and background knowledge. (Figure 1: from [21]). A theory is deemed

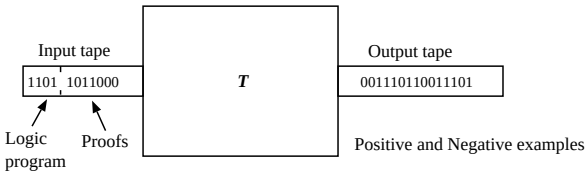

Figure 1: A Turing machine model for learning logic programs

significant if the length of the input tape (in bits) is shorter than that of the output tape (the theory and proofs are said to compress the data). This model has been empirically evaluated in [21] and shown to be better suited to learning first-order theories than the statistical measure used in [9, 10]. In the following section we describe a method of using the compression obtained from this approach to guide the progressive correction of first-order theories within a non-monotonic framework.

3 Compression-Based Non-Monotonic Induction

We incorporate the Turing machine compression model in Figure 1 within the non-monotonic learning framework developed by Muggleton and Bain ([3, 2]).

Their technique commences with an over-general logic program. This is then progressively corrected by a hierarchical decomposition strategy. At each level negated exception predicates are introduced (and generalised) to account for exceptions. Figure 2 shows an algorithm that performs the alternate operations of specialisation and generalisation characteristic of closed-world-specialisation.

It is worth noting here that:

1. As in [15], there is an assumption that the exceptions to a rule are fewer than the examples that satisfy it.

2. The call to *generalise* results in an attempt to induce a (possibly over-general) rule by a learning algorithm.

3. All rules are added to the theory. Further, all negative examples covered by an over-general clause are taken to be exceptions and the clause is specialised with a (new) abnormality predicate.

Each correction performed by the CWS algorithm is an attempt to improve the accuracy of the theory, at the expense of increasing its size. Clearly, if the correction was worthwhile, the gain in accuracy should outweigh the penalty incurred in increasing the theory size. In encoding terms, each correction increases the theory encoding on the input tape and decreases the proof encoding. In the model in Figure 1, a net decrease in the length of the input tape occurs when the correction succeeds in identifying some pattern in the errors (that is, the errors are not noise). The new theory consequently compresses the data further by exploiting this pattern. Using this feature, we evaluate the utility of updating a theory by checking for an increase in compression. We note the following consequences of using the compression model within the non-monotonic framework adopted:

1. Only compressive theories are deemed to be reliable in the model. Thus, while we can adopt the MDL principle of selecting the theory with the shortest input tape, we can be confident of not having fitted the noise only if the theory itself is compressive. Stated differently, we can be confident that highly compressive theories have avoided fitting the noise as much as possible.

2. With the closed-world assumption, all examples are covered. Consequently, the output tape has to be encoded only once. Input tapes for alternate theories are compared against this encoding.

3. Consider an over-general clause in the current theory. The proof encoding described in [21] ensures all variables of the clause are bound to ground terms. Specialising this clause involves adding a negated literal to its body. By appending this literal to the body, we are guaranteed that it will be ground. This ensures safety of the standard Prolog computation rule used by the Turing machine.

```
start:
   PosE = positive examples of target concept
   NegE = negative examples of target concept
   return learn(PosE,NegE)

learn(Pos,Neg):
   ClauseList = []
   repeat
      C = generalise(Pos,Neg)
      if C ≠ []
         PosC = positive examples covered by C
         NegC = negative examples covered by C
         Pos = Pos - PosC
         Neg = Neg - NegC
         ClauseList = ClauseList + (C,PosC,NegC)
   until C = []

   Theory = []
   foreach (Clause,PosC,NegC) in ClauseList
      if |NegC| ≠ 0
         Theory = Theory + specialise(Clause,PosC,NegC)
      else
         Theory = Theory + Clause
   return Theory

specialise(HornClause,Pos,Neg):
   hd(V1,...,Vn) = head of HornClause
   Body = body of HornClause
   ab = a new predicate symbol
   SpecialisedClause = hd(V1,...,Vn) ← Body, not(ab(V1,...,Vn))
   PosE = positive examples of ab formed from Neg
   NegE = negative examples of ab formed from Pos
   return SpecialisedClause + learn(PosE,NegE)
```

Figure 2: Non-monotonic inductive inference using closed-world-specialisation (CWS)

4. The proof encoding for each example has two parts: a choice-point specification and a proof tag. Since the negative literal appended to a clause can never create bindings, the choice-point specification remains unaltered. The size-accuracy trade-off referred to earlier therefore reduces to a trade-off between increasing theory size and decreasing tag size. Not having to recalculate the choice-point encoding for each specialisation is a major benefit as this is an extremely costly exercise.

While the aim is to obtain the most compressive subset of the clauses produced by the CWS algorithm, it is unnecessary to examine all subsets since clauses constructed as generalisations of an abnormality predicate cannot be considered independent of the parent over-general clause. For example, it makes no sense to consider the following set of clauses for explaining leap years:

normal(Year) :- year(Year), not leap4(Year).
leap400(Year) :- modulo(Year,400,0).

Despite this, there may still be an intractably large number of clause-sets to consider. Consequently, we adopt a greedy strategy of selecting clauses in order of those that give the most gain (in compression). This strategy has to confront two important issues: devising a reliable method of deciding on the "best" clause to add to the theory and the fact that adding this clause may not produce an immediate increase in compression.

A simple way to address the first problem is to select the clause that corrects the most errors. Since decreasing errors is the only way to shorten the input tape, the gains are larger for theories that make fewer errors. This works well if all clauses are of approximately the same descriptional complexity. A better estimate would account for the complexity of individual clauses as well. This can be done using average estimates of the cost of encoding predicates, functions and variables. In the experiments in the next section, this more sophisticated estimate has proved unnecessary. This is because the clauses fitting noisy data tend to correct fewer errors and therefore, considered later using the simpler estimate. For the other clauses, the gain from correcting errors dominates the loss from increased theory size.

To address the problem of local minima, it is clearly desirable to have a method of looking ahead to see if a (currently non-compressive) clause will be part of the final theory. To decide this, we calculate an estimate of the compression produced by the most accurate theory containing the clause. The clause is retained if this expected compression is better than the maximum achieved so far. Each time an actual increase in compression is produced, the theory is updated with all clauses that have been retained. Figure 3 shows how the estimate is calculated. The estimated compression will usually be optimistic because it it assumes that all errors can be compressed. However within the compression model adopted, it is extremely unlikely to get any more compression from a

estimate(Theory):

Ncorrect = number of examples correctly classified by Theory
Nmaximum = number of examples that the learner can hope to classify correctly
Outbits = length of output tape (in bits)
OldTheory = length of Theory (in bits)
OldTags = current length of correction tags (in bits)
Choices = length of choice-point encoding (in bits)

NewTheory = OldTheory $\times$ Nmaximum / Ncorrect
NewTags = length of correction tags to correctly classify Nmaximum examples
EstInbits = NewTheory + Choices + NewTags
return (Outbits - EstInbits)

Figure 3: Estimating the compression from a theory

theory that is completely correct on noisy data than from an incorrect one that leaves the noise uncompressed. Of course, one way to guarantee an optimistic estimate is to assume that there will be no increase in theory size (as opposed to the current scaled estimate). However, this gives no heuristic power and usually only prolongs a futile search for a correct theory. Figure 4 summarises the main steps in the compression-based selection of clauses as described here. The following points deserve attention:

1. At any given stage, only some clauses produced by CWS are candidates to be added to the theory (recall the earlier statement that over-general clauses have to be considered before their specialisations).

2. The "best" clause refers to the clause selected using the simple error-count measure, or the more sophisticated one that accounts for the estimated theory increase. To obtain the latter requires a knowledge of the number of predicate, function and variable symbols in the clause.

3. Consider the situation when the estimated compression from adding the "best" clause is no better than the compression already obtained. Figure 4 does not acknowledge the possibility that some of the other clauses can do better. It is possible to rectify this by progressively trying the "next best" clause until all clauses have been tried.

4. The procedure in Figure 4 is reminiscent of post-pruning in zero-order algorithms (the clauses are constructed first and then possibly discarded). A natural question that arises is whether it is possible to incorporate the

```
start:
   ClauseList = clauses produced by CWS
   return select_clauses(ClauseList)

select_clauses(ClauseList):
   Theory = PartialTheory = []
   Compression = 0
   repeat
      PotentialClauses = clauses in ClauseList that can be added to theory
      C = "best" clause in PotentialClauses
      if C ≠ []
         PartialTheory = PartialTheory + C
         NewCompression = compression of PartialTheory
         if NewCompression > Compression
            Theory = PartialTheory
            Compression = NewCompression
         else
            EstCompression = estimate(PartialTheory)
            if EstCompression ≤ Compression return Theory
   until C = []
   return Theory
```

Figure 4: Compression-based selection of clauses produced by CWS

```
% legal(WK_file,WK_rank,WR_file,WR_rank,BK_file,BK_rank)
legal(A,B,C,D,E,F) :- not ab00(A,B,C,D,E,F).
ab00(A,B,C,D,C,E) :- not ab11(A,B,C,D,C,E).
ab00(A,B,C,D,E,D) :- not ab12(A,B,C,D,E,D).
ab00(A,B,C,D,E,F) :- adj(A,E), adj(B,F).
ab00(A,B,A,B,C,D).
ab12(A,B,C,B,D,B) :- lt(A,D), lt(C,A).
ab12(A,B,C,B,D,B) :- lt(A,C), lt(D,A).
ab11(A,B,A,C,A,D) :- lt(B,D), lt(C,B).
ab11(A,B,A,C,A,D) :- lt(B,C), lt(D,B).
```

Figure 5: A complete and correct theory for KRK-legality

```
legal(A,B,C,D,E,F) :- not ab00(A,B,C,D,E,F).
ab00(A,B,C,D,C,E).
ab00(A,B,C,D,E,D).
ab00(A,B,C,D,E,F) :- adj(A,E), adj(B,F).
```

Figure 6: An "approximately correct" theory for KRK-legality

compression measure within the specialisation process The analogy to zero-order learning algorithms is whether tree pre-pruning is feasible. The answer is yes, and in practice may be preferred as it avoids inducing all clauses. The price to pay is that it may not be possible to estimate reliably the utility of a clause.

4 Empirical Evaluation

We evaluate the utility of using compression as a reliable noise detector on the standard ILP problem of learning classification rules for the KRK chess endgame [19]. However, contrary to normal practice, we chose to learn rules for KRK-legality (as opposed to KRK-illegality). This provides an extra level of exceptions for the specialisation method. Given background knowledge of the predicates *lt/2* and *adj/2*, Figure 5 shows the target theory. It is possible to achieve an accuracy of about 99.6% without accounting for the second level of exceptions. In fact, the theory shown in Figure 6 is about 98% correct.

For our experiments, we adopt a simple noise model termed the *Classification*

Noise Process (CNP) [1]. In this, a noise of η implies that (independently) for each example, the sign of the example is reversed with probability η. This is not the only random noise process possible. For example, a noise of η in our model corresponds to a class-value noise of 2η in that adopted by [22] and Donald Michie (private communication) advocates a process that preserves the underlying distribution of positive and negative examples. Finally, although the procedure described in Figure 4 is not dependent on any particular induction algorithm, the results quoted here use Golem ([20]).

Figure 7 tabulates the percentage accuracy of the most compressive theory for different noise levels. Here "accuracy" refers to accuracy on an independent (noise-free) test set of 10000 examples. Since the compression model only guarantees reliability for compressive theories, nothing can be said about those for which compression is less than 0 (irrespective of their accuracy on the test set). In Figure 7, an entry of "_" denotes that the theory obtained is non-compressive on the training data and consequently, no claim is made regarding its accuracy on the test set. The results highlight some important points. Compressive theories do appear to avoid fitting the noise to a large extent. The price for this reliability is reflected in the amount of data required. In comparison, it is possible that other techniques may require fewer examples. However, they either require various parameters to be set ([10]), use *ad hoc* constraints ([23]) or need an additional data set for pruning ([6]). Further, most of them are unable to offer any guarantee of reliability (the approach followed in [10] can select clauses above a user-set significance threshold). In this respect, our empirical results mirror PAC ([27]) results for learning with noisy data in propositional domains ([1]): with increasing noise, more examples are needed to obtain a good theory. It is also worth noting that the conditions covered by the second level of exceptions (the cases in which the White King is in between the White Rook and Black King) occur less than 4 times in every 1000 examples. This is only picked up in the noise-free data set of 10000 examples (in which there were 38 examples where the rules applied).

Extending the PAC analogy further, Figure 8 shows the results from a different perspective. For different levels of noise, this figure shows the number of training examples required for the "approximately correct" theory of Figure 6 to be compressive. For example, at least 170 examples are required to obtain a compressive theory that is 98% accurate on noise-free data. While these numbers are approximate (they are obtained by extrapolating the compression produced by the theory for the different training sets in Figure 7) they do indicate the general trend of requiring larger example sets for increasing noise levels.

	Training Set Size					
Noise (%)	100	250	500	1000	5000	10000
0	_	99.7	99.7	99.7	99.7	100
5	_	98.1	98.1	99.7	99.7	99.7
10	_	_	98.1	98.1	99.7	99.7
15	_	_	98.1	98.1	99.7	99.7
20	_	_	_	98.1	99.7	99.7
30	_	_	_	_	98.1	98.1
40	_	_	_	_	_	98.1

Figure 7: Test-set accuracy for the most compressive theory

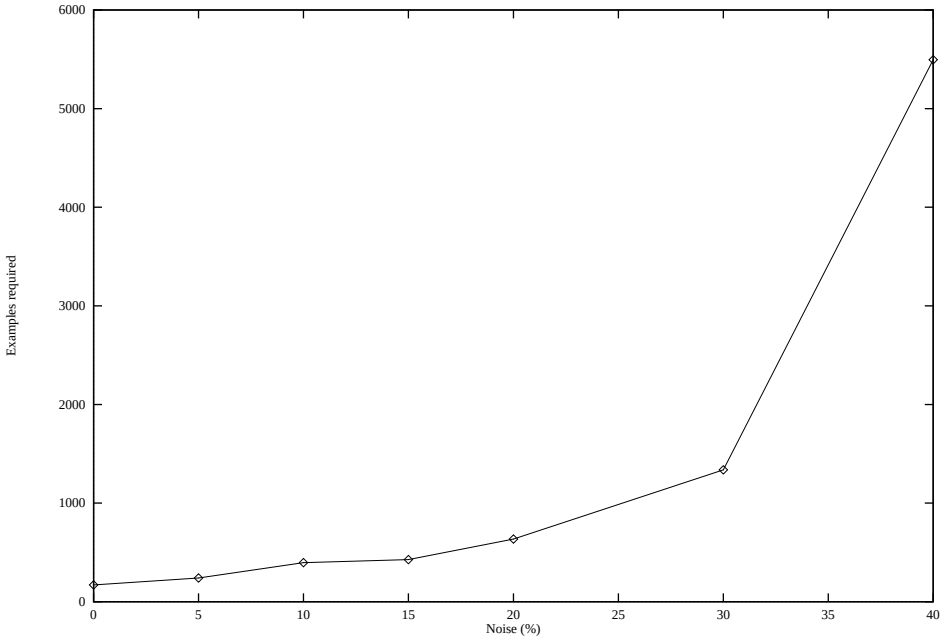

Figure 8: Examples required for a 98% correct and compressive theory

5 Conclusion

The task of distinguishing between exceptions and noise is an issue that is typically ignored in the literature on non-monotonic reasoning. It is, however, of fundamental importance for a learning program that has to construct theories using real-world data. One way to approach the problem is to see if the exceptions to the current theory exhibit a pattern. The compression model we have used in this paper uses an information-based approach to check whether the pattern detected warrants specialising the theory. While it can be formulated as an implementation of the Minimal Description Length principle, more significant is the fact that theories found to be compressive in the model are unlikely to have detected chance patterns. Our empirical results suggest that by selecting the most compressive theory, it is possible (given enough data) to reliably avoid fitting most of the noise. Clearly, it would be desirable to confirm these results with controlled experiments in other domains. In practice, the method has found interesting rules on an independent problem of pharmaceutical drug design ([12]). Finally, the results also lend support to the link between compressive theories for first-order concepts and their PAC-learnability. While various authors have shown such a connection exists ([4, 5, 14]), it would be nice to show that their concept of compression fits that used here.

Acknowledgements. The authors would like to thank Donald Michie and the ILP group at the Turing Institute for their helpful discussions and advice. This work was carried out at the Turing Institute and was supported by the Esprit Basic Research Action project ECOLES, the IED's Temporal Databases and Planning project and the SERC Rule-Base Systems Project. Stephen Muggleton is supported by an SERC post-doctoral fellowship.

References

[1] D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

[2] M. Bain. Experiments in non-monotonic learning. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 380–384, San Mateo, CA, 1991. Morgan Kaufmann.

[3] M. Bain and S. Muggleton. Non-monotonic learning. In D. Michie, editor, *Machine Intelligence 12*. Oxford University Press, 1991.

[4] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Classifying learnable geometric concepts with the Vapnik-Chervonenkis dimension. In *Proceedings of the 18th ACM Symposium on Theory of Computing*, pages 273–282, 1986.

[5] R. Board and L. Pitt. On the necessity of occam algorithms. Uiucdcs-r-89-1544, University of Illinois at Urbana-Champaign, 1989.

[6] C.A. Brunk and M.J. Pazzani. An investigation of noise-tolerant relational concept learning algorithms. In L. Birnbaum and G.C. Collins, editors, *Proceedings of the Eighth International Workshop on Machine Learning*, San Mateo, 1991. Morgan Kaufmann.

[7] R. Carnap. *The Continuum of Inductive Methods.* Chicago University, Chicago, 1952.

[8] G. Chaitin. *Information, Randomness and Incompleteness - Papers on Algorithmic Information Theory.* World Scientific Press, Singapore, 1987.

[9] P. Clark and T. Niblett. The CN2 algorithm. *Machine Learning*, 3(4):261–283, 1989.

[10] S. Dzeroski. *Handling Noise in Inductive Logic Programming.* University of Ljubljana, (M.Sc. Thesis), Ljubljana, 1991.

[11] Q. Gao and M.Li. An application of minimum description length principle to online recognition of handprinted numerals. In *IJCAI-89*, Detroit, MI, 1989. Kaufmann.

[12] R. King, S. Muggleton, and M.J.E. Sternberg. Drug design by machine learning. *submitted to Journal of the National Academy of Sciences*, 1991.

[13] A.N. Kolmogorov. Three approaches to the quantitative definition of information. *Prob. Inf. Trans.*, 1:1–7, 1965.

[14] M. Li and P.M.B. Vitanyi. Inductive reasoning and Kolmogorov complexity. In *Proceedings of the Fourth Annual IEEE Structure in Complexity Theory Conference*, pages 165–185, 1989.

[15] J. McCarthy. Applications of circumscription to formalizing common sense knowledge. *Artificial Intelligence*, 28:89–116, 1986.

[16] H. Mortimer. *The Logic of Induction.* Ellis Horwood, Chichester, England, 1988.

[17] S. Muggleton. Inductive logic programming. *New Generation Computing*, 8(4):295–318, 1991.

[18] S.H. Muggleton. A strategy for constructing new predicates in first order logic. In *Proceedings of the Third European Working Session on Learning*, pages 123–130. Pitman, 1988.

[19] S.H. Muggleton, M.E. Bain, J. Hayes-Michie, and D. Michie. An experimental comparison of human and machine learning formalisms. In *Proceedings of the Sixth International Workshop on Machine Learning*. Kaufmann, 1989.

[20] S.H. Muggleton and C. Feng. Efficient induction of logic programs. In *Proceedings of the First Conference on Algorithmic Learning Theory*, Tokyo, 1990. Ohmsha.

[21] S.H. Muggleton, A. Srinivasan, and M.E. Bain. Compression, significance and accuracy. *to appear: International Machine Learning Conference*, 1992.

[22] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[23] J.R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

[24] J.R. Quinlan and R.L. Rivest. Inferring decision trees using the Minimum Description Length principle. *Information and Computation*, 80:227–248, 1989.

[25] J. Rissanen. Modeling by Shortest Data Description. *Automatica*, 14:465–471, 1978.

[26] R.J. Solomonoff. A formal theory of inductive inference. *Information and Control*, 7:376–388, 1964.

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