

# The Accuracy of Universal Sequence Predictors

Nick Hay

3rd February 2006

## Motivation:

- Sequence predictors are interesting.
- Thus, quantifying and analysing their accuracy is an interesting thing to do.
- In particular, we care about how powerful universal predictors are.

- We define the accuracy of a predictor as the log probability it assigns to the actual outcome.
- We examine predictors  $M_T$  that model the environment as a computer  $T$  with an unknown program.

We find:

- If the true output is  $x$ ,  $M_T$  scores at least  $-K_T(x)$  and makes at most that many errors.
- $M_T$  infers the environment's program "faster" than we can tell it.
- $M_T$  scores no more than  $K_T(\nu)$  worse than any other predictor  $\nu$ , irrespective of the true output  $x$ .

This presentation emphasises exact results (no unnamed  $O(1)$  constants) and results that hold for all possible outputs (no expectations).

We first review some computational concepts, then move to the analysis.

## Computational review:

- Enumerable semimeasures
- Computers
- Computers represent enumerable semimeasures
- Universal computers
- Complexity measures

# Enumerable semimeasures

- A **semimeasure**  $\nu: X^* \rightarrow [0, 1]$  satisfies:

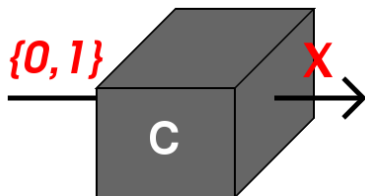
- Normalisation:

$$\nu(\epsilon) = 1$$

- Coherence:

$$\sum_{c \in X} \nu(xc) \leq \nu(x)$$

- For any string  $x$ ,  $\nu(x)$  is the probability the true string begins with  $x$ .
- Semimeasures represent probability distributions over  $X^\#$ , the set of finite and infinite strings over the alphabet  $X$ .
- An **enumerable semimeasure**  $\nu(x)$  is the limit of a computable nondecreasing sequence  $\nu_i(x)$  of semimeasures. Enumerable semimeasures can be computably approximated from below, although they need not be computable themselves.



- A **computer**  $T$  is a computable system that can request program bits and output symbols  $x_i \in X$ .
- Program bits come from an unknown program source: each binary value is equally likely.
- The probability that the computer's output starts with  $x \in X^*$ :

$$M_T(x) = \sum_{p: T(p)=x^*} 2^{-|p|}$$

where  $T(p) = x^*$  means  $p$  is the exact program read when  $x$  is output ( $p$  is a “minimal” program).

- $M_T$  is an enumerable semimeasure, and thus a distribution over  $X^\#$ .

## Theorem

*For any enumerable semimeasure  $\nu(x)$  there exists a computer  $T$  such that*

$$\nu(x) = M_T(x)$$

- **Proof idea:** The computer enumerates  $\nu$ , allocating program outputs so the output distribution is at most  $2^{-i}$  smaller than  $\nu_i$ .
- Prediction using an enumerable semimeasure is equivalent to modelling the environment as a computer.

## Definition

A computer  $U$  is **universal** if for all computers  $T$  there exists  $k_T \in \mathbb{N}$  such that for all  $p$  there exists  $q$  such that:

1.  $U(q) = T(p)$
2.  $|q| \leq |p| + k_T$

We say  $U$  simulates  $T$  with constant  $k_T$ .

## Definition

The complexity  $K_U(T)$  of a computer  $T$  relative to another  $U$  is

$$K_U(T) = \min\{k_T : U \text{ simulates } T \text{ with constant } k_T\}$$

- We introduce new complexity measures to avoid additive constants.

## Definition

The complexity  $K_U(x)$  of a string  $x$  relative to  $U$  is:

$$K_U(x) = \min\{|p| : U(p) = x\}$$

- If we define  $T_x(\epsilon) = x$ , computers which output  $x$  without reading any program bits, we have  $K_U(T_x) = K_U(x)$ .

## Definition

The complexity  $K_U(\nu)$  of a semimeasure  $\nu$  relative to  $U$  is:

$$K_U(\nu) = \min\{K_U(T) : M_T(x) = \nu(x)\}$$

- $K_U(\nu)$  is the complexity of the simplest computer  $T$  which simulates  $\nu$  (i.e. has  $\nu$  as its output distribution).

## The accuracy of universal prediction:

- The accuracy of a predictor: its score
- Example scores
- Predicting a computer's output
  - Total score
  - Partial scores
  - Total score as total error; partial score as magnitude of error
  - Total score bounds number of errors
  - Complexity of output bounds total score
  - "Easy" to predict environments
- Universal prediction

# The accuracy of a predictor: its score

- A **prediction** (the output of a **predictor**) is a probability distribution i.e. a function  $p: O \rightarrow [0, 1]$  where

$$\sum_o p(o) = 1$$

e.g. a semimeasure represents a prediction/predictor.

- The accuracy of a prediction  $p(o)$  increases with the probability it gives the correct answer  $\hat{o} \in O$ . We use the log-probability:

$$\log p(\hat{o}) \in [-\infty, 0]$$

and call it the prediction's **score** ( $\log = \log_2$ ).

- Log-probability is necessary for the partial score decomposition we introduce later.

# Example scores

$p(\hat{o})$	$\log p(\hat{o})$	Description
1	0	Correctly certain; perfect
1/2	-1	Error upper bound
1/ O	$-\log  O $	Ignorance
1/2 <sup>n</sup>	-n	Integer scores
0	$-\infty$	Complete failure

- An **error** cannot have probability greater than 1/2 for it would be more likely than all other possibilities together.
- We can always achieve a score of  $-\log |O|$  with the uniform distribution:

$$U_O(o) = \frac{1}{|O|}$$

This captures ignorance; scoring less than this is performing worse than ignorance (certainly an error).

# Predicting a computer's output

Total score

- Recall  $M_T(x)$  is the probability that computer  $T$ 's output starts with  $x$  given an unknown program. This is a prediction of  $T$ 's output.
- To measure  $M_T$ 's accuracy we need its probability that  $T$ 's output is *exactly*  $x$ , denoted  $M_T(x\downarrow)$ . This can be derived from  $M_T(x)$ . For finite  $x \in X^*$ ,

$$M_T(x\downarrow) = M_T(x) - \sum_{a \in X} M_T(xa)$$

For infinite  $x \in X^\infty$ ,

$$M_T(x\downarrow) = \lim_{n \rightarrow \infty} M_T(x_{1:n})$$

- We call  $\log M_T(x\downarrow)$  the predictor's **total score**.

# Predicting a computer's output

## Partial scores

- The total score  $\log M_T(x \downarrow)$  can be divided into **partial scores** for each sub-prediction:

$$\begin{aligned}\log M_T(x \downarrow) &= \log M_T(x) + \log M_T(\downarrow | x) \\ &= \sum_{i=1}^{|x|} \log M_T(x_i | x_{<i}) + \log M_T(\downarrow | x)\end{aligned}$$

where both steps follow from the chain rule of probability theory, and  $x_{<i} = x_1 \dots x_{i-1}$ .

- The predictor gets a score for its prediction of each element of the sequence and the sequence ending.
- Predictions of an element  $x_i$  are informed by earlier elements in the sequence  $x_{<i}$ .

# Predicting a computer's output

Total score as total error; partial score as magnitude of error

- A sub-prediction's partial score measures how much in error (i.e. how far from correctness) it is.

Score	Description
0	Perfect
-1	Half probability given to correct answer
$-\log  X $	Lower than this is worse than chance
$-\infty$	Completely wrong

- The total score measures the cumulative accuracy of all sub-predictions. It is partitioned to form the partial scores (total score as a substance; error pool).
- It can be shown that scores equal the number of bits learnt from the environment (e.g. number of bits of  $T$ 's program) in discovering what the output actually is.

# Predicting a computer's output

Total score bounds number of errors

## Theorem

*The predictor  $M_T$  makes at most  $\lfloor -\log M_T(x\downarrow) \rfloor$  scattered prediction errors.*

- **Proof:** Recall predictions in error have probability at most  $1/2$  and score at most  $-1$ . Predicting a sequence  $x$  makes  $|x| + 1$  sub-predictions:

$$\log M_T(x\downarrow) = \sum_{i=1}^{|x|} \log M_T(x_i|x_{<i}) + \log M_T(\downarrow|x)$$

At most  $\lfloor -\log M_T(x\downarrow) \rfloor$  can be in error, otherwise the RHS would be smaller than the LHS.

- This is a weak bound! If the errors are large, e.g. a partial score of  $-\log |X|$  or less, we have much fewer errors.

# Predicting a computer's output

Complexity of output bounds total score

## Theorem

*The total score  $\log M_T(x\downarrow)$  is bounded by the complexity of the output  $K_T(x)$ :*

$$-K_T(x) \leq \log M_T(x\downarrow)$$

- **Proof:** The shortest program outputting  $x$  is one of the programs outputting  $x$ :

$$2^{-K_T(x)} \leq \sum_{p:T(p)=x} 2^{-|p|} = M_T(x\downarrow)$$

## Corollary

*$M_T$  makes at most  $K_T(x)$  scattered errors in prediction.*

- Everytime  $M_T$  makes an error it learns at least one bit of  $T$ 's program!

# Predicting a computer's output

"Easy" to predict environments

- We modify  $T$  in an attempt to make it "easy to predict", to see if this increases the score.
- Take a computer  $T$  and modify it so that it tells us its program  $p$  before executing it:

$$T'(p) = f(p)T(p)$$

for some injective encoding  $f: 2^* \rightarrow X^*$ .

- $T'$  tells you exactly what you need to know, its program, to exactly predict its future behaviour.

# Predicting a computer's output

"Easy" to predict environments

## Theorem

For any program  $p$  the environment could be running,  $M_T$  never scores less than  $M_{T'}$ :

$$\log M_{T'}(T'(p)\downarrow) \leq \log M_T(T(p)\downarrow)$$

- The predictor is *more* accurate if you don't tell it the environment's program!
- This is because  $M_T$  doesn't need to infer the exact program  $p$  that  $T$  is running, only the equivalence class  $[p]$  of programs with identical outputs.
- However, we know  $M_{T'}$ 's errors are concentrated in the first  $|f(p)|$  bits, after which prediction is perfect, whereas  $M_T$ 's errors are arbitrarily scattered.

## Theorem

For  $U$  universal, any computer  $T$ , and for all  $x$ :

$$\log M_T(x\downarrow) - K_U(T) \leq \log M_U(x\downarrow)$$

- **Proof idea:** If  $T(p) = x$  then there exists  $p'$  where  $U(p') = x$  and
$$|p'| \leq |p| + K_U(T)$$
- The only predictors which score much more than universal predictor  $M_U$  are complex.
- $M_U$ 's "error pool" (to be distributed over its sub-predictions) is at most  $K_U(T)$  bits larger than  $M_T$ 's.

## Corollary

For  $U$  universal and any enumerable semimeasure  $\nu(x)$ :

$$\log \nu(x\downarrow) - K_U(\nu) \leq \log M_U(x\downarrow)$$

- $M_U$  performs at most  $K_U(\nu)$  bits worse than *any* other predictor  $\nu$ .

- We defined a predictor,  $M_U$ , which models the environment as a universal computer  $U$  with an unknown program.
- We measured the accuracy of  $M_U$ 's prediction of the environment's output  $x$  by its score:

$$\log M_U(x \downarrow)$$

- This score measures the total error made in the prediction. It is spread over sub-predictions of individual elements of  $x$ .
- The score bounds the maximum number of sub-predictions that can be in error, although it doesn't restrict where the errors occur.

Finally, we described three results:

- If the true output is  $x$  the universal predictor  $M_U$  scores at least  $-K_U(x)$

$$-K_U(x) \leq \log M_U(x \downarrow)$$

- $M_U$  has a *lower* score if it is told the exact program  $U$  is running.
- The universal predictor  $M_U$  scores no more than  $K_U(\nu)$  worse than any other predictor  $\nu$ .

$$\log \nu(x \downarrow) - K_U(\nu) \leq \log M_U(x \downarrow)$$

(The first expression is a special case of this one.)