Perception Processing for General Intelligence, Part I: Representationally Transparent Deep Learning

Ben Goertzel

July 16, 2012

Abstract

Bridging the gap between symbolic and subsymbolic representations is a – perhaps *the* – key obstacle along the path from the present state of AI technology to human-level artificial general intelligence. The companion paper (Part II) describes a novel approach to achieiving this bridging via incorporation of a subsymbolic system and a symbolic system into a integrative cognitive architecture. This paper (Part I) describes a key ingredient of this hybridization, which is also interesting in its own right: a modification of the DeSTIN deep learning based perception system, in a way that renders it "representationally transparent," meaning that when different parts of the deep learning network represent similar patterns (with similarity defined via affine transformations), this is immediately apparent via inspection of the state of the network. With DeSTIN as a case in point, it is argued that representational transparency is a desirable property for deep learning systems to have, for integration with other AI components as well as for other reasons, and that this can viably be achieved without substantially sacrificing their intelligence.

1 Introduction

Human beings carry out both *symbolic* (e.g. logical and linguistic) and *subsymbolic* (e.g. sense-perception and motor-control optimization) processing, as integral parts of their general intelligence. The practical and conceptual gap between these two types of processing has been a hot topic in the AI field for decades, and there is no commonly accepted solution to the problem. Part II of this paper [Goeon] discusses the symbolic/subsymbolic dichotomy in depth, and proposes a general solution, based on very tight integration of a subsymbolic perception/action system (the DeSTIN deep learning system [ARK09],[ARC09]) with a cognition system possessing a rich symbolic aspect (the OpenCog cognitive architecture [Goe09]). This paper carries out some critical preparatory work regarding the subsymbolic component of this integrated approach – showing how it is possible to modify DeSTIN, a deep learning based machine perception system, to display strong properties of "representational transparency," which render its internal states much more comprehensible to symbolic algorithms such as those within OpenCog.

While these modifications to DeSTIN have arisen in the context of the project of hybridizing DeSTIN and OpenCog, we believe they are also of more general interest. Similar modifications could be applied to other deep learning systems, with similar implications. We suggest that, in general, modifying deep learning systems to possess greater representational transparency will be a valuable strategy, enabling cleaner integration of deep learning systems with other AI components and more incisive analysis of the internal dynamics of deep learning systems, and potentially more efficient deep learning as well.

DeSTIN is a general framework for machine perception, and in Part II we discuss its extension to handle action as well. Here, however, we focus on the domain in which DeSTIN has currently been developed and applied: machine vision. What we describe here is a way to make a "Uniform DeSTIN", in which the internal representation of perceived visual forms is independent of affine transformations (translation, scaling, rotation and shear). This "representational transparency" means that, when Uniform DeSTIN perceives a pattern: no matter how that pattern is shifted or linearly transformed, the way Uniform DeSTIN represents that pattern internally is going to be basically the same. This makes it easy to look at a collection of DeSTIN states, obtained by exposing a DeSTIN perception network to the world at different points in time, and see the commonalities in what they are perceiving and how they are interpreting it. By contrast, in the original

version of DeSTIN (here called "classic DeSTIN"), it may take significant effort to connect the internal representation of a visual pattern and the representation of its translated or linearly transformed versions. The uniformity of Uniform DeSTIN makes it easier for humans to inspect DeSTIN's state and understand what's going on, and also (more to the point) makes it easier for other AI components to recognize patterns in sets of DeSTIN states. The latter fact is critical for the DeSTIN/OpenCog integration described in the companion paper (Part II), which rests heavily on the ability of OpenCog to recognize such patterns within reasonable computational effort.

2 DeSTIN

"Deep learning" systems – networks of simple learning elements arranged in multiple layers, with feedforward and feedback dynamics – are increasingly popular in the machine learning field. The basic idea is nothing new; in the early days of AI multilayer recurrent neural networks figured prominently. But the early practitioners lacked effective, scalable algorithms for updating deep learning networks, as well as adequate hardware to run large-scale networks. Modern deep learning systems leverage better hardware and algorithms to achieve impressive tasks, e.g. the recently reported success of a neural net deep learning system at unsupervised identification of everyday concepts from a large corpus of videos [LRM+12].

A variety of fairly different deep learning architectures have been proposed and implemented; the kind that interests us here specifically is what we call a "Compositional Spatiotemporal Deep Learning Network" (CSDLN) [Goe11]. This is a deep learning network consisting of a collection of layers of nodes, where the nodes on higher layers correspond to larger space-time regions, and the children of a node correspond to smaller regions that are largely included in their parent's correspondent region. Each node is concerned with learning patterns associated with the region of space-time to which it corresponds. The differences between CSDLN approaches have mainly to do with the processing inside each node, and the nature of information passed between nodes. The most thoroughly popularized CSDLN is Jeff Hawkins' HTM system [HB06] [GH09], but the concept significantly precedes Hawkins' work, and is also being pursued by a variety of other current researchers [TSH11] [BH10]. DeSTIN is one recent example of a CSDLN, and is the one we will focus on here.

In its original form, DeSTIN[1] is a purely subsymbolic system intended to roughly model the structure and dynamics of perception processing in the human brain (though not to model these in detail – it is not a neural model). The general DeSTIN design has been described in talks as comprising three crosslinked hierarchies, handling perception, action and reinforcement; but so far only the perceptual hierarchy (also called the "spatiotemporal inference network") has been implemented or described in detail in publications. In this section we will discuss DeSTIN as a perception system, and will discuss handling of action and reinforcement later in the context of DeSTIN-OpenCog integration.

The hierarchical architecture of DeSTIN's spatiotemporal inference network comprises an arrangement into multiple layers of "nodes" comprising multiple instantiations of an identical processing unit. Each node corresponds to a particular spatiotemporal region, and uses a statistical learning algorithm to characterize the sequences of patterns that are presented to it by nodes in the layer beneath it.

More specifically, at the very **lowest layer** of the hierarchy nodes receive as input raw data (e.g. pixels of an image) and continuously construct a belief state that attempts to characterize the sequences of patterns viewed. The **second layer**, and all those above it, receive as input the belief states of nodes at their corresponding lower layers, and attempt to construct belief states that capture regularities in their inputs. Each node also receives as input the belief state of the node above it in the hierarchy (which constitutes "contextual" information, utilized in the node's prediction process).

Inside each node, an online clustering algorithm is used to identify regularities in the sequences received by that node. The centroids of the clusters learned are stored in the node and comprise the basic visual patterns recognized by that node. The node's "belief" regarding what it is seeing, is then understood as a probability density function defined over the centroids at that node. The equations underlying this centroid formation and belief updating process are identical for every node in the architecture, and were given in

[1] Some sentences in this section were pasted with minor modifications from [GLdG+10], coauthored by Ben Goertzel, Itamar Arel and others

their original form in [ARC09], though the current open-source DeSTIN codebase reflects some significant improvements not yet reflected in the publication record.

In short, the way DeSTIN represents an item of knowledge is as a probability distribution over "network activity patterns" in its hierarchical network. An activity pattern, at each point in time, comprises an indication of which centroids in each node are most active, meaning they have been identified as most closely resembling what that node has perceived, as judged in the context of the perceptions of the other nodes in the system. Based on this methodology, the DeSTIN perceptual network serves the critical role of building and maintaining a model of the state of the world as visually perceived.

This methodology allows for powerful unsupervised classification. If shown a variety of real-world scenes, DeSTIN will automatically form internal structures corresponding to the various natural categories of objects shown in the scenes, such as trees, chairs, people, etc.; and also to the various natural categories of events it sees, such as reaching, pointing, falling. In order to demonstrate the informativeness of these internal structures, experiments have been done using DeSTIN's states as input feature vectors for supervised learning algorithms, enabling high-accuracy supervised learning of classification models from labeled image data [KAR10]. A closely related algorithm developed by the same principal researcher (Itamar Arel) has proven extremely successful at audition tasks such as phoneme recognition [ABS+11].

2.1 Beyond Gray-Scale Vision

The DeSTIN approach may easily be extended to other senses beyond gray-scale vision. For color vision, it suffices to replace the one-dimensional signals coming into DeSTIN's lower layer with 3D signals representing points in the color spectrum; the rest of the DeSTIN process may be carried over essentially without modification. Extension to further senses is also relatively straightforward on the mathematical and software structure level, though they may of course require significant additional tuning and refinement of details.

For instance, while olfaction does not lend itself well to hierarchical modeling, but audition and haptics (touch) do:

- for auditory perception, one could use a DeSTIN architecture in which each layer is one-dimensional rather than two-dimensional, representing a certain pitch. Or one could use two dimensions for pitch and volume. This results in a system quite similar to the DeSTIN-like system shown to perform outstanding phoneme recognition in [ABS+11], and is conceptually similar to Hierarchical Hidden Markov Models (HHMMs), which have proven quite successful in speech recognition and which Ray Kurzweil has argued are the central mechanism of human intelligence [Kur12]. Note also recent results published by Microsoft Research, showing dramatic improvements over prior speech recognition results based on use of a broadly HHMM-like deep learning system [HDY+12].

- for haptic perception, one could use a DeSTIN architecture in which the lower layer of the network possesses a 2D topology reflecting the topology of the surface of the body. Similar to the somatosensory cortex in the human brain, the map could be distorted so that more "pixels" are used for regions of the body from which more data is available (e.g. currently this might be the fingertips, if these were implemented using Syntouch technology [FL12], which has proved excellent at touch-based object identification). Input could potentially be multidimensional if multiple kinds of haptic sensors were available, e.g temperature, pressure and movement as in the Syntouch case.

Augmentation of DeSTIN to handle action as well as perception is also possible, and will be discussed in Part II of the paper.

3 Uniform DeSTIN

It would be possible to integrate DeSTIN in its original form with OpenCog or other AI systems with symbolic aspects, via using an unsupervised machine learning algorithm to recognize patterns in sets of states of the DeSTIN network as originally defined. However, this pattern recognition task becomes much easier if one suitably modifies DeSTIN, so as to make the commonalities between semantically similar states more obviously perceptible. This can be done by making the library of patterns recognized within each DeSTIN

node invariant with respect to translation, scale, rotation and shear – a modification we call "Uniform DeSTIN." This "uniformization" decreases DeSTIN's degree of biological mimicry, but eases integration of DeSTIN with symbolic AI methods.

3.1 Translation-Invariant DeSTIN

The first revision to the "classic DeSTIN" to be suggested here is: All the nodes on the same level of the DeSTIN hierarchy should share the same library of patterns. In the context of classic DeSTIN (i.e. in the absence of further changes to DeSTIN to be suggested below, which extend the type of patterns usable by DeSTIN), this means: the nodes on the same level should share the same list of centroids. This makes DeSTIN's pattern recognition capability translation-invariant. This translation invariance can be achieved without any change to the algorithms for updating centroids and matching inputs to centroids.

In this approach, it's computationally feasible to have a much larger library of patterns utilized by each node, as compared to in classic DeSTIN. Suppose we have a $n \times n$ pixel grid, where the lowest level has nodes corresponding to 4×4 squares. Then, there are $(\frac{n}{4})^2$ nodes on the lowest level, and on the k'th level there are $(\frac{n}{4^k})^2$ nodes. This means that, without increasing computational complexity (actually decreasing it, under reasonable assumptions), in translation-invariant Uniform DeSTIN we can have a factor of $(\frac{n}{4^k})^2$ more centroids on level k.

One can achieve a much greater decrease in computational complexity (with the same amount of centroid increase) via use of a clever data structure like a cover tree [BKL06] to store the centroids at each level. Then the nearest-neighbor matching of input patterns to the library (centroid) patterns would be very rapid, much faster than linearly comparing the input to each pattern in the list.

3.1.1 Conceptual Justification for Uniform DeSTIN

Generally speaking, one may say that: **if** the class of images that the system will see is invariant with respect to linear translations, **then** without loss of generality, we can assume that the library of patterns at each node on the same level is the same.

In reality this assumption isn't quite going to hold. For instance, for an eye attached to a person or humanoid robot, the top of the pixel grid will probably look at a person's hair more often than the bottom ... because the person stands right-side-up more often than they stand upside-down, and because they will often fixate the center of their view on a person's face, etc. For this reason, we can recognize our friend's face better if we're looking at them directly, with their face centered in our vision.

However, we suggest that this kind of peculiarity is not really essential to vision processing for general intelligence. There's no reason you can't have an intelligent vision system that recognizes a face just as well whether it's centered in the visual field or not. (In fact you could straightforwardly explicitly introduce this kind of bias within a translation-invariant DeSTIN, but it's not clear this is a useful direction.)

By and large, in almost all cases, it seems to us that in a DeSTIN system exposed to a wide variety of real-world inputs in complex situations, the library of patterns in the different nodes at the same level would turn out to be substantially the same. Even if they weren't exactly the same, they would be close to the same, embodying essentially the same regularities. But of course, this sameness would be obscured, because centroid 7 in a certain node X on level 4 might actually be the same as centroid 18 in some other node Y on level 4 ... and there would be no way to tell that centroid 7 in node X and centroid 18 and node Y were actually referring to the same pattern, without doing a lot of work.

3.1.2 Comments on Biological Realism

Translation-invariant DeSTIN deviates further from human brain structure than classic DeSTIN, but this is for good reason.

The brain has a lot of neurons, since adding new neurons was fairly easy and cheap for evolution; and tends to do things in a massively parallel manner, with great redundancy. For the brain, it's not so problematically expensive to have the functional equivalent of a lot of DeSTIN nodes on the same level, all simultaneously using and learning libraries of patterns that are essentially identical to each other. Using current computer technology, on the other hand, this sort of strategy is rather inefficient.

In the brain, messaging between separated regions is expensive, whereas replicating function redundantly is cheap. In current computers, messaging between separated regions is fairly cheap (so long as those regions are stored on the same machine), whereas replicating function redundantly is expensive. Thus, even in cases where the same concept and abstract mathematical algorithm can be effectively applied in both the brain and a computer, the specifics needed for efficient implementation may be quite different.

3.2 Mapping States of Translation-Invariant DeSTIN into Symbolic Representations

Mapping classic DeSTIN's states into a symbolic pattern-manipulation engine like OpenCog is possible, but relatively cumbersome. Doing the same thing with Uniform DeSTIN is much more straightforward.

In Uniform DeSTIN, for example, Cluster 7 means the same thing in ANY node on level 4. So after a Uniform DeSTIN system has seen a fair number of images, you can be pretty sure its library of patterns is going to be relatively stable. Some clusters may come and go as learning progresses, but there's going to be a large and solid library of clusters at each level that persists, because all of its member clusters occur reasonably often across a variety of inputs.

Define a DeSTIN state-tree as a (quaternary) tree with one node for each DeSTIN node; and living at each node, a small list of (integer pattern_code, float weight) pairs. That is, at each node, the state-tree has a short-list of the patterns that closely match a given state at that node. The weights may be assumed between 0 and 1. The integer pattern codes have the same meaning for every node on the same level.

As you feed DeSTIN inputs, at each point in time it will have a certain state, representable as a state-tree. So, suppose you have a large database of DeSTIN state-trees, obtained by showing various inputs to DeSTIN over a long period of time. Then, you can do various kinds of pattern recognition on this database of state-trees.

More formally, define a state-subtree as a (quaternary) tree with a single integer at each node. Two state-subtrees may have various relationships with each other within a single state-tree – for instance they may be adjacent to each other, or one may appear atop or below the other, etc. In these terms, one interesting kind of pattern recognition to do is: Recognize frequent state-subtrees in the stored library of state-trees; and then recognize frequent relationships between these frequent state-subtrees. The latter relationships will form a kind of "image grammar," conceptually similar and formally related to those described in [ZM06]. Further, temporal patterns may be recognized in the same way as spatial ones, as part of the state-subtree grammar (e.g. state-subtree A often occurs right before state-subtree B; state-subtree C often occurs right before and right below state-subtree D; etc.).

Anticipating ideas to be presented in Part II of this paper, the flow of activation from OpenCog back down to DeSTIN is also fairly straightforward in the context of translation-invariant DeSTIN. If relationships have been stored between concepts in OpenCog's memory and grammatical patterns between state-subtrees, then whenever concept C becomes important in OpenCog's memory, this can cause a top-down increase in the probability of matching inputs to DeSTIN node centroids, that would cause the DeSTIN state-tree to contain the grammatical patterns corresponding to concept C.

3.3 Scale-Invariant DeSTIN

The next step, moving beyond translation invariance, is to make DeSTIN's pattern recognition mostly (not wholly) scale invariant. We will describe a straightforward way to map centroids on one level of DeSTIN, into centroids on the other levels of DeSTIN. This means that when a centroid has been learned on one level, it can be experimentally ported to all the other levels, to see if it may be useful there too.

To make the explanation of this mapping clear, we reiterate some DeSTIN basics in slightly different language:

- A centroid on Level N is: a spatial arrangement (e.g. $k \times k$ square lattice) of beliefs of Level $N-1$. (More generally it is a spatiotemporal arrangement of such beliefs, but we will ignore this for the moment.)
- A belief on Level N is: a probability distribution over centroids on Level N. For heuristic purposes one can think about this as a mixture of Gaussians, though this won't always be the best model.

- Thus, a belief on Level N is: a probability distribution over spatial (or more generally, spatiotemporal) arrangements of beliefs on Level $N-1$

On Level 1, the role of centroids is played by simple $k \times k$ squares of pixels. Level 1 beliefs are probability distributions over these small pixel squares. Level 2 centroids are hence spatial arrangements of probability distributions over small pixel-squares; and Level 2 beliefs are probability distributions over spatial arrangements of probability distributions over small pixel-squares.

A small pixel-square S may be mapped into a single pixel P via a heuristic algorithm such as:

- if S has more black than white pixels, then P is black
- is S has more white than black pixels, then P is white
- if S has an equal number of white and black pixels, then use some heuristic. For instance if S is 4×4 you could look at the central 2×2 square and assign P to the color that occurs most often there. If that is also a tie, then you can just arbitrarily assign P to the color that occurs in the upper left corner of S.

A probability distribution over small pixel-squares may then be mapped into a probability distribution over pixel values (B or W). A probability distribution over the two values B and Wmay be approximatively mapped into a single pixel value – the one that occurs most often in the distribution, with a random choice made to break a tie. This tells us how to map Level 2 beliefs into spatial arrangements of pixels; and thus, it tells us how to map Level 2 beliefs into Level 1 beliefs.

But this tells us how to map Level N beliefs into Level $N-1$ beliefs, inductively. Remember, a Level N belief is a probability distribution (pdf for short) over spatial arrangements of beliefs on Level $N-1$. For example: A Level 3 belief if a pdf over arrangements of Level 2 beliefs. But since we can map Level 2 beliefs into Level 1 beliefs, this means we can map a Level 3 belief into a pdf over arrangements of Level 1 beliefs – which means we can map a Level 3 belief into a Level 2 belief. Etc.

Of course, this also tells us how to map Level N centroids into Level $N-1$ centroids. A Level N centroid is a pdf over arrangements of Level $N-1$ beliefs; a Level $N-1$ centroid is a pdf over arrangements of Level $N-2$ beliefs. But Level $N-1$ beliefs can be mapped into Level $N-2$ beliefs, so Level N centroids can be represented as pdfs over arrangements of Level N beliefs, and hence mapped into Level $N-1$ centroids.

In practice, one can implement this idea by moving from the bottom up. Given the mapping from Level 1 "centroids" to pixels, one can iterate through the Level 1 beliefs and identify which pixels they correspond to. Then one can iterate through the Level 2 beliefs and identify which Level 1 beliefs they correspond to. Etc. Each Level N belief can be explicitly linked to a corresponding level $N-1$ belief. Synchronously, as one moves up the hierarchy, Level N centroids can be explicitly linked to corresponding Level $N-1$ centroids.

Since there are in principle more possible Level Nbeliefs than Level $N-1$ beliefs, the mapping from level Nbeliefs to level $N-1$ beliefs is many-to-one. This is a reason not to simply maintain a single centroid pool across levels. However, when a new centroid C is added to the Level N pool, it can be mapped into a Level $N-1$ centroid to be added to the Level $N-1$ pool (if not there already). And, it can also be used to spawn a Level $N+1$ centroid, drawn randomly from the set of possible Level $N+1$ centroids that map into C.

Also, note that it is possible to maintain a single centroid *numbering system* across levels, so that a reference like "centroid # 175" has only one meaning in an entire DeSTIN network, even though some of these centroid may only be meaningful above a certain level in the network.

3.4 Rotation Invariant DeSTIN

With a little more work, one can make DeSTIN rotation and shear invariant as well [2]. Considering rotation first:

- When comparing an input A to a Level N node with a Level N centroid B, consider various rotations of A, and see which rotation gives the closest match.

[2]The basic idea in this section, in the context of rotation, is due to Jade O'Neill (private communication)

- When you match a centroid to an input observation-or-belief, record the rotation angle corresponding to the match.

The second of these points implies the tweaked definitions

- A centroid on Level N is: a spatial arrangement (e.g. $k \times k$ square lattice) of beliefs of Level $N-1$
- A belief on Level N is: a probability distribution over (angle, centroid) pairs on Level N.

From these it follows that a belief on Level N is: a probability distribution over (angle, spatial arrangement of beliefs) pairs on Level $N-1$

An additional complexity here is that two different (angle, centroid) pairs (on the same level) could be (exactly or approximately) equal to each other. This necessitates an additional step of "centroid simplification", in which ongoing checks are made to see if there are any two centroids C_1, C_2 on the same level so that: There exist angles A_1, A_2 so that (A_1, C_1) is very close to (A_2, C_2). In this case the two centroids may be merged into one.

To apply these same ideas to shear, one may simply replace "rotation angle" in the above by "(rotation angle, shear factor) pair."

3.5 Temporal Perception

Translation and scale invariant DeSTIN can be applied perfectly well if the inputs to DeSTIN, at level 1, are movies rather than static images. Then, in the simplest version, Level 1 consists of pixel cubes instead of pixel squares, etc. (the third dimension in the cube representing time). The scale invariance achieved by the methods described above would then be scale invariance in time as well as in space.

In this context, one may enable rectangular shapes as well as cubes. That is, one can look at a Level N centroid consisting of m time-slices of a $k \times k$ arrangement of Level $N-1$ beliefs – without requiring that $m = k$ .... This would make the centroid learning algorithm a little more complex, because at each level one would want to consider centroids with various values of m, from $m = 1, ..., k$ (and potentially $m > k$ also).

4 Interpretation of DeSTIN's Activity

Uniform DeSTIN constitutes a substantial change in how DeSTIN does its business of recognizing patterns in the world – conceptually as well as technically. To explicate the meaning of these changes, we briefly present our favored interpretation of DeSTIN's dynamics.

The centroids in the DeSTIN library represent points in "spatial pattern space", i.e. they represent exemplary spatial patterns. DeSTIN's beliefs, as probability distributions over centroids, represent guesses as to which of the exemplary spatial patterns are the best models of what's currently being seen in a certain space-time region.

This matching between observations and centroids might seem to be a simple matter of "nearest neighbor matching"; but the subtle point is, it's not immediately obvious how to best measure the distance between observations and centroids. The optimal way of measuring distance is going to depend on context; that is to say, on the actual distribution of observations in the system's real environment over time.

DeSTIN's algorithm for calculating the belief at a node, based on the observation and centroids at that node **plus** the beliefs at other nearby nodes, is essentially a way of tweaking the distance measurement between observations and centroids, so that this measurement accounts for the context (the historical distribution of observations). There are many possible ways of doing this tweaking. Ideally one could use probability theory explicitly, but that's not always going to be computationally feasible, so heuristics may be valuable, and various versions of DeSTIN have contained various heuristics in this regard.

The various ways of "uniformizing" DeSTIN described above (i.e. making its pattern recognition activity approximately invariant with respect to affine transformations), don't really affect this story – they just improve the algorithm's ability to learn based on small amounts of data (and its rapidity at learning from data in general), by removing the need for the system to repeatedly re-learn transformed versions of the same patterns. So the uniformization just lets DeSTIN carry out its basic activity faster and using less data.

4.1 DeSTIN's Assumption of Hierarchical Decomposability

Roughly speaking, DeSTIN will work well to the extent that: The average distance between each part of an actually observed spatial pattern, and the closest centroid pattern, is not too large (note: the choice of distance measure in this statement is potentially subtle). That is: DeSTIN's set of centroids is supposed to provide a compact model of the probability distribution of spatial patterns appearing in the experience of the cognitive system of which DeSTIN is a part.

DeSTIN's effective functionality relies on the assumption that this probability distribution is hierarchically decomposable – i.e. that the distribution of spatial patterns appearing over a $k \times k$ region can be compactly expressed, to a reasonable degree of approximation, as a spatial combination of the distributions of spatial patterns appearing over $(k/4) \times (k/4)$ regions. This assumption of hierarchical decomposability greatly simplifies the search problem that DeSTIN faces, but also restricts DeSTIN's capability to deal with more general spatial patterns that are not easily hierarchically decomposable. However, the benefits of this approach seem to outweigh the costs, given that visual patterns in the environments humans naturally encounter do seem (intuitively at least) to have this hierarchical property.

4.2 Distance and Utility

Above we noted that choice of distance measure involved in the assessment of DeSTIN's effective functionality is subtle. Further above, we observed that the function of DeSTIN's belief assessment is basically to figure out the contextually best way to measure the distance between the observation and the centroids at a node. These comments were both getting at the same point.

But what is the right measure of distance between two spatial patterns? Ultimately, the right measure is: the probability that the two patterns A and B can be used in the same way. That is: the system wants to identify observation A with centroid B if it has useful action-patterns involving B, and it can substitute A for B in these patterns without loss.

This is difficult to calculate in general, though. A rough proxy, which it seems will often be acceptable, is to measure the distance between A and B in terms of both

- the basic (extensional) distance between the physical patterns they embody (e.g. pixel by pixel distance)
- the contextual (intensional) distance, i.e. the difference between the contexts in which they occur

Via enabling the belief in a node's parent to play a role in modulating a certain node's belief, DeSTIN's core algorithm enables contextual/intensional factors to play a role in distance assessment.

5 Benefits and Costs of Uniform DeSTIN

We now summarize the main benefits and costs of Uniform DeSTIN a little more systematically. The key point we have made here regarding Uniform DeSTIN and representational transparency may be summarized as follows:

- Define an "affine perceptual equivalence class" as a set of percepts that are equivalent to each other, or nearly so, under affine transformation. An example would be views of the same object from different perspectives or distances.
- Suppose one has embodied agent using DeSTIN for visual perception, whose perceptual stream tends to include a lot of reasonably large affine perceptual equivalence classes.
- Then, supposing the "mechanics" of DeSTIN can be transferred to the Uniform DeSTIN case without dramatic loss of performance, Uniform DeSTIN should be able to recognize patterns based on many fewer examples than classic DeSTIN

As soon as Uniform DeSTIN has learned to recognize one element of a given affine perceptual equivalence class, it can recognize all of them. Whereas, classic DeSTIN must learn each element of the equivalence

class separately. So, roughly speaking, the number of cases required for unsupervised training of Uniform DeSTIN will be less than that for classic DeSTIN, by a ratio equal to the average size of the affine perceptual equivalence classes in the agent's perceptual stream.

Counterbalancing this, we have the performance cost of comparing the input to each node against a much larger set of centroids (in Uniform DeSTIN as opposed to classic DeSTIN). However, if a cover tree or other efficient data structure is used, this cost is not so onerous. The cost of nearest neighbor queries in a cover tree storing n items (in this case, n centroids) is $O(c^{12}logn)$, where the constant c represents the "intrinsic dimensionality" of the data; and in practice the cover tree search algorithm seems to perform quite well. So, the added time cost for online clustering in Uniform DeSTIN as opposed to DeSTIN, is a factor on the order of the log of the number of nodes in the DeSTIN tree. We believe this moderate added time cost is well worth paying, to gain a significant decrease in the number of training examples required for unsupervised learning.

Beyond increases in computational cost, there is also the risk that the online clustering may just not work as well when one has so many clusters in each node. This is the sort of problem that can really only be identified, and dealt with, during extensive practice – since the performance of any clustering algorithm is largely determined by the specific distribution of the data it's dealing with. It may be necessary to improve DeSTIN's online clustering in some way to make Uniform DeSTIN work optimally, e.g. improving its ability to form clusters with markedly non-spherical shapes. This ties in to a point raised in Part II of this paper – the possibility of supplementing traditional clusters with predicates learned by OpenCog, which may live inside DeSTIN nodes alongside centroids. Each such predicate in effect defines a (generally nonconvex) "cluster".

6 Conclusion

We have presented an extension of DeSTIN, a very capable deep learning based machine perception system. This extension, called Uniform DeSTIN, is expected to enhance DeSTIN's functionality in practical applications, via enabling it to effectively model the environment based on fewer observations. It also greatly increases the ease of integrating DeSTIN with other AI systems such as OpenCog, via increasing DeSTIN's representational transparency, so that percepts related by affine transformations will have identical or very similar representations inside DeSTIN. Part II of this paper explores one research trajectory enabled by Uniform DeSTIN: the integration of DeSTIN and OpenCog to form a hybrid system with strong potential for bridging the symbolic/subsymbolic gap that has posed so many difficulties for the AI field for so long.

Much work remains to be done, to complete the implementation and testing and integration of the ideas presented here, but we believe this is a very promising direction; and we suggest that other deep learning systems can likely be made more representationally transparent in a manner similar to what we've done with DeSTIN here, with benefits to their quality of function and their ease of integration with other AI systems.

References

[ABS+11] Itamar Arel, S Berant, T Slonim, A Moyal, B Li, and K Chai Sim. Acoustic spatiotemporal modeling using deep machine learning for robust phoneme recognition. In *Afeka-AVIOS Speech Processing Conference*, 2011.

[ARC09] I. Arel, D. Rose, and R. Coop. Destin: A scalable deep learning architecture with application to high-dimensional robust pattern recognition. *Proc. AAAI Workshop on Biologically Inspired Cognitive Architectures*, 2009.

[ARK09] I. Arel, D. Rose, and T. Karnowski. A deep learning architecture comprising homogeneous cortical circuits for scalable spatiotemporal pattern inference. *NIPS 2009 Workshop on Deep Learning for Speech Recognition and Related Applications*, 2009.

[BH10] Bundzel and 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.

[BKL06] Alina Beygelzimer, Sham Kakade, and John Langford. Cover trees for nearest neighbor. In *Proc. International Conference on Machine Learning*, 2006.

[FL12] Jeremy Fishel and Gerald Loeb. Bayesian exploration for intelligent identification of textures. *Frontiers in Neurorobotics 6-4*, 2012.

[GH09] Dileep George and Jeff Hawkins. Towards a mathematical theory of cortical micro-circuits. *PLoS Comput Biol 5*, 2009.

[GLdG+10] Ben Goertzel, Ruiting Lian, Hugo de Garis, Shuo Chen, and Itamar Arel. World survey of artificial brains, part ii: Biologically inspired cognitive architectures. *Neurocomputing*, April 2010.

[Goe09] Ben Goertzel. Opencog prime: A cognitive synergy based architecture for embodied artificial general intelligence. In *ICCI 2009, Hong Kong*, 2009.

[Goe11] B Goertzel. 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.

[Goeon] Ben Goertzel. Perception processing for general intelligence, part ii: Bridging the symbolic/-subsymbolic gap. (in preparation).

[HB06] Jeff Hawkins and Sandra Blakeslee. *On Intelligence*. Brown Walker, 2006.

[HDY+12] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, and Tara Sainathand Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 2012.

[KAR10] Tom Karnowski, Itamar Arel, and D. Rose. Deep spatiotemporal feature learning with application to image classification. In *The 9th International Conference on Machine Learning and Applications (ICMLA'10)*, 2010.

[Kur12] Ray Kurzweil. *How to Create a Mind*. Viking, 2012.

[LRM+12] Quoc V. Le, Marc'Aurelio Ranzato, Rajat Monga, Kai Chen Matthieu Devin, Greg S. Corrado, Jeffrey Dean, and Andrew Y. Ng. Building high-level features using large scale unsupervised learning. In *Proceedings of the Twenty-Ninth International Conference on Machine Learning*, 2012.

[TSH11] Mohamad Tarifi, Meera Sitharam, and Jeffery Ho. Learning hierarchical sparse representations using iterative dictionary learning and dimension reduction. In *Proc. of BICA 2011*, 2011.

[ZM06] Song-Chun Zhu and David Mumford. A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision*, 2006.