Model Growth

Jeroen van Maanen
Jeroen.van.Maanen@xs4all.nl

Abstract

The algorithmic method of inductive inference that Ray Solomonoff proposes in (Solomonoff, 1964) is not interactive. Marcus Hutter defines how to add interactivity to the inductive method based on the assumption that the environment supplies a utility function (Hutter, 2000). This paper discusses the possibility of a framework based on a utility function that is internal to the learning subject and independent of the environment. The internal utility function should measure the amount of information extracted from the interaction with the environment. The Minimum Description Length principle (MDL) proposed by Jorma Rissanen (Rissanen, 1989) supplies a framework that clearly separates a statistical model that represents the extracted information from the exact representation of the data. Algorithmic Statistics (Gács et al., 2001) should be able to bridge the gap between the algorithmic approach of Solomonoff and the statistical approach of Rissanen.

1 Introduction

Learning is not only about predicting the right answer to questions. The hard part of learning is often to ask the right questions. See also (van Maanen, 2002). Marcus Hutter defines how to add interactivity to the inductive method based on the assumption that the environment supplies a utility function. His reason for this setup is that (Hutter, 2000)

> eventually we (humans) will be the environment with which the system will communicate and *we* want to dictate what is good and what is wrong, not the other way round.

If the context would be control theory this would be a reasonable assumption, but the context is artificial intelligence. Intelligent systems determine their own priorities. Even when the environment is highly optimised for teaching the teachers have to apply techniques that try to align the learning goals of the pupils with the teaching goals of the school. Even then, many students aim for average grades that let them pass major milestones with a more or less safe margin rather than maximal grades. They optimise a function that differs significantly from the externally supplied utility function.

What would be a more natural utility function for a learning system than its own learning rate? A learning system can be seen as a system that communicates with its environment and builds an internal model of that environment. The growth of that model, or the complexity of it, can be used as an exact measure for the learning rate of the learning system. When the results on Algorithmic Statistics (Gács et al., 2001) are applied to a slightly modified version of Hutters model called AIξ[1] we can expect to find that it is possible to define a universal autonomous learning system.

2 Example

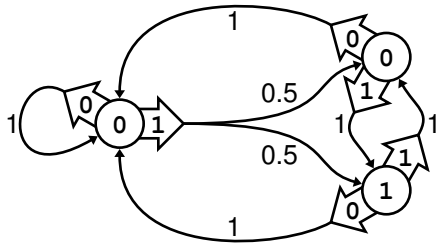
We will introduce an example of an environment that is stripped of all real-world complexities. This example is used in later sections to illustrate relevant aspects of formal models of learning. We use some abstract terminology to remind us where we would like to apply our results. With the *subject* we mean the person, animal or hypothetical device that learns. With the *environment* we mean the real or simulated environment that can be accessed by the subject. With the *interface* we mean the sensors and actors with which the subject interacts with

[1] AIξ is pronounced 'aixi' and can be written as 'AIXI' when Greek letters are not available.

the environment. The interaction between the subject and the environment results can be seen as a sequence of *events*. Each event is composed of an *action* that is performed by the subject and a *response* from the environment. We assume that actions as well as responses can be finitely encoded.

We will closely follow the notation used by Hutter. See (Hutter, 2000) for details and discussion. Without loss of generality we can assume that there are prefix-free sets of binary strings X and Y that encode responses and actions respectively. For our example we restrict ourselves to single bits: $X = Y = \mathbb{B}^1 = \mathbb{B} = \{0,1\}$. Strings over X are denoted $s = x_1 x_2 \cdots x_n$ with $x_k \in X$. $l(s) = l(x_1) + l(x_2) + \cdots + l(x_n)$ is the length of s. Analogous definitions hold for $y_k \in Y$. The elements of X and Y are called *words* rather than letters. The string $s = y_1 x_1 \cdots y_n x_n$ represents action/response pairs in chronological order. Due to the prefix property of X and Y, s can be uniquely separated into words. We further use the following abbreviations: ε is the empty string, $x_{n:m} = x_n x_{n+1} \cdots x_{m-1} x_m$ for $n \leq m$ and ε otherwise. $x_{<n} = x_{1:n-1}$. Analogous abbreviations are used for y. Furthermore we use $yx_n = y_n x_n$, $yx_{n:m} = yx_n \cdots yx_m$ and so on. We will disregard the encoding process and refer to yx_k as the k-th event. The string $yx_{<k}$ represents the entire *history* of events up to, but not including, the k-th iteration.

The example environment that is used in this paper acts like a Markov Decision Process (MDP). The behavior of the process is informally presented in the following diagram.



Each circle in the diagram represents a possible state of the environment. The word that the environment produces when it enters a state is printed inside the circle. A wide arrow is originating from each state for every action that the subject can perform. The word corresponding to the action is inscribed in the arrow. The thin arrows originating from each action arrow represent possible state transitions of the environment. Each state transition is labelled with a probability. The sum of the probabilities of the state transitions that originate from an action arrow is unity.

The essence of this example is that it has two modes: stationary and alternating, and that the subject can switch between these modes at will. The subject will see only half of the environment if it chooses the same action for every cycle. If it acts randomly it will be hard to spot the regularities. Therefore we expect that a learning subject will exhibit some interesting behavior in this environment.

For a more formal treatment of this example we introduce a notation for probabilities of responses that is similar to Hutter's. Let $t \in (Y \times X)^*$, $y \in Y$, $x \in X$, and $p \in \mathbb{R}$, then $q(t\, y\, \underline{x}) = p$ means that for all sequences of events s the probability that the next response is x, given that the history is $s\, t\, y$, equals p. An underlined variable $\underline{x}$ represents a variable and other non-underlined variables represent conditions. The value is only defined when the conditions are sufficient to determine the probability. The response has to be independent of events that are more than n cycles in the past, where n is the number of cycles in t. As the state of the environment depends only on the last few words of the history, it can be formally defined as follows:

$$\begin{aligned} q(0\,\underline{1}) &= 0 \\ q(001\,\underline{1}) &= 0.5 \\ q(101\,\underline{1}) &= 1 \\ q(111\,\underline{1}) &= 0 \end{aligned}$$

The fact that the environment happens to act like a Markov Decision Process with a finite number of states is an artefact of the example. It does not imply any general constraints on the environment or the models that the subject constructs of the environment. For example, the example could be refined with a rule that depends on more data that gives more information about $q(001\,\underline{1})$. The new rule could depend on the last response on a 1-action and the number of preceding 00 events, *e.g.*,

$$\begin{aligned} q(10\,0^{2n}\,001\,\underline{1}) &= 1/2 + 2^{-n-2} \\ q(11\,0^{2n}\,001\,\underline{1}) &= 1/2 - 2^{-n-2} \end{aligned}$$

for $n \in \mathbb{N}$, where 0^{2n} represents a string of n times the 00 event. That would rule out a finite number of states for the process that generates the environment. The refinement could be done in such a way that, for simple strategies to generate the actions, the interaction with the original MDP would have the same stationary distribution as the interaction with the more complex model.

Analogous to probabilities of responses we introduce probabilities of actions: $p(yx_{<k}\underline{y}_k)$ is the probability that the k-th action is x_k given that the history ends with $yx_{<k}$. If we look at the unmodified example we can analyse its behavior for three simple ways to choose the actions: $p(\varepsilon\underline{1}) = 0$ (always zero), $p(\varepsilon\underline{1}) = 1$ (always one), and $p(\varepsilon\underline{1}) = 1/2$. In the first case the interaction will consist of a constant sequence of pairs of zeros. In the second case the interaction will look like:

$$\cdots 10\ 11\ 10\ 11\ 10\ 11 \cdots$$

In the third case the interaction can be described as a Markov Chain over events with the following stationary distribution:

event	00	01	10	11
probability	1/2	0	1/4	1/4

3 The algorithmic approach

Ray Solomonoff proposes to use the probability that a universal Turing machine reproduces the history as basis for establishing the likelihood of new data given the history (Solomonoff, 1964). This corresponds to Bayesian prediction using all effectively enumerable probability distributions over infinite strings as a hypotheses class using a universal prior. The prior is universal in the sense that it attributes a positive probability to all hypotheses in the class and that it dominates all possible effectively enumerable priors save for a multiplicative constant that depends on the pair of priors, but not on the hypotheses.

Marcus Hutter adds actions to this framework. In his AIξ model a learning system, a subject in our terminology, has to act upon its model of the environment. The model of the environment and the strategy for choosing actions are almost symmetrically described by conditional probability distributions over histories. The AIξ model is defined in terms of chronological Turing machines instead of the monotonous Turing machines used by Solomonoff. Chronological Turing machines alternately read one word from a one-way input-tape and write one word to a one-way output-tape. The words are chosen from prefix-free sets as described in Section 2. The symmetry between the model of the environment and the strategy for generating actions is broken in the AIξ model because only the output of the environment gets special treatment. It is split in an credit part and a residue. The credit defines the utility function that the subject has to optimise. Like Solomonoff the AIξ model fixes one universal chronological Turing machine U as a reference[2]. Every chronological Turing machine M that we would like to consider as a candidate for a model of the environment can be simulated on U. In other words: there exists a program q_M such that $U(q_M y_{<k}) = M(y_{<k})$. The function $\xi(q_M) = 2^{-l(q_M)}$ can be interpreted as the probability that U will use M to simulate the environment. Using ξ as a universal prior distribution over possible models of the environment and a utility function $C_{km_k}(p, q)$ the AIξ model defines the optimal action $\dot{y}_k$ as

$$\dot{y}_k = \operatorname*{maxarg}_{y_k} \max_{p \in P_k(y_k)} \sum_{q \in Q_k} \xi(q) \cdot C_{km_k}(p, q)$$

where $P_k(y_k) = \{p \in \mathbb{B}^* \mid U(p\ \dot{x}_{<k}) = \dot{y}_{<k} y_k\}$ and $Q_k = \{q \in \mathbb{B}^* \mid U(q\ \dot{y}_{<k}) = \dot{x}_{<k}\}$. The dots above $x_{<y}$ and $y_{<k}$ indicate that these are actual values that have been exchanged between the subject and the environment. The utility function $C_{km_k}(p, q)$ computes the credits that the subject would receive on cycles k to m_k when it uses strategy p and when q is a perfectly accurate model of the environment. The AIξ model converges to models where $\xi(q)$ is replaced by another recursively enumerable prior on (programs for) chronological Turing machines. We need knowledge about the environment for a better prior as well as for an optimal universal Turing machine. Given that we are studying learning, *i.e.*, processes that increase knowledge, all prior knowledge is dangerous. We could forget to measure it beforehand and count it as obtained by the subject afterwards. Prior

[2]The prefix-free set that defines the words on the input tape has to be extended to allow for a program to be read before or together with the first word of the simulated input

knowledge can also reduce a potential learning situation to a mere search for some global maximum. Therefore we cannot expect subjects that have to *learn* about the environment while interacting with it to do much better than the AIξ model

Application to the example

To apply Hutter's approach to the example we need to specify a credit function for the output words that the environment can produce. As $X = \mathbb{B}$ every possible credit function is equivalent with either $C(x) = x$ or $\bar{C}(x) = 1 - x$.

If the credit function is $C(x) = x$ a subject that uses the AIξ model will find out quickly that the best strategy is to always choose the 1-action. Likewise, if the credit function is $\bar{C}(x) = 1 - x$ a subject that uses the AIξ model will find out quickly that the best strategy is to always choose the 0-action. In either case no further information is gathered about the transitions between the left and the right half of the state diagram. The difference between the simple example and the extended example will never become apparent in the history. If we want to 'teach' the complete environment to the subject, we need to extend it in such a way that the subject can 'tell' the environment what its model is, *i.e.*, include q_M in y_k. Then we would have to let the environment evaluate the accuracy of M and return a credit that is based on both the accuracy of M and the length of q_M. That is quite a burden to place on the environment.

4 The Minimum Description Length approach

Rissanen's Minimum Description Length principle (MDL) is explicitly formulated in terms of hypotheses (Rissanen, 1989). The process of revising evaluations of hypotheses is clearly a form of inference. MDL is based on reproducing a sequence of symbols exactly, just like Solomonoff's inductive inference procedure. MDL assumes that we are given a hypotheses class $\mathcal{C}$. Every hypothesis in the given class is a distribution over all infinite sequences of symbols from our alphabet. MDL also requires that every hypothesis H in the hypotheses class can be finitely described by a code c_H.

For learning subjects the hypotheses are functions that compute the conditional probability of responses of the environment given actions by the subject. Using the Kraft inequality (Li and Vitányi, 1993) we can construct an encoding $e_{H,y_{<k}}$ for finite sequences of symbols such that for each finite history $yx_{<k}$ we have

$$l(e_{H,y_{<k}}(x_{<k})) \approx -\log_2 H(y_1\underline{x}_1 \cdots y_{k-1}\underline{x}_{k-1})$$

Now for each hypothesis H a transcription of the symbols that were exchanged on the interface $yx_{<k}$ can be described by concatenating c_H, $y_{<k}$ and $e_H(yx_{<k})$. According to MDL the best hypothesis is the hypothesis that minimizes the length of this description. As y_k is given and thus fixed we can therefore minimize

$$l(c_H) + l(e_{H,y_{<k}}(x_{<k}))$$

Application to the example

We could take all Markov Decision Processes with finitely many states and probabilities that are rational numbers as our model class $\mathcal{C}$. Each model can be described as follows:

$$\begin{array}{c} n \\ x_{s_1} p_{0,s_1,s_2} \cdots p_{0,s_1,s_n}\ p_{1,s_1,s_2} \cdots p_{1,s_1,s_n} \\ \vdots \\ x_{s_n} p_{0,s_n,s_1} \cdots p_{0,s_n,s_{n-1}}\ p_{1,s_n,s_1} \cdots p_{1,s_n,s_{n-1}} \end{array}$$

where n is the number of states, x_{s_i} is the response of the process when entering state s_i, and p_{y_t,s_i,s_j} is the probability of a state change to s_j given state s_i and action y_t. The probability that the state will not change given an action can be found by subtracting the probabilities of the non-trivial state changes from unity.

A simple way to specify a number $n \in \mathbb{N}$ in a binary alphabet in such a way that the set of codes of numbers is prefix-free is to precede the binary notation of $n+1$ with $\lfloor \log_2 n + 1 \rfloor$ times a 0 bit. As the binary representation of a non-zero natural number always starts with a one bit we can recover a number from a string by the following procedure. First we count the leading zeros and read that many digits after the leading one bit. Then we decode the binary sting that consists of all the bits we read, the initial zeros don't harm, and subtract one. Fractions are encoded by specifying numerator and denominator.

If we follow these rules for the generating model of the example we have three states and twelve fractions to encode. The number three is encoded in five bits. Six of the fractions are zero and are encoded in two bits each. The other fractions are encoded in six bits each. The length of the model totals $5 + (1 + 2 \times 2 + 2 \times 6) + 2 \times (1 + 2 \times (2 + 6)) = 56$ bits. The trivial model requires $3 + 2(1 + 2 \times 1 \times 6) = 29$ bits. If the actions are determined by coin tosses, the generation model can be expected to be better than the trivial model after n action/response pairs, where:

$$56 + \frac{3}{4}n < 29 + n$$

So $n > 108$. It should be clear much earlier that choosing 1 actions leads to an earlier expected model growth. This can be seen by evaluating models that compress the data (temporarily disregarding the length of the encoding of the model) and evaluating the probability that such a model will become the MDL model given that it fits and given a potential strategy. If, for example, it is established that x_k is always zero when y_k is zero when $k = 10$, then the generating model becomes a candidate for the MDL hypothesis for $k = 66$ if all actions are chosen to be 1.

After the discovery the subject will continue to search for possible extensions of the model. The subject will alternate between sequences of zeros and ones of varying length to search to explore $q(001\underline{x})$. If the example is extended as described in Section 2 it will find longer and longer finite truncations of the infinite Markov Decision Process and the model that the subject has of the environment will converge to the true model.

The most important aspect of the algorithmic approach that is lacking in the MDL approach is universality. So far we have discussed a hypotheses class that contains models that are arbitrarily close to the 'true' behavior of the environment. That means that the hypotheses class contains hidden information about the environment. We would rather have the subject learn the fact that the environment acts like a Markov Decision Process by itself. Analogous to the algorithmic approach the hypotheses class should consist of all recursively enumerable semi-measures on the set of infinite binary strings. With a hypotheses class that contains all regularities that can be effectively described we could plug MDL into Hutter's framework and define a universal autonomous learning system.

5 Algorithmic statistics

In the algorithmic approach the subject evaluates descriptions of the total known history of the interaction with its environment. The lengths of these descriptions are used to determine the relative predictive value of the algorithmic processes behind those descriptions. In the MDL approach descriptions of the history consist of two parts: a description of a hypothesis and a description of the history given this hypothesis. This separation of the description in two parts is relevant, because we want to use the length of the first part to guide the actions of the subject. Murray Gell-Mann and Seth LLoyd discuss a similar separation of a description of a finite binary string into two parts (Gell-Mann and LLoyd, 1996). The first part of their description defines a set of strings that describes the regularities in the given string. This set is chosen in such a way that the given string is a 'typical' member of the set. The second part of the description pinpoints given the string in this set. They call the length of a smallest description of a string its total information and the length of a smallest description of a set, such that the string is a typical member of that set, its effective complexity. In their conclusion Gell-Mann and LLoyd even make the connection to learning systems. Given our observation about the MDL approach we come to a different analysis of the importance of these information measures to learning. A small total information just means that there are regularities to be found in the history, but if the effective complexity rises as words are added to the history the subject learns in the sense that new regularities are added to its description of the environment. Therefore the growth rate of the effective complexity of the history would be a good measure for the learning rate of the subject.

In their article about algorithmic statistics Péter Gács, John Tromp, and Paul Vitányi analyse the relation between data and models in depth (Gács et al., 2001). Although their

conclusions about the practical applicability of the universal information measures they found are sobering, it would still be worthwhile to investigate whether it is possible to avoid uncomputability limits. This could be done, for example, by introducing a cut-off on the computation time necessary to produce the history from its description. The universal information measures can also be used to prove bounds on the learning speed of subjects that use randomness to sample the hypotheses space that might be hard to prove otherwise.

6 Conclusion

To remove the external utility function from Hutter's AIξ model while preserving its universality we can take the following steps.

1. Use algorithmic statistics to define a universal form of MDL.
2. Use universal MDL models instead of the raw descriptions of the history.
3. Evaluate actions based on expected model growth instead of the externally supplied utility function.

Further research

The steps above should be formalized. The resulting formal model should be studied to find bounds on learning speed and subjectiveness. To determine the subjectiveness of the model we must ask the question: how far can universal subjects based on different universal machines be apart when interacting with the same environment? This question is especially interesting because we need to define what it means for two subjects to interact with the same environment. As the subjects act differently they might learn different things about the environment and face entirely different questions rather quickly. Another important direction would be to construct practicable algorithms that approach the universal model as closely as possible.

References

Péter Gács, John T. Tromp, and Paul M.B. Vitányi. 2001. Algorithmic statistics. *IEEETIT: IEEE Transactions on Information Theory*, 47. `http://citeseer.nj.nec.com/gacs01algorithmic.html`.

Murray Gell-Mann and Seth LLoyd. 1996. Information measures, effective complexity and total information. *Complexity*, 2:44–52.

Marcus Hutter. 2000. A theory of universal artificial intelligence based on algorithmic complexity. Technical report. 62 pages, `http://xxx.lanl.gov/abs/cs.AI/0004001`.

Ming Li and Paul M.B. Vitányi. 1993. *An Introduction to Kolmogorov Complexity and its Applications*. Springer Verlag.

Jorma J. Rissanen. 1989. *Stochastic Complexity in Statistical Enquiry*. World Scientific, Singapore.

Ray J. Solomonoff. 1964. A formal theory of inductive inference, part I. *Information and Control*, 7:1–22.

Jeroen van Maanen. 2002. Towards a formal theory of learning systems. Unpublished. `http://www.sollunae.net/zope/wiki/EnglishSummary`.