Thinking about Vector Symbolic Architectures

Ross W. Gayler
ross@rossgayler.com
Independent Researcher

2023-06-15

Introduction

Motivation

- How far would you get with knowing the definitions of the VSA operators and *nothing* else?
  - Like having axioms and no other maths knowledge
- Everyone has a web of auxiliary *beliefs* around VSA:
  - About VSA and relationships to other things they know
  - Used to reason about their VSA knowledge and its implications, so is central to applying and extending VSA
- The objective is that the auxiliary beliefs should be *productive* for applying and extending VSA

Limitations

Auxiliary beliefs are:

- Not necessarily true
- Not necessarily even true or false (e.g. metaphors)
- Not necessarily coherent
- Not stable over time. Think of them as evolving frameworks.
- Likely to be idiosyncratic (If they were canonical they would be VSA theory)

Think of them as gambles. You are betting that they will be more productive than alternatives you might entertain.

(dis)Organisation of the talk

- I am talking about my conceptual framework
  - I have no idea what yours is, because we don’t discuss it
  - This may be painfully obvious to you. I apologise.
- Framework is densely connected web of interrelated points
  - No simple, logical, explanatory path through that web
  - Any path is necessarily a random-ish ramble. I apologise.
- Pick some apparently salient points, wander in their neighbourhoods, much hand waving

Analogue computer wire interpretation

Composite label and magnitude

Q: What is a (hyper)vector? **A**: Direction + scalar magnitude

- Electric analogue computers represent values on wires. The voltage is the magnitude of the signal, the “meaning” of the signal is a label on the wire.
- Interpret the vector magnitude as the signal magnitude.
- Interpret the vector direction as the signal label.
- Labels are composite - (de)composed by VSA operators
- Labels (wires) can be created on the fly

Think of VSA systems as analogue computers

Labels can encode values

Is it too limiting for values to be scalar magnitudes? What if I want structured values?

- Composite labels can encode values, e.g. $colour \times red$ or $height \times encode(180)$
  - Interpret the label as a predicate
  - Interpret magnitude as truth value or degree of support
- To avoid -ve magnitude, put the sign in the label: $-(a)$
  - Use unary additive and multiplicative inverse operators
- May need mechanism (e.g. RELU in cleanup) to enforce this

Labels can encode complicated values

Slogan: Everything is just a vector

- In GOFAI data structures, slots/fields/keys are atomic symbols: $colour$, $height$, ...
- In VSA labels can be arbitrarily complicated, e.g. $go\_to\_kitchen\_and\_look\_in\_fridge \times beer$

  (where $go\_to\_kitchen\_and\_look\_in\_fridge$ represents an executable sensorimotor program)

  is equivalent to the agent's degree of belief that there is beer in the refrigerator.

Label symmetries and equivalences

Slogan: Every vector is just a value

- VSA "sees" the vector value, not the sequence of operations
  - Different sequences of operations can be equivalent, e.g.
    - $a + b = b + a$ (if bundling is commutative)
    - Circular convolution binding of d-dimensional vectors is equal to the bundle of d-many Hadamard bindings of permutations of the arguments
- "noise" can make unequal values effectively equal, e.g. $a + b + \cdots + y \approx a + b + \cdots + z$ (bundling capacity)

Occam’s hypervectors

- Start with the atomic vectors in their vector space
- Repeatedly apply the operators to generate expressions of increasing length
- Crowding of result vectors increases with expression length
- Every sufficiently long expression will be approximately equal to some shorter expression
- Implies a form of Occam’s Razor: The system treats any value as effectively arising from the simplest expression that approximately generates that value.

Similarities

Angular similarity and distance

- Similarity is central to reasoning about VSA
- Angle between vectors (equivalent to dot product) is appropriate because it respects the arguments as vectors
  - Vectors are defined with respect to an origin
  - Distances between points are invariant to translation
  - Angle between vectors to the same points are not invariant to translation of the origin
- Distance and angle can be equivalent if points constrained to a fixed origin, e.g. Hamming distance for binary vectors

A view from the north pole

- Imagine the hypervector representing the state of a VSA system is the north pole and you are standing there
- Look at the locations of random hypervectors. They are almost all very near the equator (quasiorthogonality)
- Similarity is defined on pairs of vectors. Consider pairs related by transforms available to the system
  - Pairs within the equatorial belt (the vast majority) have no impact on similarity relative to pole (current state)
  - Angular similarity can’t be only driver of system dynamics

Edit distance for binding/permutation

- Angular similarity is essentially about transforms within a hemi(hyper)sphere
- Angular similarity is driven by bundling
- Binding and permutation are equatorial belt transforms
- Something like an edit distance might be useful for selecting between equatorial transforms
  - Every destination is only one transform away if we allow arbitrary binding and permutation
  - Restrict to available permutations and curried bindings

Estimating latent reality

Latent reality and hypervectors

- We concentrate on the observable hypervectors
- Change our focus to unobservable reality
  - Inspired by being an applied statistician
  - Much of statistics is about inferring the state of an unobservable reality from observable measurements
- Hypervectors as observable realisations of measurements of unobservable (latent) reality
  - Try to explain the role of randomness in VSA

Elements as random measurements

- No direct access to reality (mediated by measurements)
- Imagine looking at an object
  - Random whether an atom reflects a photon to your eye
  - Random whether a photoreceptor is in the path
- Ultradimensional vector of potential measurements
  - Hypervector is a random sample of them: $\phi_i(x)$
    - ϕ_i is a function from reality x to VSA base field
  - Want to infer the reality despite randomness of sample

Properties of random measurements

- VSA doesn't "know" ϕ_i (knows the value, not the function)
- Want measurements to be individually and collectively informative about x (reality)
  - Need to depend on properties of x we want to capture
  - Need to be independent conditional on x (hash of x)
- Information is carried by covariation of $\phi_i(x)$ induced by variation in x
- No measurement is privileged (implies robustness, distributed representation, and unordered representation)

Measurements as constraints

Imagine the VSA base field is binary:

- Each specific $\phi_i(x)$ is consistent with a set of half the possible latent realities
- A set of hypervector elements specifies the intersection of those sets
  - Each element narrows the set of consistent realities

Interpret hypervector as specifying a set of indistinguishable realities rather than being a representation of a single reality

Possible implications/extensions

- More feasible search over reality because of the inexactness due to indistinguishability?
- Only need sufficient dimensionality to make the distinctions we need
  - Optimisation of dimensionality
  - Dynamic dimensionality?
- Possibility of dynamic constraints
  - Effectively adding or removing measurements
  - Dynamically creating new measurements on the fly

Statistical interpretation of algebraic terms

Statistical data structure

- Standard statistical data structure is a matrix
  - Rows are cases/observations
  - Columns are variables/features (predictors and outcomes)
- Columns structurally orthogonal in data space (bases)
- Features are often single columns (scalar values)
- Features can be groups of columns (e.g. one-hot encoding)
- Statistical modelling is about exploiting covariation of feature values induced by variation over cases (familiar?)

Terms as rotated features

- In statistical modelling, we could rotate the data space
  - Features are spread over columns
  - Works if rotated features are orthogonal in the data space
- VSA uses regression to extract predictions from hypervectors
  - Hypervectors often expressible as sum of algebraic terms
  - Algebraic terms are often orthogonal (rotated features)
    - Simple terms represent main effects
    - Bindings represent interactions

Possible implications

- Operation of VSA regression/classification systems can be understood/analysed with respect to terms in hypervector
  - E.g. Integer Echo State Network builds standard sequence representation (Interpretable as set of lagged inputs)
- Representations can be designed to achieve objectives
  - What features needed for standard regression?
  - Create algebraic terms that implement those features
    - E.g. Epileptic Seizure Challenge needed interactions of time-series features with time of day (bindings)

Indices and permutation

Element indices as unique labels

- Computer people tend to think of vector indices as consecutive integers: a_i where $i = 1, 2, \ldots$
  - This imposes more structure than necessary
- Indices only need to be unique : $i = sad, bee, hot, \ldots$
- Indices do *not* need to be ordered
  - Ordering convenient for 2D electronic implementation
  - Ordering is an imposition for 3D neural implementation
- Hypervector is a set of key-value pairs where the values are from the VSA base field (sound familiar?)

Permutation and operators

- It doesn’t make sense to talk of permuting an isolated hypervector (interpreted as set of key-value pairs) because it’s unordered
- Makes sense to talk of permutation:
  - relative to another vector,
  - when they are being combined by an operator,
  - because it’s about tracking which elements are combined

$$\{a_{a1}, a_{a2}, \ldots\} + \rho\{b_{b1}, b_{b2}, \ldots\} = \{x_{a1.b2}, x_{a2.b3}, \ldots\}$$

Possible implications

- What makes a tensor product a tensor product is the pattern of combination of the elements of the arguments and the availability of that pattern to guide tensor operations (e.g. tensor contraction)
- Key-value pairs (hypervector elements) can be represented and operated on as VSA hypervectors
- Is it possible to self-embed?
  - Implement tensor product operations with VSA?
  - Have dynamic elements (add/remove elements)?

Archival links

This presentation has been archived on Zenodo:

DOI 10.5281/zenodo.8076677 Video recording

DOI 10.5281/zenodo.8076707 Slides (PDF)

DOI 10.5281/zenodo.8076736 Source code of slides