Learning perception lattices to compare generative explanations of human-language activity stories

Bo Morgan

July 10, 2006

Abstract

Recent research in building large semantic networks of commonsense human-knowledge, such as ConceptNet, has been driven by a small number of hand-programmed rules for recognizing and placing value on specific numbers and types of semantic relationships. We propose statistical algorithms for learning these semantic relationships from partially ordered sets (posets) of symbolic data by using a modal theory of perception based on generative stacked functional graphs that are developed explicitely to deal with the problems of objective self-reflection in a representation called a **perception lattice**. We apply the same techniques for analyzing large text corpora to streams of sensor data, providing directions toward cross-modality learning, although this work is not developed in detail in this paper. We develop an efficient, $O(n \log n)$, context-dependent mutual information calculation for posets using these generative functional grammars within the perception lattice representation. Philosophical considerations of developing representations for working toward self-reflection by blurring the objective duality between algorithms and data are briefly discussed.

1 Learning to recognize semantic relationships

Perception is the process of explaining the data, what caused the data, what process generated the data. Data are explained by their generative causal regularities, which is to say their consistent relationships with other types of data. These regularities or patterns describe ranges or modalities of perception. A **mode of perception** is a consistent relationship between elements within a subset of data. This consistent relationship defines a regularity that serves to organize a larger set of data. A mode of perception can act as a manifold in the space of percepts such that percepts lie on a specific manifold or they lie off of that manifold. Given that a set of data points lie on a mode of perception, these data points contain percepts that vary along the mode. The specific percepts that define these data points are implicitly constrained by this manifold or mode.

Richards et al. (1994) have developed a Bayesian formulation of the modes of the probability densities of image features, which they have described briefly in the following excerpt:

> Our framework for understanding percepts is based on recognizing that certain image structures point reliably to particular regularities of properties in the world with which we are familiar and expect in certain contexts. In other words, these regularities have high priors in that context. ... We regard these properties as special, in that their probability density functions are "modal", whereas in contrast [other] properties ... have broad density functions.

An example of a modal percept in text processing is the generative function

$$\mathcal{G}_{is\text{-}a}(x, y) = \text{“The ”} + x + \text{“ is a ”} + y + \text{“.”} \tag{1}$$

is a common modal textual pattern that is used in much of the natural language processing community. Liu & Singh (2004) have built a semantic language network called “ConceptNet” that contains a generalization of this modal structure that includes all forms of the verb “to be” as well as other linguistic techniques (e.g. “is goal of”) that allow a general type of semantic relationship between the conceptual text variables x and y in Equation 1. In ConceptNet this is refered to as the *is-a* semantic relationship. All semantic relationships in ConceptNet are binary relationships (only take two variable arguments [x and y in this case]). ConceptNet is limited to 20 different types of hand-programmed types of English semantic relationships, which are not limited to specific parts of speech such as verb or preposition relationships but instead represent more abstract “Madlib” relations; for example,

$$\mathcal{G}_{is\text{-}used\text{-}for}(x, y) = \text{“A ”} + x + \text{“ is used for ”} + y + \text{“.”}. \tag{2}$$

See Appendix Section 9.1 for a complete list of the ConceptNet relations. We propose a theory of modal perception lattices for posets (partially ordered sets) in order to generalize the ConceptNet relation to take any number of arguments (an N-ary relation) and also support the automatic learning of these perception lattices from a raw-text corpus of activity stories consisting of lists of common goal-oriented actions in human-language. We demonstrate the effectiveness of our learning similar modal relationships sensor streams that have been categorized into symbolic posets as well. The first question that is raised by any process of generalization is the question “can the general method perform the same purpose as the specific method?” We will show the answer to this question to be “Yes, although sequential modal relations are not a replacement, but a (1) less efficient (in terms of algorithm implementation complexity, running time, and memory usage), albeight (2) more powerful (in terms of the arbitrary semantic analogies that can be found in data of an arbitrary language) when compared to ConceptNet relations.” In other words, these sequential modes are powerful learning tools that should be compiled into forms that are as efficient and easy to use as ConceptNet relations.

Orderings of data representations within LifeNet (Singh & Williams 2003) inference, such as Unicode strings on an arbitrary axis (e.g. “time”) is one of the primary functions that LifeNet serves in considering how data is arranged relative to one another, while also considering all contextual dimensions for knowledge in whatever axes the contextual relationships for data are provided. Axes’ names are allocated dynamically by the LifeNet algorithm, so at any time a user can specify new data in a relationship along a new axis and those data will then be reasoned about in those dimensions.

2 Recognizing generative functions

A **generative function** is a computational process that takes a set of arguments and returns a set of data derived from those arguments. As computational processes, generative functions assume an algebra of data processing computation. This generative functions in this paper assume an algebra of stacked poset concatenation functions, which are easily implemented on classical digital computers, but in general, generative functions could assume biological neural networks or quantum computers as other algebras of data generative computation.

Measuring relative similarities and differences between arbitrary pieces of data depends on the generative modes that define the structure of the larger context of the pieces of data

in question. Perceptual modes can be used for a number of tasks, including the detection irregular data in the context of regular data. This detection of irregular data can be used to direct the focus of a learning algorithm that is trying to develop a model of the regularities of a given set of generated data. This assumption that the data is generated is in fact a very large assumption, especially once we give a definition for generated data, but for the domain of the problem that LifeNet is applied, the assumption of generated data is argued to be a good one: different objects generate different patterns of sensor data; different language is generated by different structural patterns of language.

3 Searching for optimal generative functions

Optimal generative functions are only optimal relative to a dataset and how well the function serves to reduce the overall number of bits necessary to functionally generate the dataset. For example, the following set of two arbitrary data, $K = \{$"The sky is blue.", "The house is green."$\}$, have the following shared generative function:

$$K_{G_0} = \Lambda(x, y)\text{"The "} + x + \text{" is "} + y + \text{"."}$$

This generative function is chosen greedily with a heuristic based on the estimated compression of the the overall dataset would be accomplished by remembering this specific generative function. Only one variable is supported currently in the search for this generative function. It would be nice to have more than one variable possible, and it could possibly be done by searching up one level in the information compression/generation tree and then searching for similar subnodes between nodes in that higher layer in order to create a multiple variabled generative function heuristic search.

If we find K_{G_0} to be the highest heuristically ranked ($K_{G_{r=0}}$) generative function for the dataset, K. We can remember the generative function, K_{G_0}, and the following sets of argument data:

$$K_{G_0 A} = \{\{\text{"sky"}, \quad \text{"blue"}\}, \\ \{\text{"house"}, \quad \text{"green"}\}\}$$

With the generative function, K_{G_0}, and the arguments list, $K_{G_0 A}$, the initial dataset can be recreated or generated functionally:

$$K_{G_0}(\text{"sky"}, \text{"blue"}) = \text{"The sky is blue."} \\ K_{G_0}(\text{"house"}, \text{"green"}) = \text{"The house is green."}.$$

These generative function explanations create hierarchical lattices of generative functional explanation for perception data. A **perception lattice** is a lattice data structure that represents the generative functional explanation for a given set of data. This lattice structure is used for many algorithmic operations over perceived data. For example, a perception lattice could be used to find the most likely top-down explanation of bottom-up perceptions, or alternatively, a perception lattice could be used for the projection of low-level details given high-level evidence. We use the term "perception lattice" very similarly to the *structure lattice* in Richards et al. (1994) except for the philosophical differences discussed in Section 7 regarding the objective duality between the observer and the world and how to interpret these as self-reflective computational processes.

4 Temporal modes

The process of calculating mutual information between two pieces of data requires certain assumptions of the generative processes that could have created these data. Our generative model assumes a stacked functional model similar to a generative grammar. All of the graphs that this method currently compares and abstracts into function/argument temporal forms are hierarchical trees. In calculating analogy structures, rather than only using the nodes and edges of a predefined semantic network (Forbes similarity method), or just the edges (ConceptNet method), the method that LifeNet uses compares mutual information between nodes by considering their generative functional structures. These generative functional structures are generalized edge types that are learned based on the given computational algebra, which is in this case a stacked functional language of simple concatenation functions. We assume that Unicode strings within ConceptNet are created by a hierarchy of generative functions, which as a whole can be structure mapped against other ConceptNet concepts resulting in analogical mappings that result in variable mappings that generalize the idea of a binary ConceptNet relation to an N-ary LifeNet relation. N-ary LifeNet relations are referred to as cliques rather than as edges, which are binary cliques. These generalized forms of ConceptNet links can, for example, recognize sentence forms, such as:

$$\mathcal{G}(A, B) = \text{“The ”} \ + \ A \ + \ \text{“ is a ”} \ + \ B \ + \ \text{“.”}. \tag{3}$$

These generative functional structures can be used to calculate mutual information between two streams by considering the execution of a generative function to be an event that occurs in the context of given argument values. The context of the argument values provides a probability distribution over possible generative functions for each branch in the hierarchical generative function structure. Once this probability is defined in terms of specific probabilities for each structural change to a generative function structure, a mutual information distance function can be defined between each pair of data. LifeNet quickly calculates the correct probabilities for the generative function execution events by using a perception lattice.

LifeNet has the ability to recognize these generative function structures for temporal sequences using a limited set of functions that involve different orderings of the string concatenation function. LifeNet cannot yet recognize generative functional structures for spatial relationships.

5 Learning perception lattices from data by greedy compression search

A **greedy compression search** is a search algorithm that begins with a list of uncompressed data, L. For all of the data in L the largest contiguous repetetive section, x, of data is found. Every datum in L containing x is removed from L and split into smaller non-contiguous pieces that do not contain x. These smaller non-contiguous pieces are appended to L, and the process of removing redundant sections of data continues until no such sections exist in L, at which point L will contain the leaves of the perception lattice structure.

Simple examples of the perception lattices resulting from this greedy compression search algorithm are shown in Figure 1.

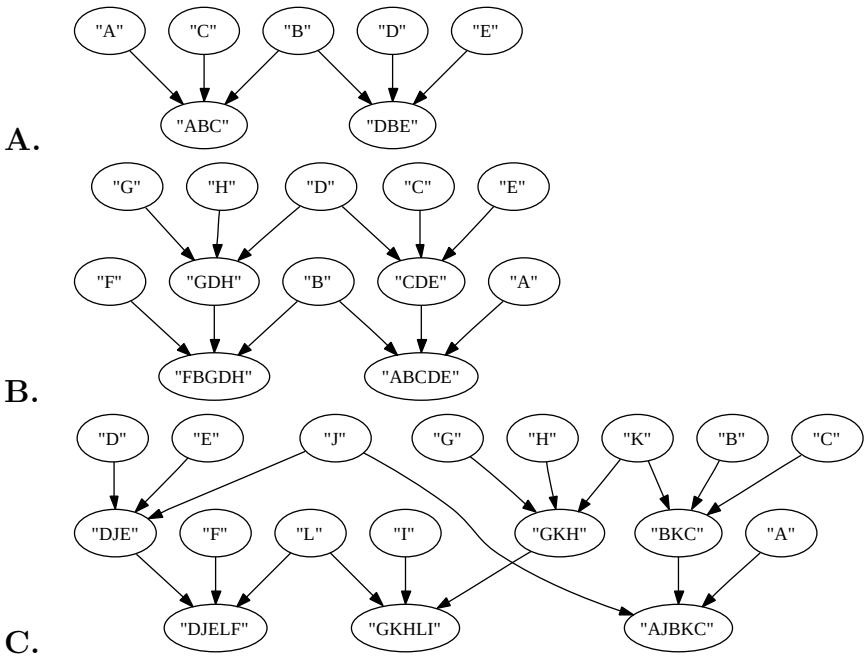

Figure 1: The simple perception lattices **A**, **B** and **C** are found by greedy compression search. The lattice in **A** is generated from the two strings "ABC" and "DBE", which contain only the character "B" as similar data. The lattice in **B** is generated from the two strings "ABCDE" and "FBGDH", which contain the characters "B" and "D" as similar data. The lattice in **C** is generated from the three strings "AJBKC", "DJELF" and "GKHLI", which share a triangular relationship in the characters "J", "L" and "K" as similar data.

Figure 1.A is a simple perception lattice that contains a single generative function,

$$\mathcal{G}_A(x, y) = x + \text{“B”} + y,$$

such that

$$\mathcal{G}_A(\text{“A”}, \text{“C”}) = \text{“ABC” and}$$
$$\mathcal{G}_A(\text{“D”}, \text{“E”}) = \text{“DBE”}.$$

Notice that the because the compression search used to create these perception lattices is greedy the generative functions that are found contain a maximum of 1 repeating phrase, which implies a maximum of 2 arguments. We have chosen a greedy compression search as an efficient proof of concept algorithm that operates over very large datasets quickly (e.g. 3000 stories each containing approximately 1000 characters in the OMICS story dataset and 300,000 sentences each containing approximately 50 characters in the ConceptNet dataset). Figure 1.B is another simple perception lattice that contains two generative functions,

$$\mathcal{G}_{B_0}(x, y) = x + \text{“B”} + y, \text{ and}$$
$$\mathcal{G}_{B_1}(x, y) = x + \text{“D”} + y,$$

such that

$$\mathcal{G}_{B_0}(\text{“A”}, \mathcal{G}_{B_1}(\text{“C”}, \text{“E”})) = \text{“ABCDE” and}$$
$$\mathcal{G}_{B_0}(\text{“F”}, \mathcal{G}_{B_1}(\text{“G”}, \text{“H”})) = \text{“FBGDH”}.$$

Similarly, for the simple perception lattice in Figure 1.C the generative functions are

$$\mathcal{G}_{C_0}(x, y) = x + \text{“J”} + y,$$
$$\mathcal{G}_{C_1}(x, y) = x + \text{“L”} + y, \text{ and}$$
$$\mathcal{G}_{C_2}(x, y) = x + \text{“K”} + y,$$

such that

$$\begin{aligned}
\mathcal{G}_{C_0}(\text{“A”}, \mathcal{G}_{C_2}(\text{“B”}, \text{“C”})) &= \text{“AJBKC”}, \\
\mathcal{G}_{C_0}(\text{“D”}, \mathcal{G}_{C_1}(\text{“E”}, \text{“F”})) &= \text{“DJELF”}, \text{ and} \\
\mathcal{G}_{C_1}(\mathcal{G}_{C_2}(\text{“G”}, \text{“H”}), \text{“I”}) &= \text{“GKHLI”}.
\end{aligned}$$

6 Efficient context-dependent mutual information calculation using generative functional grammars

LifeNet's most basic atomistic representational component is a symbol, and these symbols occur in temporal streams. A power spectrum clustering algorithm provides streams of symbols to LifeNet. Also, the story data also provides streams of textual symbols (individual Unicode characters). The greedy compression search results in a perception lattice that at it's leaves contains the atomistic posets of the experience. The leaves of the perception lattice (the parents in Figures 1 and 2) are where the lattice interfaces with the external world. For example, if we would like to use the perception lattice to find an explanation for how a given poset, x, in the world was generated, we would begin by checking which leaves in the perception lattice were used to functionally generate the poset, x. Those leaves, the few atomic posets of all experience, then form the basis for a search algorithm that flows down the generative functions of the perception lattice; note that this flow downwards in the perception lattice requires that the generative functions be reversible. Piaget (1947) emphasizes how this form of reversibility is a fundamental aspect of "successive adaptations of a sensori-motor and cognitive nature" of intelligence in the following excerpt:

> [Reversibility], as we shall see, is the essential property of the operations which characterize living logic in action. But we can see straight away that reversibility is the very criterion of equilibrium (as physicists have taught us). To define intelligence in terms of the progressive reversibility of the mobile structures which it forms is therefore to repeat in different words, that intelligence constitutes the state of equilibrium towards which tend all the successive adaptations of a sensori-motor and cognitive nature, as well as all assimilatory and accommodatory interactions between the organism and the environment.

LifeNet compares the symbol streams and calculates the similarity of these streams by using a form of mutual information. The calculation of mutual information measured in bits is as follows:

$$I(X,Y) = \sum_{y \in Y} \sum_{x \in X} P(x,y) \log_2 \frac{P(x,y)}{P(x)P(y)}. \tag{4}$$

Since our primary generative function is simply the string concatination function, we have implemented a very efficient means of calculating the mutual information between two arbitrary streams of data. Calculating extremely fast mutual information comparison functions that are experience dependent and context dependent is an extremely important and central task to building artificially intelligent computer systems that learn to perceive and act in the world.

The mutual information between two sets of posets, X and Y, depends on the modes that make up those pieces of data relative to a the perception lattice that has been generated from the universal dataset. The mutual information between sets of posets X and Y can be calculated by searching for which posets in the perception lattice exist within X and Y

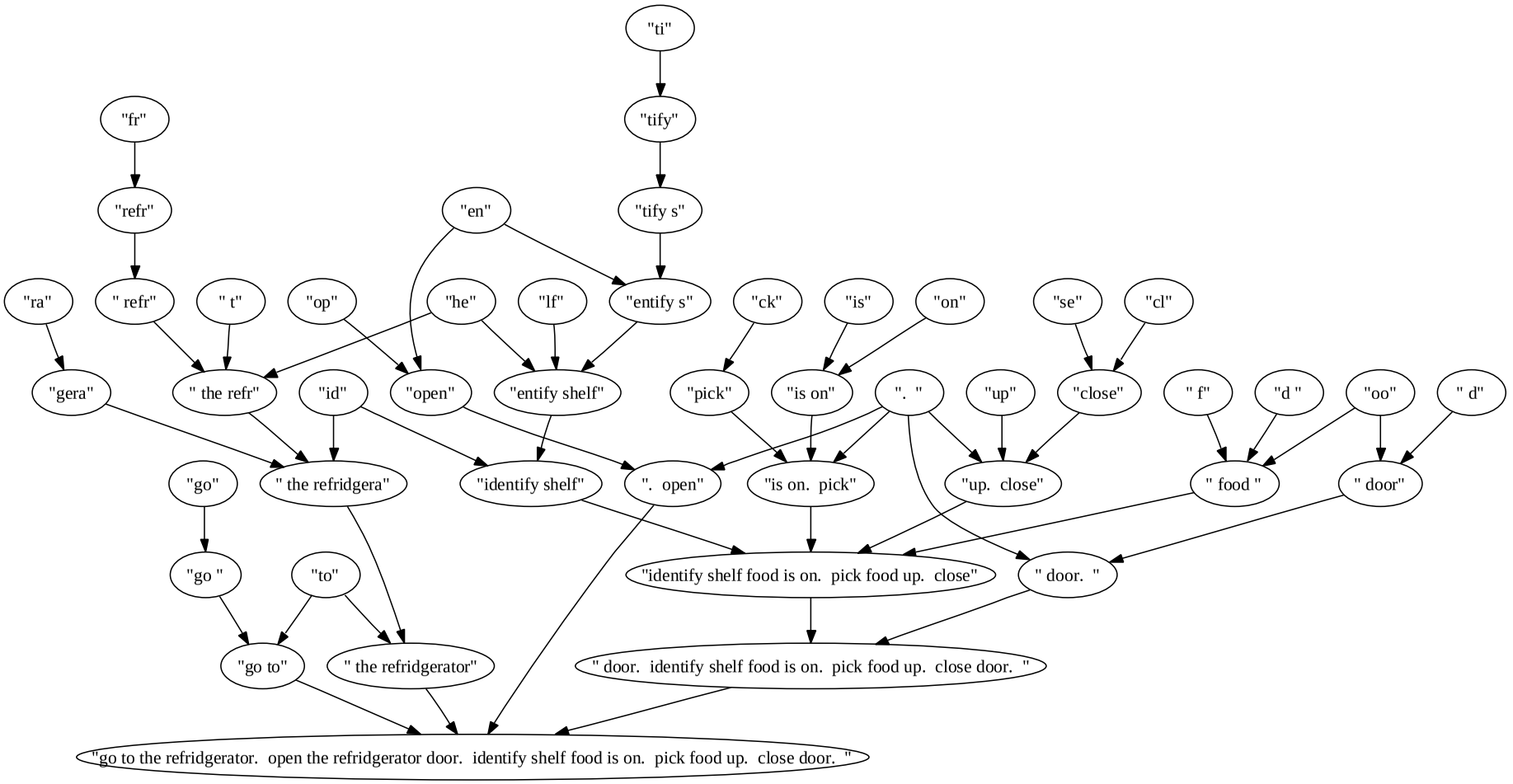

Figure 2: This example of a perception lattice was derived using just the information in this 5 step text story using a simplistic greedy compression search. Perception lattices like this one can form structures for very efficient data processing algorithms to be built. For example, a decision tree can be compiled that recognizes longest substring matches, or a belief network can be constructed to recognize hierarchical data patterns. Phrases of length one are omitted from this visualization.

separately. Let us refer to these posets that exist within the perception lattice and within X and Y as $\hat{X}$ and $\hat{Y}$ respectively. The calculation of Equation 4 directly with $|\hat{X}|\sim|\hat{Y}|\sim n$ takes time $O(n^2)$ and the calculation of $P(x,y)$ is non-trivial so this is actually a very conservative estimate; however, if we take advantage of Bayes' rule and make a few reasonable assumptions, such as treating the perception lattice as a probability network, over which an efficient belief propagation algorithm can be run, the time complexity is reduced considerably. By considering the approximation $P(\hat{Y}|\hat{X})$ instead in order to cache approximate values for each conditional probability below the time complexity can be reduced to $O(n \log n)$. The assumption that makes the mutual information calculation tractable for simple queries over large datasets with complex structure is explicitely

$$\forall x{\in}\hat{X}, y{\in}\hat{Y}\colon P(x|y) = P(x|\hat{Y}), \tag{5}$$

which is a reasonable assumption under the condition that the elements of the set of posets, $\hat{Y}$, are highly dependent variables, which will often be the case when comparing one set of dependent poset variables, $\hat{Y}$, against a second test set of poset variables, $\hat{X}$, in order to calculate the mutual information between these internally dependent clusters. Also, addressing this assumption becomes a moot point if the set, $\hat{Y}$, is a singleton set. Bayes' rule states

$$P(x,y) = P(x|y)P(y). \tag{6}$$

Substituting into Equation 4 and making the simplifying assumption that allows us to use a single application of the belief propagation algorithm, Equation 5, gives

$$I(\hat{X},\hat{Y}) = \sum_{y\in\hat{Y}}\sum_{x\in\hat{X}} P(x|\hat{Y})P(y)\log_2\frac{P(x|\hat{Y})P(y)}{P(x)P(y)} \tag{7}$$

$$= \sum_{y\in\hat{Y}}\sum_{x\in\hat{X}} P(x|\hat{Y})P(y)\log_2\frac{P(x|\hat{Y})}{P(x)} \tag{8}$$

$$= \sum_{y\in\hat{Y}} P(y)\sum_{x\in\hat{X}} P(x|\hat{Y})\log_2\frac{P(x|\hat{Y})}{P(x)}. \tag{9}$$

We notice that the factor $log_2\frac{P(x|\hat{Y})}{P(x)}$ tends to zero as x is independent of $\hat{Y}$, so if we consider the Markov blanket, $M_{y|\hat{X}}$, for each $y{\in}\hat{Y}$ with respect to $\hat{X}$ we will avoid these summations of zero. This gives

$$I(\hat{X},\hat{Y}) = \sum_{y\in\hat{Y}} P(y)\sum_{x\in M_{y|\hat{X}}} P(x|\hat{Y})\log_2\frac{P(x|\hat{Y})}{P(x)}. \tag{10}$$

Also, if we assume for the calculation of mutual information that the generative functions within the perception lattice are fully representative and exact matches, then we will have all children, C_x, of nodes, x, within the lattice to be a pure implication relationship, $\forall y{\in}C_x\colon y{\rightarrow}x$. If we consider specifically the situations where $y{\in}C_x$, Equation 10 expands to

$$I(\hat{X},\hat{Y}) = \sum_{y\in\hat{Y}} P(y)\left[\sum_{x\in M_{y|\hat{X}}\cap C_x} -\log_2 P(x) + \sum_{x\in M_{y|\hat{X}}\cap\overline{C_x}} P(x|\hat{Y})\log_2\frac{P(x|\hat{Y})}{P(x)}\right]. \tag{11}$$

We will refer to the second summation as the internal complexity, $\mathcal{C}_I(y)$, of node y.

$$\mathcal{C}_I(y) = \sum_{x \in M_{y|\hat{X}} \cap C_x} \log_2 P(x) \qquad \textbf{internal complexity} \tag{12}$$

Because internal complexity can be cached for each node, this reduces the limit of the ammortized calculation of mutual information to the following equation:

$$I(\hat{X}, \hat{Y}) = \sum_{y \in \hat{Y}} P(y) \left[-\mathcal{C}_I(y) + \sum_{x \in M_{y|\hat{X}} \cap \overline{C_x}} P(x|\hat{Y}) \log_2 \frac{P(x|\hat{Y})}{P(x)} \right] \tag{13}$$

$$= -\sum_{y \in \hat{Y}} P(y)\mathcal{C}_I(y) + \sum_{y \in \hat{Y}} \sum_{x \in M_{y|\hat{X}} \cap \overline{C_x}} P(x|\hat{Y}) \log_2 \frac{P(x|\hat{Y})}{P(x)}. \tag{14}$$

For reference, we can refer to $P(y)C_I(y)$ as the internal information, $I_I(y)$, of a node, y:

$$I_I(y) = P(y)C_I(y) \qquad \textbf{internal information} \tag{15}$$

It makes intuitive sense that the internal information within a node would be a negative quantity in the calculation of mutual information between that node and other nodes with different structures of functional generation. Therefor, to calculate the mutual information between two sets of posets, X and Y, the following optimized algorithm may be used:

$$I(\hat{X}, \hat{Y}) = -\sum_{y \in \hat{Y}} I_I(y) + \sum_{y \in \hat{Y}} \sum_{x \in M_{y|\hat{X}} \cap \overline{C_x}} P(x|\hat{Y}) \log_2 \frac{P(x|\hat{Y})}{P(x)}. \tag{16}$$

Because we have restricted the poset nodes in the perception lattice that we consider due to the Markov blanket restriction, we now define the running time complexity of the general optimized version of the mutual information calculation to be in terms of

$$|\hat{Y}| \sim n \text{ and} \tag{17}$$

$$|M_{y|\hat{X}} \cap \overline{C_x}| \sim \log n. \tag{18}$$

So, after an $O(n \log n)$ belief propagation algorithm has been run with respect to an internally dependent set of posets $\hat{Y}$, Equation 16 can be calculated for an arbitrary set of posets $\hat{X}$ within the perception lattice in $O(n \log n)$ time.

The assumption that the perception lattice is representative of the universal dataset is a very strong assumption, which can be weakened if we instead assume a finite horizon of unknown data, but we leave this calculation for future work. We expect Equation 16 to be a helpful algorithm for recognizing small subposets within large datasets that are used as different arguments within the same sets of generative functions. In text processing, this may provide a method for finding strings of language that are used in synonymous generative constructions. Within streams of sensor data posets within a perception lattice that have mutual information may represent the sensation of events that may be superficially different, but may share the same contextual causal relationships with surrounding sensor events (e.g. a metal door slamming and a glass door quietly clicking shut may both be preceded by and followed by footsteps).

7 Toward self-reflection by blurring the objective duality between algorithms and data

We use the term "perception lattice" very similarly to the *structure lattice* in Richards et al. (1994) except that instead of storing elemental preference relations in a separate lattice, which relies on the objective distinction being placed between the two types of data, *feature* and *percept* (similar to Kant's objective duality of *neumenon* and *phenomenon* respectively), our percept preferences are inherently part of the same structure as the percepts themselves. To be clear, we have not avoided the objective dualistic distinction between the observer and the world but by using the perception lattice we have merely chosen to place the distinction between the algebra of computation and the perceptual data that is to be explained by itself as structured by the assumed algebra. In this sense, our perception lattice explicitely includes a model of computation. The *Harvard architecture* of computation makes the same dualistic distinction between a computational *algorithm* and the *data*, over which this algorithm operates. Similarly, in the philosophical literature, Heidegger (1962) makes the distinction between *logos* (letting something be seen) and *phenomenon* (that which shows itself in itself). Heidegger also introduces the powerful idea of self-reflection when he introduces the idea of considering *logos* as *phenomenon*, which in the present analogy maps to considering *algorithms* as *data*, bypassing the assumptions of the traditional objectively dualistic Harvard architecture. Some programming languages, such as *Lisp* and *Python*, allow algorithms that dynamically process algorithms as data, which blur the objective duality between algorithms and data, making a rich and underexplored area of self-reflective computational research. We feel that the perception lattice lends itself to future research in self-reflective computational perception moreso than the *structure lattice*, which places the objective duality between two different types of data. For two proof-of-concept examples of social robots with multiple layers of self-reflective debugging algorithms that are processed as data for perception and action please see Singh (2005).

8 Toward abstraction using explanation-based similarity

Using generative functions as modes of perception that provide explanations for what it means for data to exist provides a means for considering how similar two pieces of data are based on whether or not their "means of existance" are similar or, in other words, how the processes that generated the data are similar. Explanation representations were used in a planning environment (Bergmann et al. 1993) in order to judge similarity in a multiple-layered graph structure that makes the distinction between *rule-nodes* and *fact-nodes* and takes advantage of *fact abstraction* and *rule abstraction* to map between different layers. The data nodes within the perception lattice could be considered *fact-nodes*, while the generative functions that form the edges of the perception lattice could be considered *rule-nodes*. Considering this mapping of terminology, perhaps a similar method of *fact abstraction* could be employed where multiple data nodes could be mapped to an abstract data node, and similarly, a sublattice of generative functions and data nodes can be compiled to abstract generative functions and data nodes.

9 Appendix

9.1 ConceptNet Semantic Relationships

1. *conceptually-related-to*
2. *superthematic-k-line*
3. *thematic-k-line*
4. *capable-of*
5. *is-a*
6. *effect-of*
7. *location-of*
8. *capable-of-receiving-action*
9. *motivation-of*
10. *desire-of*
11. *property-of*
12. *used-for*
13. *last-subevent-of*
14. *part-of*
15. *subevent-of*
16. *defined-as*
17. *desirous-effect-of*
18. *made-of*
19. *prerequisite-event-of*
20. *first-subevent-of*

Glossary

generative function	Computational process that takes a set of arguments and returns a set of data derived from those arguments. As computational processes, generative functions assume an algebra of data processing computation. This generative functions in this paper assume an algebra of stacked poset concatenation functions, which are easily implemented on classical digital computers, but in general, generative functions could assume biological neural networks or quantum computers as other algebras of data generative computation, 2
greedy compression search	Search algorithm that begins with a list of uncompressed data, L. For all of the data in L the largest contiguous repetetive section, x, of data is found. Every datum in L containing x is removed from L and split into smaller non-contiguous pieces that do not contain x. These smaller non-contiguous pieces are appended to L, and the process of removing redundant sections of data continues until no such sections exist in L, at which point L will contain the leaves of the perception lattice structure, 4
mode of perception	Consistent relationship between elements within a subset of data. This consistent relationship defines a regularity that serves to organize a larger set of data. A mode of perception can act as a manifold in the space of percepts such that percepts lie on a specific manifold or they lie off of that manifold. Given that a set of data points lie on a mode of perception, these data points contain percepts that vary along the mode. The specific percepts that define these data points are implicitly constrained by this manifold or mode, 1

perception lattice	Lattice data structure that represents the generative functional explanation for a given set of data. This lattice structure is used for many algorithmic operations over perceived data. For example, a perception lattice could be used to find the most likely top-down explanation of bottom-up perceptions, or alternatively, a perception lattice could be used for the projection of low-level details given high-level evidence, 3

References

Bergmann, R., Pews, G. & Wilke, W. (1993), 'Explanation-based similarity: A unifying approach for integrating domain knowledge into case-based reasoning for diagnosis and planning tasks'.

Heidegger, M. (1962), *Being and Time*, Harper San Francisco.

Liu, H. & Singh, P. (2004), 'Conceptnet: A practical commonsense reasoning toolkit', *B.T. Technology Journal* **22**.

Piaget, J. (1947), *The Psychology of Intelligence*, Littlefield, Adams and Co.

Richards, W., Jepson, A. & Feldman, J. (1994), *Perception as Bayesian Inference: Priors, preferences, and categorical percepts*, chapter 3.

Singh, P. (2005), EM-ONE: An Architecture for Reflective Commonsense Thinking, PhD thesis, Massachusetts Institute of Technology.

Singh, P. & Williams, W. (2003), 'Lifenet: a propositional model of ordinary human activity', *Proceedings of the Workshop on Distributed and Collaborative Knowledge Capture* **DC-KCAP**.