Technical Appendices

Michael Timothy Bennett[1]
[0000−0001−6895−8782]

The Australian National University
michael.bennett@anu.edu.au
http://www.michaeltimothybennett.com/

Abstract. The following is a list of definitions [pp. 2-5], a frequently asked questions section containing examples of how these definitions may be applied [pp. 6-8], a description of two experiments performed using the code accompanying this appendix [pp. 9-10], an anecdotal example of The Hall of Mirrors analogy [p. 11], and a list of proofs concerning the aforementioned definitions [p. 12].

1 List of definitions

Definitions 1, 2 and 3 are taken from [1]:

Definition 1 (environment).

- *We assume a set Φ whose elements we call* ***states****, one of which we single out as the* ***present state***[1]*.*
- *A* ***declarative program*** *is a function $f : \Phi \to \{true, false\}$, and we write P for the set of all declarative programs. By an* ***objective truth*** *about a state ϕ, we mean a declarative program f such that $f(\phi) = true$.*

Definition 2 (implementable language).

- $\mathfrak{V} = \{V \subset P : V \ is \ finite\}$ *is a set whose elements we call* ***vocabularies****, one of which*[2] *we single out as* ***the vocabulary*** $\mathfrak{v}$ *for an implementable language.*
- $L_\mathfrak{v} = \{l \subseteq \mathfrak{v} : \exists \phi \in \Phi \ (\forall p \in l : p(\phi) = true)\}$ *is a set whose elements we call* ***statements****. $L_\mathfrak{v}$ follows from Φ and $\mathfrak{v}$. We call $L_\mathfrak{v}$ an* ***implementable language****.*
- *$l \in L_\mathfrak{v}$ is a* ***true statement*** *iff the present state is ϕ and $\forall p \in l : p(\phi) = true$.*
- *The* ***extension of a statement*** *$a \in L_\mathfrak{v}$ is $Z_a = \{b \in L_\mathfrak{v} : a \subseteq b\}$.*
- *The* ***extension of a set of statements*** *$A \subseteq L_\mathfrak{v}$ is $Z_A = \bigcup_{a \in A} Z_a$.*

(Notation) *Z with a subscript is the extension of the subscript*[3]*. Lower case letters represent statements, and upper case represent sets of statements.*

Definition 3 ($\mathfrak{v}$-task). *For a chosen $\mathfrak{v}$, a task α is $\langle S_\alpha, D_\alpha, M_\alpha \rangle$ where:*

- *$S_\alpha \subset L_\mathfrak{v}$ is a set whose elements we call* ***situations*** *of α.*
- *S_α has the extension Z_{S_α}, whose elements we call* ***decisions*** *of α.*
- *$D_\alpha \subset Z_{S_\alpha}$ is a set whose elements we call* ***correct decisions*** *of α.*
- *$M_\alpha = \{l \in L_\mathfrak{v} : Z_{S_\alpha} \cap Z_l = D_\alpha\}$ is a set whose elements we call* ***models*** *of α.*

$\Gamma_\mathfrak{v}$ is the set of all tasks for our chosen $\mathfrak{v} \in \mathfrak{V}$.

(Notation) *If $\omega \in \Gamma_\mathfrak{v}$, then we will use subscript ω to signify parts of ω, meaning one should assume $\omega = \langle S_\omega, D_\omega, M_\omega \rangle$ even if that isn't written.*

(How a task is completed) *Assume we've a $\mathfrak{v}$-task ω and a hypothesis $\boldsymbol{h} \in L_\mathfrak{v}$ s.t.*

1. *we are presented with a situation $s \in S_\omega$, and*
2. *we must select a decision $z \in Z_s \cap Z_{\boldsymbol{h}}$.*
3. *If $z \in D_\omega$, then z is correct and the task is complete. This occurs if $\boldsymbol{h} \in M_\omega$.*

[1] Each state is just reality from the perspective of a point along one or more dimensions. States of reality must be separated by something, or there would be only one state of reality. For example two different states of reality may be reality from the perspective of two different points in time, or in space and so on.

[2] The vocabulary $\mathfrak{v}$ we single out represents the sensorimotor circuitry with which an organism enacts cognition - their brain, body, local environment and so forth.

[3] e.g. Z_s is the extension of s.

1.1 Induction definitions

Definitions 4, 5, 6, 7 and 8 are taken from [2]:

Definition 4 (probability). *We assume a uniform distribution over* $\Gamma_{\mathfrak{v}}$.

Definition 5 (generalisation). *A statement* l *generalises to* $\alpha \in \Gamma_{\mathfrak{v}}$ *iff* $l \in M_\alpha$. *We say* l *generalises from* α *to* $\mathfrak{v}$*-task* ω *if we first obtain* l *from* M_α *and then find it generalises to* ω.

Definition 6 (child and parent). *A* $\mathfrak{v}$*-task* α *is a child of* $\mathfrak{v}$*-task* ω *if* $S_\alpha \subset S_\omega$ *and* $D_\alpha \subseteq D_\omega$. *This is written as* $\alpha \sqsubset \omega$. *If* $\alpha \sqsubset \omega$ *then* ω *is then a parent of* α.

Definition 7 (proxy for intelligence). *A proxy is a function parameterized by a choice of* $\mathfrak{v}$ *such that* $q_{\mathfrak{v}} : L_{\mathfrak{v}} \to \mathbb{N}$. *The set of all proxies is* Q.

(Weakness) *The weakness of a statement* l *is the cardinality of its extension* $|Z_l|$. *There exists* $q_{\mathfrak{v}} \in Q$ *such that* $q_{\mathfrak{v}}(l) = |Z_l|$.

(Description length) *The description length of a statement* l *is its cardinality* $|l|$. *Longer logical formulae are considered less likely to generalise [3], and a proxy is something to be maximised, so description length as a proxy is* $q_{\mathfrak{v}} \in Q$ *such that* $q_{\mathfrak{v}}(l) = \frac{1}{|l|}$.

Definition 8 (induction). α *and* ω *are* $\mathfrak{v}$*-tasks such that* $\alpha \sqsubset \omega$. *Assume we are given a proxy* $q_{\mathfrak{v}} \in Q$, *the complete definition of* α *and the knowledge that* $\alpha \sqsubset \omega$. *We are not given the definition of* ω. *The process of induction would proceed as follows:*

1. *Obtain a hypothesis by computing a model* $\mathbf{h} \in \arg\max_{m \in M_\alpha} q_{\mathfrak{v}}(m)$.
2. *If* $\mathbf{h} \in M_\omega$, *then we have generalised from* α *to* ω.

1.2 ASI and AGI definitions

Definitions 9, 10, 11 and 12 are taken from [1]:

Definition 9 (artificial general intelligence). *If* $\boldsymbol{h} \in L_{\mathfrak{v}}$ *is our AGI mechanism's hypothesis and* $\alpha \in \Gamma_{\mathfrak{v}}$ *its knowledge*[4], *then* $\boldsymbol{h} \in \arg\max_{m \in M_\alpha} |Z_m|$.

Definition 10 (task mapper). *Let* $\Gamma_{\mathfrak{V}} = \bigcup_{\mathfrak{k} \in \mathfrak{V}} \Gamma_{\mathfrak{k}}$ *be the set of all tasks across all vocabularies. Let* λ *be a function* $\lambda : \mathfrak{V} \to \Gamma_{\mathfrak{V}}$ *that takes a vocabulary and returns a particular task in that vocabulary.* λ *lets us represent a version of the same task in different vocabularies. We call* λ *a* ***task mapper****, and* Λ *the set of all task mappers.*

[4] This assumes either that knowledge only consists of an ostensive definition of what "good enough" is, or that positive and negative feedback are expressed through the implementable language (as declarative programs), and that each situation expresses a threshold with respect to what is considered "good enough".

Definition 11 (utility of intelligence). *Every task $\gamma \in \Gamma_{\mathfrak{V}}$ has a utility of intelligence value, computed by a function $\epsilon : \Gamma_{\mathfrak{V}} \to \mathbb{N}$ such that $\epsilon(\gamma) = \arg\max_{m \in M_\gamma}(|Z_m| - |D_\gamma|)$.*

Definition 12 (artificial super intelligence). *Assume is a task mapper λ represents our ASI's knowledge. If $\boldsymbol{h}$ is our ASI's hypothesis then our ASI uses vocabulary $\mathfrak{v}$ s.t.*

$$\mathfrak{v} \in \arg\max_{\mathfrak{v} \in \mathfrak{V}} \epsilon\left(\lambda(\mathfrak{v})\right) \text{ and } \boldsymbol{h} \in \arg\max_{m \in M_{\lambda(\mathfrak{v})}} |Z_m|$$

If $\mathfrak{K} \subseteq \mathfrak{V}$ is the set of vocabularies for which ϵ has been computed so far, then an anytime computable alternative is $\mathfrak{v} \in \arg\max_{\mathfrak{v} \in \mathfrak{K}} \epsilon(\lambda(\mathfrak{v}))$

1.3 Causal definitions

Definitions 9 and 10 are taken from [4]:

Definition 13 (identity). *If $a \in L_{\mathfrak{v}}$ is an intervention [5] to force $c \in L_{\mathfrak{v}}$, then $k \subseteq a - c$ may function as an identity undertaking the intervention if $k \neq \emptyset$.*

Definition 14 (higher and lower level statements). *A statement $c \in L_{\mathfrak{v}}$ is higher level than $a \in L_{\mathfrak{v}}$ if $Z_a \subset Z_c$, which is written as $a \sqsubset c$.*

1.4 Symbol emergence definitions

Definition 15 (organism). *An organism $\mathfrak{o}$ is a quintuple $\langle \mathfrak{v}_\mathfrak{o}, \mathfrak{e}_\mathfrak{o}, \mathfrak{s}_\mathfrak{o}, n_\mathfrak{o}, f_\mathfrak{o} \rangle$, and the set of all such quintuples is $\mathfrak{O}$ where:*

- *$\mathfrak{v}_\mathfrak{o}$ is a **vocabulary** we single out as belonging to this organism. We assume a corresponding set of statements $L_{\mathfrak{v}_\mathfrak{o}}$ constructed from $\mathfrak{v}_\mathfrak{o}$.*
- *We assume a $\mathfrak{v}_\mathfrak{o}$-task ζ wherein S_ζ is every situation in which $\mathfrak{o}$ has made a decision, and D_ζ contains every such decision. Given the set $\Gamma_{\mathfrak{v}_\mathfrak{o}}$ of all tasks, $\mathfrak{e}_\mathfrak{o} = \{\omega \in \Gamma_{\mathfrak{v}_\mathfrak{o}} : \omega \sqsubset \zeta\}$ is a set whose members we call **experiences**.*
- *A **symbol system** $\mathfrak{s}_\mathfrak{o} = \{\alpha \in \Gamma_{\mathfrak{v}_\mathfrak{o}} : \text{there exists } \omega \in \mathfrak{e}_\mathfrak{o} \text{ where } M_\alpha \cap M_\omega \neq \emptyset\}$ is a set whose members we call **symbols**. $\mathfrak{s}_\mathfrak{o}$ is the set of every task to which it is possible to generalise from an element of $\mathfrak{e}_\mathfrak{o}$.*
- *$n_\mathfrak{o} : \mathfrak{s}_\mathfrak{o} \to \mathbb{N}$ is a function we call **preferences**.*
- *$f_\mathfrak{o} : \mathfrak{s}_\mathfrak{o} \to \mathfrak{f}_\mathfrak{o}$ is a function, and $\mathfrak{f}_\mathfrak{o} \subset L_{\mathfrak{v}_\mathfrak{o}}$ a set whose elements we call **feelings**, being the reward, qualia and so forth, from which we assume preferences arose[5].*

Definition 16 (interpretation). *We say that an organism $\mathfrak{o}$ **experiences** $s \in L_{\mathfrak{v}_\mathfrak{o}}$ to mean that $\mathfrak{o}$ must choose $\alpha \in \mathfrak{s}_\mathfrak{o}$ such that $s \in S_\alpha$, and then **interprets** s using α by making a decision $d \in Z_s \cap Z_{M_\alpha}$. The process is as follows:*

1. *We say s **signifies** a symbol $\alpha \in \mathfrak{s}_\mathfrak{o}$ if $s \in S_\alpha$.*

[5] Note that this assumes qualia, preferences and so forth are part of physical reality, which means they are sets of declarative programs.

2. $\mathfrak{s}_{\mathfrak{o}}^{s} = \{\alpha \in \mathfrak{s}_{\mathfrak{o}} : s \in S_{\alpha}\}$ *is the set of all symbols which* s *signifies.*
3. *If* $\mathfrak{s}_{\mathfrak{o}}^{s} \neq \emptyset$ *then* s ***means something*** *to the organism in the sense that there are feelings and preferences which can ascribed to symbols in* $\mathfrak{s}_{\mathfrak{o}}^{s}$ *which* s *signifies.*
4. *If* s *means something, then* $\mathfrak{o}$ *chooses* $\alpha \in \arg\max_{\omega \in \mathfrak{s}_{\mathfrak{o}}^{s}} n_{\mathfrak{o}}(\omega)$ *with which to interpret* s.
5. *Interpretation is selecting a decision* $d \in Z_s \cap Z_{M_\alpha}$, *which is the effect of* s *upon* $\mathfrak{o}$.

Definition 17 (affect). *To* ***affect*** *an organism is to cause it to make a different decision than it otherwise would have (in the sense of counterfactual).*

- *(circumstances) If* $\mathfrak{o}$ *interprets* $s \in L_{\mathfrak{v}_{\mathfrak{o}}}$ *as a decision* d, *then a* ***statement*** $v \subset s$ ***affects*** $\mathfrak{o}$ *if* $\mathfrak{o}$ *interprets* $z = s - v$ *as* $g \neq d$.
- *(information) A* ***symbol*** $\alpha \in \mathfrak{s}_{\mathfrak{o}}$ *affects* $\mathfrak{o}$ *if there exists* $s \in L_{\mathfrak{v}_{\mathfrak{o}}}$ *s.t.* $\mathfrak{o}$ *interprets* s *as* $d \in D_\alpha$, *but if* α *were subtracted from* $\mathfrak{s}_{\mathfrak{o}}^{s}$ *then either* s *would not mean something to* $\mathfrak{o}$, *or* s *would be interpreted as* $g \neq d$.
- *(another organism) An* ***organism*** $\mathfrak{k}$ *affects* $\mathfrak{o}$ *if* $\mathfrak{o}$ *would have made a decision* d, *but as a result of a decision* c *made by* $\mathfrak{k}$, $\mathfrak{o}$ *makes decision* $g \neq d$.

Definition 18 (meaning:). *Two symbols* $\alpha \in \mathfrak{s}_{\mathfrak{k}}$ *and* $\omega \in \mathfrak{s}_{\mathfrak{o}}$ *are roughly equivalent to mean that the feelings, experiences and thus preferences associated are in some sense roughly the same for each organism (meaning* $\alpha \approx \omega$ *if* $f_{\mathfrak{k}}(\alpha) \approx f_{\mathfrak{o}}(\omega)$ *and* $n_{\mathfrak{k}}(\alpha) \approx n_{\mathfrak{o}}(\omega)$*).*

$\mathfrak{k}$ *means* $\alpha \in \mathfrak{s}_{\mathfrak{k}}$ *by deciding* u *affecting* $\mathfrak{o}$ *iff* $\mathfrak{k}$ *intends in deciding* u*:*
1. that $\mathfrak{o}$ *interpets the situation at hand with* $\omega \in \mathfrak{s}_{\mathfrak{o}}$ *s.t.* $\omega \approx \alpha$,
2. $\mathfrak{o}$ *recognize this intention, for example by predicting it according to*

$$\gamma_{\mathfrak{o}}^{\mathfrak{k}} \in \arg\max_{\alpha \in \mathfrak{K}} |Z_\alpha| \text{ where } \mathfrak{K} = \arg\max_{\alpha \in \Gamma_{\mathfrak{o}}^{\mathfrak{k}}} n_{\mathfrak{o}}(\alpha), \Gamma_{\mathfrak{o}}^{\mathfrak{k}} = \{\omega \in \Gamma_{\mathfrak{v}_{\mathfrak{o}}} : M_{\zeta_{\mathfrak{o}}^{\mathfrak{k}}} \cap M_\omega \neq \emptyset\}$$

3. and (1) occur on the basis of (2), because $\mathfrak{k}$*'s intent is to co-operate and so it will interpret the situation at hand using what it has inferred of* $\mathfrak{o}$*'s intent.*

Prerequisites for the comprehension of meaning follow from he above:

1. *Organisms must be able to* ***affect one another****.*
2. *Organisms must have similar* ***feelings****, and*
3. *similar* ***experiences****, so* $\mathfrak{s}_{\mathfrak{o}}$ *and* $\mathfrak{s}_{\mathfrak{k}}$ *contain roughly equivalent symbols.*
4. *Similar* ***preferences*** *then inform the correct inference of intent.*
5. *Finally, all this assumes organisms are* ***reasonably performant****.*

2 Frequently Asked Questions

2.1 How would you apply this to solve a typical regression problem?

Say we have a finite[6] set of input values $X \subset \mathbb{R}$ and output values $Y \subset \mathbb{R}$, and $g : X \to Y$ be a function we wish to model. $G = \{(x, y) \in X \times Y : g(x) = y\}$, and we call G the ground truth. Let $Train \subset G$ be a training set, and $Test \subset G$ be a test set. We are given $Train$ and $Test$, and our goal is to infer G. Typically, machine learning could be used to obtain an approximation of G from $Train$ by doing the following:

1. Fit a function f (e.g. a neural net) such that $\forall (x, y) \in Train\,(f(x) \approx y)$[7].
2. Measure test accuracy as $\frac{|\{(x,y) \in Test : y \approx f(x)\}|}{|Test|}$.
3. If accuracy good enough, use f to make predictions $P = \{(x, y) \in X \times Y : f(x) = y\}$ and hope that $\frac{|\{(x,y) \in P : y \approx g(x)\}|}{|P|} \approx 1$.

This is how we would solve the problem using machine learning normally. Now let us represent this as a task and solve it that way.

1. We start by creating an implementable language.
   (a) Begin by defining the vocabulary V of the implementable language $\langle V, L \rangle$[8]. We need to create V first and then L, because we cannot actually create Φ[9].
   (b) To obtain L, we must first write a program $converter : X \cup Y \to 2^V$ which converts members of X and Y into sets of declarative programs, and another program $converter^{-1} : 2^V \to X \cup Y$ which reverses that process (meaning an isomorphism between $X \cup Y$ and 2^V).
   (c) Next we define $pair\_converter : X \times Y \to 2^V$ such that $\forall (x, y) \in X \times Y$, $pair\_converter((x, y)) = converter(x) \cup converter(y)$ and likewise the inverse $pair\_converter^{-1}((converter(x) \cup converter(y))) = (x, y)$.
   (d) A set Q can be created as

$$Q = \{v \in 2^V : \exists (x, y) \in X \times Y\,(pair\_converter((x, y)) = v)\}$$

   We can then use Q to create L as each member of Q must be a subset of an objective totality $h \in H$ (even though we haven't needed to explicitly define H), meaning $L = \{l \in 2^V : \exists v \in Q\ (l \subseteq v)\}$.
2. Now we can use $converter$ and $pair\_converter$ to define a task $\langle S, D, M \rangle$.
   (a) First we compute $S = \{s \in L : \exists (x, y) \in Train\,(converter(x) = s)\}$.
   (b) Second we create the set of correct decisions

$$D = \{d \in L : \exists (x, y) \in Train\,(pair\_converter((x, y)) = d)\}$$

[6] We assume finite X and Y for practical reasons, for example that the real numbers we can represent as floating point values in a computer are constrained by the number of bits used.

[7] $a \approx b$ just means that there exists a very small number c and a measure of distance d such that $d(a, b) < c$, meaning the distance $d(a, b)$ between a and b is less than c

[8] The choice of V permit an isomorphism between $X \times Y$ and 2^V

[9] We must obtain L from directly V using the structure inherent in the values of X and Y (e.g. X and Y represent real numbers in different parts of memory in a computer, not the same part, and so we can create statements in L describing values from both without creating problems).

(c) Finally to create $M \subset L$ by excluding members of L according to the definition:

$$M = \{m \in L : Z_S \cap Z_m \equiv D, \forall z \in Z_m \ (z \subseteq \bigcup_{d \in D} d)\}$$

3. We now have a set of models M and situations S which we can treat as constraints, and can define a program $search : S, M \to D$ which, given any situation and model, returns a decision in D. We can now measure the accuracy of a given model $m \in M$.
   (a) First we compute $S_{Test} = \{s \in L : \exists (x, y) \in Test\,(converter(x) = s)\}$.
   (b) Compute $D_{Test} = \{l \in L : \exists s \in S_{Test}\,(search(s, m) = l)\}$.
   (c) Convert to real numbers by computing:

$$P = \{(x, y) \in X \times Y : \gamma(x, y)\}$$

   where $\gamma(x, y)$ means

$$\exists d \in D_{Test}\left(pair\_converter^{-1}(d) = (x, y)\right)$$

   (d) Measure accuracy as:

$$\frac{|\{(x, y) \in Test : \exists (a, b) \in P\,((x, y) \approx (a, b))\}|}{|Test|}$$

4. Assuming test accuracy is acceptable, we can use this to predict the ground truth G.
   (a) Compute $S_X = \{l \in L : \exists x \in X\,(converter(x) = l)\}$, which is the set of all situations in which we need to make a decision.
   (b) Choose a model $m \in M$.
   (c) Compute $D_{Predicted} = \{l \in L : \exists s \in S_X\,(search(s, m) = l)\}$.
   (d) Convert to real numbers by computing

$$G_{Predicted} = \{(x, y) \in X \times Y : \delta(x, y)\}$$

   where $\delta(x, y)$ means

$$\exists d \in D_{Predicted}\left(pair\_converter^{-1}(d) = (x, y)\right)$$

2.2 Example of an implementable language

- There exist 4 bits bit_1, bit_2, bit_3 and bit_4, to which each $\phi \in \Phi$ assigns a value.
- $V = \{a, b, c, d, e, f, g, h, i, j, k, l\}$ is a subset of all logical tests which might be applied to these 4 bits:

• $a : bit_1 = 1$	• $d : bit_4 = 1$	• $g : bit_3 = 0$	• $j : bit_1 = bit_3$
• $b : bit_2 = 1$	• $e : bit_1 = 0$	• $h : bit_4 = 0$	• $k : bit_2 = bit_4$
• $c : bit_3 = 1$	• $f : bit_2 = 0$	• $i : j \wedge k$	• $l : i \vee bit_2 = 1$

- $L = \{\{a, b, c, d, i, j, k, l\}, \{e, b, c, d, k, l\}, \{a, f, c, d, j\}, \{e, f, c, d\}, \{a, b, g, d, k, l\}, \{e, b, g, d, i, j, k, l\}, \{a, f, g, d\}, \{e, f, g, d, j\}, \{a, b, c, h, j, l\}, \{a, b, g, h, l\}, \{e, b, c, h, l\}, \{a, f, c, h, i, j, k, l\}, \{e, f, c, h, k\}, \{e, b, g, h, j\}, \{a, f, g, h, k\}, \{e, f, g, h, i, j, k, l\}\}$

2.3 Example of a task ω

- $S_\omega = \{\{a,b\},\{e,b\},\{a,f\},\{e,f\}\}$
- $D_\omega = \{\{a,b,c,d,i,j,k,l\},\{e,b,g,d,i,j,k,l\},\{a,f,c,h,i,j,k,l\},\{e,f,g,h,i,j,k,l\}\}$
- $M_\omega = \{\{i\},\{j,k\},\{i,j,k\},\{i,l\}...\}$

2.4 Example of a child-task α of ω

- $S_\alpha = \{\{a,b\},\{e,b\}\}$
- $D_\alpha = \{\{a,b,c,d,i,j,k,l\},\{e,b,g,d,i,j,k,l\}\}$
- $M_\alpha = \{\{i,j,k,l\},\{b,d,j\},...\}$
  - Weakest model $\mathbf{m} = \{i,j,k,l\}$
  - Strongest model $\mathbf{e} = \{b,d,j\}$
  - $Z_\mathbf{m} = \{\{a,b,c,d,i,j,k,l\},\{e,b,g,d,i,j,k,l\},\{a,f,c,h,i,j,k,l\},\{e,f,g,h,i,j,k,l\}\}$
  - $Z_\mathbf{e} = \{\{a,b,c,d,i,j,k,l\},\{e,b,g,d,i,j,k,l\}\}$

3 Experiments

Context for and a detailed analysis of these experiments (see "experiments.py") is given in [2]. What is given here is merely a brief technical report on the two experiments. In these two experiments, a toy program computes models to 8-bit string prediction tasks (binary addition and multiplication).

Implementable language: There were 256 states, one for every possible 8-bit string. The statements in L were expressions regarding those 8 bits that could be written in propositional logic ($\neg$, $\wedge$ and $\vee$).

Task: A task was specified by choosing $D \subset L$ such that all $d \in D$ conformed to the rules of either binary addition (for the first experiment) or multiplication (for the second experiment) with 4-bits of input, followed by 4-bits of output.

3.1 Trials

Each of the two experiments (addition and multiplication) involved repeated trials trials (sampling results). The parameters of each trial were "operation" (a function), and an even integer "number_of_trials" between 4 and 14 which determined the cardinality of the set D_k (defined below). Each trial was divided into training and testing phases.

Training phase:

1. A task T_n was generated:
   (a) First, every possible 4-bit input for the chosen binary operation was used to generate an 8-bit string. These 16 strings then formed D_n.
   (b) A bit between 0 and 7 was then chosen, and S_n created by cloning D_n and deleting the chosen bit from every string (meaning S_n was composed of 16 different 7-bit strings, each of which could be found in an 8-bit string in D_n).
2. A child-task $T_k = \langle S_k, D_k, M_k \rangle$ was sampled from the parent task T_n. Recall, $|D_k|$ was determined as a parameter of the trial.
3. From T_k two models (rulesets) were generated; a weakest c_w, and a MDL c_{mdl}.

Testing phase: For each model $c \in \{c_w, c_{mdl}\}$:

1. The extension Z_c of c was then generated.
2. A prediction D_{recon} was then constructed s.t. $D_{recon} = \{z \in Z_c : \exists s \in S_n \ (s \subset z)\}$.
3. D_{recon} was then compared to the ground truth D_n, and results recorded.

Between 75 and 256 trials were run for each value of the parameter $|D_k|$. Fewer trials were run for larger values of $|D_k|$ due to restricted availability of hardware. The results of these trails were then averaged for each value of $|D_k|$.

3.2 Measurements

Rate at which models generalised completely: Generalisation was deemed to have occurred where $D_{recon} = D_n$. The number of trials in which generalisation occurred was measured, and divided by n to obtain the rate of generalisation for c_w and c_{mdl}. Error was computed as a Wald 95% confidence interval.

Average extent to which models generalised: Even where $D_{recon} \neq D_n$, the extent to which models generalised could be ascertained. $\frac{|D_{recon} \cap D_n|}{|D_n|}$ was measured and averaged for each value of $|D_k|$, and the standard error computed.

Table 1: Results for Binary Addition

	c_w				c_{mdl}			
$\|D_k\|$	Rate	±95%	AvgExt	StdErr	Rate	±95%	AvgExt	StdErr
6	.11	.039	.75	.008	.10	.037	.48	.012
10	.27	.064	.91	.006	.13	.048	.69	.009
14	.68	.106	.98	.005	.24	.097	.91	.006

Table 2: Results for Binary Multiplication

	c_w				c_{mdl}			
$\|D_k\|$	Rate	±95%	AvgExt	StdErr	Rate	±95%	AvgExt	StdErr
6	.05	.026	.74	.009	.01	.011	.58	.011
10	.16	.045	.86	.006	.08	.034	.78	.008
14	.46	.061	.96	.003	.21	.050	.93	.003

4 Bing chat anecdote exemplifying The Hall of Mirrors

These images pertain to The Hall of Mirrors analogy given in [6].

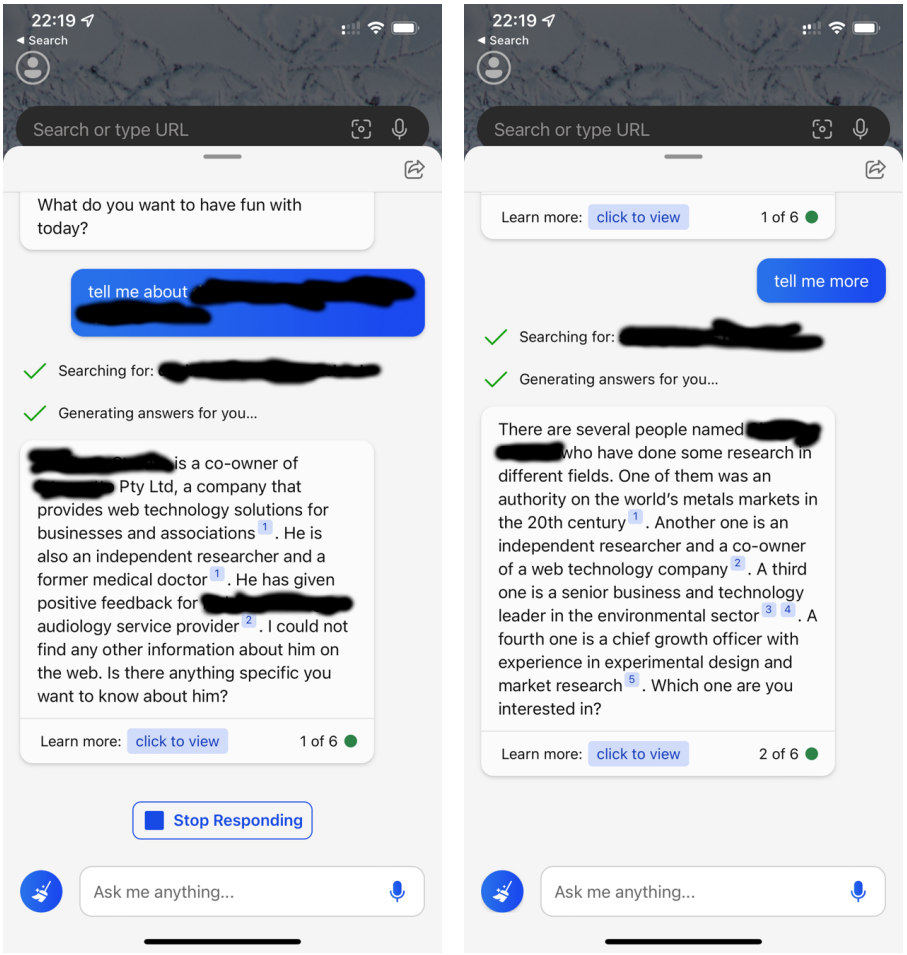

Fig. 1: Note that all these descriptions refer to the same person. Bing has confused the different aspects of an object for different objects.

5 List of proofs

These proofs support the claims of [7]. Longer and more detailed versions of these and other proofs are given in [2].

Proposition 1 (sufficiency). *Weakness is a proxy sufficient to maximise the probability that induction generalises from α to ω.*

Proof: You're given $\alpha = \langle S_\alpha, D_\alpha, M_\alpha \rangle$ and a hypothesis $\mathbf{h} \in M_\alpha$. Let $\omega = \langle S_\omega, D_\omega, M_\omega \rangle$ be the parent to which we wish to generalise:

1. The set of statements which *might* be decisions addressing situations in S_ω and not S_α, is $\overline{Z_{S_\alpha}} = \{l \in L : l \notin Z_{S_\alpha}\}$.
2. For any given $\mathbf{h} \in M_\alpha$, the set of decisions $\mathbf{h}$ implies which fall outside the scope of what is required for the known task α is $\overline{Z_{S_\alpha}} \cap Z_{\mathbf{h}}$.
3. Increasing $|Z_{\mathbf{h}}|$ increases[10] $|\overline{Z_{S_\alpha}} \cap Z_{\mathbf{h}}|$, because $\forall z \in Z_m : z \notin \overline{Z_{S_\alpha}} \rightarrow z \in Z_{S_\alpha}$.
4. $2^{|\overline{Z_{S_\alpha}}|}$ is the number of tasks which fall outside of what it is necessary for a model of α to generalise to, and $2^{|\overline{Z_{S_\alpha}} \cap Z_{\mathbf{h}}|}$ is the number of those tasks to which a given $\mathbf{h} \in M_\alpha$ does generalise.
5. The probability that a model $\mathbf{h} \in M_\alpha$ generalises to the unknown parent task ω is

$$p(\mathbf{h} \in M_\omega \mid \mathbf{h} \in M_\alpha, \alpha \sqsubset \omega) = \frac{2^{|\overline{Z_{S_\alpha}} \cap Z_{\mathbf{h}}|}}{2^{|\overline{Z_{S_\alpha}}|}}$$

$p(\mathbf{h} \in M_\omega \mid \mathbf{h} \in M_\alpha, \alpha \sqsubset \omega)$ is maximised when $|Z_{\mathbf{h}}|$ is maximised.

Proposition 2 (necessity). *To maximise the probability induction generalises from α to ω, it is necessary to use weakness, or a function thereof, as proxy.*

Proof: Let α and ω be defined exactly as they were in the proof of prop. 1.

1. If $\mathbf{h} \in M_\alpha$ and $Z_{S_\omega} \cap Z_{\mathbf{h}} = D_\omega$, then it must be he case that $D_\omega \subseteq Z_{\mathbf{h}}$.
2. If $|Z_{\mathbf{h}}| < |D_\omega|$ then generalisation cannot occur, because that would mean that $D_\omega \not\subseteq Z_{\mathbf{h}}$.
3. Therefore generalisation is only possible if $|Z_m| \geq |D_\omega|$, meaning a sufficiently weak hypothesis is necessary to generalise from child to parent.
4. The probability that $|Z_m| \geq |D_\omega|$ is maximised when $|Z_m|$ is maximised. Therefore to maximise the probability induction results in generalisation, it is necessary to select the weakest hypothesis.

To select the weakest hypothesis, it is necessary to use a weakness based proxy.

[10] Monotonically.

References

[1] M. T. Bennett. *Enactivism & Objectively Optimal Super-Intelligence*. 2023. URL: https://arxiv.org/abs/2302.00843.

[2] M. T. Bennett. *The Optimal Choice of Hypothesis Is the Weakest, Not the Shortest*. 2023. URL: arxiv.org/abs/2301.12987.

[3] J. Rissanen. "Modeling By Shortest Data Description*". In: *Autom.* 14 (1978), pp. 465–471.

[4] M. T. Bennett. *Emergent Causality & the Foundation of Consciousness*. 2023. URL: arxiv.org/abs/2302.03189.

[5] J. Pearl and D. Mackenzie. *The Book of Why: The New Science of Cause and Effect*. 1st. New York: Basic Books, Inc., 2018.

[6] M. T. Bennett. *How to Compute Meaning & Lovecraftian Horrors*. 2023.

[7] M. T. Bennett. "Symbol Emergence and the Solutions to Any Task". In: *Artificial General Intelligence*. Cham: Springer, 2022, pp. 30–40.