Integrating Deep Learning Based Perception with Probabilistic Logic via Frequent Pattern Mining

Ben Goertzel[1,2], Ted Sanders, and Jade O'Neill[2]

[1] Novamente LLC
[2] School of Design, Hong Kong Polytechnic University

Abstract. The bridging of the gap between 1) subsymbolic pattern recognition and learning algorithms and 2) symbolic reasoning algorithms, has been a major issue for AI since the early days of the field. One class of approaches involves integrating subsymbolic and symbolic systems, but this raises the question of how to effectively translate between the very different languages involved. In the approach described here, a frequent subtree mining algorithm is used to identify recurrent patterns in the state of a hierarchical deep learning system (DeSTIN) that is exposed to visual stimuli. The relationships between state-subtrees and percepts are then input to a probabilistic logic system (OpenCog's Probabilistic Logic Networks), which conducts uncertain inferences using them as axioms. The core conceptual idea is to use patterns in the states inferred by a perceptual hierarchy, as inputs to an uncertain logic system. Simple illustrative examples are presented based on the presentation of images of typed letters to DeSTIN. This work forms a component of a larger project to integrate perceptual, motoric and cognitive processing within the integrative OpenCog cognitive architecture.

1 Introduction

The bridging of the symbolic and subsymbolic aspects of intelligence has been a challenge for the AI field from the beginning. One promising approach to the proble m is the hybridization of symbolic and subsymbolic components within an integrative architecture. With this in mind, in a paper presented at last year's AGI conference [1], a novel approach to integrating symbolic and subsymbolic AI was outlined, centered on the use of pattern mining to bridge the gap between a subsymbolic, hierarchical deep learning system and a symbolic logical reasoning system. As an instantiation of this general concept, it was suggested that it might be fruitful to apply a frequent subgraph mining algorithm to the set of states of the DeSTIN deep learning algorithm, and then use the mined subgraphs as inputs to a system such as the OpenCog integrative cognitive architecture which incorporates logical inference. Since that time, the details of this latter suggestion have been worked out, and implemented in software. Here we review some of the particulars of this integration, and give some very simple illustrative examples obtained using the relevant software. While simple, the particulars given here serve as a practical example of what it really means to use pattern mining as a bridge between subsymbolic and symbolic components of a cognitive architecture. We believe this constitutes significant conceptual and practical progress toward the resolution of one of the deep nagging issues at the core of the AI field.

1.1 DeSTIN and OpenCog

The work described here involves the integration of two separate AI systems, both currently implemented in open-source software:

– **OpenCog**, an integrative architecture for AGI [2] [3], which is centered on a "weighted, labeled hypergraph" knowledge representation called the Atomspace, and features a number of different, sophisticated cognitive algorithms acting on the Atomspace. Some of these cognitive algorithms are heavily symbolic in focus (e.g. a probabilistic logic engine); others are more subsymbolic in nature (e.g. a neural net like system for allocating attention and assigning credit). However, OpenCog in its current form cannot deal with high-dimensional perceptual input, nor with detailed real-time control of complex actuators. OpenCog is now being used to control intelligent characters in an experimental virtual world, where the perceptual inputs are the 3D coordinate locations of objects or small blocks; and the actions are movement commands like "step forward", "turn head to the right." For OpenCog code and documentation see [1].
– **DeSTIN** [4],[5], a deep learning system consisting of a hierarchy of processing nodes, in which the nodes on higher levels correspond to larger regions of space-time, and each node carries out prediction regarding events in the space-time region to which it corresponds. Feedback and feedforward dynamics between nodes combine with the predictive activity within nodes, to create a complex nonlinear dynamical system whose state self-organizes to reflect the state of the world being perceived. The core concepts of DeSTIN are similar to those of Jeff Hawkins' Numenta system [6] [7], Dileep George's work [2], and work by Mohamad Tarifi [8], Bundzel and Hashimoto [9], and others. In the terminology introduced in [10], DeSTIN is an example of a Compositional Spatiotemporal Deep Learning System, or CSDLN. However, compared to other CSDLNs, the specifics of DeSTIN's dynamics have been designed in what we consider a particularly powerful way, and the system has shown good results on small-scale test problems [11]. So far DeSTIN has been utilized only for vision processing, but a similar proprietary system has been used for auditory data as well; and DeSTIN was designed to work together with an accompanying action hierarchy. For DeSTIN code see [3].

These two systems were not originally designed to work together, but we will describe a method for achieving their tight integration. For space reasons, we

[1] `http://opencog.org`
[2] `http://vicariousinc.com`
[3] `https://github.com/tpsjr7/destin_ted_temp`

will not, however, describe either of the systems in any detail. The reader is referred to the above references, or for a quick review, to the online review of DeSTIN/OpenCog integration available at the URL [4].

One particular aspect of OpenCog is especially relevant here. OpenCog's primary tool for handling declarative knowledge is an uncertain inference framework called Probabilistic Logic Networks (PLN). The complexities of PLN are the topic of a lengthy technical monograph [12]. A key point to note is that, as a logic, PLN is broadly integrative: it combines certain term logic rules with more standard predicate logic rules, and utilizes both fuzzy truth values and a variant of imprecise probabilities called *indefinite probabilities*. PLN mathematics tells how these uncertain truth values propagate through its logic rules, so that uncertain premises give rise to conclusions with reasonably accurately estimated uncertainty values. Also, PLN can be used in either forward or backward chaining mode, or using more innovative control strategies, such as those reliant on integration with other OpenCog components. The PLN inferences described below will involve mainly term logic Inheritance and Similarity relationships, and will utilize the OpenCog notation described online at [5].

2 Using Subtree Mining to Bridge the Gap between DeSTIN and PLN

The core technical idea explored in the present paper is to apply Yun Chi's Frequent Subtree Mining software [13] [6] to mine frequent patterns from a data-store of trees representing DeSTIN states. In this application, each frequent subtree represents a common visual pattern. This approach may also be extended to include additional quality metrics besides frequency, e.g. interaction information [14] which lets one measure how surprising a subtree is.

First we briefly describe the overall architecture into which the use of frequent subtree mining to bridge DeSTIN and PLN is intended to fit. This architecture is not yet fully implemented, but is a straightforward extension of the current OpenCog architecture for processing data from game worlds [15], and is scheduled for implementation later in 2013 in the course of a funded project involving the use of DeSTIN and OpenCog for humanoid robot control. The architecture is visually illustrated online at `http://wp.goertzel.org/?page_id=518`. The components intervening between DeSTIN and OpenCog, in this architecture, are:

- **DeSTIN State DB:** Stores all DeSTIN states the system has experienced, indexed by time of occurrence
- **Frequent Subtree Miner:** Recognizes frequent subtrees in the database of DeSTIN states, and can also filter the frequent subtrees by other criteria such as information-theoretic surprisingness. These subtrees may sometimes span multiple time points.

[4] `http://wp.goertzel.org/?p=404`
[5] `http://wiki.opencog.org/w/OpenCogPrime:AtomNotation`
[6] available for download at `http://www.nec-labs.com/~ychi/publication/software.html`

- **Frequent Subtree Recognizer:** Scans DeSTIN output, and recognizes frequent subtrees therein. These subtrees are the high level visual patterns that make their way from DeSTIN to OpenCog.
- **Perception Collector:** Linearly normalizes the spatial coordinates associated with its input subtrees, to compensate for movement of the camera. Filters out perceptions that didnt change recently (e.g. a static white wall), so that only new visual information is passed along to OpenCog. Translates the subtrees into Scheme files representing OpenCog logical Atoms.
- **Experience DB:** Stores all the normalized subtrees that have actually made their way into OpenCog
- **Semantic Feedback:** Allows the semantic associations OpenCog makes to a subtree, to be fed back into DeSTIN as additional inputs to the nodes involved in the subtree. This allows perception to make use of cognitive information.

2.1 The Importance of Semantic Feedback

One aspect of the above architecture not yet implemented, but worthy of note, is semantic feedback. Without the semantic feedback, we expect to be able to emulate human object and event recognition insofar as they are done by the human brain in a span of less than 500ms or so. In this time frame, the brain cannot do much sophisticated cognitive feedback, and processes perceptual data in an essentially feedforward manner. On the other hand, properly tuned semantic feedback along with appropriate symbolic reasoning in OpenCog, may allow us to emulate human object and event recognition as the human brain does it when it has more time available, and can use its cognitive understanding to guide vision processing.

A simple example of this sort of symbolic reasoning is analogical inference. Given a visual scene, OpenCog can reason about what the robot has seen in similar situations before, where its notion of similarity draws not only on visual cues but on other contextual information: what time it is, what else is in the room (even if not currently visible), who has been seen in the room recently, etc.

For instance, recognizing familiar objects that are largely occluded and in dim light, may be something requiring semantic feedback, and not achievable via the feedforward dynamics alone. This can be tested in a robot vision context via showing the robot various objects and events in various conditions of lighting and occlusion, and observing its capability at recognizing the objects and events with and without semantic feedback, in each of the conditions.

If the robot sees an occluded object in a half-dark area on a desk, and it knows that a woman was recently sitting at that desk and then got up and left the room, its symbolic analogical inference may make it more likely to conclude that the object is a purse. Without this symbolic inference, it might not be able to recognize the object as a purse based on bottom-up visual clues alone.

3 Some Simple Experiments with Letters

To illustrate the above ideas in an elementary context, we now present results of an experiment using DeSTIN, subtree mining and PLN together to recognize patterns among a handful of black and white images comprising simple letter-forms. This is a "toy" example, but exemplifies the key processes reviewed above. During the next year we will be working on deploying these same processes in the context of robot vision.

3.1 Mining Subtrees from DeSTIN States Induced via Observing Letterforms

Figure 1 shows the 7 input images utilized; Figure 2 shows the centroids found on each of the layers of DeSTIN (note that translation invariance was enabled for these experiments); and Figure 3 shows the most frequent subtrees recognized among the DeSTIN states induced by observing the 7 input images.

The centroid images shown in Figure 2 were generated as follows. For the bottom layer, centroids were directly represented as 4x4 grayscale images (ignoring the previous and parent belief sections of the centroid). For higher-level centroids, we proceeded as follows:

– Divide the centroid into 4 sub-arrays. An image is generated for each sub-array by treating the elements of the sub-array as weights in a weighted sum of the child centroid images. This weighted sum is used superpose / blend the child images into 1 image.
– Then these 4 sub-array images are combined in a square to create the whole centroid image.
– Repeat the process recursively, till one reaches the top level.

The weighted averaging used a p-power approach, i.e. replacing each weight w_i with $w_i^p/(w_1^p + ... + w_n^p)$ for a given exponent $p > 0$. The parameter p toggles how much attention is paid to nearby versus distant centroids. In generating Figure 2 we used $p = 4$.

The relation between subtrees and input images, in this example, was directly given via the subtree miner as:

```
tree #0 matches input image: 4 6
tree #1 matches input image: 1 2
tree #2 matches input image: 3 5
tree #3 matches input image: 0 1 2 4
tree #4 matches input image: 3 5
tree #5 matches input image: 4 5
tree #6 matches input image: 1 2
tree #7 matches input image: 0 3 4
```

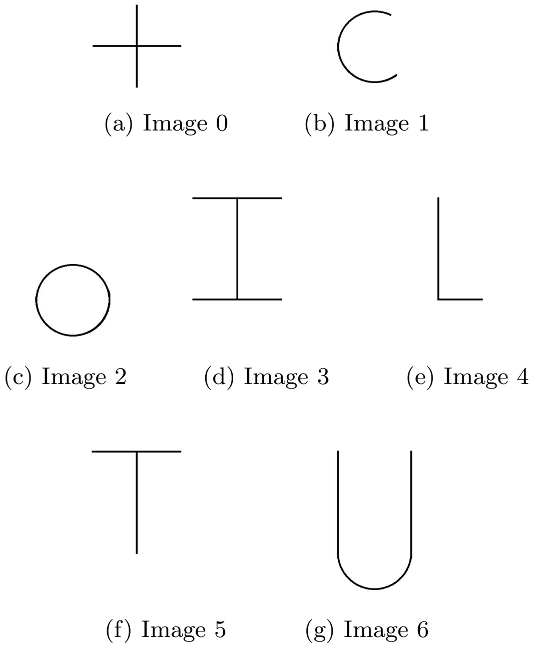


Fig. 1. Simple input images fed to DeSTIN for the experiment reported here

3.2 Mining Subtrees from DeSTIN States Induced via Observing Letterforms

The subtree-image relationships listed above may be most directly expressed in PLN syntax/semantics via

```
Evaluation contained_in (Tree0 Image4)
Evaluation contained_in (Tree0 Image6)
Evaluation contained_in (Tree1 Image1)
Evaluation contained_in (Tree1 Image2)
Evaluation contained_in (Tree2 Image3)
Evaluation contained_in (Tree2 Image5)
Evaluation contained_in (Tree3 Image0)
Evaluation contained_in (Tree3 Image1)
Evaluation contained_in (Tree3 Image2)
Evaluation contained_in (Tree3 Image4)
Evaluation contained_in (Tree4 Image3)
Evaluation contained_in (Tree4 Image5)
Evaluation contained_in (Tree5 Image4)
Evaluation contained_in (Tree5 Image5)
Evaluation contained_in (Tree6 Image1)
Evaluation contained_in (Tree6 Image2)
Evaluation contained_in (Tree7 Image0)
```

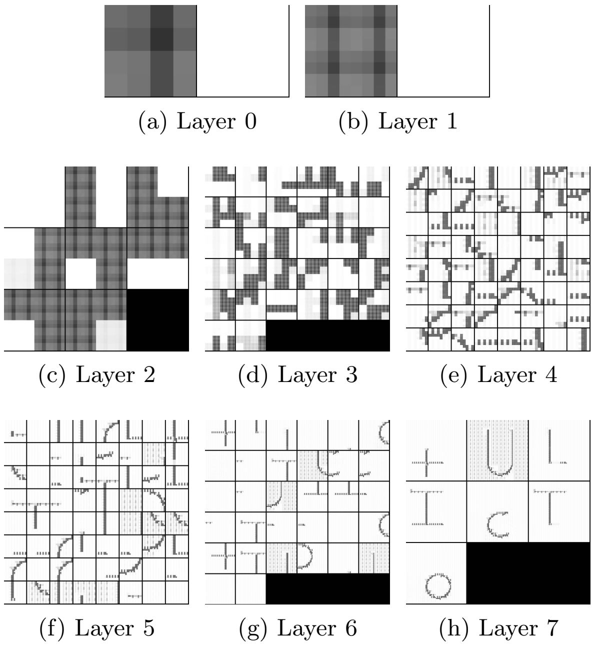


Fig. 2. Example visualization of the centroids on the 7 layers of the DeSTIN network. Each picture shows multiple centroids at the corresponding level. Higher level centroids are visualized as p'th power averages of lower level centroids, with p=4.

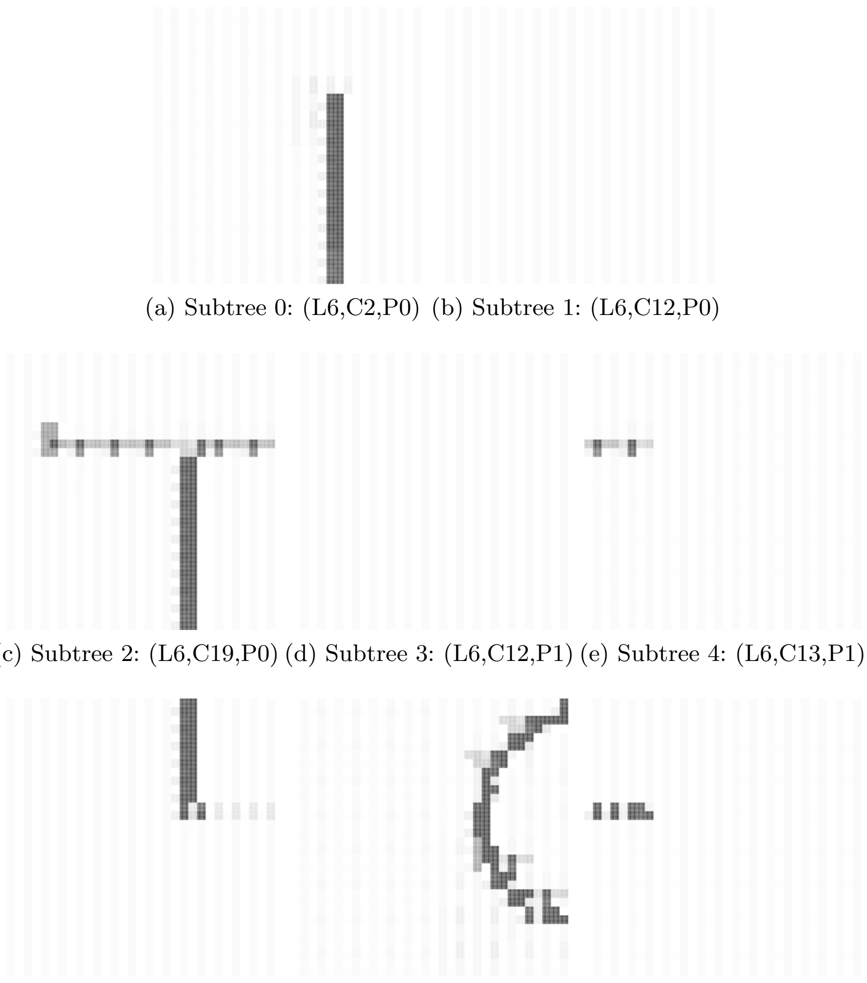
```
Evaluation contained_in (Tree7 Image3)
Evaluation contained_in (Tree7 Image4)
```

But while this is a perfectly natural way to import such relationships into OpenCog, it is not necessarily the most convenient form for PLN to use to manipulate them. For some useful inference chains, it is most convenient for PLN to translate these into the more concise form

```
Inheritance Image4 hasTree0
Inheritance Image6 hasTree0
...
Inheritance Image3 hasTree7
Inheritance Image4 hasTree7
```

PLN performs the translation from Evaluation into Inheritance form via the inference steps

```
Evaluation contains (Tree0 Image4)
==> \\ definition of SatisfyingSet
Member Image4 (SatisfyingSet (Evaluation contains(Tree0 *)) )
== \\ definition of hasTree0
Member Image4 hasTree0
==> \\ M2I, Member to Inheritance inference
Inheritance Image4 hasTree0
```



(a) Subtree 0: (L6,C2,P0) (b) Subtree 1: (L6,C12,P0)

(c) Subtree 2: (L6,C19,P0) (d) Subtree 3: (L6,C12,P1) (e) Subtree 4: (L6,C13,P1)

(f) Subtree 5: (L6,C1,P2) (g) Subtree 6: (L6,C5,P2) (h) Subtree 7: (L6,C20,P3)

Fig. 3. Example subtrees extracted from the set of DeSTIN states corresponding to the input images given above. Each subtree is associated with a triple (level, centroid, position). The position is one of the four level n squares making up a level $n-1$ centroid. In this simple exmaple, all these frequent subtrees happen to be from Level 6, but this is not generally the case for more complex images. some of the centroids look like whitespace, but this is because a common region of whitespace was recognized among multiple input images.

Finally, given the Inheritance relations listed above, PLN can draw some simple conclusions fairly directly, such as:

```
Similarity Image1 Image2 <1, .375>
Similarity Image3 Image5 <.5, .444>
```

The PLN truth values above are given in the form " ¡strength, confidence¿ ", where strength is in this case effectively a probability, and confidence represents a scaling into the interval $[0, 1]$ of the amount of evidence on which that strength value is based. The confidence is calculated using a "personality parameter" of $k = 5$ (k may vary between 1 and ∞, with higher numbers indicating less value attached to each individual piece of evidence. For example the truth value strength of 1 attached to "Similarity Image1 Image2" indicates that according to the evidence provided by these subtrees (and ignoring all other evidence), Image1 and Image2 are the same. Of course they are not the same – one is a C and another is an O – and once more evidence is given to PLN, it will decrease the strength value of this SimilarityLink. The confidence value of .375 indicates that PLN is not very certain of the sameness of these two letters.

What conclusion can we draw from this toy example, practically speaking? The conclusions drawn by the PLN system are not useful in this case – PLN thinks C and O are the same, as a provisional hypothesis based on this data. But this is not because DeSTIN and PLN are stupid. Rather, it's because they have not been fed enough data. The hypothesis that a C is an occluded O is actually reasonably intelligent. If we fed these same systems many more pictures, then the subtree miner would recognize many more frequent subtrees in the larger corpus of DeSTIN states, and PLN would have a lot more information to go on, and would draw more conmmonsensically clever conclusions. We will explore this in our future work.

We present this toy example not as a useful practical achievement, but rather as a very simple illustration of the process via which subsymbolic knowledge (as in the states of the DeSTIN deep learning architecture) can be mapped into abstract logical knowledge, which can then be reasoned on via a probabilistic logical reasoning engine (such as PLN). We believe that the same process illustrated so simplistically in this example, will also generalize to more realistic and interesting examples, involving more complex images and inferences. The integration of DeSTIN and OpenCog described here is being pursued in the context of a project aimed at the creation of a humanoid robot capable of perceiving interpreting and acting in its environment with a high level of general intelligence.

References

1. Goertzel, B.: Perception processing for general intelligence: Bridging the symbolic/Subsymbolic gap. In: Bach, J., Goertzel, B., Iklé, M. (eds.) AGI 2012. LNCS, vol. 7716, pp. 79–88. Springer, Heidelberg (2012)
2. Goertzel, B., et al.: Opencogbot: An integrative architecture for embodied agi. In: Proc. of ICAI 2010, Beijing (2010)
3. Goertzel, B., Pitt, J., Wigmore, J., Geisweiller, N., Cai, Z., Lian, R., Huang, D., Yu, G.: Cognitive synergy between procedural and declarative learning in the control of animated and robotic agents using the opencogprime agi architecture. In: Proceedings of AAAI 2011 (2011)

4. Arel, I., Rose, D., Karnowski, T.: A deep learning architecture comprising homogeneous cortical circuits for scalable spatiotemporal pattern inference. In: NIPS 2009 Workshop on Deep Learning for Speech Recognition and Related Applications (2009)
5. Arel, I., Rose, D., Coop, R.: Destin: A scalable deep learning architecture with application to high-dimensional robust pattern recognition. In: Proc. AAAI Workshop on Biologically Inspired Cognitive Architectures (2009)
6. Hawkins, J., Blakeslee, S.: On Intelligence. Brown Walker (2006)
7. George, D., Hawkins, J.: Towards a mathematical theory of cortical microcircuits. PLoS Comput. Biol. 5 (2009)
8. Tarifi, M., Sitharam, M., Ho, J.: Learning hierarchical sparse representations using iterative dictionary learning and dimension reduction. In: Proc. of BICA 2011 (2011)
9. Bundzel, Hashimoto: Object identification in dynamic images based on the memory-prediction theory of brain function. Journal of Intelligent Learning Systems and Applications 2(4) (2010)
10. Goertzel, B.: Integrating a compositional spatiotemporal deep learning network with symbolic representation/reasoning within an integrative cognitive architecture via an intermediary semantic network. In: Proceedings of AAAI Symposium on Cognitive Systems (2011)
11. Karnowski, T., Arel, I., Rose, D.: Deep spatiotemporal feature learning with application to image classification. In: The 9th International Conference on Machine Learning and Applications, ICMLA 2010 (2010)
12. Goertzel, B., Ikle, M., Goertzel, I., Heljakka, A.: Probabilistic Logic Networks. Springer (2008)
13. Chi, Y., Xia, Y., Yang, Y., Muntz, R.R.: Mining closed and maximal frequent subtrees from databases of labeled rooted trees. IEEE Trans. Knowledge and Data Engineering (2005)
14. Bell, A.J.: The co-information lattice. In: Proc. ICA 2003 (2003)
15. Goertzel, B., Pennachin, C., et al.: An integrative methodology for teaching embodied non-linguistic agents, applied to virtual animals in second life. In: Proc. of the First Conf. on AGI. IOS Press (2008)