Vector Symbolic Sub-objects Classifiers as Manifold Analogues

Renato Faraone*†, Peter Sutor*, Cornelia Fermüller, and Yiannis Aloimonos

2024 International Joint Conference on Neural Networks (IJCNN) | 979-8-3503-5931-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/IJCNN60899.2024.10651219

***Abstract*—Vector Symbolic Architectures (VSAs) generally consist of a hyper-algebra that is defined on points in a space of vectors. However, such a space is typically defined in usual ways, such as Euclidean Spaces for real vectors, Hamming Spaces for binary vectors, and so on. In any empirical setting, such as Artificial Intelligence, Machine Learning, Data Science, etc., observations in these spaces tend to produce subspaces of these broader spaces. As a result, these empirically-derived subspaces enable VSAs to make predictions through how severely the subspaces deviate from what is expected. Thus, it is desirable to be able to understand how such subspaces behave. As an analogy, when one observes terrain, they should find good "roads and bridges" to navigate that terrain. In Category Theory, the idea of Topos describes how such roads and bridges should be placed. In this paper, we explore the relationship between VSAs and Topoi. Namely, we show how a Topos-like representation can be constructed from empirical observations of vectors and demonstrate this on a practical example using Hyperdimensional Computing (HDC) on dense binary hypervectors. Our results indicate that a Topos can be effectively constructed for a dataset and the resulting space of vectors is biased to reflect the compositional aspects of the data. We conclude that such a Topos can be used to better guide the construction of VSAs for downstream tasks, in lieu of the original space of vectors, much like manifolds in Topology.**

***Index Terms*—VSA, Vector Symbolic Architectures, HDC, Hyperdimensional Computing, Topos Theory, Topoi, Category Theory, Cartesian Closed Categories, Hypervector**

I. Introduction

Vector Symbolic Architectures (VSAs) are equipped with algebraic operations on high-dimensional spaces of vectors. More specifically, in Hyperdimensional Computing (HDC), properties of these high dimensional spaces are leveraged to perform computational tasks, encode data, and train Artificial Intelligence (AI) models. In general, no assumptions are made on these spaces of vectors, and randomness is utilized to bind symbolic meaning into these spaces. Using the VSA algebra to compute in this space gives rise to emergent topological properties of the space that are naturally derived from the empirical data. However, the hypervectors inside these topologies are used as-is, without actually trying to learn the structure these hypervectors reside in. As an analogous example, if you are trying to understand the topology of a table, empirical observations of points on the surface of the table should eventually make it obvious that the surface is better approximated by a plane. Such external reasoning on topologies of hypervector subspaces are not typically performed in VSA/HDC. From a typical VSA's point of view, points above the table are just as likely to occur - even though the empirical data would suggest this can't occur.

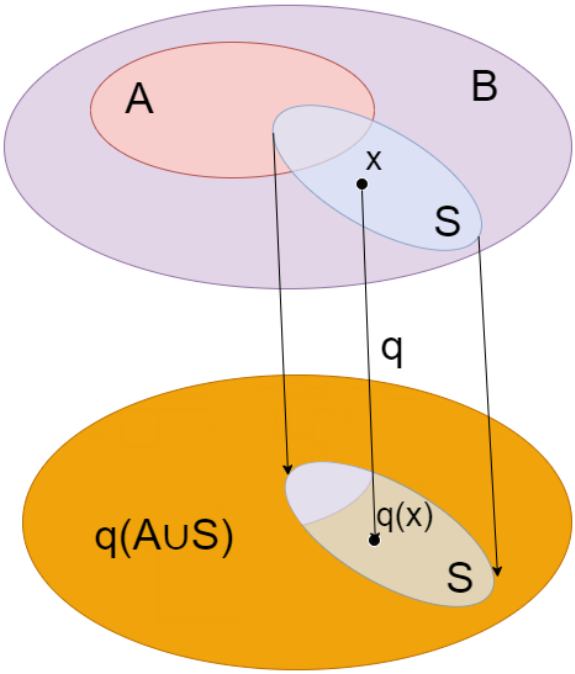


Fig. 1. Let A be an open subset of B. Given a point $x \in B$, choose a neighborhood S around x on which q is a local homeomorphism. Unlike the usual set theoretical classification options of $x \in A$ or $x \notin A$, the topological context affords us the ability to classify according to how *close* x is to A. This is the heart of using sub-object classification in a topological view of the vectors in a VSA. The Topos enables generalizing the compositional transformations shown here to alter the usual metric we use to measure distance between vectors. So, we can alter the topology of our space to suit our empirical observations; placing roads and bridges where needed.

The observation of these properties can be linked to meaningful correlations between the original data samples. Among other applications, this could offer a variety of custom strategies for problems in Explainable AI.

In this paper, we attack the problem of approximating hyperdimensional topologies extracted from empirical data through Topos Theory. By design, Topoi are well-suited towards understanding the compositionality and networking of an evolving topology. We show how a Topos can be constructed from empirically-derived hypervectors algorithmically and demonstrate how such a structure differs from the standard space of vectors. Using genetically constructed hypervectors, we employ the Topos on a practical example of a biased space of

* These authors contributed equally.
† Corresponding author: renatofaraone@gmail.com
A. Faraone is with the Department of Mathematics in University of Parma, Parma, PM, Italy.
P. Sutor, C. Fermüller, and Yiannis Aloimonos are with the Department of Computer Science in University of Maryland, College Park, MD, USA.

Algorithm 1 Topos Generation

procedure TOPOSGENERATOR(H_{set})
  $T \leftarrow$ TOPOS(H_{set})
  $T \leftarrow$ LOCALNEIGHBORHOODS(T)
  return T
end procedure

procedure TOPOS($H_{set}, localRoot = root$)
  if $|H_{set}| == 1$ **then**
    $localRoot.h \leftarrow h \in H_{set}$
    return
  end if
  $P \leftarrow$ SEPARATION($H_{set}, localRoot$)
  for i from 1 to $|P|$ **do**
    $child \leftarrow$ MAKECHILD($localRoot$)
    TOPOS($P[i], child$)
  end for
  return
end procedure

procedure LOCALNEIGHBORHOODS(T)
  for i from HEIGHT(T) - 1 to 1 **do**
    $N_{(i)} \leftarrow$ LEVELSET(i, M)
    for $N \in N_{(i)}$ **do**
      PUSHOUT(N)
    end for
  end for
end procedure

ancestor and descendant hypervectors. Our results indicate that this can be effectively constructed for a dataset of hypervectors and the resulting structure accurately reflects the compositional nature of the source data - in this case, genetic lineages of hypervectors. We conclude that Topoi can be used to better guide the construction of VSAs within their internal algebras for use in downstream tasks, in lieu of the original space of vectors. This is reminiscent of the relationship between Manifolds in Topology, and Neural Networks in Machine Learning.

II. Background

We assume the reader is familiar with VSAs and HDC. Namely, the notion of *binding*, *bundling*. Primarily, our work will focus on the bundling aspect itself, which usually serves as the "learning" operation. The practical example of a Topos in the Results section will assume dense binary hypervectors. However, the algorithm for Topos creation is not specific to this, and can be done on any hyperdimensional space. Indeed, the algorithm can be done with a broad class of functions, that can be task dependent. As such, it suffices for the reader to be familiar with the use of bitwise XOR for binding and for measuring Hamming Distance (the count of 1 bits after XOR), and the Consensus Sum over hypervectors, which is simply the hypervector formed by taking the most occurring bit value across the terms in each component. For those unfamiliar with such formulations of HDC, [1] is an excellent starting reference for what is presented in this paper. Since our work can be extended to other formulations of VSA/HDC paradigms, [2] is a great survey for reference. Instead, we provide more background on Topos Theory, which is more likely to be unfamiliar to the reader. This is provided in Appendix A. At a high-level, the construction provided in the pseudo code describes the behaviour of a Sub-object classifier, which is the key component of a Topos. This provides a compositional interpretation of the entire space of hypervectors. The main definitions are that of Sub-Object Classifier (A.10) and of Topos (A.11). Both depend to the unifying concept of (co)limit, that also offers the link with Hyperstructures. In the appendix, the reader is given some hopefully motivating examples of these constructions.

In terms of related works, the usage of Topos Theory appears to be quite novel in VSAs/HDC. Mostly, usages are indirect, such as the proscriptive Sub-Object Classification in the Life-Long Learning approach presented in [3] - we differ by generating the Topos from empirical samples of the space. Some works are in the same vein, however, and will described here for further reading. In [4], the authors similarly explore the usage of network topologies in HDC. By cloning hypervectors, the authors were able to quickly perform dynamic programming in HDC-based AI for exploratory robots. The authors analyzed the effects of differing network topologies on this dynamic programming. Another relevant work can be found in [5], in which the authors attempt to take word embedding models (which are a type of VSA-like structure themselves) and learn their underlying topology, in order to isolate context from complex phrases and sentences. While the authors themselves do not reference the term Topos, the compositional algorithm they present seems to approximate Sub-Object classification. Indeed, there are striking similarities to that work and what is shown in Fig. 1. Outside of direct applications of VSAs/HDC to Category Theory, it is also well-understood how Category Theory applies to compositional distributive semantic spaces, such as in Linguistics [6]. While not strictly linguistic in nature, the self-encoding aspects of HDC and Vector-Symbolic Representations tend to manifest as distributional semantics. Furthermore, the genetic working problem we present here for analyzing the performance of Topoi is well-founded in evolutionary biology, genetics, and their tie-ins to Category Theory [7], [8] and Neural Networks [9].

III. Methodology

In this section, we describe our algorithm for Topos generation, describe our practical example involving genetically-derived hypervectors, and our experimental methodology.

A. Topos Generation

Broadly speaking, our Topos generation can be described as a recursive top-down sub-object classification. An example of this type of algorithm can be seen in the TOPOS function in Alg. 1. Given a predefined SEPARATION function, we recursively cluster a dataset of hypervectors, called H_{set}, into child nodes of a tree-like data structure, according to a

partitioning P of H_{set} that partitions our dataset into Sub-Object classes. The separation function itself is task-specific. However, we will present and use a general separation function in our experiments, shown in Alg. 2.

Since we are also interested in creating a meaningful composition of local neighborhoods that behave like general space of vectors, a key part of Alg. 1 is LOCALNEIGHBORHOODS, which takes the level set of our tree from the bottom up, and attempts to collapse Sub-Object classes into well-behaved local neighborhoods through application of a function PUSHOUT on each node of the tree. Once again, like the separation function, the pushout function is predefined and specific to the space of vectors and/or is task-specific. We will present and use a general pushout function in our experiments, shown in Alg. 2. These local neighborhoods "capture" specific classes of hypervectors, and are the terminal points of the tree. For example, Fig. 2 represents these as a parenthesization of a Topos, with 2 Sub-Object classes per node. The complete tree has been collapsed through the pushout function to group the hypervectors into classes. The number here indicates how many hypervectors in our dataset fall into that class.

The usage of the Topos can be summarized via utility functions, such as those in Alg. 3. Namely, the primary usage of the Topos is to classify hypervectors into the correct Sub-Object collection, i.e. a path to the correct node in the tree describing the Topos. This is done via the FIND function. Once classified, a local metric can be made to measure the distance between the two Sub-Object collections, such as the LOCALMETRIC function described in Alg. 3. We can define a geodesic-like distance between two specific hypervectors by using both the local metric function in the Topos and the local neighborhood distance function, which is typically the default distance metric for you space of hypervectors (e.g., Euclidean Distance, Cosine Distance, Hamming Distance, etc.). Once again, these utility functions are both specific to the space of vectors and the task at hand. We present and use general versions of these utility functions in Alg.3.

B. Practical Working Example

In order to show how our construction would be used in a practical VSA/HDC setting, we present now the working example to use throughout the remainder of this paper. The goal is to demonstrate the usage of a Topos and to evaluate its differences from the original space the VSA was defined on.

We consider the practical problem of understanding genetic lineage in ancestors and descendants. This is conceptualized as a hypervector representing DNA, which is then passed on to descendants. Specifically, we consider dense binary hypervectors for this problem, for the sake of simplicity. Given a random hypervector a, its descendant $d(a) = \text{FLIP}(a + b, \epsilon)$, where b is another random hypervector, and FLIP is a function that random flips the bits in the positions where a 1 is specified in ϵ. Essentially, FLIP adds noise to the result, which can be specified by percent of noise parameter that is allowed. This parameter controls how chaotic the DNA of the descendant is.

Furthermore, we will use a working example of such genetic lineage throughout the rest of the paper, in order to

Algorithm 2 A General Separation/Pushout Algorithm

procedure SEPARATION($H_{set}, localRoot$)
  $C_{set} \leftarrow \emptyset$
  $F_{set} \leftarrow \emptyset$
  $H^{(local)} \leftarrow$ AVERAGEHYPERVECTOR(H_{set})
  $localRoot.h = H^{(local)}$
  $upper \leftarrow |H^{(local)}|$
  $lower \leftarrow 0$
  $ball \leftarrow \lfloor \frac{upper+lower}{2} \rfloor$
  repeat
    $prev \leftarrow ball$
    $(C_{set}, F_{set}) \leftarrow$ PARTITION($H_{set}, H^{(local)}, ball$)
    if $|C_{set}| < |F_{set}| - 1$ **then**
      $temp \leftarrow \lfloor \frac{upper+ball}{2} \rfloor$
      $upper \leftarrow ball$
      $ball \leftarrow temp$
    else if $|C_{set}| > |F_{set}| + 1$ **then**
      $temp \leftarrow \lfloor \frac{lower+ball}{2} \rfloor$
      $lower \leftarrow ball$
      $ball \leftarrow temp$
    end if
    if $prev == ball$ **then**
      break
    end if
  until $|C_{set}| == |F_{set}| \pm 1$
  return $[C_{set}, F_{set}]$
end procedure

procedure PUSHOUT(N)
  $H_{set} = \emptyset$
  for $child \in N.children$ **do**
    if ISLEAF($child$) == False **then**
      return
    end if
    $H_{set} \leftarrow H_{set} \cup \{child.h\}$
  end for
  $H \leftarrow$ AVERAGEHYPERVECTOR(H_{set})
  $D \leftarrow \text{H}(H_{set}, H)$
  $ball \leftarrow \text{MIN}(D)$
  if ISDISAMBIGUABLE(H_{set}, H) **then**
    $N \leftarrow$ NEWNODE($N.parent$)
    $N.ball \leftarrow ball$
    $N.h \leftarrow H$
    $N.local = H_{set}$
  end if
end procedure

procedure ISDISAMBIGUABLE(H_{set}, H)
  for h in H_{set} **do**
    $h_{best} \leftarrow \text{argmin}_{x \in H_{set}}(\text{H}(x, H_{set}))$
    if $h_{best} \neq h$ **then**
      return False
    end if
  end for
  return True
end procedure

Algorithm 3 Topos Utility Functions

procedure HYPERDESIC(H_1, H_2)
  $R^{(1)} \leftarrow$ FIND(H_1)
  $R^{(2)} \leftarrow$ FIND(H_2)
  $d \leftarrow$ LOCALMETRIC($R^{(1)}, R^{(2)}$)
  $d \leftarrow d + \min(H(H_1, R^{(1)}.h), R^{(1)}.ball)$
  $d \leftarrow d + \min(H(H_2, R^{(2)}.h), R^{(2)}.ball)$
  return d
end procedure

procedure LOCALMETRIC($R^{(1)}, R^{(2)}$)
  $(R^{(1)}, P_1) \leftarrow$ FIND($H_1, getPath = True$)
  $(R^{(2)}, P_2) \leftarrow$ FIND($H_2, getPath = True$)
  $P = P_1 \Delta P_2$
  return DISTANCE(P)
end procedure

procedure DISTANCE(P)
  $s \leftarrow P.head$
  $e \leftarrow P.tail$
  $c \leftarrow s$
  $sum \leftarrow 0$
  while $c \neq e$ **do**
    $sum \leftarrow sum +$
      $H(c.localRoot.h, c.next.localRoot.h)$
    $c \leftarrow c.next$
  end while
  return sum
end procedure

demonstrate properties of the Topos. Consider the 45 children generated in Fig. 3. We can see through their coloration, how the descendant function d would be used compositionally to generate the children and their ancestors. These children were generated using hypervectors of length 2^{14} and noise in up to 20% of the those bits.

C. Dense Binary Hypervector Separation and Pushout

For our practical working example, we need to specify the SEPARATION and PUSHOUT functions. We use the simplest versions of both of these for dense binary hypervectors. In reality, these functions can be tailor made to the task at hand. Alg. 2 describes pseudo-code for both of these functions. In simple terms, the separation is contingent upon identifying "close" and "far" subsets of our hypervector dataset H_{set}. An attempt is made to make the close and far subsets as equal in size as possible, though this cannot always be done, especially if they all have the same Hamming Distances from the H_{set}'s average hypervector, which is computed via a simple Consensus Sum superposition. Furthermore, the pushout function simply identifies and groups hypervectors that can be superposed and then subsequently identified via the superposition. Namely, given superposition s and a target term $t \in L$, where L is the set of hypervectors in the local space, the $\min_x(H(x, s)) = t$ for all $t \in L$. This tells us that for this class of hypervectors in L, their Hamming Distances approximates what is expected with random hypervectors in their local neighborhood.

TABLE I
AVERAGE HYPERVECTOR DISTANCES IN ANCESTORS AND DESCENDANTS

	Descendant-Descendant	Ancestor-Descendant
Random+Hamming	8013.07	8190.18
Hamming	7056.73	6390.26
Random+Topos	8013.07	5868.98
Topos	5893.94	4490.54

Average hypervector distances for the working example in Fig. 3. Column 1 measures the average distances between each descendant. Column 2 measures the average distances between each ancestor and each of the 45 descendants. The rows compare Hamming Distance versus Topos Distance, with and without replacing descendants with random hypervectors.

D. Experiments

To measure the differences between Topoi hypervector spaces and classical spaces of hypervectors, we can measure the distribution of distances in the former vs. the latter between ancestors and descendants. Namely, the terminal descendants, such as the 45 children in Fig. 3. We are interested in how the two spaces differ in their distribution of distances. There are two metrics to track; the average Hamming Distance between the children, and the average Hamming Distance between the children and their ancestors. This can be compared to random data for both the Topoi Distance and Hamming Distance. We would expect a good Topos to identify genetic relationships better than the classic space of hypervectors.

IV. RESULTS

In this section, we summarize our results on the practical working problem described in the Methodology section.

A. Working Example

To elucidate the functionality of the Topos, we investigate the differences in distances between the Topos' "metric" and the Hamming Distance. We examine these distributions of distances both with random data and the actual descendants shown in Fig. 3. The Topos that is constructed for these descendants is shown in Fig. 2. This Topos is approximately symmetric in how it classifies our set of descendant hypervectors. There appear to be two major classes, as shown by the leaf nodes marked "`(11)`". The next two major classes are marked by the "`(6)`" and "`(5)`". The symmetric nature implies that the Topos is classifying the children similarly to a Normal Distribution. This is expected, since genetic variance here can be constructive or deconstructive, and the random perturbations will cause this to occur in a Normal way, as dense binary hypervectors follow a Bernoulli Coin-Flip Distribution, and will be approximately Normal as the hypervector size approaches infinity. In this case, the size is 2^{14}, and so it well approximates it.

In Fig. 4, the distances of the descendants are compared to random data in both regular Hamming Distance and the Topos' Distance. We can see that the Topos maintains the properties

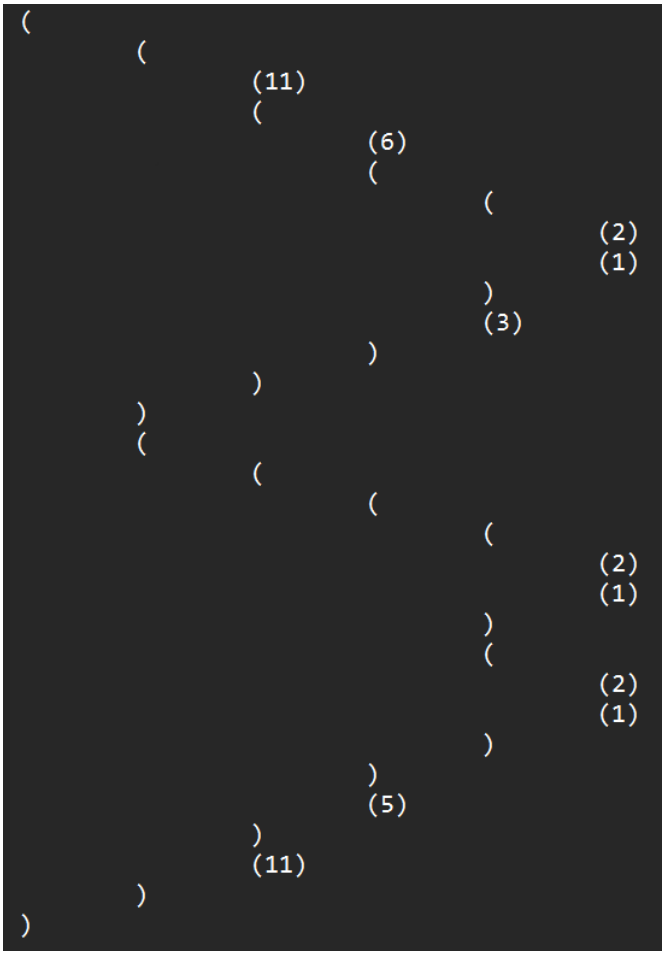


Fig. 2. A Topos for the Working Example: A parenthesization for the Topos from the working example in Fig. 3. The 45 descendant hypervectors are grouped into the above compositionally connected Sub-Object classes. Leaf nodes end with a local neighborhood that approximates Hamming Distance within a Hamming Ball around the superposition of the vectors that fall into that class. The numbers in these nodes, such as "(11)", mean that 11 descendants fall into this class.

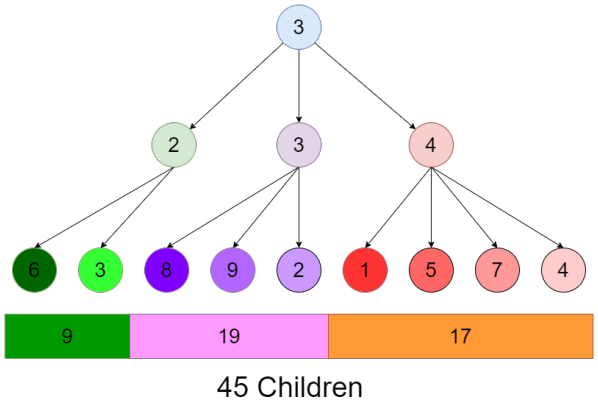


Fig. 3. Working Example of Genetically Generated Descendants: Descendants inherit their parent's DNA by averaging their random hypervector with their parent's. Then, a random percent of bits up to a given noise ratio are flipped, so that each sibling is different. The color codes signify these alterations from ancestral hypervectors.

of random dense hypervectors, as the plots match between both distances, but it uncovers much more genetic structure than regular Hamming Distance, shown by the darker colors that imply lower Hamming Distance. The same phenomenon can be seen in Fig. 5, where the same measurements are made between ancestors and the children. Note that even the oldest ancestor is significantly closer to the children than regular Hamming Distance, which looks random in comparison. The averages of these distances are shown in Table I, which reflect these results in the average. In fact, the random data behaves exactly as expected in both distances, with the subspace of genetically related hypervectors being significantly closer.

B. Randomized Trials

We further study this behavior by running 1000 randomized trials of Topos generation, acquiring similar data as shown in Table I. Our genetic lineage was randomly generated to be up to 4 ancestors (height of 4) with up to 5 children per ancestor. We ran 10 trials for each power in vector size from 2^5 up to 2^{14} and for each noise percentage in increments of 5% from 5% to 50%. The goal was to study how well each distance metric handles the extremes of noise (with 50% noise being nearly random for even large hypervectors) at each vector size from the small to the hyperdimensional. Our results are plotted for hypervector size in Fig. 6 and Fig. 7. The same phenomena are observed as with our working example. Indeed, each graph has the same general trend, despite the hypervector size. Interestingly, as the noise approaches 50%, the Topos begins categorizing ancestors the same as the descendants, while the regular Hamming Distance maintains a large gap between the two. This indicates that the Topos has a better compositional understanding of the relationship between the descendants and between their ancestors than regular Hamming Distance.

V. Discussion

Our results indicate that the learned Topos has gained some compositional understanding of the genetic relationships in our practical working problem. The randomized trials demonstrate that this is the general trend across even less than hyperdimensional sizes of hypervectors. Additionally, the Topos is quite resilient to noise altering its compositional power. We do note that noise causes a wider distribution of descendants which can then influence the average distance of random vectors, as the sparse examples presented will bias the whole space. All in all, the Topos is behaving as expected.

These results open a plethora of possible future works. Our results imply that a model can benefit from the compositional understanding of its input space in downstream tasks using a Topos instead of the standard space of vectors that the model is defined in. A basic use of this would be guided superposition, instead of naive superposition of all terms. Additionally, the separation and pushout functions used for our experiments are just the most basic methods of creating a Topos for the Hamming Space, and they certainly are not representative of what can be done for the other typical VSA/HDC structures. It remains to be seen which functions are best suited for these. Furthermore, in the interest of explainability of AI, the

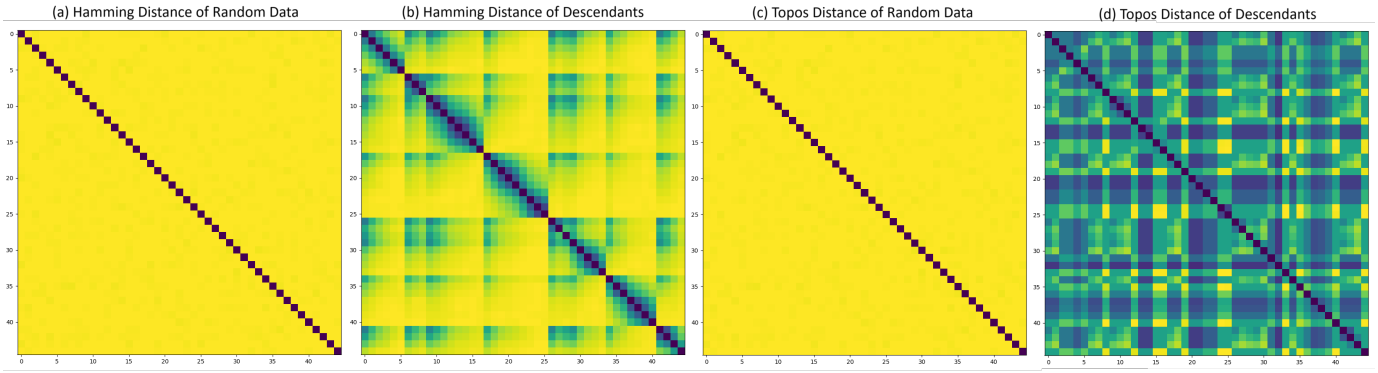


Fig. 4. Example Descendant vs. Descendant Distances: Shows the distances between the 45 descendant of the example in Fig. 3. Yellow symbolizes orthogonality, and darker colors symbolize closeness. Plot (a): Randomly generated descendants and their pairwise Hamming Distances. Plot (b): Descendants and the Hamming Distances between each. Plot (c): Topos Distance between the same random descendants in plot (a). Plot (d): Topos Distance between the same descendants as plot (b). While some of the genetic structure is apparent in plot (b), many distance relatives appear random (yellow), even though they share the same ancestry. In plot (d), the Topos reveals much deeper genetic similarity than (b), without compromising the randomness of unrelated vectors as shown in plots (a) and (c).

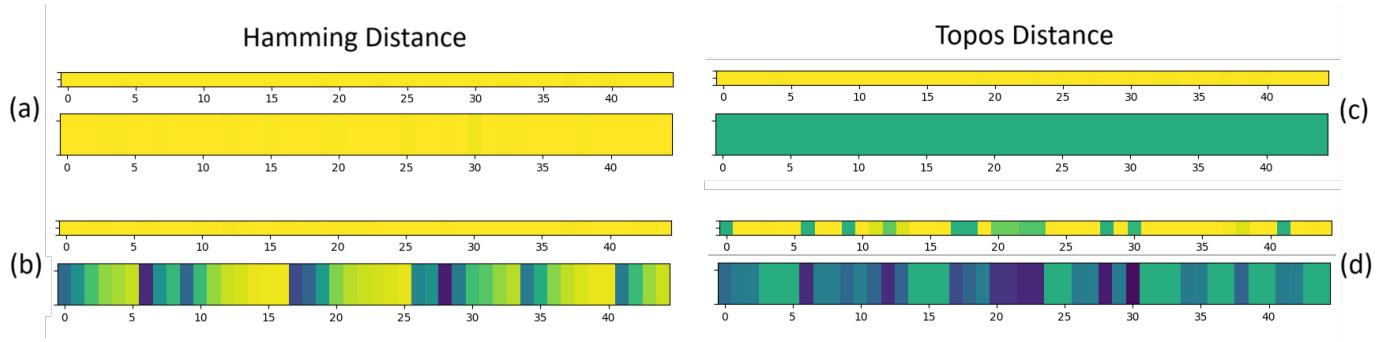


Fig. 5. Example Ancestor vs. Descendant Distances: Shows the distances between the 45 descendants and their ancestors in Fig. 3. Yellow symbolizes orthogonality, and darker colors symbolize closeness. Since each descendant has 2 generations of ancestors before their parents, we show both generations. Plot (a): Randomly generated descendants and the Hamming Distance between them and the actual ancestors. Plot (b): Hamming Distance between descendants and their ancestors. Plot (c): Topos distance between the same random descendants in plot (a). Plot (d): Topos Distance between descendants and their ancestors. We can see that plot (a) is completely random, while plot (b) shows similarity in at least the closer generation of ancestors. Plot (c) shows that the Topos puts ancestors in a non random and unusual close subspace of the hypervectors. Plot (d) shows that genetic similarity is preserved better in the first generation and even in the second generation, which doesn't happen in plot (b).

separation and pushout can be used to discover meaningful categorizations learned by an AI using VSAs/HDC. Some of these properties are easily observable in our results, but task-specific version of these functions still remain an open question. An even more interesting observation is that the separation and pushout functions themselves can be learned by an AI or Machine Learning model. This opens the door to allowing VSAs to not only receive learned encodings as hypervectors from other trained models, as is typical in VSA/HDC, but also use trained models to compositionally create its own algebra for each given task. Finally, the results in Fig. 6 and Fig. 7 imply that the most suitable Topos for a given input can be selected based off of the Hamming Distance to Sub-Object classifiers in the Topoi. This opens up further avenues of exploration in Anomaly Detection, Federated Learning, and more.

VI. Conclusion

We discussed the clear relationship between how VSAs and HDC function and Topoi. We conclude this paper with some thought provoking insight. In some sense, it can be said, that the hypervectors allow us to approximate the functionality of Deep Neural Networks (DNNs) as an algebra (typically done via learning the output embeddings such as in [10], [11], [12]) defined by HDC operations. Our first results indicate that Toposes can be used to approximate the higher-order relationships of hypervectors through its Sub-Objects classifier. In [13] and [14] it is presented how to formulate the fundamental operations carried out by Deep Neural Networks in a purely categorial framework. Rephrasing standard VSA constructions through the lens of Topos Theory could afford us some new insights on how to translate Deep Learning Models into HDC.

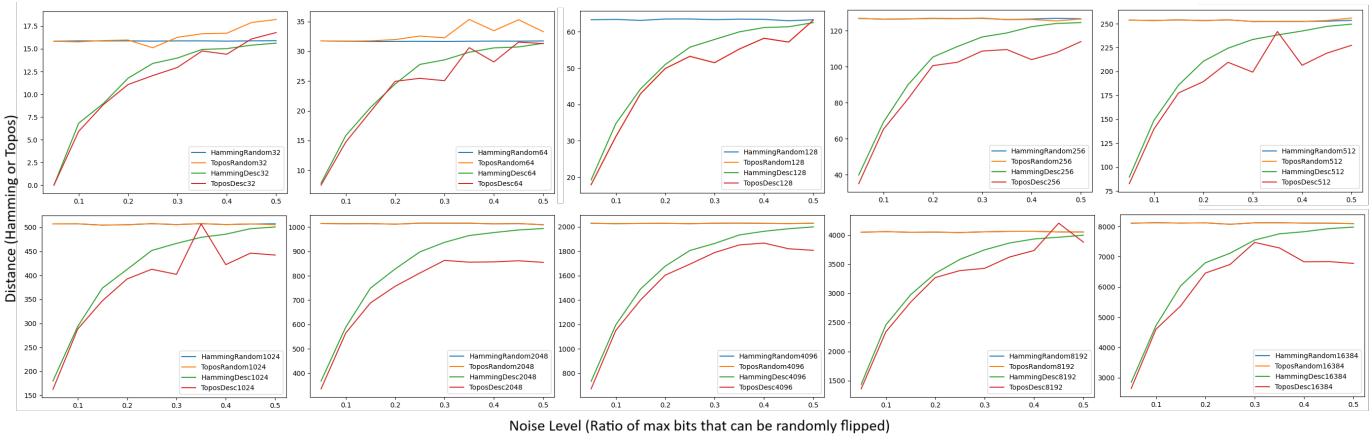


Fig. 6. Descendant vs. Descendant Distances Average Trial Results: Shows the average distances across multiple randomly generated lineages between descendants, much like in Fig. 4. Each subplot is a different power of 2, starting from left to right, and top to bottom at 2^5 up to 2^{14}. The vertical axis is the Distance (Hamming or Topos) and the horizontal axis is the level of noise in increments of 5%. We can see that no matter the hypervector size or the genetic ancestry, the hypervector spaces have similar distributions in each graph across increasing noise levels. The distance is generally smaller in the Topos, which tends to identify genetic relationships better than Hamming Distance, despite the sparse samples of descendants.

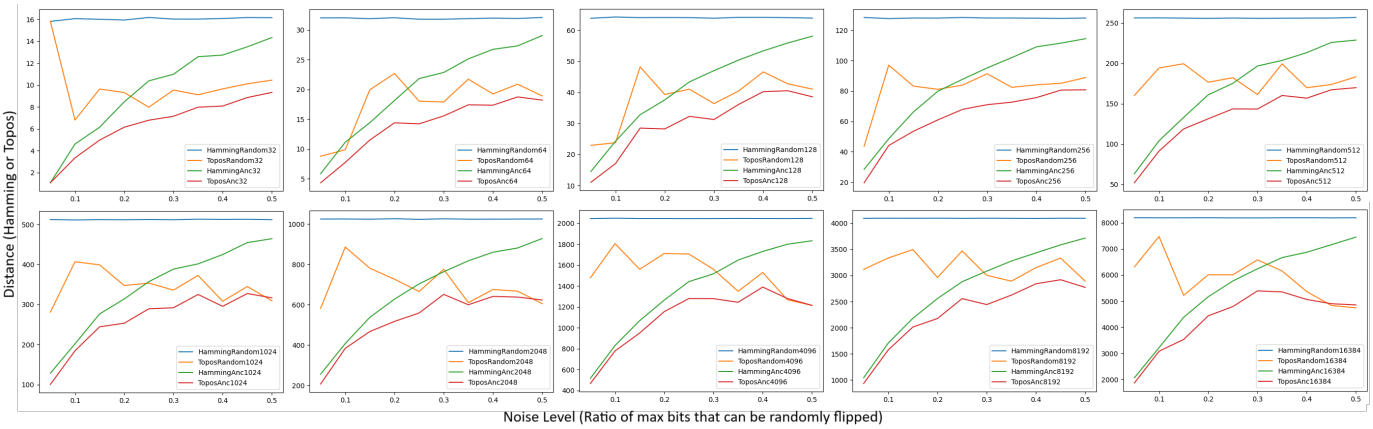


Fig. 7. Ancestor vs. Descendant Distances Average Trial Results: Shows the average distances across multiple randomly generated lineages between descendants and their ancestors, much like in Fig. 5. Each suplot is a different power of 2, starting from left to right and top to bottom at 2^5 up to 2^{14}. The vertical axis is the Distance (Hamming or Topos) and the horizontal axis is the level of noise in increments of 5%. We can see that regardless of the hypervector size or ancestry, similar distributions occur in each graph across increasing noise levels. The distance is significantly smaller in the Topos than in Hamming Distance, as the Topos identifies genetic similarities. With high noise, the Topos loses the ability to identify between ancestors and random data, as this is empirically derived from observations during training.

APPENDIX
TOPOI FROM FIRST PRINCIPLES

The monograph [15] is a standard and comprehensive introduction to Topos Theory, suited both for readers already experienced with Category Theory and its applications to Computer Sciences and readers with no previous exposure to such concepts.

The founding work of Grothendieck proposes Topoi as a unifying concept for mathematical theories, a contemporary and comprehensive account is given by [16], where their role as bridges between mathematical universes is largely explored.

Informally, underlying a Category $\mathcal{C}$ is just a pair of collections $\mathbf{Obj}(\mathcal{C}), \mathbf{Arr}(\mathcal{C})$, whose elements are *objects* and *arrows*, together with an associative *composition scheme* $\circ$. Each arrow f comes equipped with the information of a *source* $\mathrm{dom}\, f$ and a *target* $\mathrm{cod}\, f$ objects, while each object a comes in pair with a special arrow: the *identity* on that object ι_a. This allows us, to a great extent, to forget about objects entirely. In order to further describe the arrows of a Category, we recall some standard taxonomy.

Definition A.1 (Monics and Epics): An arrow $f : a \to b$ is **monic** if for all pairs of *parallel arrows* $g, h : c \rightrightarrows a$ we have

$$f \circ g = f \circ h \Rightarrow g = h. \tag{1}$$

Dually, f is **epic** when, for all $g, h : b \rightrightarrows c$,

$$g \circ f = h \circ f \Rightarrow g = h. \tag{2}$$

In the realm of Sets, one concludes that an arrow (function) is monic/epic iff it is injective/surjective iff it is left/right-invertible. In Category Theory, since we do not describe objects element-wise, the notion of being isomorphic replaces the stricter requirement of having actual equality.

Definition A.2 (Isos): An arrow $f : a \to b$ is **iso** if there exists $g : b \to a$ such that

$$f \circ g = \iota_b, \qquad g \circ f = \iota_a. \tag{3}$$

When this is the case, we also write $a \approx b$ and say that they are **isomorphic**.

Note that isos are always both monic and epic, but the contrary is, in general, not guaranteed: take the monoid of the Naturals $(\mathbf{N}, 0, +)$ as a one-object category with arrows representing the integers, then any arrow is monic and epic but only 0 is an iso. Categories where the contrary holds, such as **Grp** (groups and their homomorphisms), are called *balanced*.

We pause for a second to note that while "monic/epic" and "left/right-invertible" make sense for sets but also arbitrary categories, the concepts of injectivity and surjectivity are not immediately translated to categorial semantics. Indeed, they seem entirely element-based and their arrow-theoretic formulations is one of the cornerstones of Topos Theory.

Usually, there can be many *parallel* arrows between the same source and target. Objects that are connected in precisely one way to every other one are of fundamental interest.

Definition A.3 (Initial and Terminal Objects): An object x is **initial** if there is exactly one arrow $p : x \to b$ for any object b. Dually, it is **terminal** if there is exactly one arrow $q : a \to x$ for each a. An object which is both is called a **zero** object.

Initial and Terminal objects are desirable inhabitants for any suitable notion of "Universe of Sets". A Category with an initial object is sometimes termed *pointed*. A *degenerate* category is one where all objects are isomorphic, although in Topos Theory this terminology is typically used more generally to simply mean that it has a zero object. This precludes a Category from being a Topos, with the only exception being the *Trivial Topos* (the one comprising of one object and its identity arrow only). Observe that arrows with domain/codomain a terminal/initial object are always monic/epic and that initial/terminal/zero objects are unique only up to (unique) isomorphism.

We now turn our attention to the *Extensionality* principle of Set Theory, which asserts that sets with the same elements are actually the same set, following again [15]. Its categorial form addresses the extent to which we can distinguish parallel arrows in a given category. Note that a Category in which there is at most one arrow between each pair of objects is essentially just a *Preorder*.

Definition A.4 (Generators): An object s is called a **generator** (or **separator**) if, for each pair of arrows $f, g : a \rightrightarrows b$,

$$\forall e : s \to a (f \circ e = g \circ e) \Rightarrow f = g. \tag{4}$$

An initial object can be a generator only in case the Category is already just a preorder. A terminal object instead, is required to be a generator in order to have a *Well-pointed Topos*. Even though **Grp** has a zero object, it also has infinitely many non-isomorphic generators. This shows how the idea of well-pointedness makes sense for arbitrary Categories and not just Toposes, and is in fact an open research topic. Hence, the following definition is somewhat non-standard.

Definition A.5 (Well-pointedness): A Pointed Category is Well-pointed if it has a non-zero terminal generator.

The main consequence of Well-pointedness is that one can think of the monics $e : s \to a$ as[1] *generalised elements* of a.

[1]One is, of course, also *free* to think the same holds in the Trivial Topos.

Our next goal is to translate the ideas of *bundling* and *binding* in categorial terms. In [17] it is given a comprehensive and in depth account of the binding problem. It turns out, that its arrow reformulation leads to the idea of *colimits*. This is one of the motivations of many works of Baas, such as [18], on the foundational role of *Hyperstructures*.

A (*Covariant*) *Functor* F between Categories $\mathcal{C}, \mathcal{D}$, written $F : \mathcal{C} \to \mathcal{D}$, maps objects of $\mathcal{C}$ to objects of $\mathcal{D}$ and arrows to arrows in a way that respects the composition schemes and their units, i.e. it preserves *commuting triangles*:

$$F(f \circ g) = F(f) \circ F(g), \tag{5}$$

$$F(\iota_a) = \iota_{F(a)}. \tag{6}$$

Now, in a similar way as we can think as a set function as a special kind of relation (its *graph*), a functor is a special case of the more general concept of a *Diagram*. Instead of attempting a formal definition of diagrams, which are the very essence of Category Theory, we focus on some concrete examples that are relative to this paper and Hyper-Dimensional Computing in general.

The Theory of *Nets*, was one of the main motivations behind early Category Theory. It was developed in [19] in order to generalise the idea of *sequences* and *limit points* in Topology so that an arbitrary topological space could be entirely recovered by studying the convergence of its nets (recall that this is not possible by simply using sequences).

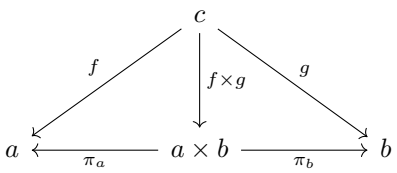
Example A.1 (Directed Sets): As already mentioned, any preorder $(P, \sqsubseteq)$ can be made into a category $\mathcal{P}$ with $\mathbf{Obj}(\mathcal{P}) = P$ and one arrow $f : a \to b$ whenever $a \sqsubseteq b$. Now, given two elements a, b of P, an **upper bound** is an element c satisfying $a \sqsubseteq c$ and $b \sqsubseteq c$. A preorder where upper bounds exist for all pair of elements is called a (**upward**) **directed set**. A category where isomorphism actually reduces to equality is termed *skeletal*, so that a **Poset** is simply a skeletal preorder (i.e. a *symmetric one*). A **Totally Ordered Set** is a poset with *path-connected* objects, that is: we always either have $a \sqsubseteq b$ or vice versa (i.e. *trichotomy holds*).

Definition A.6 (Nets): A functor $S : \mathcal{P} \to \mathcal{C}$, where $\mathcal{P}$ is a directed set, is called a **net** in $\mathcal{C}$.

Example A.2 (Joins and Meets): An element c of a preorder P is **maximal** if whenever $c \sqsubseteq a$, we also have $a \sqsubseteq c$. It is called a **greatest element** if it is terminal. Dually, we obtain **lower bounds**, **minimal elements** and **smallest elements**. Note that a preorder with a greatest/smallest element is trivially upward/downward directed. The **join/meet** of a, b is, when it exists, the smallest/greatest element of the directed (sub)set of upper/lower bounds of a and b, and is denoted by, resp., $a \vee b$ and $a \wedge b$.

In an arbitrary Category, this generalises to the idea of (co)products: given objects a, b, a *span* (sometimes, *internal object*) $a \times b$ is an object together with two maps (the *projections*) $\pi_a : a \times b \to b, \pi_b : a \times b \to b$. A span which is *universal*, i.e. such that given any other span c with arrows

$f, g : c \rightrightarrows a, b$ we have a unique $f \times g : c \to a \times b$ making



(7)

a *commutative diagram*, is called the *product* of a and b. As usual, reversing the direction of all arrows leads us to the dual concept of *cospans* and, specifically, the *coproduct* of a, b, which is denoted $a + b$. Thanks to associativity, if a category has products for each pair of objects, it is always possible to extend the definition to any (finite) number n of objects $\{a_i\}_{i \leq n}$, denoted by $\prod_{i \leq n} a_i$, with the terminal object as the limiting case when $n = 0$.

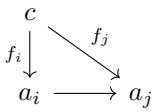
Looking back at Sets, one recognises that the product is given by the familiar notion of the *Cartesian product*. Moreover, given two sets a, b, the collection of arrows from a to b is the set b^a. Categories such that the families of parallel arrows carry the same structure as its objects are termed *closed*.

Definition A.7 (Exponentials): Let $a \times b$ be the product of objects a, b, we say that an object c together with an arrow $v : c \times a \to b$, the **evaluation**, is the **exponential** of a and b, denoted b^a, if for any other $w : d \times a \to b$ there exists a unique $t : d \to c$ such that

$$v \circ (t \times \iota_a) = w. \tag{8}$$

Definition A.8 (CCC): A **Cartesian Closed Category** is a Category with all finite products (including a terminal object) and all exponentials.

Now, suppose that we were given not only some objects $\{a_i\}_{i \leq n}$ but also a (finite) collection of arrows between them. A *cone over* this diagram is just an object c together with one arrow $f_i : c \to a_i$ for any index i such that all triangles



(9)

commute. Reversing directions, we find the *cocones under* a diagram.

Definition A.9 (Completeness): A universal cone for a diagram, if it exists, is called its **limit**. A category is **finitely complete** if every finite diagram has a limit. Duality yields **colimits** as universal cocones and **finite cocompleteness**.

Example A.3 (Bounded Lattices): A **Bounded Lattice** is a finitely complete and finitely cocomplete poset.

In order to have a Topos, we need all finite (co)limits. For completeness, the following diagrams will suffice in a CCC.

Example A.4 (Equalizers and Coequalizers): Given a parallel pair of arrows, $f, g : a \rightrightarrows b$ their (**co**)**equalizer** is, if it exists, their (co)limit.

Theorem A.1: A Cartesian Closed Category is finitely complete if and only if it has all equalizers.

Example A.5 (Pullback and Pushouts):

- A **pullback** is the limit of a cospan.
- A **pushout** is the colimit of a span.

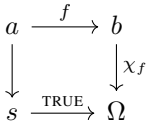
A proof of the following classical result (actually, of its dual) can be found in [20].

Theorem A.2: A category is finitely cocomplete if and only if it is pointed and has all pushouts.

Thus far, we have seen how the unifying concepts of functors, diagrams and (co)limits permits us to create new objects from old ones. These constructions take the place, and generalise, the *Pairing* and *Union* axioms from Set Theory. The last ingredient needed to obtain a Topos, will take care of *Power-sets*.

First of all, as we think of elements of an object a as the monics with source the terminal object (at least if the category is well-pointed), a *subobject* of a is any monic with target a; the collection of subobjects of a is denoted $\mathbf{Sub}(a)$. For sets, there is an immediate correspondence between subsets and functions with value in $2 := \{0, 1\}$, namely $\mathcal{P}(X) \approx 2^X$.

Definition A.10: [Sub-objects Classifier] An object Ω of a category with terminal object s, together with an arrow $\text{TRUE} : s \to \Omega$, is called a **subobjects classifier** whenever for all monics $f : a \to b$ there is precisely one arrow $\chi_f : b \to \Omega$, the **characteristic arrow**, making



(10)

a pullback. In general, the arrows of type $s \to \Omega$ are called the **truth-values**.

With all of this in place, we can finally give a proper definition of a Topos.

Definition A.11: [Elementary Toposes] A (**Elementary**) **Topos** is a Finitely Cocomplete Cartesian Closed Category with a subobjects classifier and all equalizers.

The explicit mention of equalizers is needed because we are following the current modern definition of CCCs, as given in [21], that requires all finite products but not all finite limits.

It is instructive to draw a comparison with the classical definition of topological space, to better understand the new perspective given by Toposes.

Example A.6 (General Topology): Usually, one presents a **Topological Space** as a set T together with a *collection* $\mathcal{T}$ of **open** subsets containing $\emptyset, X$ and **closed** under *finite intersections* and *arbitrary unions*. We can now rephrase this as saying that $\mathcal{T}$ is a bounded lattice with union for the join and meet for the intersection with the additional presence *arbitrary* joins. The *points* of a topological space are not the objects. The connection is, of course, much deeper and, in fact, the study of sheaves and bundles from Algebraic Geometry served as the inspiration for much of the early development of the Theory of Toposes.

Now that we have sketched a path to Topoi, we conclude by giving a glimpse of how one can reason "inside" a Topos and how this relates to Manifold Learning.

According to the constructivist view of mathematics, truth is not absolute but rather context-dependant: the truth-value of a sentence varies according to the state of knowledge regarding its matter. Indeed, the theory of topoi is deeply connected to *Kripke Frames*. An interpretation of classical *Vector Spaces*

in these terms is discussed in [22], which further motivate the investigation of how models for *Modal Logic* can be realised as VSAs.

To make things easier, we can think of a frame as being simply any poset $(P, \sqsubseteq)$. Given a set X and a *formula* ϕ, the *Separation Scheme* allows us to consider the set $\{x \in X : \phi(x)\}$ of elements of X that *satisfy* it. If truth is supposed to persist in time, we can instead consider the P-indexed collection of formulas which have been proven to hold at instant $p \in P$:

$$\{x \in X : \phi_p(x)\}.$$

The *algebras of subobjects* in the case of sets are *Boolean Algebras*, which are known to be the *models* of *Classical Propositional Logic*. The models for *Intuitionistic Logic* are instead provided by

Definition A.12 (Heyting Algebras): A **Heyting Algebra** is a bounded lattice with all exponentials. The exponential of objects a, b is referred to as **implication** and denoted by $a \Rightarrow b$.

This construction is essentially the weakest possible such that *modus ponens* is *sound*.

Definition A.13 (Hereditary Sets): Consider a poset $(P, \sqsubseteq)$, a subset $Q \subseteq P$ is **hereditary** (or **upward closed**) if $q \sqsubseteq p$ implies $p \in Q$ whenever $q \in Q$. For any element $p \in P$, the **principal** hereditary set $\{q \in Q : p \sqsubseteq q\}$ is denoted by $\lceil p \rceil$.

Example A.7: Denoting by P^+ the collection of its hereditary (sub)sets, we have actually defined a topology, whose collection of open sets forms a Heyting Algebra: the exponential of S, Q is obtained by taking the interior of $(P - S) \cup Q$, where $P - S$ is the complement of S.

The reader interested with the paradigms of *Lattice-based Computing*, always throughout the lens of Category Theory, can find an extensive treatment in [23]. This line of research was initiated by the groundbreaking work of Lawvere, starting from [24], where a detailed description of *metric spaces* in terms of *Enriched Categories* is given. In the paper, it is also made extensive use of the notion of *Closed Categories*, where the collections of arrows carry the structure of objects themselves, and *Monoidal Categories*, which are equipped with an abstract *tensor product* (the categorial notion of bundling), both generalising CCCs.

ACKNOWLEDGMENT

The authors would like to thank Simuli, Inc. for their collaboration and support in this research, as well as for motivating research into the problem.

REFERENCES

[1] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognitive computation*, vol. 1, pp. 139–159, 2009.

[2] D. Kleyko, D. Rachkovskij, E. Osipov, and A. Rahimi, "A survey on hyperdimensional computing aka vector symbolic architectures, part ii: Applications, cognitive models, and challenges," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–52, 2023.

[3] P. Sutor, D. Summers-Stay, and Y. Aloimonos, "A computational theory for life-long learning of semantics," in *Artificial General Intelligence: 11th International Conference, AGI 2018, Prague, Czech Republic, August 22-25, 2018, Proceedings 11*. Springer, 2018, pp. 217–226.

[4] N. McDonald, R. Davis, L. Loomis, and J. Kopra, "Aspects of hyperdimensional computing for robotics: transfer learning, cloning, extraneous sensors, and network topology," in *Disruptive Technologies in Information Sciences V*, vol. 11751. SPIE, 2021, pp. 21–33.

[5] P. Sutor, Y. Aloimonos, C. Fermuller, and D. Summers-Stay, "Metaconcepts: Isolating context in word embeddings," in *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, 2019, pp. 544–549.

[6] E. Grefenstette and M. Sadrzadeh, "Experimental support for a categorical compositional distributional model of meaning," *arXiv preprint arXiv:1106.4058*, 2011.

[7] P. Huneman, "Neutral spaces and topological explanations in evolutionary biology: Lessons from some landscapes and mappings," *Philosophy of Science*, vol. 85, no. 5, p. 969–983, 2018.

[8] I. Baianu, R. Brown, G. Georgescu, and J. Glazebrook, "Complex non-linear biodynamics in categories, higher dimensional algebra and łukasiewicz–moisil topos: transformations of neuronal, genetic and neoplastic networks," *Axiomathes*, vol. 16, no. 1-2, pp. 65–122, 2006.

[9] I. C. Baianu, "Lukasiewicz-Topos Models of Neural Networks, Cell Genome and Interactome Nonlinear Dynamic Models. Les Categories des Connexions Cellulaire dans le Topos sur les Algebres des Logiques Lukasiewicz pour les Neuronnes et le Genome Cellulaire compris dans un Interactome Dynamique," Illinois Univ., Urbana-Champaign, IL, Tech. Rep., 2004. [Online]. Available: https://cds.cern.ch/record/746663

[10] P. Sutor, D. Yuan, D. Summers-Stay, C. Fermuller, and Y. Aloimonos, "Gluing neural networks symbolically through hyperdimensional computing," in *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2022, pp. 1–10.

[11] A. Mitrokhin, P. Sutor, D. Summers-Stay, C. Fermüller, and Y. Aloimonos, "Symbolic representation and learning with hyperdimensional computing," *Frontiers in Robotics and AI*, vol. 7, p. 63, 2020.

[12] M. Nazemi, A. Fayyazi, A. Esmaili, and M. Pedram, "Synergiclearning: Neural network-based feature extraction for highly-accurate hyperdimensional learning," in *Proceedings of the 39th International Conference on Computer-Aided Design*, 2020, pp. 1–9.

[13] Toposes and S. of Neural Networks, "Jean-claude belfiore, daniel bennequin," *Huawei Advanced Wireless Technology Lab*, 2022.

[14] F. Z. Neil Ghani, Paul Wils, "Categorical foundations of gradient-based learning," *Proc ACM Program Lang, Vol 1*, 2021.

[15] R. Goldblatt, *Topoi: The Categorial Analysis of Logic*. Dover Publications, 2006.

[16] O. Caramello, *Theories, Sites, Toposes*. Oxford University Press, 2018.

[17] J.-P. V. Andrée Ehresmann, *Memory Evolutive Systems; Hierarchy, Emergence, Cognition*. Elsevier, Studies in Multidisciplinarity, Volume 4, 2015.

[18] N. Baas, "Hyperstructures-a framework for emergence, hierarchies and complexity." *Proceedings Congres sur l'emergence dans les modeles de la cognition, Telecom, Paris, 67-93*, 1992.

[19] H. L. S. Eliakim Hastings Moore, "A general theory of limits," *American Journal of Mathematics, Vol 44, No 2 (April 1922), 102-121*, 1922.

[20] G. S. Horst Herrlich, *Category Theory*. Sigma Series in Pure Mathematics - Volume 1, 2007.

[21] P. Johnstone, *Sketches of an Elephant: A Topos Theory Compendiumm*. Oxford science publications, tba.

[22] G. Greco, F. Liang, M. Moortgat, and A. T. Alessandra Palmigiano, "Vector spaces as kripke frames," *IfCoLoG Journal of Logics and their Applications, Vol 7, pp 853–873*, 2020.

[23] W. T. Dirk Hofmann, Gavin Seal, *Monoidal Topology, A Categorical approach to Order, Metric and Topology*. Cambridge, Encyclopedia of Mathematics and Its Applications 15, 2014.

[24] W. Lawvere, "Metric spaces, generalised logic, and closed categories," *Reprints in Theory and Applications of Categories, No 1, 1-37*, 1973.