To appear in *Connectionist-Symbolic Integration*, Ed. R. Sun and F. Alexandre, Lawrence Erlbaum Associates, 1997.

17

STRUCTURE MATCHING AND TRANSFORMATION WITH DISTRIBUTED REPRESENTATIONS

Tony A. Plate

School of Mathematical and Computing Sciences
Victoria University of Wellington

1 INTRODUCTION

One of the barriers to integrating connectionist and symbolic computation is the difficulty of representing recursive structure in a distributed fashion. 4) discusses this problem and proposes a framework in which "reduced descriptions" are used to represent parts and wholes in a part-whole hierarchy. Hinton's framework requires that a number of vectors, each a part and together forming a whole, be compressed (reduced) into a single vector of the same dimension as the original vectors. This reduced vector can in turn be used as a part in the representation of some greater whole. The reduction must be reversible so that one can move in both directions in a part-whole hierarchy, i.e., reduce a set of vectors to a single vector (a whole to a part), and expand a single vector to a set of vectors (a part to a whole). In this way, compositional structure can be represented. For reduced descriptions to be truly useful they should be systematically related to their components, so that information about the components can be gleaned from the reduced description without requiring its expansion. It is this property that distinguishes reduced descriptions from pointer-based methods for representing recursive structure, in which the pointer is the "reduced description" but is arbitrary and is unrelated to the components of the object it points to.

Holographic Reduced Representations (HRRs) (10) provide an implementation of Hinton's reduced descriptions. In HRRs, circular convolutions are used to construct associations of vectors. These representations of associations differ from those produced by the more familiar outer-product associative operator (as in Smolensky's (14) Tensor products) in that they are vectors of the same dimensionality as the vectors participating in the associations. This allows the easy construction of representations of objects with recursive structure. HRRs inherit many of the attractive properties of distributed representations of objects, including the ability to represent similar objects by similar representations. In fact, if the objects in two HRRs have similar representations, then the two HRRs will be similar, and the similarity will further depend on the similarity of the structural arrangement of the objects. The most interesting properties of HRRs are related to structural operations that can be performed without unpacking – matching and transformation are described in this chapter.

The next section of this chapter gives a brief overview of HRRs. For more details, consult 10) or 9). The third section gives an example of structure matching with HRRs: obtaining fast estimates of the analogical similarity of structured objects. The fourth section describes how HRRs can be transformed structurally without unpacking. The final section compares HRRs to other connectionist representations for complex structure and discuss some general issues.

2 HOLOGRAPHIC REDUCED REPRESENTATIONS

HRRs are a distributed representation for recursive propositional (or frame-like) structures. Objects, predicate labels, and roles are represented by n-dimensional vectors of real numbers. Instantiated propositions are encoded as the sum of role-filler bindings. A binding is the circular convolution (denoted by $\circledast$) of a role and a filler. The circular convolution of two n-dimensional vectors $\mathbf{x}$[1] and $\mathbf{y}$ ($\mathbf{z} = \mathbf{x} \circledast \mathbf{y}$) is n-dimensional and has elements

$$z_i = \sum_{k=0}^{n-1} x_k y_{j-k}$$

where subscripts are modulo-n. Circular convolution can be considered as a multiplication operation for vectors, and has algebraic properties in common with scalar and with matrix multiplication: it is associative, commutative, distributive over addition, an identity vector exists, and most vectors have inverses. Circular convolution can also be computed efficiently via the Fast Fourier Transform in $O(nlogn)$ time (see 12) for algorithms).

[1] Bold-face names, e.g., $\mathbf{x}$ are used for vectors.

For example, a simple propositional representation of "Spot bit Jane" is

$$\mathbf{P_{bite}} = \langle \mathbf{bite} + \mathbf{bite_{agt}} \circledast \mathbf{spot} + \mathbf{bite_{obj}} \circledast \mathbf{jane} \rangle,$$

where **bite** is a predicate label, $\mathbf{bite_{agt}}$ and $\mathbf{bite_{obj}}$ are roles, and **jane** and **spot** are fillers. The angle brackets $\langle \cdot \rangle$ indicate that the vector is normalized to have a Euclidean length of one. Bindings do not get confused when added together: $\mathbf{bite_{agt}} \circledast \mathbf{spot} + \mathbf{bite_{obj}} \circledast \mathbf{jane}$ is quite distinct from $\mathbf{bite_{agt}} \circledast \mathbf{jane} + \mathbf{bite_{obj}} \circledast \mathbf{spot}$. Even though some components are associations (the bindings) and some are objects (the predicate label), they are all n-dimensional vectors and can be superimposed straightforwardly. Thus, $\mathbf{P_{bite}}$, the representation for the entire proposition, is also an n-dimensional vector. This makes it simple to use propositions as fillers for roles in other propositions. For example, in the proposition for "Spot bit Jane, which caused Jane to run away from Spot." ($\mathbf{P_{cause}}$, below), $\mathbf{P_{bite}}$ fills the antecedent role of the top-level cause proposition:

$$\begin{aligned}
\mathbf{P_{bite}} &= \langle \mathbf{bite} + \mathbf{bite_{agt}} \circledast \mathbf{spot} + \mathbf{bite_{obj}} \circledast \mathbf{jane} \rangle \\
\mathbf{P_{flee}} &= \langle \mathbf{flee} + \mathbf{flee_{agt}} \circledast \mathbf{jane} + \mathbf{flee_{from}} \circledast \mathbf{spot} \rangle \\
\mathbf{P_{objects}} &= \langle \mathbf{jane} + \mathbf{spot} \rangle \\
\mathbf{P_{cause}} &= \langle \mathbf{cause} + \mathbf{P_{objects}} + \mathbf{P_{bite}} + \mathbf{P_{flee}} \\
&\quad + \mathbf{cause_{antc}} \circledast \mathbf{P_{bite}} + \mathbf{cause_{cnsq}} \circledast \mathbf{P_{flee}} \rangle
\end{aligned}$$

The unbound representations for the objects ($\mathbf{P_{objects}}$) and lower-level HRRs ($\mathbf{P_{bite}} + \mathbf{P_{flee}}$) are included in $\mathbf{P_{cause}}$ in order to make the representation for a proposition have some similarity to those objects and other propositions composed of those objects; these components are not necessary for the representation of the structure.

Bindings can be decoded by convolving with the approximate inverse[2] of the role (or filler) vector. For example, the filler in $\mathbf{bite_{agt}} \circledast \mathbf{spot}$ can be found by convolving it with $\mathbf{bite^{*}_{agt}}$ ($\mathbf{x}^*$ denotes the approximate inverse of $\mathbf{x}$). The result is a noisy version of **spot**, which must be cleaned up using some sort of auto-associative memory in which the possible fillers have been stored. This clean-up memory must perform closest-match retrieval and must store all vectors, both objects and structures, which can result from decoding a HRR in the system.

[2]The exact inverse of a vector under convolution usually exists, but is numerically unstable except under certain conditions. Thus it is usually preferable to use the approximate inverse.

When multiple bindings are superimposed, as in $\mathbf{P_{bite}}$, the other bindings do not interfere with the decoding other than by making it more noisy. The approximate inverse of a vector under convolution is simple to compute: it is a permutation of elements: $\mathbf{x_i^*} = \mathbf{x_{n-i}}$, where subscripts are modulo-n (this is equivalent to reversal of the elements followed by rotation by one position). Thus, like binding, decoding (without cleanup) can be computed in $O(n \log n)$ time.

As a binding operation, convolution has two very useful properties: it is randomizing and similarity preserving. Convolving a role and a filler effectively randomizes with respect to the role and the filler: the binding will not be similar to either the role or the filler (except in special cases). On the other hand, convolution preserves similarity between bindings with similar components: if **jane** is similar to **fred** then $\mathbf{jane} \circledast \mathbf{role}$ will be similar to $\mathbf{fred} \circledast \mathbf{role}$ to approximately the same degree.

The distributed representations for the base-level components, i.e., objects, roles, and labels, are based on random vectors of independently distributed elements with mean zero and variance $1/n$. Representations of similar entities (e.g., **jane** and **fred**) are composed of a base type (**person**) and unique random identifying vectors ($\mathbf{id_{jane}}$ and $\mathbf{id_{fred}}$):

$$\begin{aligned} \mathbf{jane} &= \langle \mathbf{person} + \mathbf{id_{jane}} \rangle \\ \mathbf{fred} &= \langle \mathbf{person} + \mathbf{id_{fred}} \rangle. \end{aligned}$$

The mixing proportions of the various components can be changed to make objects of the same type more or less similar.

HRRs require high-dimensional vectors to work well – even small toy problems require vectors with 512 or 1,024 elements.[3] However, this is not as much of a drawback as it might seem, for two reasons. One reason is that the scaling is very good, thus larger problems do not require much higher dimensionality – tens or hundreds or thousands of objects can be represented using vectors with 2,048 or 4,192 elements. The other reason is that the HRR construction and decoding operations are very fast: binding and decoding can be computed in $O(n \log n)$ time. Clean-up (i.e., finding the closest matching vector in memory) is the only slow operation, but is highly amenable to parallel implementation.

[3] Vectors here have dimensions which are powers of 2 only because this makes FFT implementation of convolution simple and efficient.

3 ESTIMATING ANALOGICAL SIMILARITY VIA DOT PRODUCTS OF HRRS

The superficial features of HRRs can reflect their structural composition: if the entities in structures are similar, then the HRRs are similar to the degree that their structures are similar. This allows the estimation of degree of structural match by the dot product of vectors (HRRs). The sensitivity of this representation to various aspects of structural similarity is illustrated using the examples shown in Table 1. These examples are elaborations on an example in 16). The episodes **E1**–**E5** have different degrees of similarity to the probe on attributes (first-order features), role alignment, and higher-order structure.

Probe:		Spot bit Jane, causing Jane to flee from Spot.	Average dot product with probe		
Episodes for comparison:			Expt1	Expt2	Expt3
E1	LS	Fido bit John, causing John to flee from Fido.	0.70	0.63	0.81
E2	AN^{cm}	Fred bit Rover, causing Rover to flee from Fred.	0.47	0.47	0.69
E3	AN	Felix bit Mort, causing Mort to flee from Felix.	0.39	0.39	0.61
E4	SF	John fled from Fido, causing Fido to bite John.	0.47	0.44	0.53
E5	FOR	Mort bit Felix, causing Mort to flee from Felix.	0.39	0.39	0.39

Table 1 Results of Experiments 1, 2 and 3.

In these episodes Jane, John, and Fred are people; Spot, Fido and Rover are dogs; Felix is a cat; and Mort is a mouse. All of these are *objects*, represented by *token* vectors. Tokens of the same type are considered to be similar to each other, but not to tokens of other types. Bite, flee, and cause are *relations* (i.e., propositions). The argument structure of the cause relation, and the patterns in which objects fill multiple roles constitutes the *higher-order* structure.

The second column classifies the relationship between each episode and the probe using Gentner et al.'s (3) types of similarity: LS (Literal Similarity) shares relations, object features, and higher-order structure; AN (Analogy, also called True Analogy) shares relations and higher-order structure, but not object features; SF (Surface Feature Similarity, also called Mere Appearance) shares relations and object features, but not higher-order structure; FOR (First-Order Relations or False Analogy) shares relations only. AN^{cm} denotes a *cross-mapped* analogy – it involves the same types of objects as the probe, but the types of corresponding objects are swapped.