# Predictive processing is Turing complete: A new view of computation in the brain

*Chris Thornton*
Centre for Research in Cognitive Science
University of Sussex
Brighton
BN1 9QJ
UK
c.thornton@sussex.ac.uk

July 5, 2016

**Abstract**

Increasingly, the brain is conceived to be an engine of prediction consolidated in a multilayer model that is probabilistic and generative. According to one recent proposal, the operational basis of all functionality is then 'predictive processing.' The question of the computational power of this regime then becomes of interest. Does predictive processing subserve computation in general? Does it provide the means of implementing any function whose values can be computed by an algorithm? This paper shows the existing specification is imprecise in some respects. But with the ambiguities resolved in a plausible way, the regime can be shown to be Turing complete. Its capacity to implement a universal Turing machine implies an ability to replicate the behavior of any general-purpose computer. The thesis that functionalities of the brain stem from predictive processing need not imply a limited capacity to compute.

Keywords: predictive processing, predictive coding, hierarchical prediction machine, Bayesian brain, information theory, cognitive informatics

## 1 Introduction

There is increasing enthusiasm for what Clark calls 'the emerging unifying vision of the brain as an organ of prediction using a hierarchy of generative models' (Clark, 2013, p. 185). Part of a long tradition emphasizing the role of prediction in perception (von Helmholtz, 1860/1962; James, 1890/1950; Tolman, 1948; Lashley, 1951; Mackay, 1956), this approach is now advancing on a broad range of fronts (Rao and Ballard, 1999; Lee and Mumford, 2003; Rao and Ballard, 2004; Knill and Pouget, 2004; Friston, 2005; Hohwy et al., 2008; Jehee and

Ballard, 2009; Friston, 2010; Huang and Rao, 2011; Brown et al., 2011; Clark, 2016). Given action can be viewed as prediction put into a behavioral form, the proposal can be seen as unifying interpretive and behavioral functionality (Brown et al., 2011; Friston et al., 2009).[1] It is also ideally positioned to use information theory (Shannon, 1948; Shannon and Weaver, 1949) as a way of explaining what is achieved. By improving performance in prediction, the agent renders the world less surprising, effectively gaining information (Cover and Thomas, 2006; Friston et al., 2012).

Clark's proposal (e.g. Clark, 2013, 2016) characterizes function organized in this way as 'predictive processing.' According to Clark's scheme, the brain is 'fundamentally an inner engine of probabilistic prediction' (Clark, 2016, pp. 27-28) that is 'constantly trying to guess at the structure and shape of the incoming sensory array' (Clark, 2016, p. 3). Predictions are seen to derive from a 'multilayer probabilistic generative model' (Clark, 2016, p. 4). Behavior reflects not only the way predictions are made, but the way prediction error is reduced. Predictive processing stems from the way prediction-making and error-reduction interact.

The question of computational power then becomes of interest. The capacity of this form of processing to mediate computation needs clarification. Are there computational tasks that cannot be accomplished in this way? If so, should we assume the brain deals with these in some other way? Or can predictive processing be seen as a fully sufficient medium of calculation? The latter is the more economical option. But to place it on a firm footing, the functional power of predictive processing needs to be pinned down precisely. Are there bounds on what can be achieved in this way? Or does the regime provide the means of implementing any function whose values can be computed by an algorithm? To put the question more formally: Is predictive processing a Turing complete model of computation?

To answer the question, we require an operational specification to work from. We need to know what it would mean to build a predictive processing system, and how the system would work. Unfortunately, a specification at this level of detail is not yet available. Although the description provided by Clark and others is detailed, it falls short of being operationally precise. Clark himself classifies the scheme as a 'relatively abstract theoretic model' (Clark, 2016, p. 298), and a 'mid-level organizational sketch' (Clark, 2016, p. 2). This is a fair assessment. Some aspects of what is envisaged are described with complete precision. Others less so.

The fundamental posits of the approach are unambiguous, however. The proposal incorporates the commitments of the Bayesian brain hypothesis, that 'the brain codes and computes weighted probabilities' (Clark, 2016, p. 41), and that probabilistic inference plays an important role (cf. Pouget et al., 2013). But, importantly, the proposal also claims that processing is mediated by a 'multilayer probabilistic generative model' (Clark, 2016, p. 4). This description

[1] The assumption underlying this is that 'the best ways of interpreting incoming information via perception, are deeply the same as the best ways of controlling outgoing information via motor action' (Eliasmith, 2007, p. 7).

could be satisfied in a range of ways, some more complex that others. For purposes of operationalizing the scheme, it is necessary to specify what would minimally be entailed.

Arguably the simplest example of this kind of model that Clark cites is a hierarchical Bayesian model (Clark, 2016, pp. 172-175). A minimal constitution for the model can be established on this basis. Each layer in a hierarchical Bayesian model comprises a set of probability-bearing states (e.g., variable states). The between-layer structural connections are defined by conditional probabilities, such that the conditionality is downward in all cases. Each conditional probability has its conditional state at one layer, and its conditioned state in the layer below. Allowing that a state in one layer can conditionalize more than one state in the layer below, the mapping from layer to layer is one-to-many. The model is thus hierarchical.

A model in this form is clearly both probabilistic and multilayer. Is it also generative? To satisfy the requirement for generativity, a model must embody predictive functionality of a certain type. Clark describes what is needed as follows:

> An important feature of the internal models that power such [predictive processing] approaches is that they are *generative* in nature. That is to say, the knowledge (model) encoded at an upper layer must be such as to render activity in that layer capable of predicting the response profiles at the layer below. (Clark, 2016, p. 93)

A hierarchical Bayesian model satisfies this by virtue of its ability to mediate top-down inference. In this process, a conditional probability, and the probability of the state on which it depends, are combined to derive a conditioned probability for a state at the level below. In Bayesian terms, this is a simple form of inference, accomplished without use of Bayes' rule. It is sometimes called 'forward inference', a term which is confusing for the present context as the flow of information would be classified neuroscientifically as 'backward'. Also problematic is the fact that this form of inference derives a prior from a prior, a process which seems on first sight to be a contradiction in terms. To get around these problems, some theorists characterize the process as the 'pulling down' of priors (e.g. Hohwy, 2013, p. 33). Clark sticks to the idea that priors are inferred. As he comments, one of the advantages of using a model in this form

> is that it allows the system to infer its own priors (the prior beliefs essential to the guessing routines) as it goes along. It does this by using its best current model—at one level—as the source of the priors for the level below. (Clark, 2013, p. 3)

Since a hierarchical Bayesian model can accomplish prediction by forward inference, it satisfies the stipulated requirements. It is a valid example of a multilayer probabilistic generative model. This does not rule out that a model of this kind might be much more complex. The role played by lateral (within

layer) connectivity is particularly emphasized in (Clark, 2016), for example.[2] For present purposes, what is important is that the model used in predictive processing might be of just this form.

Given this way of specifying the model itself, a particular form of processing is then implied. It is at this point that questions begin to arise. One relates to the upward flow of information (which is the 'forward' flow from the neuroscientific perspective). This is assumed to exploit the data-compression strategy of predictive coding.[3] As Clark says,

> What is most distinctive about the predictive processing proposal ... is that it depicts the forward flow of information as solely conveying error, and the backward flow as solely conveying predictions. (Clark, 2016, p. 38)

What flows down the hierarchy are predictions; what flows up is prediction error. Clark sees this arrangement as reducing the amount of information that needs to be signalled in the upward direction. As he says, the 'information that needs to be communicated "upward" under all these [predictive processing] regimes is just the prediction error: the divergence from the expected signal' (Clark, 2013, p. 183).

For implementation purposes, we need to determine how the error is calculated, and how it is communicated. Clark states that error is calculated in an information-theoretic way. In (Clark, 2016), he writes[4]

> Prediction error here reports the 'surprise' induced by a mismatch between the sensory signals encountered and those predicted. More formally—and to distinguish it from surprise in the normal, experientially loaded sense—this is known as surprisal (Clark, 2016, p. 25).

This pins down the measurement quite precisely. The problem is that it is not clear how an error signal calculated in this way could play the envisaged role. Except in trival scenarios, such as predicting a scalar value, a prediction error in this form does not itself indicate how the error can be reduced. The problem is easy to illustrate. Imagine telling someone that their translation of a sentence is 30% in error. This gives little indication of how the error can be reduced. The error signal in predictive processing, it seems, must give an indication of how the error can be reduced. We have to assume that it encapsulates some identification of 'that which is not predicted.' Clark seems to acknowledge this when he asserts (at a different point) that the 'unpredicted

[2] Clark writes that 'in the standard implementation of PP higher level 'representation units' send predictive signals laterally (within level) and downwards (to the next level down) thus providing priors on activity at the subordinate level' (Clark, 2016, p. 143).

[3] Clark defines predictive processing to be 'the use of [predictive coding] in the very special context of hierarchical (i.e., multilevel) systems deploying probabilistic generative models' (Clark, 2016, pp. 25-26).

[4] Citing Tribus (1961).

parts of the input (errors) travel up the hierarchy, leading to the adjustment of subsequent predictions' (Clark, 2016, p. 30).

A second issue relates to the relationship between error-signalling and Bayesian inference. Given we are assuming the model takes the form of a Bayesian hierarchy, application of Bayes' rule also produces an upward flow of information. Derivation of posteriors is accomplished by inverting conditional probabilities. The probability of a state at one layer, and a prior on a state at the level above, are combined using Bayes' rule to derive a posterior for the state in question. (If downward inference is characterized as the 'pulling down' of priors, this complementary process involves their 'pulling up'.) For operational purposes we then have to decide whether error-signalling in predictive processing entirely *replaces* upward Bayesian inference, or whether it is integrated with it in some way.

The main question regarding the upward flow of information is, however, how it is implemented. Is some additional apparatus required? If so, what form does this take? Clark remains fairly open on this issue; but he argues that there has to be some functional separation between the apparatus that mediates prediction-making, and the apparatus that mediates error-signalling:

> However it may (or may not) be realized ... predictive processing demands *some* form of functional separation between encodings of prediction and of prediction error. (Clark, 2016, p. 39, original emphasis)

A question remains, however. Given inference using Bayes' rule produces an upward flow of information, it is conceivable that this could be the medium in which error is signalled. For present purposes, such an arrangement would be doubly attractive, since it would give Bayesian inference a well-defined role in the scheme while also eliminating the need for a separate signalling apparatus. For purposes of deriving a well-defined implementation, this issue needs to be settled in some way.

Another question relates to the use of precision weighting. It is assumed that the reliability and salience of error signals is assessed, and that these assessments 'determine the *weighting* (precision) given to different aspects of the prediction error signal at different levels of processing' (Clark, 2016, p. 146). The degree to which the system strives to reduce a particular error is then understood to be controlled by the weighting of the error. An attraction of this scheme is its capacity to explain properties and pathologies of the mind.[5] But questions are raised regarding the distinction between precision-weighting and ordinary inference.

One problem is the threat of an infinite regress. Does it make sense to posit weightings on error signals, unless we also posit weightings on weightings, weightings on weightings on weightings, and so on? What justifies cutting off the process at the first iteration? Clark argues that 'Obviously, no system can afford

[5]Precisions appear to be a good way to explain attention, for example. Clark suggests that 'predictive processing depicts attention as increasing the gain on select prediction errors' (Clark, 2016, p. 77).

to engage in endless spirals of 'computational self-doubt' in which it attempts to estimate its confidence in its own assignments of confidence' (Clark, 2016, p. 201). The difficulty is that, from the operational point of view, adopting this particular cutoff seems arbitrary.

The question of whether any additional apparatus is needed also crops up again. Assuming the system strives to reduce relatively greater error to a relatively greater degree, the effect of using precision weighting might be achieved by amplifying errors according to their *imprecision*. Imagine a prediction is made for states in a particular layer, and that this produces an error with a certain precision. Assume that certain states of this layer are used to represent different precisions. To reproduce the effect of precision weighting, it would then suffice to augment the original prediction with a designation of whatever states represents zero precision. The error of the prediction will then scale with the precision, and the degree to which it is reduced will be modulated accordingly. On this argument, precisions might be implemented in terms of the multilayer apparatus already assumed to exist.

Last but not least, there is the issue of how downward and upward flows of information interact. According to Clark, predictive processing is able 'flexibly to combine top-down and bottom-up flows of information within the multilayer cascade' (Clark, 2016, pp. 25-26). What happens in the case of conflicts—where downward and upward flows produce different probabilities for the same state—is not stipulated in any detail. In Clark's view, there are many possibilities. As he says, there are many 'possible ways of combining top-down predictions and bottom-up sensory information' (Clark, 2016, p. 298). Here, some assumption has to be introduced about the way conflicts are resolved.

To derive an operational formulation of predictive processing, all these questions need to be answered in some way. The constitution of the error signal has to be settled in light of the requirement that this must indicate how the error is to be reduced. Whether signalling of error can be mediated by Bayesian inference needs to be determined. A position has to be taken in regard to the implementation of precision weightings. And the way in which downward/upward conflicts are resolved must be clarified. Even with all these issues settled, there remains one fundamental obstacle. Within Bayesian theory, probabilistic inference—even the simpler forward variety—is computationally intractable. Combination of derived probabilities is accomplished by taking their product. This is a process that, past a certain point of complexity, produces values too small to be represented.

The intractability of Bayesian inference is a problem affecting all work in the Bayesian-brain tradition. While 'Most, if not all, of the computations performed by the brain can be formalized as instances of probabilistic inference' (Pouget et al., 2013, p. 1176), the intractability of the process means that 'unconstrained Bayesian inference is not a viable solution for computation in the brain' (Knill and Pouget, 2004, p. 718). Clark fully acknowledges the problem, nothing that 'Complex real-world problems demand the use of approximations to truly optimal forms of probabilistic inference (Clark, 2016, p. 298). Any operational formulation of predictive processing must address this issue in some way. It must

introduce an approximation of optimal inference that satisfies the requirement to be computationally tractable.

The proposal set out below uses an information-theoretic approach to derive a scheme that satisfies all the requirements that arise. Bayesian theory is often seen to be the most natural way of modeling probabilistic prediction. But information theory can also provide an effective formalization. It is possible to define a metric which measures the informational value of a prediction of a known outcome, given the outcome and its informational value are both known (Thornton, Forthcoming). Using this metric, Bayesian inference can be recast as information maximization. This way of implementing probabilistic inference ensures the process 'respects Bayesian principles' (Clark, 2016, p. 39) — the inference that is implemented is approximately optimal. But there is no loss of tractability. The approach also leads to practical answers for the questions raised above. It mandates a particular way of dealing with the operational ambiguities affecting bottom-up inference. It resolves the issue of whether error-signalling can be subsumed within (upward) Bayesian inference. And it leads to a way of implementing precision weighting without introducing a separate apparatus.

With predictive processing operationalized in this way, it becomes possible to examine the formal properties of the regime. Its computational power can be put to the test. What is demonstrated below is that, in this implementation, the regime is capable of implementing a universal Turing machine. This establishes that predictive processing is Turing complete: the regime has the ability to replicate the behavior of a general-purpose computer. Accordingly, the claim that functionalities of the brain stem from predictive processing has no negative implications for the question of how brains compute. Rather, the approach becomes a way of explaining how computation is accomplished.

The remainder of the paper sets out the proposal in detail. Section 2 introduces the metric of predictive payoff, and examines its relationship to other measures from the Shannon framework. Section 3 shows how the metric leads to a tractable way of implementing predictive processing. Some illustrative examples are presented. Section 4 then tackles the task of demonstrating the computational power of the regime. Its capacity to simulate a Turing machine is demonstrated, and an example simulation is presented. Finally, Section 5 discusses the degree to which the predictive processing proposal answers the long-standing question of how brains compute. An appendix is also added which presents a more complex example of Turing-machine computation simulated by predictive processing.

## 2 Informational modeling of prediction

For purposes of modeling probabilistic prediction, it is normally Bayesian theory we turn to. But Shannon information theory (Shannon, 1948; Shannon and Weaver, 1949) can also provide an effective treatment (Thornton, 2014, Forthcoming). The basic form of this is easily illustrated. Imagine someone produces a weather prediction (i.e., a forecast) in the form of a distribution that gives a

20% chance of rain (probability 0.2), and an 80% chance of no rain (probability 0.8). Given knowledge of the outcome and its informational value, what is the informational value of the predictive distribution? How should we calculate the informational value of the forecast once we know the outcome?

The way the value is calculated informally is clear enough, at least where the outcomes occur with equal probability. If the outcome is rain, the prediction is considered fairly bad (i.e., misinformative). Otherwise, it is considered fairly good (i.e., informative). Such assessments are implicitly graded, and there are well-defined extreme cases. If the predictive distribution gives a 50/50 chance of either outcome, it is judged entirely uninformative regardless of the outcome. Otherwise, the perceived informativeness of the prediction is seen to either increase or decrease, depending on the probability given to the correct outcome. If the balance of probability favours the correct outcome, the more it does so, the more informative the prediction is considered to be. Conversely, if the balance of probability favours the wrong outcome, the more it does so, the more *misinformative* the prediction is seen to be. The perceived informativeness of a predictive distribution can be positive, negative or non-existent in this way.

The weather-forecasting scenario can be used as an illustration. Predicting a 20% chance of rain (i.e., awarding a probability of 0.2) is considered rather misinformative if the outcome turns out to be rain, but not as misinformative as predicting a 10% chance. If the forecast gave a 40% chance of rain, it would be considered less misinformative, but more so than one specifying a 45% chance. If, on the other hand, the forecast places the balance of probability on the correct outcome, the judgements then scale in the opposite way. Forecasting a 60% chance of rain is considered informative, but not so much as forecasting a 80% chance. There is also a well-defined extreme at both ends of the scale. Given it rains, forecasting a 100% chance of rain is maximally informative, while forecasting a 0% chance is maximally misinformative.

This informal method of evaluation is clearly rather precise. Can it be placed on a mathematical footing? Is there an information-theoretic version of the calculation? It is tempting to assume the evaluation can be performed simply by deriving the entropy (uncertainty) of the predictive distribution.[6] This conforms to the general principle that entropy is the means of determining informational value. But the approach clearly fails. For one thing, an entropy measurement applied to the distribution takes no account of the informational value of the outcome itself. For another, it fails to distinguish between the best and worst cases. A predictive distribution that places all probability on the correct outcome has exactly the same entropy (namely zero) as one which places all probability on the wrong outcome.

To get closer to what we need, it is necessary to proceed in a deductive way. Let $x$ be an outcome taken from a choice of any size[7], with $x_{\mathrm{T}}$ being the outcome that occurs, and $x_{\mathrm{F}}$ being an outcome that does not occur. (Allowing the choice

[6] The following section deals with utilization of KL divergence, and scoring functions from decision theory such as the Brier score.

[7] For present purposes, a choice is defined to be a set of outcomes from which precisely one occurs.

to be of any size means there may be more than one non-occurring outcome.) Let $Q(x)$ be the predicted probability of outcome $x$, or more precisely the probability of *predicting* outcome $x$. Finally, let $I_p(x)$ denote the informational value of the prediction of outcome $x$.

According to these definitions, the average informational value of $Q$ as a source of prediction is a weighted sum:

$$I(Q) = \sum_x Q(x) I_p(x) \tag{1}$$

Using $Q_u$ to name a distribution which gives all outcomes the same probability, it can be asserted that

$$I(Q_u) = 0 \tag{2}$$

The uniform distribution $Q_u$ cannot have any informational value as a prediction since it simply asserts what is anyway known, that there is a choice of outcomes.

On the grounds that predicting the correct outcome is informationally equivalent to observing it, a constraint can also be added which links the informational value of the correct outcome to the informational value of correctly predicting it:

$$I_p(x_\mathrm{T}) = I(x_\mathrm{T}) = -\log_2 P(x_\mathrm{T}) \tag{3}$$

Here, $P(x_\mathrm{T})$ is the objective probability of $x_\mathrm{T}$ occurring. On this basis, the informational value of the occurring outcome becomes $-\log_2 P(x_\mathrm{T})$ bits, as does the informational value of predicting it correctly. By deduction from (1), (2) and (3), it then follows that the informational value of predicting the occurring outcome must balance the summed informational value of predicting non-occurring outcomes. This is necessary to satisfy the constraint that the informational value of the uniform distribution $Q_u$ is zero. It must be the case that

$$I_p(x_\mathrm{T}) = -\sum_{x_\mathrm{F}} I_p(x_\mathrm{F}). \tag{4}$$

Combining (3) and (4), the informational value of predicting any outcome can then be defined thus:

$$I_p(x) = \begin{cases} -\log_2 P(x) & \text{if } x = x_\mathrm{T} \\ \frac{P(x)}{1-P(x_\mathrm{T})} \log_2 P(x) & \text{otherwise} \end{cases} \tag{5}$$

Given we know the outcome that occurs, this definition can be used to find the informational value of predicting any outcome within the choice. It allows $I_p(x)$ to be calculted for any $x$. The quantity defined by Eq. 1 is then well-defined. It is the average value of distribution $Q$ for predicting the outcome, taking into account the outcome that occurs, and the informational values that arise. This average is termed the *predictive payoff* of the distribution.

It will be seen that predictive payoff behaves exactly as expected on intuitive grounds. Its value is positive, negative or zero in precisely the cases where a prediction is judged to be informative, misinformative, or uninformative. Recall the original example, where $Q$ gives probability 0.8 to no rain, but the outcome turns out to be rain. This predictive distribution is judged rather misinformative. The mathematical evaluation corroborates the judgement. Given the two outcomes have equal objective probability, each has an informational value of $-\log_2 \frac{1}{2} = 1$ bit. The predictive payoff of the distribution is then

$$0.2 \times 1 + 0.8 \times -1 = -0.6 \text{ bits}$$

The average informational value is negative, reflecting the judgement that the distribution is misinformative.

Examining alternative scenarios confirms that predictive payoff always reflects informativeness in the way we would expect. If the forecast gives a 70% chance of rain and there is rain, the judgement is 'fairly informative' and the payoff is a 0.4 bit gain. If there is rain following a forecast giving only a 10% chance, the judgement is 'highly misinformative' and the payoff is a 0.8 bit loss. If the forecast gives a 50% chance of rain, the judgement is 'completely uninformative' and the payoff is neither gain nor loss. Evaluations range from positive $\rightarrow$ zero $\rightarrow$ negative in a perfect mirror-image of the way judgements range from 'informative' $\rightarrow$ 'uninformative' $\rightarrow$ 'misinformative.'

Predictive payoff can be measured in arbitrarily complex situations involving any number of outcomes. Consider, for example, a doctor who gives a 75% chance of a certain test producing a negative result, a 15% chance of it producing a positive result, and a 10% chance of it producing no result. Here, the predictive probabilities are 0.75, 0.15 and 0.1 respectively. Assuming objective probabilities are uniform, the informational value of an outcome is $-\log_2 \frac{1}{3} \approx 1.58$ bits. If the outcome is a positive result, the predictive payoff of the doctor's forecast is then

$$0.15 \times 1.58 - \frac{0.75}{0.85} \times 1.58 - \frac{0.1}{0.85} \times 1.58 \approx -0.43$$

Given the doctor's strong prediction of a negative result, the outcome of a *positive* result would lead us to judge the prediction as fairly misinformative. The evaluation corroborates the judgement. The prediction is found to produce a loss of around 0.43 bits in relation to the outcome.

Evaluation in the face of a four-way choice can also be illustrated. Imagine a housing agent who gives a 60% chance of selling a property for more than the asking price, a 10% chance of selling for the asking price, a 20% chance of selling for less than the asking price, and a 10% chance of not selling at all. The implied probabilities are then 0.6, 0.1, 0.2 and 0.1 respectively. Given the objective probabilities are uniform, the informational value of each outcome is $-\log_2 \frac{1}{4} = 2$ bits. In the case of a sale above the asking price, we would judge the forecast to be fairly informative—this is the outcome most strongly predicted. Again, mathematical evaluation corroborates the judgement. Since

$$0.6 \times 2 - \frac{0.1}{0.4} \times 2 - \frac{0.2}{0.4} \times 2 - \frac{0.1}{0.4} \times 2 \approx 0.93$$

The predictive payoff is found to be a gain of approximately 0.93 bits.

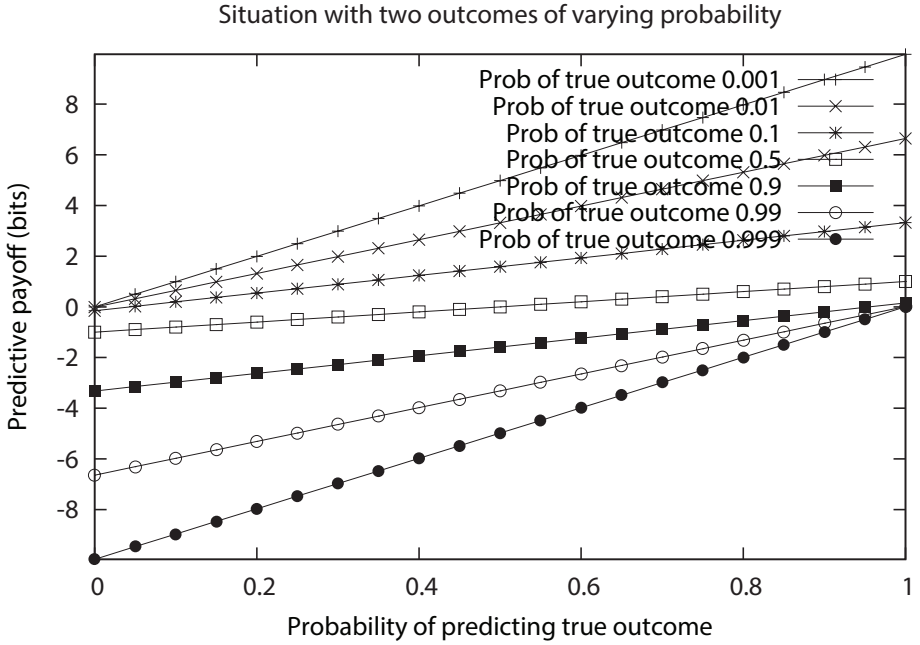



Figure 1: Predictive payoff under varying conditions.

All these examples take outcomes to be objectively equiprobable. This assumption simplifies the calculation. But it is important to remember that predictive payoff is also affected by the informational values of the predicted outcomes. It is a function of the predictive distribution, the outcome, and the probabilities with which outcomes occur. Other things being equal, correctly predicting an outcome of higher informational value produces a greater predictive payoff. The graph of Figure 1 shows how predictive payoff varies across a range of situations. The situation in which the two outcomes are equiprobable is represented by the central line ('Prob of true outcome 0.5').

## 2.1 Relation to KL-divergence and other metrics

How does predictive payoff fit into the Shannon framework more generally? The metric has not been previously defined, but there are several within the framework with which it is potentially compared. Predictive payoff depends largely on the relationship between two distributions: the objective distribution

which defines the probability (and thus informational value) of each outcome, and the subjective distribution which defines the probability of predicting each outcome. Predictive payoff can be compared against other ways of quantifying the informational relationship between two distributions, then.

Mutual information is a metric of this type. This quantifies the informational relationship between two random variables, taking into account their individual distributions (Cover and Thomas, 2006). The measure quantifies how much one distribution tells us about the other. Unfortunately, it also references the joint distribution, which plays no part in the calculation of predictive payoff. In calculating predictive payoff, the joint distribution is assumed not to be known. Mutual information and predictive payoff are incommensurable for this reason. The same applies to conditional entropy and cross-entropy. The former is defined in terms of a conditional distribution, and the latter in terms of a set of observations. Neither figure in the calculation of predictive payoff.

One measure that can be compared is Kullback-Leibler (KL) divergence. This quantifies the relationship between two distributions without referring to any additional data. Given probability distributions $P$ and $Q$, the KL divergence of $P$ from $Q$ is the information lost when $Q$ is used to approximate $P$ (Kullback and Leibler, 1951).[8] The KL divergence of distributions $P$ and $Q$ is

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_i \ln\left(\frac{P(i)}{Q(i)}\right) P(i)$$

Distributions that are identical have a KL divergence of zero, and the value rises to infinity as they diverge. A relationship with predictive payoff can then be discerned. Taking the objective distribution to be uniform, and the best predicting distribution to be one which places all probability on the occurring outcome, it will be seen that predictive payoff always decreases as the predicted and best predicting distributions diverge. KL divergence varies in the opposite direction, decreasing as predictive quality increases.

Predictive payoff has as its maximum the informational value of the occurring event, and as its minimum the corresponding negative. A value of zero also identifies the special case of a neutral (uninformative) prediction. KL divergence maps this range into zero to infinity, and inverts it. For purposes of measuring predictive payoff, it has several drawbacks therefore. It cannot deal with variation in objective probability. The qualitative distinction between good, bad and neutral predictions is not made. Critically, the quantity identified is *not* the informational payoff attained. A relationship exists, but there are significant differences. The graph of Figure 2 shows how values of KL divergence and predictive payoff compare in the two-outcome scenario.

Beyond the Shannon framework, predictive payoff can be related to scoring functions in decision theory. These are also a way of evaluating probabilistic forecasts, and they behave much like KL divergence. Consider the situation

[8]The measure has an intimate relationship with log loss. The log loss sustained by mispredicting a binary outcome is also the KL divergence of the suggested distribution from a distribution which gives all probability to the realized outcome (Mackay, 2003).

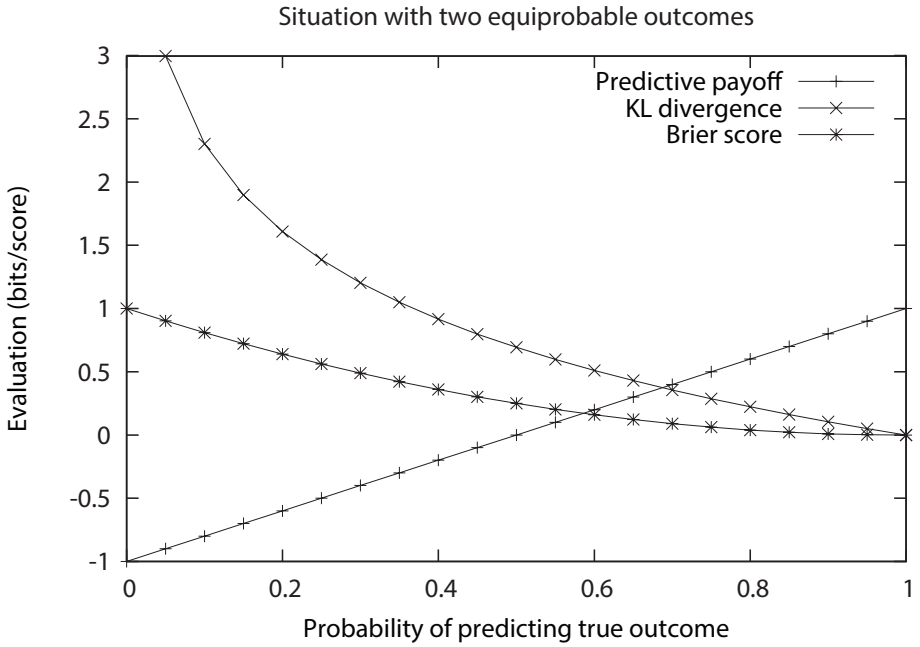



Figure 2: Comparison of predictive payoff, KL divergence and Brier score.

where a weather forecaster gives an 80% chance of rain, but there is no rain. Since predictive payoff reflects the probability given to the true outcome, its value in this case would be negative. We can also apply a scoring rule to evaluate the forecast with respect to this outcome. We might use the Brier rule (Brier, 1950) for example. This is defined by

$$BS = \frac{1}{N}\sum_{i=1}^{N}(p_t - o_t)^2$$

where $N$ is the number of forecasts made, $p_t$ is the probability forecast at time $t$, and $o_t$ is 1 if the forecasted outcome occurs and 0 otherwise. It will be seen that, in the case of a single forecast, the Brier score also varies monotonically with the probability given to the true outcome, with a score of 0 being awarded in the best case (all probability allocated to the outcome that occurs) and a score of 1 in the worst case (all probability given to the outcome that does not occur). Again, the effect is to map predictive payoff onto a non-negative, inversely varying quantity. Figure 2 shows how the this measure compares to predictive payoff and KL divergence in the case of two equiprobable outcomes.[9]

[9]In practice, scoring functions are used to evaluate a series of forecasts with respect to an observed probability distribution over events. A scoring rule is termed 'proper' if it is

# 3 Predictive processing

Bayesian theory is often used to model probabilistic prediction. But, as has been seen, Shannon information theory offers a useful alternative. For present purposes, the Shannon approach has several advantages. Using the metric of predictive payoff, we can calculate the informational value of any probabilistic prediction, subject to knowing the outcome and its informational value. Probabilistic inference (by means of Bayes' rule) can then be recast as maximization of information. To illustrate: imagine we have some hypotheses, each of which awards probability to potential items of data. Using Bayes' rule, we can determine the posterior probability of each hypothesis with respect to some observed data. Generally speaking, the optimal hypothesis is the one that best predicts these data. This is the process of maximum *a posteriori* (MAP) inference. In the information theoretic counterpart, we find the hypothesis whose summed predictive payoff with respect to the observed data is maximized. On the assumption that this is also the hypothesis that best predicts the data, the two methods produce the same result.

There are mathematical differences that need to be recognized, however. Both calculations reference the probability each hypothesis awards to each item of data. But whereas the Bayesian calculation also attends to prior probabilities for both hypotheses and data, the informational calculation references only informational values. If all priors and informational values are set to 1 (i.e., probability 1 in the case of a prior, and 1 bit in the case of an information value) posterior probability and predictive payoff are then numerically identical for each item of data. The payoff is the probability awarded, counted as a quantity of information. The posterior is the same value treated as a probability. The difference comes in the way the evaluations are combined. A hypothesis that awards probability to multiple observed data has multiple posteriors and payoffs. The overall posterior is then calculated as the product of the individual posteriors. The overall payoff, on the other hand, is their sum.

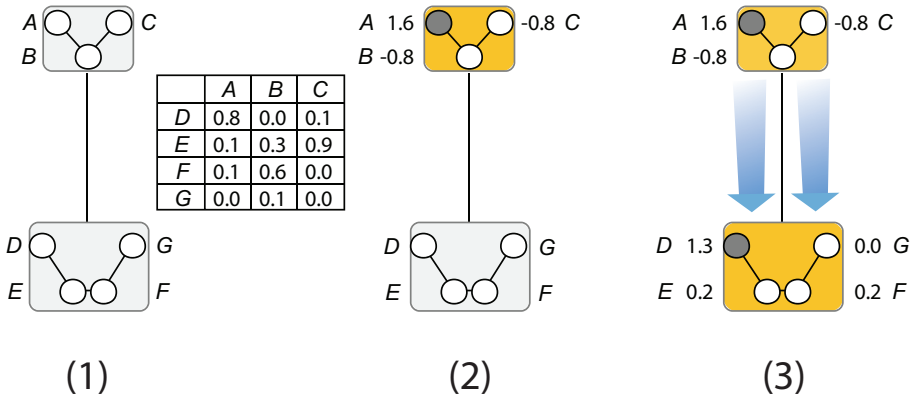
In many situations, the result will be the same. The hypothesis with maximum posterior probability is likely to be the one with maximum payoff. But divergence cannot be ruled out. Say we have 100 observed data, all of which have prior probability 1, and an information value of 1 bit. Imagine a hypothesis that awards maximum probability to 99 of the observed data, but zero probability to the 100th. This hypothesis predicts the observed data extremely well, and the summed payoff will be close to the maximum accordingly. But due to the 'single mistake', the posterior probability falls to zero. Under the Bayesian calculation, this near perfect hypothesis is found to have a value no better than that of the worst hypothesis of all: one that gives all observed data zero probability.

The information-theoretic version of probabilistic inference is an approximation of Bayesian inference, then. But as such it has two attractions. It is mandated by the principles of information theory; and it is computationally

maximized when forecasted probabilities are equal to true probabilities, and 'locally proper' if this maximum is unique.

tractable. Because it combines evaluations by summation rather than multiplication, the problem of disappearingly small evaluations is avoided. Informational modeling of probabilistic prediction solves the main problem presently faced, then. It gives probabilistic inference a computationally tractable form. Operationalization can proceed on this basis.

The informational approach also yields up specific answers to the questions raised in the introduction. The 'multilayer probabilistic generative model' that predictive processing uses can take the minimal form previously envisaged—a hierarchical structure defined by Bayesian (conditional) probabililties. But the probability-bearing states on which the conditionals are defined are now replaced by information-bearing outcomes, and the way these are organized into choices becomes relevant. The generative functionality of the model still derives from application of conditional probabilities. What is generated, however, are distributions of information rather than probability.




| | A | B | C |
|---|---|---|---|
| D | 0.8 | 0.0 | 0.1 |
| E | 0.1 | 0.3 | 0.9 |
| F | 0.1 | 0.6 | 0.0 |
| G | 0.0 | 0.1 | 0.0 |

Figure 3: Top-down information flow.

As an illustration, consider Figure 3. Panel (1) shows a simple, two-layer, hierachical Bayesian model. The rounded rectangles represent two discrete variables, and the enclosed circles represent their possible states. The adjacent conditional probability table (CPT) lists the conditional probabilities that link states of the upper variable ($A$, $B$ and $C$) to states of the lower variable ($D$, $E$, $F$ and $G$). In this table, conditioned states are arranged in rows, and conditioning states in columns. The conditional probability of $D$ given $C$ is 0.1 for example. This is a conventional hierarchical Bayesian model that could be used to derive probabilistic inferences in the usual way.

For purposes of turning the model into an operational predictive processing system, the state of each variable is considered to predict the corresponding outcome of a choice; e.g., state $A$ is considered to predict (with probability 1) outcome $A$ from the choice $A/B/C$. Given occurrence of such an outcome and knowledge of its informational value, the predictive payoff of any predict-

ing states can then be calculated. By the same token, we can represent the occurrence of such an outcome by *setting* informational values to the payoffs in question. The situation of panel (2) shows the configuration representing occurrence of the outcome predicted by $A$. The numbers adjacent to the states $A$, $B$ and $C$ are the informational values acquired in the case of this outcome occurring. Notice the most highly valued state—the one predicting the occurring outcome—is also filled for emphasis. (This graphical convention is used in all cases below.) An outcome that acquires greater (positive) informational value than any of its alternatives is said to be 'cued', and its representing circle is filled. The enclosing rectangle representing the choice is also shaded to a degree that signifies the recency of the cueing.

With the model configured to represent occurrence of $A$, there is the potential to update the informational values of states in the lower layer according to predictions emanating from the upper layer. For each state of the lower variable we can determine the informational value it is predicted to have in light of the informational values of the upper states, and the conditional probabilities they impose. On this basis, each state $x$ in the lower choice acquires a value of $I_d(x)$, where

$$I_d(x) = \frac{1}{|U|} \sum_{y \in U} P(x|y) I_p(y),$$

given $U$ is the set of predicting outcomes. Probabilistic predictions can give rise to downward flows of information in this way. Panel (3) of Figure 3 shows the situation that results. To keep the terminology as simple as possible, it is assumed henceforth that informational values are updated automatically, whenever possible, and the distinction between states and predicted outcomes is dispensed with. From here on, the states of a variable will be referred to directly as the outcomes of a choice.

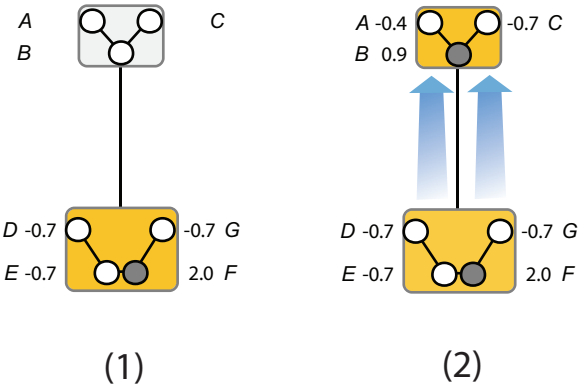



Figure 4: Bottom-up information flow.

If the model is set to represent occurrence of an outcome from the lower choice, a flow of information in the opposite direction arises. The informational values of outcomes in the upper layer are updated according to their predictive payoff for outcomes of the lower layer. Information flows upward from lower to upper outcomes. Figure 4 shows the effect of the occurrence of $F$. The outcome from the upper layer which best predicts $F$ acquires the greatest value (0.9 bits approximately).[10]

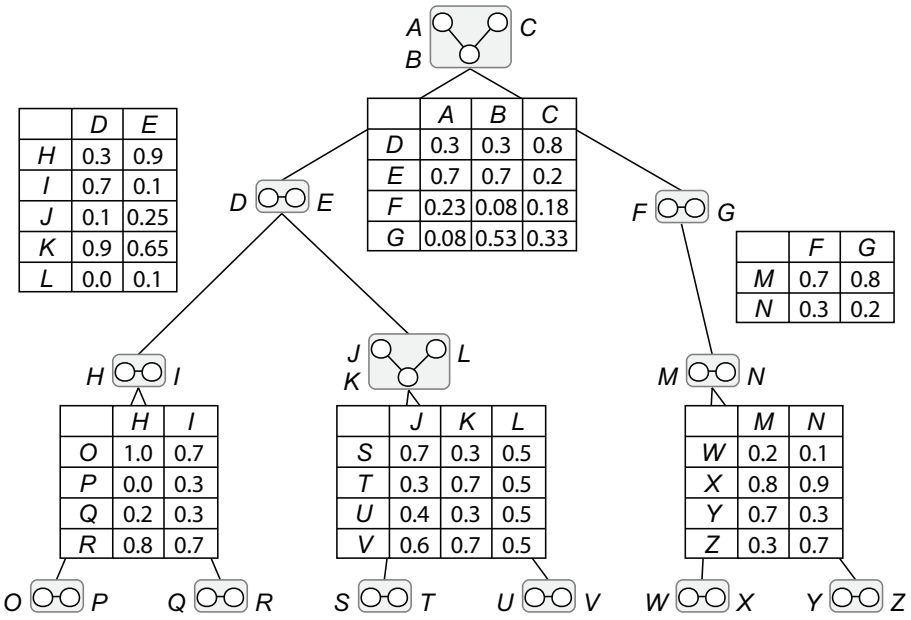



Figure 5: Four-layer probabilistic model.

A more complex setup is depicted in Figure 5. This shows a hierarchical model of four layers, using the same conventions as before. With more structure in the model, more complex flows can arise. Figure 6 illustrates the downward flow that results from cueing $B$. In view of the assigned value of $B$ (2.0 bits) and the probability it awards to $D$ and $E$, the latter two outcomes acquire (by Eq. 6) values of 0.6 and 1.4 bits respectively. With these established, values at the layer below are then updated. Outcomes $H$ and $I$ acquire values of 1.3 and 0.1 bits respectively, while $J$, $K$ and $L$ acquire 0.4, 0.9 and 0.1 bits respectively. The downward flow continues down this and all other branches, eventually awarding values to all outcomes of all choices.

This model can also exhibit a mixture of downward and upward flows. An

[10] As it identifies the best predicting outcome from the upper layer, this behavior approximately (but tractably) replicates identification of the best predicting hypothesis by Bayesian MAP inference.

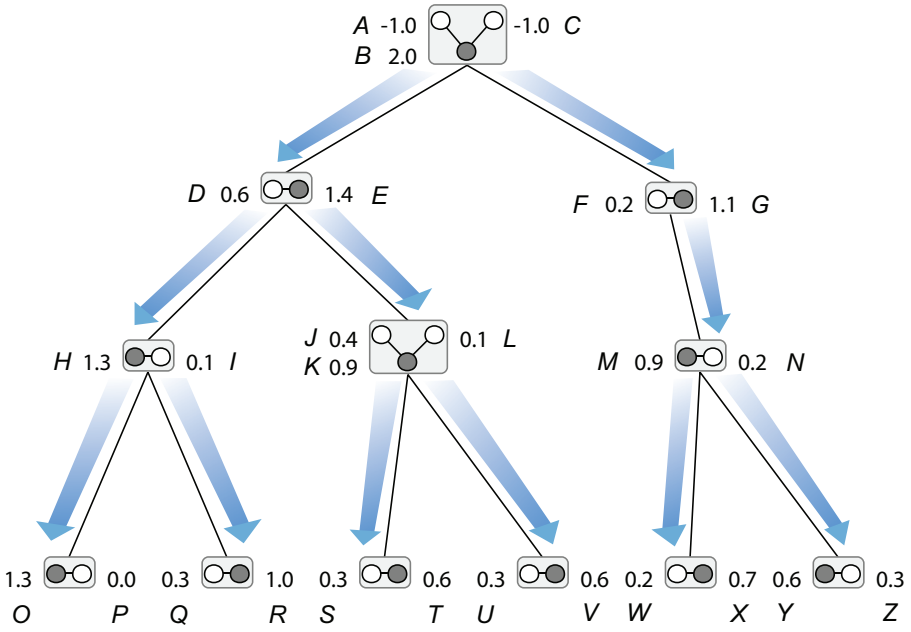



Figure 6: Generative information flow in the four-layer model.

upward flow gives rise to a downward flow as a knock-on effect, if it resets values in a choice that has descendants elsewhere. Figure 7 illustrates an example. Cueing $V$ produces an upward flow that rises first to $J/K/L$, then to $D/E$ and finally to $A/B/C$. The awarding of new values to outcomes in these three choices then produces three different downward flows, with the eventual effect of giving values to all outcomes in the model.

A hierarchical Bayesian model can be turned into an operational predictive processing system, then, simply by assuming that informational values are updated whenever possible. This way of operationalizing the scheme also resolves some of the ambiguities noted in the introduction. One problem, recall, relates to communication of prediction error. According to Clark, the upward flow of information in predictive processing should involve no more than the commnication of an error signal. In (Clark, 2013), he states that the 'information that needs to be communicated "upward" under all [predictive processing] regimes is just the prediction error: the divergence from the expected signal' (Clark, 2013, p. 183). The operational difficulties that stem from this have already been noted. Given the model's structure is defined by conditional probabilities that have their conditioning state at the level above the conditioned state—the arrangement presently assumed—Bayesian inference is itself a bottom-up process. Using Bayes' rule to derive a posterior gives probability to a state at one level, dependent on the probabilities of states at the level below. Bayesian inference

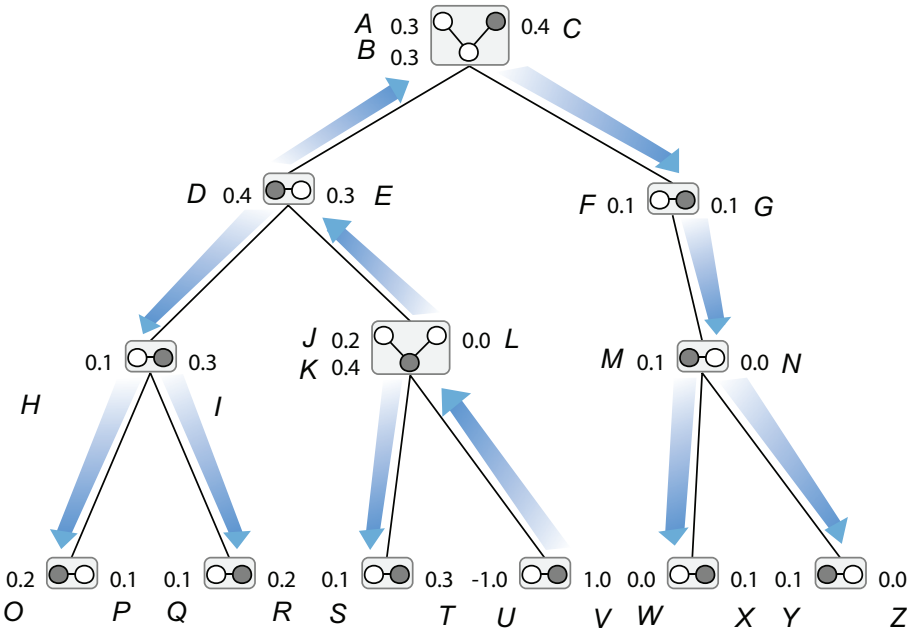



Figure 7: Mixed information flow in the four-layer model.

mediates the 'pulling up' of priors in this way. If the upward flow is mediated by error-signalling alone, Bayesian inference has no role.

A second problem is that an error-signal in the form of a surprisal value—the arrangement Clark envisages—could not play the anticipated role. Setting aside trivial predictions, such as predicting a scalar value, a prediction error does not itself indicate how the error can be reduced. An error signal will generally be ambiguous for this purpose. For the sake of operational viability we have to take as definitive, then, Clark's more general (and more recent) statement that 'unpredicted parts of the input (errors) travel up the hierarchy, leading to the adjustment of subsequent predictions' (Clark, 2016, p. 30). It has to be assumed that what flows in the upward direction is some specification of what is not yet predicted.

On this basis, the informational formulation can be seen to kill two birds with one stone. Notice, first, the sense in which the upward flow *implicitly* conveys an error signal. An outcome that mispredicts a state of affairs at the layer below acquires a *negative* information value, whose size reflects the degree of misprediction. What is communicated upward in this case is, in effect, an error signal. Taking 'what is not predicted' to be that which is potentially predicted by other outcomes of the same choice, their acquisition of less negative values also implicitly 'corrects' the error. In this sense, the upward flow of signed information both communicates and corrects prediction error. At the same

time, the upward flow realizes approximate but tractable Bayesian inference. The informational formulation satisfies several requirements, then. The upward flow conveys an error signal and gives rise to implicit error correction, while also mediating tractable Bayesian inference.

The problem of conflicts is also naturally resolved in the informational approach. Given information can flow either upward or downward within the model, it is possible for the two processes to come into conflict. There may be two ways to update the value of the same outcome, one deriving from predictions made in the layer above, and the other deriving from predictions made for the layer below. But given the competing values are quantities of information, a conflict of this kind can be resolved by adopting whichever value is greater. Given two ways of deriving the informational value of an outcome, it is natural to choose the maximum.

In one way or another, then, the problems encountered in operationalizing predictive processing are all overcome by adopting the informational model of probabilistic prediction. A multilayer probabilistic generative model that is (1) structurally defined by conditional probabilities, (2) has information-bearing outcomes as its basic states, and is (3) mandated to update information values whenever possible, yields the combination of prediction and error-correction that predictive processing requires. Formulating the regime in this way also overcomes the computational problems associated with Bayesian inference. Probabilistic prediction modeled as a Bayesian process is intractable. Modeled as an informational process, it is not.

## 3.1 Computational power

Using the informational model of probabilistic prediction, devices can be constructed that perform predictive processing in a tractable way. With this way of implementing the scheme set out, it is possible to move on to the second objective, which is to assess its computational power. Some of the ways the informational approach facilitates this may already be apparent. A predictive processing system of this type processes information rather than probability. The medium is that of conventional computation. The way we view what is accomplished is also more convenient. The awarding of informational value to outcomes can be considered a form of output. Cueing of outcomes by external action can be viewed as a form of input. A system of this type can thus be seen as an information-processing machine which maps input to output in the usual way.

A simple illustration of computation by predictive processing[11] is provided by Figure 7. On the left of the figure, a short computer program is listed. This comprises a conditional instruction which sets variable $E$ true if both $A$ and $B$ are true; otherwise, $F$ is set true if both $B$ and $G$ are true. The right side of the figure shows a predictive processing system which simulates this process by propagation of information. Key to the layout is the correspondence between

[11] Henceforth, 'predictive processing' refers to the informational implementation set out.

outcomes and program variables. The program uses the variables $A$, $B$, $E$, $F$ and $G$. For each of these, there is a corresponding outcome in a two-way choice of the predictive processing system. All conditional probabilities are shown in the central table. The top-right cell, for example, shows the predicted probability of outcome $A$ conditional on outcome $D$.

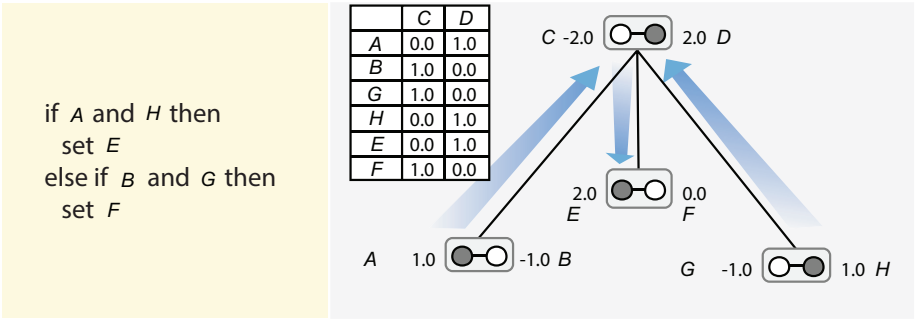



Figure 8: (1) A computer program and (2) a functionally equivalent predictive processing system.

Taking true variables to equate to positively valued outcomes, the predictive processing system perfectly replicates the conditional branch. The figure illustrates the case where $A$ and $H$ are both set true. On this basis, each of the corresponding outcomes acquires a value of 1 bit[12]. The response is an upward information flow that awards $C$ a value of 2.0 bits, and $D$ a value of -2.0 bits. Subsequent to this, information flows downward to $E$ and $F$, giving the former but not the latter a positive value. This conforms to the way the program sets variable $E$ true, but not $F$. Predictive processing perfectly replicates the computation specified by the program.

Implementing conditional branching by means of predictive processing is quite straightforward, then. What is required is a choice that includes, for each conditional case, an outcome that predicts the relevant conditions. Given the most highly-valued outcome will then be the one that best predicts conditions arising, the updating of its value has the effect of 'applying' the test. Ongoing information flows can then be the means of performing the desired 'actions'. Functionality of this type can also be generalized. The approach of Figure 8, in which the true/false distinction is captured purely by the positive/negative distinction, is simplest. But statistical conditional branches are no more difficult to implement. Given each outcome of the conditionalizing choice predicts the

[12]Outcomes are assumed to be objectively equiprobable unless otherwise specified. Representing the occurrance of an outcome from a two-way choice then entails giving it a value of $-\log_2 \frac{1}{2} = 1$ bit, and the non-occurring counterpart a value of -1 bit.

relevant conditions, the outcome of highest value will indicate the case that is best supported statistically. Ongoing information flows can then be the means of accomplishing the most justified action. Predictive processing offers two ways of branching conditionally, then, one implicitly digital, and the other inherently statistical.

| | *A0* | *A1* |
|---|---|---|
| *W0* | 0.7 | 0.3 |
| *W1* | 0.3 | 0.7 |
| *A* | 0.7 | 0.7 |
| *B* | 0.1 | 0.1 |
| *C* | 0.1 | 0.1 |
| *D* | 0.1 | 0.1 |

| | *B0* | *B1* |
|---|---|---|
| *W0* | 0.7 | 0.3 |
| *W1* | 0.3 | 0.7 |
| *A* | 0.1 | 0.1 |
| *B* | 0.7 | 0.7 |
| *C* | 0.1 | 0.1 |
| *D* | 0.1 | 0.1 |

| | *C0* | *C1* |
|---|---|---|
| *W0* | 0.7 | 0.3 |
| *W1* | 0.3 | 0.7 |
| *A* | 0.1 | 0.1 |
| *B* | 0.1 | 0.1 |
| *C* | 0.7 | 0.7 |
| *D* | 0.1 | 0.1 |

| | *D0* | *D1* |
|---|---|---|
| *W0* | 0.7 | 0.3 |
| *W1* | 0.3 | 0.7 |
| *A* | 0.1 | 0.1 |
| *B* | 0.1 | 0.1 |
| *C* | 0.1 | 0.1 |
| *D* | 0.7 | 0.7 |

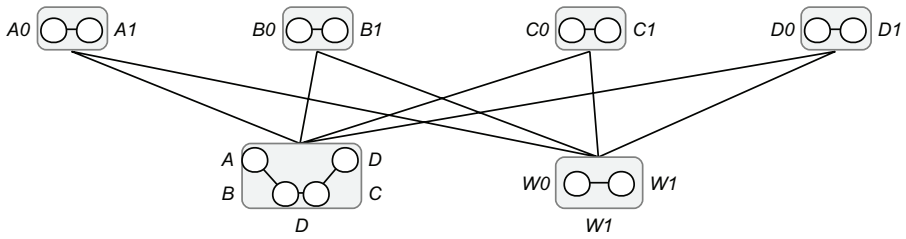



Figure 9: A four-cell predictive-processing memory for binary values.

Exploitation of conditional branches plays a fundamental role in any computation. But equally criticial is use of addressable memory. To replicate this, predictive processing has to be deployed in a more complex way. Figure 9 illustrates a simple case. The model here is a two-layer hierarchy, in which each of the four top-layer choices is considered to represent a memory cell capable of storing a binary value. Outcome *A0* represents binary 0 stored in cell *A*, outcome *C1* represents binary 1 stored in cell *C*, and so on. In the lower layer, outcomes *W0* and *W1* represent the binary value to be stored (where *W0* denotes a zero to be stored, and *W1* a 1). Outcomes *A*, *B*, *C* and *D* denote the cell in which the value is to be stored.

Notice the conditional probabilities ensure the outcomes for each memory cell predict the corresponding address and value. Outcome *A0* predicts address *A* and binary digit 0, for example. Cueing a particular address and binary value ensures the outcome which predicts this combination acquires greatest informational value. But as all predictions are made relatively weakly, it is only the outcomes of the correct choice which achieve positive values. Updating the outcomes of a choice in a way that gives negative value to them all is counterproductive by definition: the system can be assumed not to do this. The effect of presenting an address and a digit to be stored is thus to cue a particular outcome of a particular choice, namely the outcome representing storage of the digit in the correct cell. Panel (1) of Figure 10 illustrates storing a 0 in cell B;

Panel (2) illustrates storing a 1 in cell C.

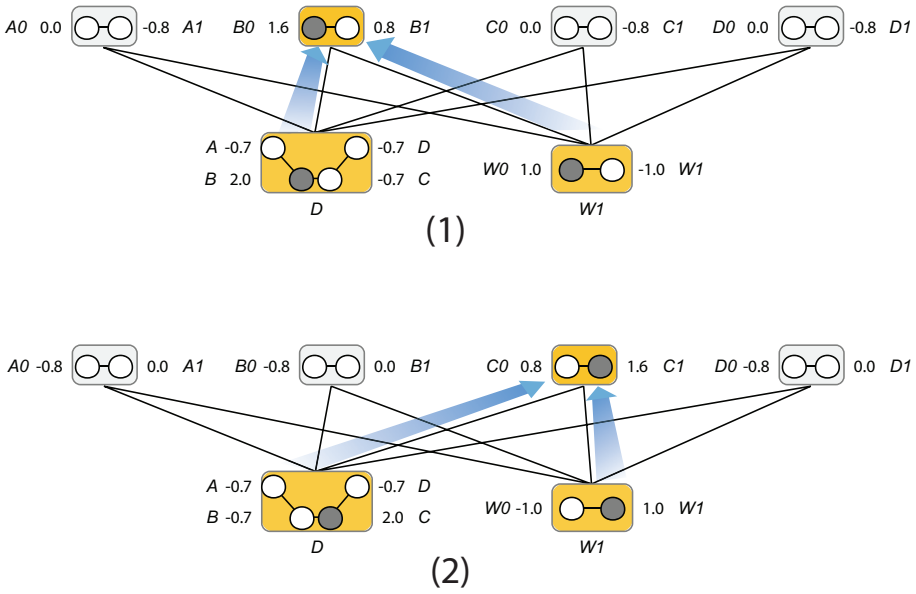



Figure 10: Storage of binary values into named memory cells.

Another critical ingredient of computation is sequencing. Application of an algorithm is achieved by carrying out actions in a particular order. Predictive processing has no central clock around which functionality of this sort can be synchronized. But it does provide ways in which sequencing can be achieved. The regime is defined to update informational values whenever possible. One way of getting actions applied in a given sequence is thus to configure their predictive relationships so as to ensure the desired executation sequence is the sequence in which updates are made. This is essentially the strategy used in the system of Figure 10.

A second approach involves distinguishing between propagational possibilities and conditional probabilities. Up to now, it has been assumed that every conditional probability can be the medium of either upward or downward propagation. It is within the general mandate of predictive processing to be more specific, however. A particular conditional probability can be defined to mediate one form of propagation only. By constraining all the probabilities in a system to mediate upward propagation, for example, we obtain a system in which the sequence of hierarchical levels determines the sequence in which actions are performed.

The general outlook for simulating generic computation by predictive processing is not unpromising, then. The means of implementing sequential execution, addressable memory and conditional branches are all ready to hand. Some

fundamental aspects of computation can be reproduced. But is this endowment sufficient for computation in general? Can we conclude that the regime has the capacity to implement any algorithm? The answer, it turns out, is that it can. Given the present way of operationalizing the regime, predictive processing can be shown to be capable of simulating a Turing machine of any complexity. On this basis, it can be identified as Turing complete.

## 3.2 Turing machine simulation

A Turing machine is a device that combines an addressable memory with a state-transition table. Very simple in its construction, it is assumed, by means of the Church-Turing thesis, to have the power to compute any function which can be computed (Turing, 1950; Abelson and Sussman, 1985). Although the device has no practical use, it is useful in the present context as a theoretical reference point. Showing that predictive processing can simulate a Turing machine of any complexity demonstrates the regime is as powerful computationally as any general-purpose computer.

A Turing machine uses a 'tape' as a storage device. This is divided into cells, each of which can store a symbol. The tape passes through a read/write-head, with the symbol immediately under the head being accessible by the machine at any one time. The machine also has a state variable, and a set of transition rules. Each rule has a condition specifying a required state and symbol. Coupled to this is an action which specifies a new state, a new symbol, and a left or right move of the tape. If the symbol required by a rule is the symbol currently read from the tape, and the required state exists, the corresponding action is executed. This produces a new state, a new symbol in the current cell of the tape, and the specified movement of the tape. The behavior of the machine is the sequence of operations that ensues given some initial state, tape and transition table.

The potential to simulate a Turing machine by predictive processing can then be demonstrated. Notice the machine depends entirely on the three functionalities identified in the previous section: conditional branching, addressable memory, and sequencing. To convert a Turing machine into a predictive processing system, we turn the transition rules into a choice implementing a conditional branch, and the tape into an addressable memory. Effects of sequential execution can then be realized by any one of the strategies outlined above.

Consider, for example, a Turing machine that computes the exclusive-or relation (XOR). This machine turns a tape representing a combination of boolean values into a tape representing their exclusive-or. Each cell of the tape can contain either a 'T' representing true, a 'F' representing false, or a '#' representing a space. A tape of the form 'T F' then represents the combination true and false, while one of the form 'T T' represents the combination true and true. Computing the exclusive-or of the values on the tape can then be defined as the task of replacing the first value on the tape with a space, and the final value with an 'F' if the original values are the same, and 'T' if they are different.

Figure 11 represents a Turing machine that operates in this way. The upper

| Defined transitions | | | | | |
|---|---|---|---|---|---|
| *ID* | State label | Symbol | New state | New symbol | Tape move |
| T0 | *X* | F | *Y* | # | *L* |
| T1 | *X* | T | *Z* | # | *L* |
| T2 | *Y* | F | *H* | F | *R* |
| T3 | *Y* | T | *H* | T | *R* |
| T4 | *Z* | F | *H* | T | *R* |
| T5 | *Z* | T | *H* | F | *R* |

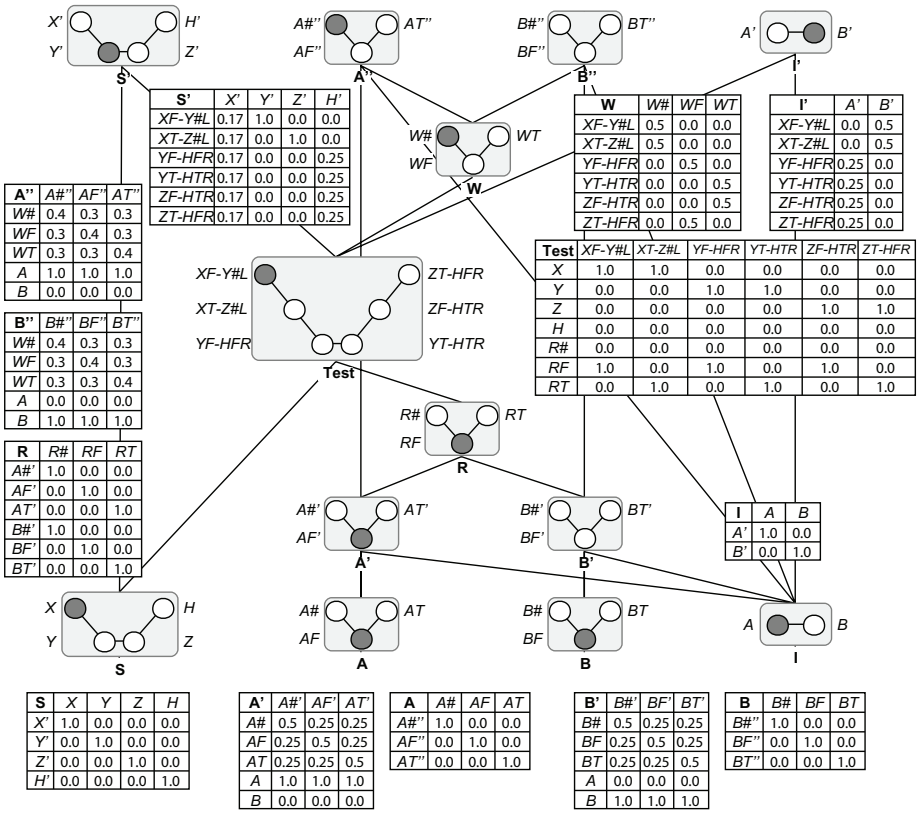
| Execution sequence | | |
|---|---|---|
| *start* | *X* | [T] F |
| T1 | *Z* | # [F] |
| T4 | *H* | [#] T |

Figure 11: The XOR Turing Machine.

table lists the machine's transition rules. The lower table shows a complete execution. Each row in the lower table shows the state and tape at a particular point of processing. The first row shows the starting configuration; each subsequent row shows the configuration reached as a result of the transition named in the first column. The machine begins in the state labeled $X$, and terminates on reaching the state labeled $H$, which is the halt state.

The processing runs as follows. Initially, the machine is in state labeled $X$, with the read/write-head placed over the first cell of the tape. This is the *start* configuration, seen in the first row of the lower table. If the symbol at this position is seen to to be T, transition T1 is performed, causing the symbol to be replaced by a #, and making $Y$ the new state label. The tape is moved one cell to the left, effectively shifting the head one cell to the right. If, in state labeled $Y$, the current symbol is seen to be F, T4 is performed, replacing the symbol with T, and entering the $H$ (halt) state. The final tape is then '# T', which correctly identifies the XOR value for the combination 'T F'. The other three combinations are dealt with in a similar way. The machine computes XOR correctly for all inputs. (A video demonstrating all four computations is available at "www.sussex.ac.uk/Users/cjt/demos/PP-TMs.mp4".)

A predictive-processing version of this Turing machine is set out in Figure 12. All conditional probabilities are defined to carry upward propagation only; information flows from bottom to top. The large choice in the center (labelled **Test**) implements the transition table. Each outcome in this choice represents a particular transition. For example, outcome $XF$-$Y\#L$ represents the rule that in state $X$ reading an F, the machine should transition to state $Y$, writing a #,




| S' | X' | Y' | Z' | H' |
|---|---|---|---|---|
| XF-Y#L | 0.17 | 1.0 | 0.0 | 0.0 |
| XT-Z#L | 0.17 | 0.0 | 1.0 | 0.0 |
| YF-HFR | 0.17 | 0.0 | 0.0 | 0.25 |
| YT-HTR | 0.17 | 0.0 | 0.0 | 0.25 |
| ZF-HTR | 0.17 | 0.0 | 0.0 | 0.25 |
| ZT-HFR | 0.17 | 0.0 | 0.0 | 0.25 |

| A'' | A#'' | AF'' | AT'' |
|---|---|---|---|
| W# | 0.4 | 0.3 | 0.3 |
| WF | 0.3 | 0.4 | 0.3 |
| WT | 0.3 | 0.3 | 0.4 |
| A | 1.0 | 1.0 | 1.0 |
| B | 0.0 | 0.0 | 0.0 |

| B'' | B#'' | BF'' | BT'' |
|---|---|---|---|
| W# | 0.4 | 0.3 | 0.3 |
| WF | 0.3 | 0.4 | 0.3 |
| WT | 0.3 | 0.3 | 0.4 |
| A | 0.0 | 0.0 | 0.0 |
| B | 1.0 | 1.0 | 1.0 |

| R | R# | RF | RT |
|---|---|---|---|
| A#' | 1.0 | 0.0 | 0.0 |
| AF' | 0.0 | 1.0 | 0.0 |
| AT' | 0.0 | 0.0 | 1.0 |
| B#' | 1.0 | 0.0 | 0.0 |
| BF' | 0.0 | 1.0 | 0.0 |
| BT' | 0.0 | 0.0 | 1.0 |

| W | W# | WF | WT |
|---|---|---|---|
| XF-Y#L | 0.5 | 0.0 | 0.0 |
| XT-Z#L | 0.5 | 0.0 | 0.0 |
| YF-HFR | 0.0 | 0.5 | 0.0 |
| YT-HTR | 0.0 | 0.0 | 0.5 |
| ZF-HTR | 0.0 | 0.0 | 0.5 |
| ZT-HFR | 0.0 | 0.5 | 0.0 |

| I' | A' | B' |
|---|---|---|
| XF-Y#L | 0.0 | 0.5 |
| XT-Z#L | 0.0 | 0.5 |
| YF-HFR | 0.25 | 0.0 |
| YT-HTR | 0.25 | 0.0 |
| ZF-HTR | 0.25 | 0.0 |
| ZT-HFR | 0.25 | 0.0 |

| Test | XF-Y#L | XT-Z#L | YF-HFR | YT-HTR | ZF-HTR | ZT-HFR |
|---|---|---|---|---|---|---|
| X | 1.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Y | 0.0 | 0.0 | 1.0 | 1.0 | 0.0 | 0.0 |
| Z | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 1.0 |
| H | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| R# | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| RF | 1.0 | 0.0 | 1.0 | 0.0 | 1.0 | 0.0 |
| RT | 0.0 | 1.0 | 0.0 | 1.0 | 0.0 | 1.0 |

| I | A | B |
|---|---|---|
| A' | 1.0 | 0.0 |
| B' | 0.0 | 1.0 |

| S | X | Y | Z | H |
|---|---|---|---|---|
| X' | 1.0 | 0.0 | 0.0 | 0.0 |
| Y' | 0.0 | 1.0 | 0.0 | 0.0 |
| Z' | 0.0 | 0.0 | 1.0 | 0.0 |
| H' | 0.0 | 0.0 | 0.0 | 1.0 |

| A' | A#' | AF' | AT' |
|---|---|---|---|
| A# | 0.5 | 0.25 | 0.25 |
| AF | 0.25 | 0.5 | 0.25 |
| AT | 0.25 | 0.25 | 0.5 |
| A | 1.0 | 1.0 | 1.0 |
| B | 0.0 | 0.0 | 0.0 |

| A | A# | AF | AT |
|---|---|---|---|
| A#'' | 1.0 | 0.0 | 0.0 |
| AF'' | 0.0 | 1.0 | 0.0 |
| AT'' | 0.0 | 0.0 | 1.0 |

| B' | B#' | BF' | BT' |
|---|---|---|---|
| B# | 0.5 | 0.25 | 0.25 |
| BF | 0.25 | 0.5 | 0.25 |
| BT | 0.25 | 0.25 | 0.5 |
| A | 0.0 | 0.0 | 0.0 |
| B | 1.0 | 1.0 | 1.0 |

| B | B# | BF | BT |
|---|---|---|---|
| B#'' | 1.0 | 0.0 | 0.0 |
| BF'' | 0.0 | 1.0 | 0.0 |
| BT'' | 0.0 | 0.0 | 1.0 |

Figure 12: Emulation of the XOR Turing machine.

and moving the tape left. The rightmost choice in the bottom layer represents the address of the current cell of the tape, where $A$ is the address of the first cell, $B$ the address of the second. The two central choices in this layer represent the cells themselves. The outcomes $A\#$, $AF$ and $AT$ represent there being (respectively) a #, F or T in cell $\mathbf{A}$. The outcomes $B\#$, $BF$ and $BT$ serve the same roles for cell $\mathbf{B}$. The four outcomes of the leftmost choice ($X$, $Y$, $Z$ and $H$) represent the possible state labels, with $H$ labeling the halt state.

The remaining constituents of the system implement the read/write functionality. Predictions are configured so that the current symbol is 'stored' into either $\mathbf{A}'$ or $\mathbf{B}'$, then 'read' into $\mathbf{R}$. Given the highest valued outcome of **Test** denotes the appropriate transition, ongoing information flow then has the effect of setting the next state (outcome of $\mathbf{S}'$) and next address (outcome of $\mathbf{I}'$) in the appropriate way. At the same time, the symbol to be written is 'placed' in $\mathbf{W}$, and then 'stored' into the addressed cell of the memory comprised of $\mathbf{A}''$ and $\mathbf{B}''$. Since each outcome in the bottom layer predicts the corresponding outcome

in the top layer, the hierarchy is circular. Ongoing upward flow of information has the effect of copying outcomes from the top layer to their counterparts in the bottom layer. The cycle then repeats.

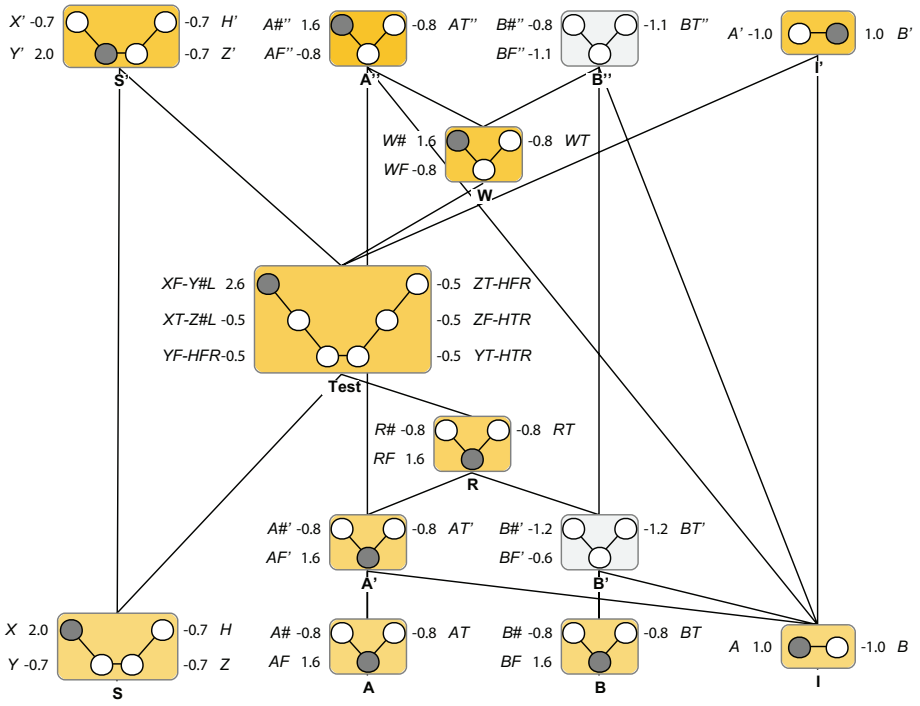



Figure 13: The system at the end of the first cycle in the XOR simulation.

Figure 13 shows the state of the system at the end of the first cycle, and Figure 14 its state at the end of the second. Propagation from this point immediately 'copies' symbol T into cell $B$, and 'selects' state $H$. Since this is the halt state, processing terminates. The final tape is then '# T', denoting true. This is the correct result for the input 'T F' (true and false).

The way this system updates the memory address exploits the fact that the tape has only two cells. After a left move of the tape, the new address can only be $B$, while after a right move, it can only be $A$. Hence, the new address can be set directly (by upward propagation) from the present outcome of the **Test** choice. In general, this shortcut cannot be used. Where a given move of the tape can produce more than one new address, it is necessary to introduce a conditional test to establish what the new address should be. This must take into account the required move and the *present* address (outcome of **I**). Appendix A sets out a more complex simulation which illustrates use of this modification.

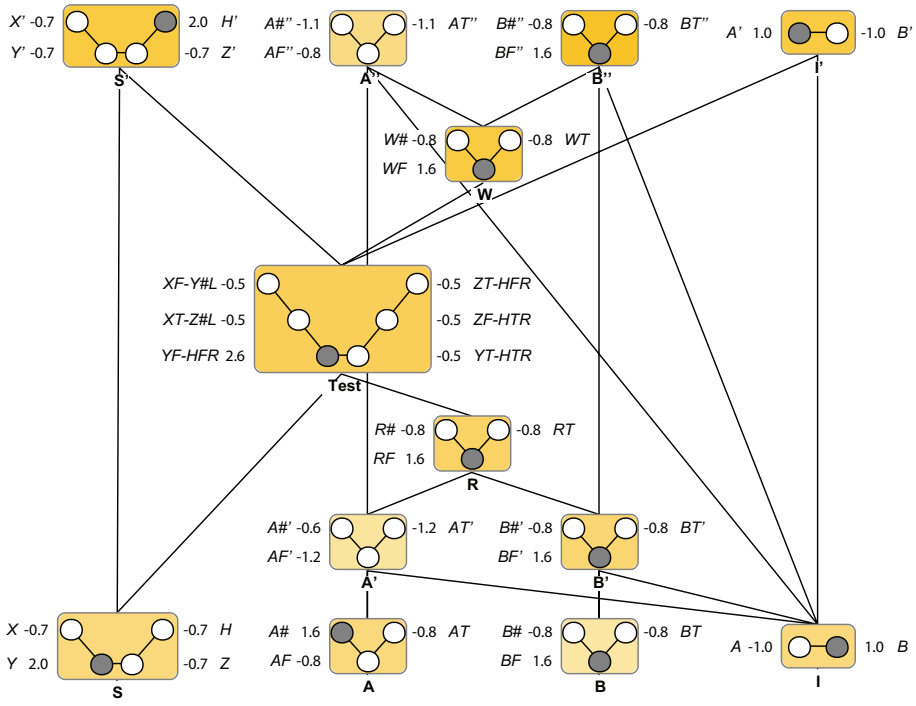



Figure 14: The system immediately preceding termination.

The more elaborate example of Appendix A also helps to illustrate the generality of the scheme. The general design of Figures 12-14 can be used to simulate a Turing machine of any complexity. There is no limit on the size of the transition table that can be accommodated. A set of $N$ transition rules can always be represented as a choice with $N$ outcomes. There is also no limit on the length of the tape. A tape containing $N$ cells can be represented as a memory based on $N$ choices. (The counterpart of a machine assumed to have an infinitely long tape is a system assumed to have infinitely many memory cells.) Regardless of its complexity, then, any Turing machine can be represented in this way. This includes the universal Turing machine, which simulates any other machine given its initial configuration. Accordingly, predictive processing can be said to be Turing complete. Its computational power is equal to that of any known computational device.

# 4 Discussion

It is important to be clear about what is and is not demonstrated above. Plausibly operationalized, predictive processing can simulate any Turing machine,

and thus any computer. The claim that this is the form of processing used by the brain does not beg the question of how computation is achieved, therefore. It offers an answer. This does not imply that the brain computes by simulating a Turing machine, of course. That would be an absurd claim. A predictive processing system that computed in such a way would be failing to exploit the propabilistic dimension of the regime altogether. A high-cost probabilistic medium would be used for purposes of implementing low-cost deterministic calculations. That computation in the brain is accomplished in such a profligate way is not what is proposed. Rather, predictive processing is noted to be a medium in which tasks of a general, computational nature can be accomplished.

To what extent does this lead to a new way of looking at brain functionality? The question of how computation is achieved in neural hardware has long been a subject of debate (Gazzaniga, 2010). On the assumption that the brain is capable of producing results which require computation, a capacity to derive these results is obviously implied. But the idea of this functionality growing out of any conventional implementation is implausible. The functional apparatus provided by the brain seems entirely inappropriate. Digital computation, as we know it, requires a precise distinction to be maintained between process and data. It requires conditional branching to be carried out according to reliably representated discrete states. For these tasks, the noisy hardware of the brain seems profoundly ill-suited (Hasson et al., 2015). As one of the innovators of computational technology, von Neumann, once observed, computation in the brain seems to be a logical impossibility. In von Neumann's opinion, any process of computation implemented in neural hardware would inevitably be 'swept away' by statistical noise (von Neumann, 1958).

More formally, von Neumann identified the obstacle as the problem of precision. As he saw it, the nervous system is simply incapable of achieving the numeric precision needed to support reliable calculation.

> [T]he nervous system transmits numerical data ... by periodic or nearly periodic trains of pulses ... [U]nder these conditions ... only precision levels of 2-3 decimals are possible ... no known computing machine can operate reliably and significantly on such low precision level' (von Neumann, 1958, pp. 76-77).

Reasoning in this way, von Neuman concluded that the brain must be using a language, arithmetic and logic 'radically different from those invented by humans' (Piccinini, 2003, p. 331).[13]

The desire to shed light on how the brain computes is not a key objective of the predictive processing proposal. Yet, on the present view, Clark's

[13]This is not the only reason for thinking computation in the brain must involve something radically unlike conventional digital processing. Another argument derives from the observation that 'too many entities turn out to be computers' (Copeland, 1996, p. 335). Since any physical system can be seen as computing an input/output function, all manner of things can be viewed as computers. A bucket of water, or even a rock, can be viewed as such (Shagrir, 2006). For this reason as well, the theory that the brain is able to compute by virtue of being a conventional computer is open to question (Putnam, 1988; Searle, 1992).

scheme does pave the way for a new account. With the informational formulation adopted, predictive processing is found to have the power of a Turing machine, and thus of any digital computer.[14] Accordingly, the proposal can be a way of explaining how the brain computes. But a side-effect of this is to generalize considerably the notion of what computation *is*. Predictive processing, as conceived above, differs radically from the processing of a von Neumann or Turing machine. There is an important commonality, however. Conventional computation uses the distinction between truth values as a way of differentiating flows of execution. The dichotomy exploited is between boolean *true* and *false*. In informational predictive processing, it is the distinction between positive and negative information which plays this role. A fundamental dichotomy is exploited, but the informational arrangement has the advantage of accommodating an inherently noisy medium of calculation. The underlying functionality of a predictive processing system is probabilistic and statistical. But with the distinction between positive and negative information superimposed, it becomes inherently digital. How the brain computes without being swept away by noise is potentially explained in this way.

This way of looking at computation also has the effect of naturalizing the process to some degree. That it is an isolated case no longer seems so obvious. Instead of seeing computation as a specialzed form of mathematics that happens to have a remarkable range of applications, we can see it as part of a family of mechanisms, that includes probabilistic prediction and statistical inference. The relationship between computational processing and basic information husbandry then becomes more evident. Both can be seen as ways of exploiting the operational possibilities that stem from a capacity to quantify information. Computation and exploiting information are naturally seen as end-points of a single continuum.

[14] The demonstration that a medium of inference can also be a model of computation offers a new perspective on Littman et al.'s (2001) result that computational complexity of a problem is not reduced when formulating it as an inference problem.

# A Emulation of an incrementing machine

| Defined transitions | | | | | |
|---|---|---|---|---|---|
| *ID* | State | Symbol | New state | New symbol | Tape move |
| T0 | *X* | # | *Y* | 1 | *L* |
| T1 | *X* | 0 | *Y* | 1 | *L* |
| T2 | *X* | 1 | *X* | 0 | *R* |
| T3 | *Y* | # | *H* | # | *R* |
| T4 | *Y* | 0 | *Y* | 0 | *L* |
| T5 | *Y* | 1 | *Y* | 1 | *L* |

| Execution sequence | | |
|---|---|---|
| *start* | *X* | # # 1 [1] # |
| T2 | *X* | # # [1] 0 # |
| T2 | *X* | # [#] 0 0 # |
| T0 | *Y* | # 1 [0] 0 # |
| T4 | *Y* | # 1 0 [0] # |
| T4 | *Y* | # 1 0 0 [#] |
| T3 | *H* | # 1 0 [0] # |

Figure 15: The incrementing Turing Machine.

The design of Figure 12 allows a minimal Turing machine—one using only two tape cells—to be represented as a predictive processing system. To simulate a machine with a longer tape, it is necessary to augment the architecture. With more tape cells in play, the new address in each cycle depends not only on the desired tape move, but also on the current address. A conditionalizing choice, whose outcomes predict particular move+address combinations, is thus required. With this introduced, the new address can be derived by information propagation in the usual way.

Consider the Turing machine of Figure 15. This accomplishes the task of incrementing a binary number. The machine is represented using the same conventions as before. The defining transitions appear in the upper table, while the lower table shows a complete execution sequence. On the tape, a binary number is represented using 1s and 0s, with # representing an unused cell.

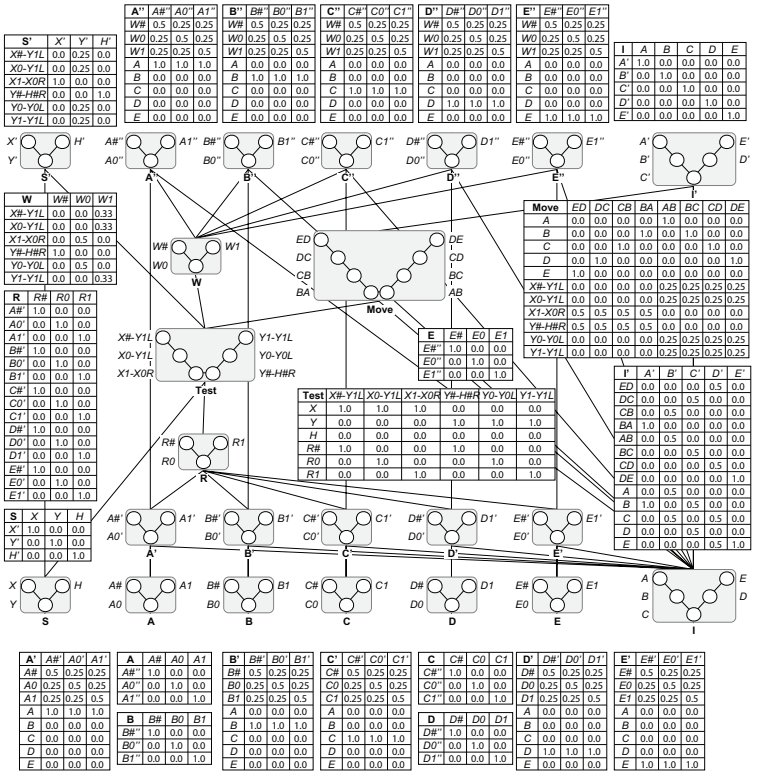


Figure 16: Simulation of the incrementing Turing machine.

Initially, the machine is in state $X$, with the tape '# # 1 1 #' representing binary 11 (decimal 3). The execution sequence that ensues contains six transitions in all, with four of the defined transitions being used. The final tape obtained is '# 1 0 0 #', representing binary 4. This is the correct output: binary 3 incremented by one. (The video at "www.sussex.ac.uk/Users/cjt/demos/PP-TMs.mp4" shows several other execution sequences.)

Represented as a predictive processing system, the machine takes the form shown in Figure 16. The general design of this simulation remains the same as before. Each cell of the tape is represented by a choice in the bottom layer. These are labelled **A**, **B**, **C**, **D** and **E**. The remainder of the architecture is identical to that used for the XOR simulation, except for the outcomes of the condionalizing choice, and the introduction of the **Move** choice. Each outcome of this predicts a particular move+address combination. Predictions made by the outcomes of $\mathbf{I}'$ then ensure upward propagation 'sets' the correct address for the next cycle.

Figure 17 shows the evaluations that arise at the end of the first cycle. The

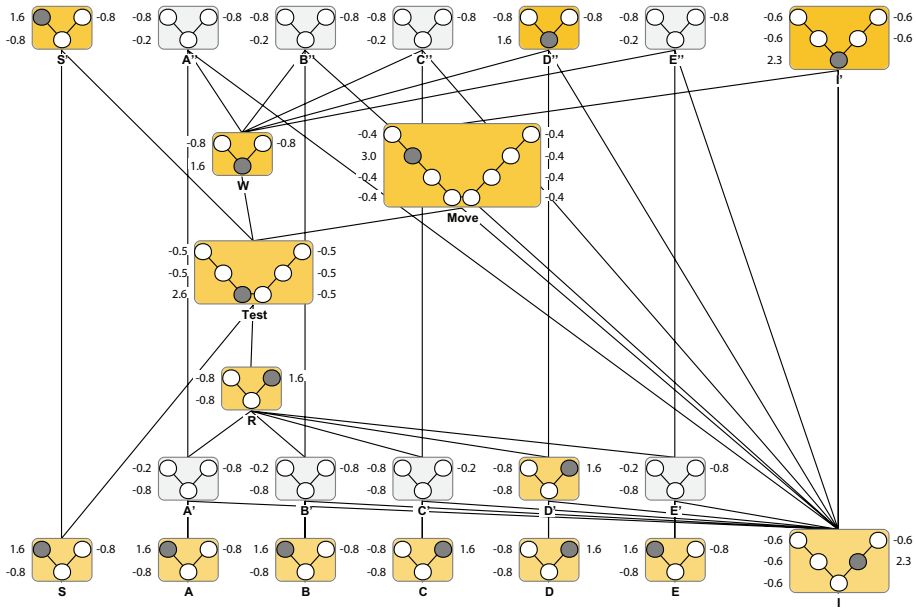



Figure 17: End of first cycle in the incrementing simulation.

symbol in the fourth cell (**D**) has been 'stored' to **D′**, and then 'copied' to **R**. The appropriate transition has been identified as *X1-X0R* (i.e., in state X reading a 1, transition to state X writing a 0, and moving the tape right). The ongoing result is that **W** is set to represent the symbol to be written, and **Move** to represent the address-specific transition *DC* (i.e., transition from address *D* to *C*). Finally, the new address (outcome of **I′**), the new symbol for cell **D**, and the new state (*X*) are all set. Since the hierarchy is circular—bottom-layer outcomes predict top-layer outcomes—upward propagation from this point on immediately resets choices in the bottom layer to the correct outcomes for the start of the next cycle.

Figure 18 portrays the system at the end of the second cycle, and Figure 19 at termination, which occurs immediately after the end of cycle 6. The system successfully increments any binary number that can be represented using the available memory cells. It increments any 1, 2, or 3 digit binary number. The static illustrations of Figures 17-19 are not an ideal way of describing the processing performed. A better approach is to show the system in action. The computational behavior is then more easily appreciated. A video demonstrating the behavior of this and the XOR simulation is available at "www.sussex.ac.uk/Users/cjt/demos/PP-TMs.mp4".

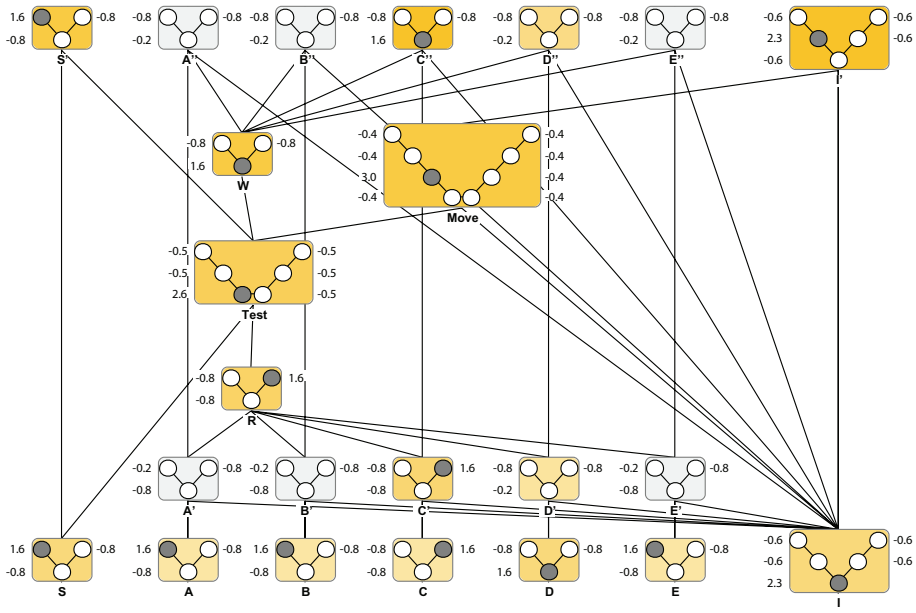



Figure 18: End of first cycle in the incrementing simulation.

# References


Abelson, H. and Sussman, G. J. (1985). *Structure and Interpretation of Computer Programs*, Cambridge, Massachusetts: MIT Press.

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly weather review*, *78* (pp. 1-3).

Brown, H., Friston, K. and Bestamnn, S. (2011). Active inference, attention and motor preparation. *Frontiers in Psychology*, *2*, 218.

Clark, A. (2013). Whatever next? Predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences*, *36* (pp. 181-253).

Clark, A. (2016). *Surfing Uncertainty: Prediction, Action, and the Embodied Mind*, Oxford: Oxford University Press.

Copeland, B. J. (1996). What Is Computation? *Synthese*, *108*, No. 3 (pp. 335-359).

Cover, T. M. and Thomas, J. A. (2006). *Elements of Information Theory 2nd Edition*, New York: John Wiley and Sons.

Eliasmith, C. (2007). How to Build a Brain: from Function to Implementation. *Synthese*, *159* (pp. 373-388).

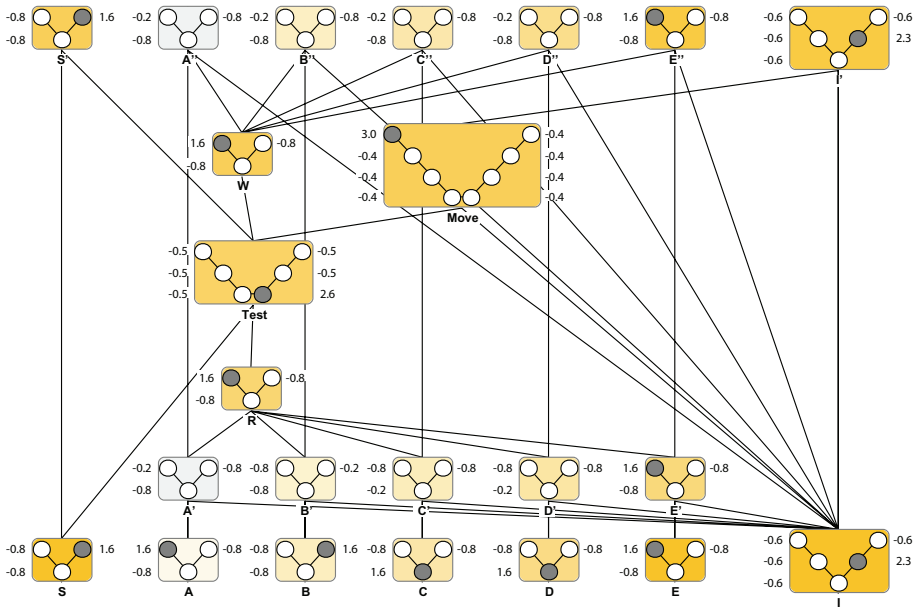



Figure 19: Termination of the incrementing simulation.

Friston, K. J., Daunizeau, J. and Kiebel, S. J. (2009). Reinforcement Learning or Active Inference. *PLoS One*, *4*, No. 7 (pp. 1-13).

Friston, K., Thornton, C. and Clark, A. (2012). Free-energy Minimization and the Dark Room Problem. *Frontiers in Perception Science*.

Friston, K. (2005). A Theory of Cortical Responses. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, *360*, No. 1456 (pp. 815-836).

Friston, K. J. (2010). The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, *11*, No. 2 (pp. 127-138).

Gazzaniga, M. S. (2010). Neuroscience and the correct level of explanation for understanding mind. *Trends in Cognitive Science*, *14* (pp. 291-292).

Hasson, U., Chen, J. and Honey, C. J. (2015). Hierarchical process memory: memory as an integral component of information processing. *Trends in Cognitive Sciences*, *19*, No. 6 (pp. 304-313).

Hohwy, J., Roepstorff, A. and Friston, K. (2008). Predictive Coding explains Binocular Rivalry: An Epistemological Review. *Cognition*, *108*, No. 3 (pp. 687-701).

Hohwy, J. (2013). *The Predictive Mind*, Oxford University Press.

Huang, Y. and Rao, R. (2011). Predictive coding. *Wiley Interdisciplinary Reviews: Cognitive Science*, *2* (pp. 580-93).

James, W. (1890/1950). *The Principles of Psychology (Vol. 1)*, New York: Dover.

Jehee, J. F. M. and Ballard, D. H. (2009). Predictive feedback can account for biphasic responses in the lateral geniculate nucleus. *PLoS (Public Library of Science) Computational Biology*, *5*, No. 5.

Knill, D. C. and Pouget, A. (2004). The Bayesian brain: the role of uncertainty in neural coding and computation. *Trends in Neuroscience*, *27*, No. 12 (pp. 712-19).

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, *22* (pp. 79-86).

Lashley, K. S. (1951). The Problem of Serial Order in Behavior. In Jeffries (Ed.), *Cerebral Mechanisms in Behavior* (pp. 112-136), New York, NY: John Wiley & Sons.

Lee, T. S. and Mumford, D. (2003). Hierarchical Bayesian Inference in the Visual Cortex. *Journal of Optical Society of America, A*, *20*, No. 7 (pp. 1434-1448).

Littman, M., Majereik, S. and Pitassi, T. (2001). Boolean satisfiability. *Journal of Automated Reasoning*, *27* (pp. 251-296).

Mackay, D. (1956). Towards an information-flow model of human behaviour. *Br. J. Psychol*, *43* (pp. 30-43).

Mackay, D. J. C. (2003). *Information Theory, Inference, and Learning Algorithms*, Cambridge: Cambridge University Press.

Piccinini, G. (2003). Book Review: John von Neumann, The Computer and the Brain, 2nd edition. *Minds and Machines*, *13* (pp. 327-332), 2.

Pouget, A., Beck, J., Ma, W. J. and Latham, P. (2013). Probabilistic brains: Knowns and unknowns. *Nature Neuroscience*, *16*, No. 9 (pp. 1170-1178).

Putnam, H. (1988). *Representation and Reality*, Cambridge, MA: MIT Press.

Rao, R. P. N. and Ballard, D. H. (1999). Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, *2*, No. 1 (pp. 79-87).

Rao, R. P. N. and Ballard, D. H. (2004). Probabilistic Models of Attention based on Iconic Representations and Predictive Coding. In Itti, Rees and Tsotsos (Eds.), *Neurobiology of Attention*, Academic Press.

Searle, J. R. (1992). *The Rediscovery of the Mind*, Cambridge, MA: MIT Press.

Shagrir, O. (2006). Why we view the brain as a computer. *Synthese*, *153* (pp. 393-416).

Shannon, C. and Weaver, W. (1949). *The Mathematical Theory of Communication*, Urbana, Illinois: University of Illinois Press.

Shannon, C. E. (1948). A Mathematical Theory of Communication. *Bell System Technical Journal*, *27* (pp. 379-423 and 623-656).

Thornton, C. (2014). Infotropism as the underlying principle of perceptual organization. *Journal of Mathematical Psychology*, *61* (pp. 38-44).

Thornton, C. (Forthcoming). Predictive processing simplified: The infotropic machine. *Brain & Cognition*.

Tolman, E. C. (1948). Cognitive Maps in Rats and Men. *Psychological Review*, *55* (pp. 189-208).

Tribus, M. (1961). *Thermodynamics and thermostatics: An introduction to energy, information and states of matter, with engineering applications*, D. Van Nostrand.

Turing, A. (1950). Computing Machinery and Intelligence. *Mind*, No. 59 (pp. 433-460).

von Helmholtz, H. (1860/1962). In Southall (Ed.), *Handbuch der physiologischen Optik, vol. 3*, Dover.

von Neumann, J. (1958). *The Computer and the Brain: Mrs Hepsa Ely Silliman Memorial Lectures*, New Haven: Yale University Press.