Analogical Reasoning Within a Conceptual Hyperspace

Howard Goldowsky , **Vasanth Sarathy**
Tufts University
{howard.goldowsky, vasanth.sarathy}@tufts.edu

Abstract

We propose an approach to analogical inference that marries the neuro-symbolic computational power of complex-sampled hyperdimensional computing (HDC) with Conceptual Spaces Theory (CST), a promising theory of semantic meaning. CST sketches, at an abstract level, approaches to analogical inference that go beyond the standard predicate-based structure mapping theories. But it does not describe how such an approach can be operationalized. We propose a concrete HDC-based architecture that computes several types of analogy classified by CST. We present preliminary proof-of-concept experimental results within a toy domain and describe how it can perform category-based and property-based analogical reasoning.

1 Introduction and Motivation

"Analogies are partial similarities between different situations that support further inferences." [Gentner, 1998] The well-known formulation

$$A : B :: C : X \tag{1}$$

represents an analogy. Analogical reasoning entails many key aspects of human cognition and involves several key processes: *retrieval* (given C, find A and B), *mapping* (determine a structural correspondence between A and B to find X, by applying the correspondence to C), and *inference* (using A to advance the concept C). In this paper, we focus on the task of *mapping* – more specifically, the task of identifying a relationship between A and B, and then applying the identified relationship to characterize X. The particular challenge with mapping is that there is often a large number of potential relationships between A and B, and these relationships may themselves be compositional and graded in nature, as well as span the symbolic/sub-symbolic representational divide. Finding the *salient* relationships – the ones between A and B relevant to C – is a combinatorially hard problem.

Approaches to solving the mapping problem have been either connectionist or symbolic, and both types of models attempt to identify structural correspondence (graph isomorphisms) between concepts [Gayler and Levy, 2009]. Connectionist approaches such as ACME and DRAMA [Eliasmith and Thagard, 2001] are essentially "localist" [Page, 2000] in nature, which means concept representations/symbols, although connected within networks, remain localized to single nodes. Purely symbolic approaches, on the other hand (such as Structure Mapping Theory [Gentner, 1983; Crouse *et al.*, 2021]), explicitly incorporate the geometric graph structure using a predicate-based representation. Both types of analogy engines remain constrained by their representations.

Localist-connectionist approaches require a decomposition and sequential symbolic traversal of the source and target structures, substantially increasing their time complexity [Gayler and Levy, 2009]. Purely symbolic approaches struggle to compute analogies that need to isolate salient conceptual properties of objects [Gärdenfors and Osta-Vélez, 2023]. Salience requires a distance metric that just does not exist within a symbol-dominated space.

Neural networks offer more of a "distributed" connectionist approach. Unfortunately, however, their concept embeddings do not necessarily possess the structural and compositional aspects with which to perform analogical mapping. Moreover, neural network models fail to generate many types of analogies outside the distributions that characterize their training sets, and they fail to provide the underlying hierarchical structure to support analogical inference [Pavlus, 2021; Lewis and Mitchell, 2024]. A cognitive framework with the ability to simultaneously and seamlessly represent these so-far mutually exclusive connectionist and symbolic computational paradigms has been lacking [Lieto *et al.*, 2017].

More recently, researchers have explored hyperdimensional computing (HDC), synonymously known as vector-symbolic architecture (VSA)[1] as a paradigm for capturing a number of neurally plausible cognitive phenomena, including analogical mapping [Hersche *et al.*, 2023; Kanerva, 1997; Plate, 2003; Gayler, 2004; Blouw *et al.*, 2016]. HDC is a representational and inferential paradigm in which data structures can be represented with high-dimensional vectors, thus bridging the symbolic/subsymbolic gap.

The study of analogical inference using HDC is relatively new. Many open questions exist. In [Maudgalya *et al.*, 2020], for example, the salient aspects of the $A : B$ relationship were given, but it was assumed that the structure underlying A, B

[1] To ensure clarity, we will use the term hyperdimensional computing or HDC in the rest of this paper.

and C were binary. The question of how to identify more salient aspects of structure and allow for a more nuanced or graded characterization of concepts remains open.

In this paper, we begin answering this question with a crucial insight: that the HDC framework could operationalize the cognitive science theory of Conceptual Spaces [Gärdenfors, 2000]. For over a quarter century, Conceptual Spaces Theory (CST) has been an attractive guide for research in cognitive science [Douven and Gärdenfors, 2019], neuroscience [Bellmund *et al.*, 2018], and artificial intelligence [Zenker and Gärdenfors, 2015]. Its strength lies in its inclusive ability to model how sensory observations flow through an agent's initial connectionist layer of observation processing, through a layer of geometric concept space, and ultimately into a form of symbolic representation often used for reasoning – thus sharing many properties with HDC.

CST offers guidance on analogical mapping. Under CST, analogical inference requires a distance metric within a concept space. This requirement implies at least five necessary and simultaneously present capabilities of an agent [Osta-Vélez and Gärdenfors, 2022]:

1. It must continuously accept sensory observations in real-time and encode these signals into working memory;
2. It must afford an algebraic calculus over distance metrics within working memory;
3. It must afford a logical calculus in working memory;
4. It must cross-reference the same concepts during sensory observation, geometric processing, and symbolic processing;
5. It must provide interaction with long-term memory in an associative capacity.

Neither structure mapping nor neural architectures alone support these five requirements. We argue that HDC can support these requirements and provide concrete guidance for how concepts can be represented and analogized. We call this model a "conceptual hyperspace." CST offers a set of algorithms to solve several types of analogies [Osta-Vélez and Gärdenfors, 2022]. In this paper, we provide a proof-of-concept for how these algorithms can be implemented with HDC: how concepts can be encoded, how salient relationships can be identified, and how mapping can be performed. We also perform experiments in toy domains to illustrate the approach.

2 Background

In this section, we introduce both the Conceptual Spaces Theory and the Hyperdimensional Computing paradigm, which we will then combine in the next sections.

2.1 Conceptual Spaces

The Conceptual Spaces framework [Gärdenfors, 2000; Gärdenfors, 2014] adopts a prototype theory of concept representation [Murphy, 2002], which models concept prototypes as points within a geometric metric space [Bellmund *et al.*, 2018]. This space is constructed from one or more property dimensions of the represented concepts. Properties constitute direct sensory observations or hierarchical abstractions built from sensory observations. One or multiple integral properties constitute a domain. The integral color domain, for example, consists of three property dimensions along the positive Real number line: hue, brightness, and saturation. The weight domain consists of a single property dimension along the positive Real number line. Concepts are convex regions within the space. For concepts that span multiple domains, the domains can be correlated or weighted in various ways.[2]

Figure 1 illustrates the color domain used as a running example throughout this paper. The prototype for the color concept PURPLE, for example, falls at the location $HUE = 315°$, $SATURATION = 87$, and $BRIGHTNESS = 53$, within a cylindrical frame of reference. Three other color prototypes and their locations are also shown. The similarity between each of these colors can be modeled by their respective prototype distances from each other.

2.2 Hyperdimensional Computing (HDC)

HDC uses hypervectors for computation. Hypervectors are random high-dimensional (1,000+) vectors that combine hierarchically to produce new hypervectors of the same dimension. The hypervectors entail binary ($\{0, 1\}^d$), bipolar ($\{-1, +1\}^d$), real ($\mathcal{R}^d$), or complex ($\mathbb{C}^d$) samples. A tradeoff typically exists between processing speed and computational power for each sample type. We pick complex samples for our experiments, because they entail all other types and are the most computationally powerful [Plate, 2000]. As neuromorphic hardware achieves greater fidelity, however, this tradeoff may change. Researchers have suggested that complex hypervectors map to neural cell assemblies, where the phase of each sample represents the phase of their neuronal spikes [Orchard and Jarvis, 2023].

A crucial advantage of high-dimensionality is that the likelihood of two random hypervectors being orthogonal is extremely high. This penchant for orthogonality means hypervectors are capable of encoding scalars and representing bases within a latent representation space. Compositions can be represented in this space without much overlap, while remaining robust to noise. Additionally, through various operations, atomic concepts can be composed symbolically to define new abstract concepts. The HDC community has developed operations to manipulate these data structures, allowing the creation of a flexible symbolic algebra over the vector space. Several of these operations will be used to compute analogical inference. We represent concepts within a latent space using random complex hypervectors of length equal to 10^4.

Complex-sampled HDC uniquely affords mechanisms for artificial intelligence such as traditional data structures like trees and graphs [Kleyko *et al.*, 2022], navigation [Komer and Eliasmith, 2020], probabilistic modeling [Furlong and Eliasmith, 2024], reinforcement learning [Ni *et al.*, 2023], models of Grid and Place cells [Bartlett *et al.*, 2023; Dumont

[2] We adopt from Conceptual Spaces Theory only the definitions we need to show the efficacy of HDC applied to analogical inference. The interested reader can learn more about CST in [Gärdenfors, 2000; Gärdenfors, 2014].

and Eliasmith, 2020], etc. All of these mechanisms rely on complex-sampled HDC's ability to form kernels out of hypervectors [Frady *et al.*, 2022]. As we will show in the following sections, we model concepts within conceptual hyperspace as three-dimensional radial basis kernel functions. Section 3 outlines our general approach, and Section 4 focuses more on its details.

3 Proposed Approach: Conceptual Hyperspaces

In this section, we begin formally defining the problem setting and our proposed approach for solving the analogical mapping problem.

3.1 Problem Definition

We return to our compositional analogy of Eq. 1. Here, A and B are "source" concepts and C and X are "target" concepts [Gentner, 1998]. The mapping task is to find X that satisfies the underlying analogical relationship, namely that the relationship between the source concepts matches the relationship between the target concepts. Consider a domain D that is a vector subspace (of, say, $\mathbb{R}^n$) with k bases. Here, we can initially represent the concepts $A, B, C \leq D$ as themselves being subspaces within D, and therefore representable within D. For example, we can think of the "color" domain comprising $k = 3$ bases – hue, saturation and brightness – and the concept of red can be thought of as shades of "red" falling within the subspace of color that an agent might consider to be reddish. Because concepts are often vaguely defined, we leverage prototype theory and capture prototypes within concepts, which are meant to represent the concept more precisely. Here, prototypes are individual vectors or points in a domain, such as $\mathbf{p}_X \in D$. Thus, when computing analogies between concepts A, B, C, we use prototypes[3] to compute $\mathbf{p}_A : \mathbf{p}_B :: \mathbf{p}_C : \mathbf{p}_X$. Since prototypes are vectors in D, they have projections onto each of the bases of D, thereby representing the extent to which a prototype extends along that basis. Compositionally, this is a useful notion allowing us to capture how much hue, saturation and brightness the prototypical red has. The D, together with its bases, allow us to represent concepts as convex regions in D and prototypes as points in D. We can now define an analogical mapping problem in terms of conceptual spaces as follows:

Definition 1. ***Analogical Mapping Problem:*** *Given* $\mathbf{p}_A \in A, \mathbf{p}_B \in B$ *and* $\mathbf{p}_C \in C$*, determine* $\mathbf{p}_X \in X$*, including projections of* $\mathbf{p}_X$ *along each of* k *bases of domain* D *such that concepts* $A, B, C, X \leq D$*.*

An implicit prerequisite in Def. 1 is that the source and target concepts can all be represented within domain D using a set of salient k basis, which we will discuss in the next section together with an approach for solving this task.

3.2 Analogical Mapping Algorithm

Algorithms 1 to 4 describe the proposed approach to using HDC to solve the analogical mapping problem. Overall, the approach is to encode the prototypes into hyperspace, search for the analogical mapping in hyperspace, and then decode the hypervector to obtain the prototype of the desired target concept. More broadly, our approach has two general steps: (1) ensure saliency requirements are satisfied to construct the computation, and (2) perform the computation of the analogy.

Algorithm 1 Overall

```
Input: $\mathbf{p}_A \in A, \mathbf{p}_B \in B$ and $\mathbf{p}_C \in C$
Output: $\mathbf{p}_X \in X$
$\mathbf{a}, \mathbf{b}, \mathbf{c} \leftarrow \mathit{encode}([\mathbf{p}_A, \mathbf{p}_B, \mathbf{p}_C])$
$\mathbf{x} \leftarrow \mathit{find}([\mathbf{a}, \mathbf{b}, \mathbf{c}])$
$\mathbf{p}_X \leftarrow \mathit{decode}(\mathbf{x})$
return $\mathbf{p}_X$
```

Two assumptions are made in this algorithm:

1. That k bases for a domain[4] D has been obtained. Such a bases set captures shared properties of concepts A, B, C. In the case of our color example, all the concepts share a common set of three basis. In other property domains[5] this may require additional processing, the discussion of which is beyond the scope of this paper. See [Gärdenfors, 2000] for more insight.
2. That we already have prototypes $\mathbf{p}_A, \mathbf{p}_B, \mathbf{p}_C$ selected for the concepts. This may require retrieval from long-term memory.

To encode the prototypes, we propose using a Fractional Power Encoding, which allows us to capture gradations along each of our basis in the domain. Algorithm 2 shows us how to encode by first generating basis hypervectors for each of the k dimensions of the domain, D. These are randomly sampled complex hypervectors from a Gaussian distribution. To encode, we exponentiate these hypervectors with normalized prototype property values and then bind the k hypervectors together for each prototype. This operation produces a new hypervector of the same d dimensions and serves as a 3D radial basis function kernel in conceptual hyperspace.[6]

The approach taken to solve an analogical inference problem within conceptual space depends on the type of analogy being calculated. For analogies confined to object categories, CST recommends the Parallelogram model [Rumelhart and Abrahamson, 1973]. We implement this model in hyperspace, as shown in Algorithm 3, via binding operations with hypervectors.

$\mathbf{x}$ represents a latent point in k-dimensional conceptual hyperspace, implemented by a d-dimensional hypervector in bound superposition. But we don't know the exact location in k-space (within domain D) of the prototype for concept X, because this information is distributed within the hypervector and not human interpretable. The challenge of "factorizing" the components of the hypervector stems from the combina-

[3] We could just as well use non-prototypical points to compute, but we stick to prototypes in this section, because it conforms to our running example within the color domain.

[4] The term "domain" here refers to a vector subspace, not the CST-specific term.

[5] Here we refer to the CST-specific definition of "domain."

[6] by "zip" on line 9, we mean iterator of tuples where they are incremented in pairs in a lock-step manner

Algorithm 2 encode

```
1:  Input: $\mathbf{p}_A, \mathbf{p}_B, \mathbf{p}_C$
2:  Output: $\mathbf{F}$
3:  $k \leftarrow \dim(D)$
4:  $\mathbf{B} \leftarrow sampleBasesHypervectors(k)$
5:  $\hat{\mathbf{p}_A}, \hat{\mathbf{p}_B}, \hat{\mathbf{p}_C} \leftarrow normalize(\mathbf{p}_A, \mathbf{p}_B, \mathbf{p}_C)$
6:  initialize $\mathbf{F}$ to contain the encoded versions of $\hat{\mathbf{p}_A}, \hat{\mathbf{p}_B}, \hat{\mathbf{p}_C}$
7:  for $\hat{\mathbf{p}_i} \in [\hat{\mathbf{p}_A}, \hat{\mathbf{p}_B}, \hat{\mathbf{p}_C}]$ do
8:      initialize $\mathbf{f}_i$
9:      for $\mathbf{m}, j \in zip(\mathbf{B}, \hat{\mathbf{p}_i})$ do
10:         $\mathbf{m}^* \leftarrow exp(\mathbf{b}_i, j)$ {FPE exponentiation}
11:         $\mathbf{f}_i \leftarrow \mathbf{f}_i \circledast \mathbf{m}^*$ {binding operation}
12:     end for
13:     $\mathbf{F} \leftarrow append(\mathbf{f}_i)$
14: end for
15: return $\mathbf{F}$
```

Algorithm 3 find (using parallelogram method)

```
Input: $\mathbf{a}, \mathbf{b}, \mathbf{c}$
Output: $\mathbf{x}$
$\mathbf{x} \leftarrow (\mathbf{c} \circledast \mathbf{a}^{-1}) \circledast \mathbf{b}$
return $\mathbf{x}$
```

torial explosion that occurs from explicitly encoding all possible locations in hyperspace and comparing each one against $\mathbf{x}$. To address this issue, researchers have recently developed so-called resonator networks that enable factorization without such an exhaustive search [Frady *et al.*, 2020].

Algorithm 4 decode

```
Input: $\mathbf{x}, resolution$
Output: $\mathbf{p}_X$
$k \leftarrow \dim(D)$
$\mathbf{Q} \leftarrow makeCodebook(resolution, k)$
$\mathbf{p}_x \leftarrow resnet(\mathbf{x}, \mathbf{Q})$
return $\mathbf{p}_X$
```

A first step in factorization or decoding is in making a "code book" for each of the k bases of domain D. A code book, $\mathbf{Q}$, is a set of reference hypervectors built from each basis in the domain. If the domain is $\mathbb{R}^k$, then there are k codebooks. We discretize each dimension into discrete points, depending on our desired resolution. For example, when handling colors, we can normalize each of the k dimensions into a range of [-10,10] and then assign one point for each 0.5 increment, creating 41 locations along each of our 3 basis dimensions of the color domain. If we assign a hypervector to each of these locations, we obtain three code books, each containing 41 d-dimensional hypervectors. We then employ a resonator network to iteratively discover the underlying factors for $\mathbf{p}_X$.

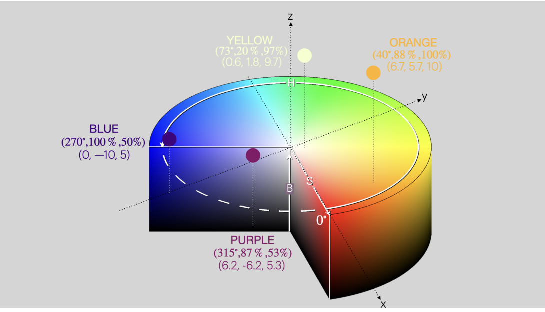

Figure 1: Color domain showing the location of prototype points for PURPLE, BLUE, ORANGE, and YELLOW.

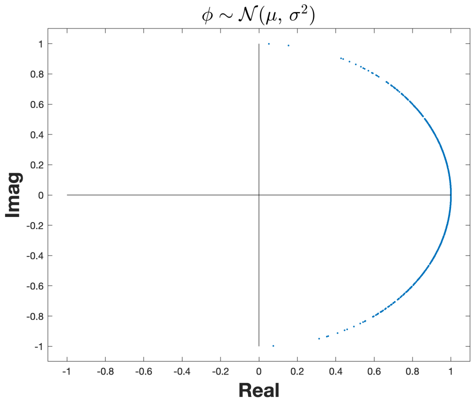

Figure 2: A complex-sampled hypervector of dimension $d = 1000$ samples initialized to a Gaussian phase distribution around the unit circle: $\phi \sim \mathcal{N}(\mu, \sigma^2)$, where here $\mu = 0$ and $\sigma = \frac{\pi}{7}$. We experimented with various σ values.

4 The Conceptual Hyperspace

4.1 Encoding Concepts in Hypervectors

To detail our approach using HDC for encoding conceptual hyperspace, we return to our running example of a composed analogy, this time associated with colors.

$$PURPLE : BLUE :: ORANGE : X$$

Figure 1 shows the geometry of this composed analogy within the color domain. The example starts by building three three-dimensional concept regions (for the operands purple, blue, and orange) within the color domain. Since the color domain is a three-dimensional space, we begin by initializing three *basis* hypervectors, which define the space (Algorithm 2, Line 4).

A hypervector, $\mathbf{x}$, in conceptual hyperspace initializes to a Gaussian phase distribution around the unit circle:

$$\mathbf{x} \in \mathbb{C}^d, \text{where sample } x_j = e^{i\phi_j},$$

with phases $\phi_j \sim \mathcal{N}(\mu, \sigma^2)$, where μ is the mean phase and σ the standard deviation in radians. See Figure 2. Bases hypervectors with large enough σ initialize to orthogonal.

The agent must encode the color properties into a concept space within its working memory. A visual preprocessing step extracts a concept's property values from either a specific object in the environment or from prototypes in long-term memory (steps beyond the scope of this work). For the rest of this paper, we will presume to use prototypes rather than observations. The three color property values then get encoded into three hypervectors, one for each basis, which represent prototype locations along their respective dimensions (for example, encoding the amount of hue, saturation, and brightness for each color). This encoding is done via a process called *Fractional Power Encoding* (Algorithm 2, Line10) [Komer *et al.*, 2019]. The prototype locations may also get stored into temporary variables to be used later. Both FPE and variable storage apply one of HDC's most fundamental operations, called *binding*.

The binding process is simultaneously a symbolic variable operation and a signal processing operation. Binding two HDC [7] hypervectors performs a circular convolution in Fourier space, represented by the operator $\circledast$, which is a sample-wise multiplication of the two hypervectors. Given two hypervectors $\mathbf{x} \in \mathbb{C}^d$ and $\mathbf{y} \in \mathbb{C}^d$, the result of their binding is

$$\mathbf{z} = \mathbf{x} \circledast \mathbf{y}. \tag{2}$$

Figure 3 illustrates how binding adds the phases of each respective pair of samples, given by $\phi_{z_i} = \phi_{x_i} + \phi_{y_i}$. $\mathbf{z}$ results in a hypervector orthogonal to both $\mathbf{x}$ and $\mathbf{y}$ if $\mathbf{x}$ and $\mathbf{y}$ start orthogonal to each other.

Unbinding is the inverse of binding and is performed by binding with the complex-conjugate of one operand. For example, given Equation 2,

$$\mathbf{x} = \mathbf{z} \circledast \mathbf{y}^{-1},$$

where $\mathbf{y}^{-1}$ is the complex-conjugate of $\mathbf{y}$, making the phase relationship between samples $\phi_{x_i} = \phi_{z_i} - \phi_{y_i}$. Unbinding is used to dereference a value from a variable. Binding is the base operation for FPE. FPE encodes a scalar into a hypervector. Given a hypervector, $\mathbf{x} \in \mathbb{C}^d$, and prototype location scalar, $p \in \mathbb{R}$, $\mathbf{z}$ encodes p via exponentiation:

$$\mathbf{z} = \mathbf{x}^p = \mathbf{x} \circledast \mathbf{x} \circledast ... \circledast \mathbf{x} \text{ (p times)}. \tag{3}$$

In Equation 3, the hypervector $\mathbf{x}$ appears p times. p can be any Real value [Komer and Eliasmith, 2020]. Figure 4 shows what happens to the phase of each sample of $\mathbf{x}$ after an FPE encoding. The initial phase gets multiplied by p, and the encoding process behaves as p sequential bindings, even if p is not an integer.

By binding together our three fractionally power encoded basis hypervectors (Algorithm 2, Line11) , we build a latent concept prototype within the color domain of conceptual hyperspace. For example, the final concept encoding for $PURPLE$ looks like this:

$$PURPLE = \mathbf{x}^{6.2} \circledast \mathbf{y}^{-6.2} \circledast \mathbf{z}^{5.3}, \tag{4}$$

where we establish Cartesian locations $p_1 = 6.2$, $p_2 = -6.2$, and $p_3 = 5.3$ for the "prototype location" based on the following pre-processing steps (Algorithm 2, Line 5), where $\beta = 10$ is a scaling constant, and we use the values $HUE = 315°$, $SATURATION = 87$, and $BRIGHTNESS = 53$:

$$p_1 = cos(HUE) \cdot SATURATION/\beta,$$
$$p_2 = sin(HUE) \cdot SATURATION/\beta,$$
$$p_3 = BRIGHTNESS/\beta.$$

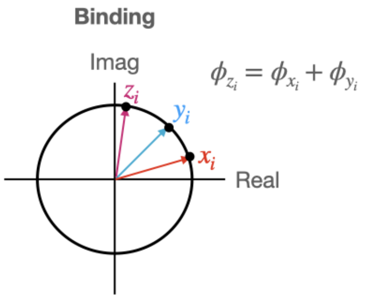

Figure 3: Result of one sample-wise binding operation between two complex hypervectors.

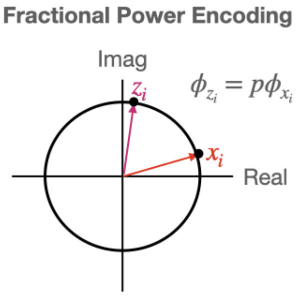

Figure 4: Result of fractional power encoding for one sample within a complex hypervector.

Because we initialized each basis to a Gaussian distribution, our concept in hyperspace, shown in Equation 4, retains the computational properties of a latent three-dimensional radial basis kernel embedded within a single d-dimensional complex-sampled hypervector.

4.2 Analogical Mapping Algorithm for Category-based analogies

Researchers have classified the semantic relationships present in analogy (e.g., the colon in $A : B$) into many different types [Collins and Burstein, 1987; Osta-Vélez and Gärdenfors, 2022] that can broadly fall into "category-based" and "property-based." There are other classifications like event-based, part-whole, and causal; but for this paper we focus on just the first two.

Recall our example from above.

$$PURPLE : BLUE :: ORANGE : X$$

We solve for X, after first guaranteeing A, B, and C satisfy the algorithm-specific pre-processing requirements. For this algorithm, there are two pre-processing requirements:

[7] Throughout this paper we simply refer to complex-sampled hypervectors as "HDC" hypervectors. A synonymous name for hypervectors with complex sample type is Fourier Holographic Reduced Representation (FHRR).

1. Check that concepts A, B, and, C are labeled in long-term memory as a common superordinate category. Labelling and label checking can be accomplished through a variety of methods, including Kanerva's distributed record [Kanerva, 1997], graphs, tree structures, or state machines, [Kleyko *et al.*, 2022]. Each of these symbolic data structures can be built with HDC and efficiently store properties of prototypes within long-term memory or perform other symbolic logic. The detailed implementation of this long-term memory is beyond the scope of this paper and a work in progress.
2. Find the dimensions over which each A and B have common properties. This step can also be solved using HDC methods and is beyond the scope of this paper.

In this example, all operands are already within the color category (which also happens to be a single domain), so no algorithm-specific pre-processing needs to happen.

Since all concepts are now known to be within the same category, it is straightforward to find the location for the prototype of X. We apply the Parallelogram model (Algorithm 3, Line 3) [Rumelhart and Abrahamson, 1973]:

$$\mathbf{p}_X = \mathbf{p}_C - \mathbf{p}_A + \mathbf{p}_B \tag{5}$$

Encoding these locations into hypervectors by applying Equation 5 within a conceptual hyperspace and applying the appropriate HDC operations, gives us

$$\mathbf{x} = (\mathbf{o} \circledast \mathbf{p}^{-1}) \circledast \mathbf{b}, \tag{6}$$

where we have abbreviated the colors to the first letter of their names. It's worth noting a few things at this point. We know from the statistical literature that the convolution of two Gaussian distributions adds their means [Jaynes, 2003]; and because each complex hypervector lives within the Fourier domain, the sample-wise multiplication of the binding operator performs a convolution. All elements of the operand hypervectors compute within bound superposition of three encoded basis per concept. The result hypervector, $\mathbf{x}$, remains entangled within bound superposition and remains random. Nevertheless, the phase changes induced by the algebraic operations used to compute the analogy maintain their statistical fidelity within the answer.

While we have access to the hypervector elements that comprise our answer's prototype, we do not yet know its encoded latent Cartesian location (p_1, p_2, p_3). This location is deeply encoded into the aforementioned bound superposition. To find an estimate of the encoded location within each basis hypervector, we employ an HDC resonator network [Frady *et al.*, 2020] whose job is to find factors like these within bound superposition. Most mathematical details about how resonator networks work are beyond the scope of this paper. What is important, however, is this: If we build a code book that contains all possible encoded hypervector factors as a reference, then the resonator network can quickly factor the bound superposition. The network can actually take advantage of statistical properties afforded by this bound superposition rather than get hindered by them. The number of iterations needed to find factors scales linearly $\mathcal{O}(M)$, where M is the search space size. Kent's PhD thesis [Kent, 2020] shows how resonator networks factor bound superposition problems faster than any other known method.

The resonator network "collapses" the bound superposition from a learned or innate range of typical property values, resulting in an estimate for the prototype location, $\mathbf{p}_X$, when it returns references to the set of factors. In this experiment, our resonator network took two iterations to find the correct encoded basis factors for an estimate of $\mathbf{p}_{YELLOW}$ (Algorithm 4, Line 5). Our model's code book included 41 different possible encoded basis hypervectors for each dimension, spanning each basis' range of possible normalized values from -10 to $+10$, at increments of 0.5. In our experiment, the resonator network returned the location $p_1 = 0.5$, $p_2 = 2.0$, and $p_3 = 9.5$, which equates to un-normalized values of $HUE = 75°$, $SATURATION = 75$, and $BRIGHTNESS = 95$. This is as close as it could theoretically approach the prototype location for YELLOW, given the 0.5 normalized resolution of our code books.

Solving a Property-based Analogy

This type of analogy compares a semantic relationship between categories and properties. Recall the general representation for analogy:

$$A : B :: C : X$$

A property-based example provided by [Osta-Vélez and Gärdenfors, 2022] is

$$APPLE : RED :: BANANA : X.$$

The steps to solving this composed analogy are the following:

1. Identify the salient property dimensions that correspond to concept or property B. In this example, this would be the bases dimensions associated with RED (i.e. the color domain).
2. Find the location within the identified salient dimensions closest to prototype location C. In this example, this would be the location along the hue, saturation, and brightness bases associated with concept $BANANA$ (i.e. $YELLOW$).

In general, the agent will often need to search for salient properties. This search requires a distance metric. For example, in this problem the agent needs to search for what bases constitute RED. The agent stores bases in long-term memory, because these bases hypervectors are needed to build code books. To determine the salient bases associated with RED, the agent computes a distance metric between all domain bases hypervectors stored in memory and the domain basis hypervector used to encode RED. The closest domain hypervector in memory to RED's domain hypervector is the salient one. By using the properties of HDC Gaussian kernels [Frady *et al.*, 2022], a simple dot-product between hypervectors measures their relative distance from each other within conceptual hyperspace. [8] This brings us to the final part of our introduction to HDC. Similarity between complex-sampled unitary

[8] Each concept or basis hypervector is also a Gaussian kernel within a formal Reproducing Kernel Hilbert Space (RKHS). This fact affords the dot-product as a distance metric.

hypervectors is defined as the mean of the cosine of angle differences between corresponding samples [Vine and Bruza, 2010]. This definition equates to the inner product between two complex hypervectors. Given $\mathbf{x} \in \mathbb{C}^d, \mathbf{y} \in \mathbb{C}^d$, the relative distance between $\mathbf{x}$ and $\mathbf{y}$ is $similarity(\mathbf{y}, \mathbf{x}) = \mathbf{y}^{-1} \cdot \mathbf{x}$. There could be myriad types of search depending on analogy type. This example shows only one type of search. But given the algebraic, symbolic, and metric operations afforded by HDC, many different types of search are possible.

5 Discussion

The Importance of HDC as a Modeling Tool

Why should we care about using a neurally-plausible analogy engine? Why can't we just perform our analogical number crunching with a base-10 number system? The traditional approach would certainly be more straightforward.

The reason is because the more traditional approaches are stuck at Marr's algorithmic level [Marr, 1982]. Constraining our representations to be neurally-plausible adds scientific value to our model. If we can achieve analogical inference by using a model between Marr's algorithmic and implementation levels, which we propose here, then we've reduced the search space for an algorithm-plus-implementation towards human-level intelligence.

At the engineering level, we concede that for the toy model presented here, a neurally-plausible conceptual hyperspace seems more complex than a traditional conceptual space using traditional vectors. But the brain does not use a von Neumann architecture. As we scale this model to include more cognitive functionality that would require more resources, we could potentially build it within a spiking neural network or other power-efficient neuromorphic hardware. Constraining our AI models to neural-plausibility affords hope for human-level cognitive efficacy at scale.

The Origin of Property Dimensions

The origin of property dimensions do not yet seem to be deeply grounded in theory. Therefore, we plan to pursue the best ways to model these. In this paper, we treat property dimensions as orthogonal bases within a conceptual hyperspace, which act as building blocks for intrinsic domains. This conveniently works out from both a signal processing and cognitive science perspective. The signal processing theory literally requires orthogonality to afford kernel construction. If it turned out that the brain built a hierarchy of property dimensions from a finite set of orthogonal basis dimensions then this would also be elegantly satisfying for cognitive science. [Wierzbicka, 1996] provides a finite set of semantic primitives over all languages, which seems like a good place to start building such a conceptual hyperspace model grounded on a finite set of atomic property dimensions. Through HDC operations, we could then generate hierarchical property dimensions on the fly that correspond with analogy algorithm requirements.

Neuroscience experiments that take place in fMRI machines show evidence that the Entorhinal cortex-hippocampus system quickly builds highly specific hierarchical concept regions. For example, the regions reported in [Bellmund *et al.*, 2018] are "Neck length" versus "Leg length." We can possibly use HDC to teach an agent how to build the appropriate property dimensions. HexSSP, which are HDC kernel hypervectors that model Place and Grid cells, introduced by [Bartlett *et al.*, 2023], may afford learnable resolution sizes for property dimensions. We would like to integrate this learning of property dimensions with efficient resonator network code book design, as well.

Exploration of Kernel Functions

Playing with the bandwidth on our radial basis kernel functions allows us to shape the similarity regions within conceptual hyperspace. Given the flexibility of HDC, radial basis functions are not the only kernel function at our disposal, however. Playing with different kernel types and their respective parameters afford myriad concept region shapes [Frady *et al.*, 2022] for a variety of semantic similarity. Most kernel similarities adhere to the (arguably) soft CST requirement of maintaining domain convexity [Hernández-Conde, 2017]. But if we'd like to model non-convexity, then that's possible too. It's possible (albeit inelegant) to symbolically label any concept by binding to it an additional hypervector, which can be stripped off before signal processing begins. [Balkenius and Gärdenfors, 2016] discuss radial basis function network models for learning within the context of motor movement, reasoning, and other applications. The agent can potentially learn the appropriate kernels to use along with their respective parameters.

Conceptual Hyperspace as a Generative Model

HDC allows us to build novel concept regions in a generative manner. If the answer to an analogy problem generates a concept location that does not, say, already have a linguistic or symbolic label within a minimal distance of an existing prototype, then the agent has an opportunity to be creative. The way we've modeled a concept space with kernel functions does not require the entire space to be tiled with concepts. Any time a new concept is produced, the agent can use the kernel properties of HDC to quickly find the new concept's distance to all existing prototypes and decide what it wants to do – create a new prototype, merge with an existing concept, implementing a sort of exemplar model via additive HDC superposition (an HDC operation not covered in this paper), or do nothing, allowing the agent to retain a more holistic conceptualization. In this manner, HDC has the potential to extend CST into a generative framework not constrained to a rigid theory of prototypes.

We know that analogy is the "Fuel and Fire of Thinking" as Douglas Hofstadter [Hofstadter and Sander, 2013] likes to say. Therefore we know that analogical inference likely plays a role in all forms of cognition. This is a core principle that will guide us as we move forward with this research.

References

[Balkenius and Gärdenfors, 2016] Christian Balkenius and Peter Gärdenfors. Spaces in the brain: From neurons to meanings. *Frontiers in Psychology*, 7, 2016.

[Bartlett *et al.*, 2023] Madeleine Bartlett, Kathryn Simone, Dumont Nicole Sandra-Yaffa, P. Michael Furlong, Chris

Eliasmith, and Terrence C. Stewart. Improving reinforcement learning with biologically motivated continuous state representations. *Proceedings of the 21st International Conference on Cognitive Modeling*, 2023.

[Bellmund *et al.*, 2018] Jacob L. S. Bellmund, Peter Gärdenfors, Edvard I. Moser, and Christian F. Doeller. Navigating cognition: Spatial codes for human thinking. *Science*, 362(6415):eaat6766, 2018.

[Blouw *et al.*, 2016] Peter Blouw, Eugene Solodkin, Paul Thagard, and Chris Eliasmith. Concepts as semantic pointers: A framework and computational model. *Cognitive Science*, 40(5):1128–1162, 2016.

[Collins and Burstein, 1987] Allan Collins and Mark Burstein. *A Framework for a Theory of Mapping:*. Fort Belvoir, VA, December 1987.

[Crouse *et al.*, 2021] Maxwell Crouse, Constantine Nakos, Ibrahim Abdelaziz, and Ken Forbus. Neural analogical matching. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(1):809–817, May 2021.

[Douven and Gärdenfors, 2019] Igor Douven and Peter Gärdenfors. What are natural concepts? a design perspective. *Mind and Language*, (3):313–334, 2019.

[Dumont and Eliasmith, 2020] Nicole Sandra-Yaffa Dumont and Chris Eliasmith. Accurate representation for spatial cognition using grid cells. In *Annual Meeting of the Cognitive Science Society*, 2020.

[Eliasmith and Thagard, 2001] Chris Eliasmith and Paul Thagard. Integrating structure and meaning: A distributed model of analogical mapping. *Cognitive Science*, 25(2):245–286, 2001.

[Frady *et al.*, 2020] Edward Paxon Frady, Spencer J. Kent, Bruno A. Olshausen, and Friedrich T. Sommer. Resonator networks, 1: An efficient solution for factoring high-dimensional, distributed representations of data structures. *Neural Computation*, 32:2311–2331, 2020.

[Frady *et al.*, 2022] E. Paxon Frady, Denis Kleyko, Christopher J. Kymn, Bruno A. Olshausen, and Friedrich T. Sommer. Computing on functions using randomized vector representations (in brief). In *Proceedings of the 2022 Annual Neuro-Inspired Computational Elements Conference*, NICE '22, page 115–122, New York, NY, USA, 2022. Association for Computing Machinery.

[Furlong and Eliasmith, 2024] P. Michael Furlong and Chris Eliasmith. Bridging cognitive architectures and generative models with vector symbolic algebras. *Proceedings of the AAAI Symposium Series*, 2:262–271, 01 2024.

[Gärdenfors and Osta-Vélez, 2023] Peter Gärdenfors and Matías Osta-Vélez. Reasoning with concepts: A unifying framework. *Minds and Machines*, 1(3):451–485, 2023.

[Gayler and Levy, 2009] Ross Gayler and Simon Levy. A distributed basis for analogical mapping. January 2009.

[Gayler, 2004] Ross W. Gayler. Vector symbolic architectures answer jackendoff's challenges for cognitive neuroscience. (arXiv:cs/0412059), December 2004. arXiv:cs/0412059.

[Gentner, 1983] Dedre Gentner. Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7(2):155–170, 1983.

[Gentner, 1998] Dedre Gentner. *Analogy*, page 107–113. Oxford: Blackwell, 1998.

[Gärdenfors, 2000] Peter Gärdenfors. *Conceptual Spaces: The Geometry of Thought*. MIT press, 2000.

[Gärdenfors, 2014] Peter Gärdenfors. *The Geometry of Meaning*. MIT press, 2014.

[Hernández-Conde, 2017] José V. Hernández-Conde. A case against convexity in conceptual spaces. *Synthese*, 194(10):4011–4037, 2017.

[Hersche *et al.*, 2023] Michael Hersche, Mustafa Zeqiri, Luca Benini, Abu Sebastian, and Abbas Rahimi. A neuro-vector-symbolic architecture for solving raven's progressive matrices. *Nature Machine Intelligence*, 5:363–375, 2023.

[Hofstadter and Sander, 2013] Douglas Hofstadter and Emanuel Sander. *Surfaces and Essances*. Basic Books, 2013.

[Jaynes, 2003] E. T. Jaynes. *Probability theory: The logic of science*. Cambridge University Press, Cambridge, 2003.

[Kanerva, 1997] Pentti Kanerva. Fully distributed representation. 1997.

[Kent, 2020] Spenser Kent. *Multiplicative Coding and Factorization in Vector Symbolic Models of Cognition*. PhD thesis, University of California Berkeley, 2020.

[Kleyko *et al.*, 2022] Denis Kleyko, Mike Davies, Edward Frady, Pentti Kanerva, Spencer Kent, Bruno Olshausen, Evgeny Osipov, J.M. Rabaey, Dmitri Rachkovskij, Abbas Rahimi, and Friedrich Sommer. Vector symbolic architectures as a computing framework for emerging hardware. *Proceedings of the IEEE*, 110:1538–1571, 10 2022.

[Komer and Eliasmith, 2020] Brent Komer and Chris Eliasmith. Efficient navigation using a scalable, biologically inspired spatial representation. In *Annual Meeting of the Cognitive Science Society*, 2020.

[Komer *et al.*, 2019] Brent Komer, Terrence C. Stewart, Aaron R. Voelker, and Chris Eliasmith. A neural representation of continuous space using fractional binding. In *Annual Meeting of the Cognitive Science Society*, 2019.

[Lewis and Mitchell, 2024] Martha Lewis and Melanie Mitchell. Using counterfactual tasks to evaluate the generality of analogical reasoning in large language models. *ArXiv*, abs/2402.08955, 2024.

[Lieto *et al.*, 2017] Antonio Lieto, Antonio Chella, and Marcello Frixione. Conceptual spaces for cognitive architectures: A lingua franca for different levels of representation. *Biologically Inspired Cognitive Architectures*, 19:1–9, 2017.

[Marr, 1982] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt and Co., Inc., USA, 1982.

[Maudgalya *et al.*, 2020] Navneedh Maudgalya, Bruno Olshausen, and Spencer Kent. Vector symbolic visual analogies. (arXiv:cs/0412059), 2020. arXiv:cs/0412059.

[Murphy, 2002] Gregory L. Murphy. *The Big Book of Concepts*. MIT press, 2002.

[Ni *et al.*, 2023] Yang Ni, Danny Abraham, Mariam Issa, Yeseong Kim, Pietro Mercati, and Mohsen Imani. Efficient off-policy reinforcement learning via brain-inspired computing. In *Proceedings of the Great Lakes Symposium on VLSI 2023*, GLSVLSI '23, page 449–453, New York, NY, USA, 2023. Association for Computing Machinery.

[Orchard and Jarvis, 2023] Jeff Orchard and Russell Jarvis. Hyperdimensional computing with spiking-phasor neurons. In *Proceedings of the 2023 International Conference on Neuromorphic Systems*, ICONS '23, New York, NY, USA, 2023. Association for Computing Machinery.

[Osta-Vélez and Gärdenfors, 2022] Matías Osta-Vélez and Peter Gärdenfors. Analogy as a search procedure: a dimensional view. *Journal of Experimental & Theoretical Artificial Intelligence*, 0(0):1–20, 2022.

[Page, 2000] Mike Page. Connectionist modelling in psychology: A localist manifesto. *Behavioral and Brain Sciences*, 23(4):443–467, August 2000.

[Pavlus, 2021] John Pavlus. The computer scientist training ai to think with analogies. *Scientific American*, August 2021.

[Plate, 2000] Tony A. Plate. Analogy retrieval and processing with distributed vector representations. *Expert Systems*, 17(1):29–40, 2000.

[Plate, 2003] Tony Plate. *Holographic Reduced Representation: Distributed Representation for Cognitive Structures*. January 2003.

[Rumelhart and Abrahamson, 1973] David E. Rumelhart and Adele A. Abrahamson. A model for analogical reasoning. *Cognitive Psychology*, 5:1–28, 1973.

[Vine and Bruza, 2010] Lance De Vine and Peter Bruza. Semantic oscillations: Encoding context and structure in complex valued holographic vectors. In *AAAI Fall Symposium: Quantum Informatics for Cognitive, Social, and Semantic Processes*, 2010.

[Wierzbicka, 1996] Anna Wierzbicka. *Semantics: Primes and Universals*. Oxford University Press, 1996.

[Zenker and Gärdenfors, 2015] Frank Zenker and Peter Gärdenfors. *Applications of Conceptual Spaces*. Springer, 2015.