Adversarial Sequence Prediction

Bill HIBBARD
University of Wisconsin - Madison

Abstract. Sequence prediction is a key component of intelligence. This can be extended to define a game between intelligent agents. An analog of a result of Legg shows that this game is a computational resources arms race for agents with enormous resources. Software experiments provide evidence that this is also true for agents with more modest resources. This arms race is a relevant issue for AI ethics. This paper also discusses physical limits on AGI theory.

Keywords. Sequence prediction, AI ethics, physical limits on computing.

Introduction

Schmidhuber, Hutter and Legg have created a novel theoretical approach to artificial general intelligence (AGI). They have defined idealized intelligent agents [1, 2] and used reinforcement learning as a framework for defining and measuring intelligence [3]. In their framework an agent interacts with its environment at a sequence of discrete times and its intelligence is measured by the sum of rewards it receives over the sequence of times, averaged over all environments. In order to maximize this sum, an intelligent agent must learn to predict future rewards based on past observations and rewards. Hence, learning to predict sequences is an important part of intelligence.

A realistic environment for an agent includes competition, in the form of other agents whose rewards depend on reducing the rewards of the first agent. To model this situation, this paper extends the formalism of sequence prediction to a competition between two agents. Intuitively, the point of this paper is that agents with greater computational resources will win the competition. This point is established by a proof for agents with large resources and suggested by software experiments for agents with modest resources. This point has implications for AI ethics.

1. Sequence Prediction

In a recent paper Legg investigates algorithms for predicting infinite computable binary sequences [4], which are a key component of his definition of intelligence. He proves that there can be no elegant prediction algorithm that learns to predict all computable binary sequences. However, as his Lemma 6.2 makes clear, the difficulty lies entirely with sequences that are very expensive to compute. In order to discuss this further, we need a few brief definitions. N is the set of positive integers, $B = \{0, 1\}$ is a binary alphabet, B^* is the set of finite binary sequences (including the empty sequence), and B^∞ is the set of infinite binary sequences. A *generator* g is a program for a universal Turing machine that writes a sequence $w \in B^\infty$ to its output tape, and we write $w =$

$U(g)$. A *predictor p* is a program for a universal Turing machine that implements a total function $B^* \rightarrow B$. We say that a predictor *p learns to predict* a sequence $x_1\ x_2\ x_3\ \ldots \in B^\infty$ if there exists $r \in N$ such that $\forall n > r$, $p(x_1\ x_2\ x_3\ \ldots\ x_n) = x_{n+1}$. Let $C \subset B^\infty$ denote the set of computable binary sequences computed by generators. Given a generator g such that $w = U(g)$, let $t_g(n)$ denote the number of computation steps performed by g before the n^{th} symbol of w is written.

Now, given any computable monotonically increasing function f: $N \rightarrow N$, define $C_f = \{w \in C \mid \exists g.\ U(g) = w$ and $\exists r \in N,\ \forall n > r.\ t_g(n) < f(n)\}$. Then Lemma 6.2 can be stated as follows:

Paraphrase of Legg's Lemma 6.2. Given any computable monotonically increasing function f: $N \rightarrow N$, there exists a predictor p_f that learns to predict all sequences in C_f. This is a bit different than Legg's statement of Lemma 6.2, but he does prove this statement.

Lloyd estimates that the universe contains no more than 10^{90} bits of information and can have performed no more than 10^{120} elementary operations during its history [5]. If we take the example of $f(n) = 2^n$ as Legg does, then for $n > 400$, $f(n)$ is greater than Lloyd's estimate for the number of computations performed in the history of the universe. The laws of physics are not settled so Lloyd may be wrong, but there is no evidence of infinite information processes in the universe. So in the physical world it is reasonable to accept Lemma 6.2 as defining an elegant universal sequence predictor. This predictor can learn to predict any sequence that can be generated in our universe. But, as defined in the proof of Lemma 6.2, this elegant predictor requires too much computing time to be implemented in our universe. So this still leaves open the question of whether there exist sequence predictors efficient enough to be implemented in this universe and that can learn to predict any sequence that can be generated in this universe. It would be useful to have a mathematical definition of intelligence that includes a physically realistic limit on computational resources, as advocated by Wang [6].

2. Adversarial Sequence Prediction

One of the challenges for an intelligent mind in our world is competition from other intelligent minds. The sequences that we must learn to predict are often generated by minds that can observe our predictions and have an interest in preventing our accurate prediction. In order to investigate this situation define an *evader e* and a *predictor p* as programs for a universal Turing machine that implement total functions $B^* \rightarrow B$. A pair e and p play a game [7], where e produces a sequence $x_1\ x_2\ x_3\ \ldots \in B^\infty$ according to $x_{n+1} = e(y_1\ y_2\ y_3\ \ldots\ y_n)$ and p produces a sequence $y_1\ y_2\ y_3\ \ldots \in B^\infty$ according to $y_{n+1} = p(x_1\ x_2\ x_3\ \ldots\ x_n)$. The predictor p wins round n+1 if $y_{n+1} = x_{n+1}$ and the evader e wins if $y_{n+1} \neq x_{n+1}$. We say that the predictor *p learns to predict* the evader e if there exists $r \in N$ such that $\forall n > r$, $y_n = x_n$ and we say the evader *e learns to evade* the predictor p if there exists $r \in N$ such that $\forall n > r$, $y_n \neq x_n$.

Note that an evader whose sequence of output symbols is independent of the prediction sequence is just a generator (the evader implements a function $B^* \rightarrow B$ but is actually a program for a universal Turing machine that can write to its output tape while ignoring symbols from its input tape). Hence any universal predictor for evaders will also serve as a universal predictor for generators.

Also note the symmetry between evaders and predictors. Given a predictor p and an evader e, define an evader e' by the program that implements p modified to complement the binary symbols it writes to its output tape and define a predictor p' by the program that implements e modified to complement the binary symbols it reads from its input tape. Then p learns to predict e if and only if e' learns to evade p'.

Given any computable monotonically increasing function f: $N \to N$, define E_f = the set of evaders e such that $\exists r \in N$, $\forall n > r$. $t_e(n) < f(n)$ and define P_f = the set of predictors p such that $\exists r \in N$, $\forall n > r$. $t_p(n) < f(n)$. We can prove the following analogy to Legg's Lemma 6.2, for predictors and evaders.

Propostition 1. Given any computable monotonically increasing function f: $N \to N$, there exists a predictor p_f that learns to predict all evaders in E_f and there exists an evader e_f that learns to evade all predictors in P_f.

Proof. Construct a predictor p_f as follows: Given an input sequence $x_1\ x_2\ x_3\ \ldots\ x_n$ and prediction history $y_1\ y_2\ y_3\ \ldots\ y_n$ (this can either be remembered on a work tape by the program implementing p_f, or reconstructed by recursive invocations of p_f on initial subsequences of the input), run all evader programs of length n or less, using the prediction history $y_1\ y_2\ y_3\ \ldots\ y_n$ as input to those programs, each for $f(n+1)$ steps or until they've generated $n+1$ symbols. In a set W_n collect all generated sequences which contain $n+1$ symbols and whose first n symbols match the input sequence $x_1\ x_2\ x_3\ \ldots\ x_n$. Order the sequences in W_n according to a lexicographical ordering of the evader programs that generated them. If W_n is empty, then return a prediction of 1. If W_n is not empty, then return the $n+1^{\text{th}}$ symbol from the first sequence in the lexicographical ordering.

Assume that p_f plays the game with an evader $e \in E_f$ whose program has length l, and let $r \in N$ be the value such that $\forall n > r$. $t_e(n) < f(n)$. Define $m = \max(l, r)$. Then for all $n > m$ the sequence generated by e will be in W_n. For each evader e' previous to e in the lexicographical order ask if there exists $r' \geq \max(m,$ length of program implementing $e')$ such that $t_{e'}(r'+1) < f(r'+1)$, the output of e' matches the output of e for the first r' symbols, and the output of e' does not match the output of e at the $r'+1^{\text{th}}$ symbol. If this is the case then this e' may cause an error in the prediction of p_f at the $r'+1^{\text{th}}$ symbol, but e' cannot cause any errors for later symbols. If this is not the case for e', then e' cannot cause any errors past the m^{th} symbol. Define r'' to be the maximum of the r' values for all evaders e' previous to e in the lexicographical order for which such r' exist (define $r'' = 1$ if no such r' values exists). Define $m' = \max(m, r''+2)$. Then no e' previous to e in the lexicographical order can cause any errors past m', so the presence of e in W_n for $n > m'$ means that p_f will correctly predict the n^{th} symbol for all $n > m'$. That is, p_f learns to predict e.

Now we can construct an evader e_f using the program that implements p_f modified to complement the binary symbols it writes to its output tape. The proof that e_f learns to evade all predictors in P_f is the same as the proof that p_f that learns to predict all evaders in E_f, with the obvious interchange of roles for predictors and evaders. □

This tells us that in the adversarial sequence prediction game, if either side has a sufficient advantage in computational resources to simulate all possible opponents then it can always win. So the game can be interpreted as a computational resources arms race.

Note that a predictor or evader making truly random choices of its output symbols, with 0 and 1 equally likely, will win half the rounds no matter what its opponent does.

But Proposition 1 tells us that an algorithm making pseudo-random choices will be defeated by an opponent with a sufficient advantage in computing resources.

3. Software Experiments

Adversarial sequence prediction is a computational resources arms race for algorithms using unrealistically large computational resources. Whether this is also true for algorithms using more modest computational resources can best be determined by software experiments. I have done this for a couple algorithms that use lookup tables to learn their opponent's behavior. The size of the lookup tables is the measure of computational resources. The predictor and evader start out with the same size lookup tables (a parameter can override this) but as they win or lose at each round the sizes of their lookup tables are increased or decreased. The software includes a parameter for growth of total computing resources, to simulate non-zero-sum games. Occasional random choices are inserted into the game, at a frequency controlled by a parameter, to avoid repeating the same outcome in the experiments. The software for running these experiments is available on-line [8].

Over a broad range of parameter values that define the specifics of these experiments, one opponent eventually gets and keeps all the computing resources. Thus these experiments provide evidence that adversarial sequence prediction is an unstable computational resources arms race for reasonable levels of computational resources.

Interestingly, the game can be made stable, with neither opponent able to keep all the resources, by increasing the frequency of random choices. It is natural and desirable that simple table-lookup algorithms should be unable to predict the behavior of the system's pseudo-random number algorithm. But more sophisticated algorithms could learn to predict pseudo-random sequences.

The adversarial sequence prediction game would make an interesting way to compare AGI implementations. Perhaps future AGI conferences could sponsor competitions between the AGI systems of different researchers.

4. AI Ethics

Artificial intelligence (AI) is often depicted in science fiction stories and movies as a threat to humans, and the issue of AI ethics has emerged as a serious subject [9, 10, 11]. Yudkowsky has proposed an effort to produce a design for AGI whose friendliness toward humans can be proved as it evolves indefinitely into the future [12]. Legg's blog includes a debate with Yudkowsky over whether such a proof is possible [13]. Legg produced a proof that it is not possible to prove what an AI will be able to achieve in the physical world, and Yudkowsky replied that he is not trying to prove what an AI can achieve in the physical world but merely trying to prove that the AI maintains friendly intentions as it evolves into the indefinite future. But intentions must be implemented in the physical world, so proving any constraint on intentions requires proving that the AI is able to achieve a constraint on the implementation of those intentions in the physical world. That is, if you cannot prove that the AI will be able to achieve a constraint on the physical world then you cannot prove that it will maintain a constraint on its intentions.

Adversarial sequence prediction highlights a different sort of issue for AI ethics. Rather than taking control from humans, AI threatens to give control to a small group of humans. Financial markets, economic competition in general, warfare and politics include variants of the adversarial sequence prediction game. One reasonable explanation for the growing income inequality since the start of the information economy is the unstable computational resources arms race associated with this game. Particularly given that in the real world algorithm quality is often an important computational resource. As the general intelligence of information systems increases, we should expect increasing instability in the various adversarial sequence prediction games in human society and consequent increases in economic and political inequality. This will of course be a social problem, but will also provide an opportunity to generate serious public interest in the issues of AI ethics.

References

[1] J. Schmidhuber. The Speed Prior: A New Simplicity Measure Yielding Near-Optimal Computable Predictions. In J. Kivinen and R. H. Sloan, editors, Proceedings of the 15th Annual Conference on Computational Learning Theory (COLT 2002), Sydney, Australia, Lecture Notes in Artificial Intelligence, pages 216--228. Springer, 2002. http://www.idsia.ch/~juergen/coltspeed/coltspeed.html

[2] Hutter, M. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Springer, Berlin, 2004. 300 pages. http://www.idsia.ch/~marcus/ai/uaibook.htm

[3] Hutter, M. and S. Legg. Proc. A Formal Measure of Machine Intelligence. *15th Annual Machine Learning Conference of Belgium and The Netherlands (Benelearn 2006)*, pages 73-80. http://www.idsia.ch/idsiareport/IDSIA-10-06.pdf

[4] Legg, S. Is there an Elegant Universal Theory of Prediction? Technical Report No. IDSIA-12-06. October 19, 2006. IDSIA / USI-SUPSI Dalle Molle Institute for Artificial Intelligence. Galleria 2, 6928 Manno, Switzerland. http://www.idsia.ch/idsiareport/IDSIA-12-06.pdf

[5] Lloyd, S. Computational Capacity of the Universe. Phys.Rev.Lett. 88 (2002) 237901. http://arxiv.org/abs/quant-ph/0110141

[6] Wang, P. Non-Axiomatic Reasoning System --- Exploring the essence of intelligence. PhD Dissertation, Indiana University Comp. Sci. Dept. and the Cog. Sci. Program, 1995. http://www.cogsci.indiana.edu/farg/peiwang/PUBLICATION/wang.thesis.ps

[7] http://en.wikipedia.org/wiki/Game_theory

[8] http://www.ssec.wisc.edu/~billh/g/asp.html

[9] Hibbard, W. Super-Intelligent Machines. Computer Graphics 35(1), 11-13. 2001. http://www.ssec.wisc.edu/~billh/visfiles.html

[10] Bostrom, N. Ethical Issues in Advanced Artificial Intelligence. *Cognitive, Emotive and Ethical Aspects of Decision Making in Humans and in Artificial Intelligence, Vol. 2, ed. I. Smit et al.*, Int. Institute of Advanced Studies in Systems Research and Cybernetics, 2003, pp. 12-17. http://www.nickbostrom.com/ethics/ai.html

[11] Goertzel, B. Universal Ethics: The Foundations of Compassion in Pattern Dynamics. October 25, 2004. http://www.goertzel.org/papers/UniversalEthics.htm

[12] Yudkowsky, E. (2006) Knowability of FAI. http://sl4.org/wiki/KnowabilityOfFAI

[13] Legg, S. Unprovability of Friendly AI. September 15, 2006. http://www.vetta.org/?p=6