Solving factorization problems with VSAs and resonator networks
Seems easy to me...

Alcorn et al. (2019) Strike (with) a Pose: Neural Networks Are Easily Fooled by Strange Poses of Familiar Objects. ArXiv

How to incorporate new “machinery” to solve these problems?
DISENTANGLING IMAGES WITH LIE GROUP TRANSFORMATIONS AND SPARSE CODING

Ho Yin Chau$^{1,2}$  Frank Qiu$^{2,3}$  Yubei Chen$^{1,2,4}$  Bruno Olshausen$^{1,2}$

Generative models with transformations
Must infer multiple parameters
VSAs to represent transformation parameters
What did we learn?

2D Translation

Rotation + Scaling

Why learn when you can engineer?
-- Still have to build-in dimensionality of transform

Fourier

Log-polar
Vector-Symbolic Architectures (VSA): a framework for computing with distributed representations

\[ a, b, c \ldots = \{-1, +1\}^N \]

\[ N = 10,000 \]

\[ a \cdot b : \text{similarity} \]

\[ (\text{dot product}) \]

\[ a + b : \text{set operation} \]

\[ \text{(element-wise sum)} \]

\[ a \odot b : \text{binding operation} \]

\[ \text{(element-wise multiply)} \]

\[ \rho(a) : \text{ordering operation} \]

\[ \text{(cyclic shift)} \]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary sparse distributed code (Rachkovskij, 2001)</td>
<td>Hyperdimensional Computing (Kanerva, 2009)</td>
<td>Matrix binding with additive terms (Gallant &amp; Okaywe, 2013)</td>
<td>-------------------------------------------------</td>
</tr>
</tbody>
</table>

\[ a, b, c \ldots = \{-1, +1\}^N \]

\[ N = 10,000 \]

\[ a \cdot b : \text{similarity} \]

\[ (\text{dot product}) \]
VSAs and abstract algebra

• What makes VSA interesting/why did I want to study VSA?
  • VSA includes a binding operation, which is missing in most neural network models
• Abstract algebra defines multiplication as an operation that distributes over addition
  • This immediately leads to the combinatoric nature of multiplication
  • \((a + b + c) \times (d + e + f) = ad + ae + af + bd + be + bf + cd + ce + cf\)
    • 2 product with 3 sums \(\rightarrow 3^2\) terms
HD computing is a field ...and more?

• A field has addition/subtraction and multiplication/division

• Multiplication distributes over addition
  • Distributivity is key to multiplication
  • Combinatorics of distributivity \((a + b + c) \times (x + y + z)\)

• Binding distributes over addition
  • \(x \odot (y + z) = x \odot y + x \odot z\)

• Permutation also has distributive property
  • \(\rho(x + y) = \rho(x) + \rho(y)\)

• Permutation also distributes across binding
  • \(\rho(x \odot y) = \rho(x) \odot \rho(y)\)
Combinatoric representations in connectionism

• The need to represent conjunctions and other types of combinatoric data structures lead connectionists to develop new operations in neural networks.
  • Superposition catastrophe

• Tensor products were natural extension of requirements for representing combinatoric structure in neural networks
  • Dimensionality increases with combinatorics

• VSAs utilize randomness to represent combinatoric structures in vector spaces of fixed dimensionality
  • VSA binding is equivalent to tensor products through lens of compressed sensing.
When are VSAs useful?

• Exploit the combinatoric structure
  • The combinatorics of data structures greatly exceeds the dimensionality of the representation space
    • Trigrams of letters: $26^3 > 17K$, want to reduce dimensions to less than 17K

• Avoid large superpositions
  • Sparsity in the data – only a few of these combinatoric possibilities are represented at a time.
10,000 < 27^3 = 19,683

Sparse statistics = less noise

Language Vectors = histogram of letter-trigrams
Factorization in perception

(Adelson, 2000)
Scene understanding and inverse graphics are factoring problems

\{cube, size, x_0, y_0, z_0, \theta_{XY}, \theta_{XZ}, \theta_{YZ}, \ldots\}
\{sphere, radius, x_1, y_1, z_1, \ldots\}

Disentangle `what' and `where'
Representing simple scenes

Figure 3: Generating a vector symbolic encoding of a visual scene.
Factorization problems are common in VSAs

Let \( b = x \otimes y \otimes z \)

\[ x \in X := \{x_0, x_1, \ldots, x_n\} \]
\[ y \in Y := \{y_0, y_1, \ldots, y_n\} \]
\[ z \in Z := \{z_0, z_1, \ldots, z_n\} \]

Problem: You are given \( b \), what are \( x \), \( y \) and \( z \)?
Consider the following energy function

\[ E = -\mathbf{b} \cdot (\mathbf{x} \otimes \mathbf{y} \otimes \mathbf{z}) \]

\[ \mathbf{x} = \sum_{i=1}^{n} \alpha_i \mathbf{x_i}, \quad \mathbf{y} = \sum_{i=1}^{n} \beta_i \mathbf{y_i}, \quad \mathbf{z} = \sum_{i=1}^{n} \gamma_i \mathbf{z_i} \]
Consider the following energy function

\[ \sum_{i=1}^{100} \alpha_i \beta_i \gamma_i x_i \otimes y_i \otimes z_i + \ldots + \alpha_i \beta_j \gamma_k x_i \otimes y_j \otimes z_k + \ldots + \alpha_n \beta_n \gamma_n x_n \otimes y_n \otimes z_n \]

\[ E = -b \cdot (x \otimes y \otimes z) \]

\[ x = \sum_{i=1}^{100} \alpha_i x_i, \quad y = \sum_{i=1}^{100} \beta_i y_i, \quad z = \sum_{i=1}^{100} \gamma_i z_i \]
Resonator networks for factoring distributed representations of data structures

\[ s = x \odot y \odot z + \text{noise} \]

\[ x \in X: \{ \pm 1 \}^{N \times D_x} \]
\[ y \in Y: \{ \pm 1 \}^{N \times D_y} \]
\[ z \in Z: \{ \pm 1 \}^{N \times D_z} \]

Strategy: Search in superposition
\[ \hat{x}(t) = f(XX^T(s \odot \hat{y}(t - 1) \odot \hat{z}(t - 1))) \]
\[ \hat{y}(t) = f(YY^T(s \odot \hat{x}(t - 1) \odot \hat{z}(t - 1))) \]
\[ \hat{z}(t) = f(ZZ^T(s \odot \hat{x}(t - 1) \odot \hat{y}(t - 1))) \]

Use guesses in \( \hat{x} \), \( \hat{y} \) to infer \( \hat{z} \)

Constrain \( \hat{z} \) guesses to codebook \( Z \)

Combination of attractor networks and multiplication/binding
Performance of the resonator network
Tree search as factorization problem

\[
\begin{align*}
\text{tree} &= a \circ \text{left} \circ \rho(\text{left}) \circ \rho^2(\text{left}) \\
&+ b \circ \text{left} \circ \rho(\text{right}) \circ \rho^2(\text{left}) \\
&+ c \circ \text{right} \circ \rho(\text{right}) \circ \rho^2(\text{left}) \\
&+ d \circ \text{right} \circ \rho(\text{right}) \circ \rho^2(\text{right}) \circ \rho^3(\text{left}) \\
&+ e \circ \text{right} \circ \rho(\text{right}) \circ \rho^2(\text{right}) \circ \rho^3(\text{right}) \\
&+ f \circ \text{left} \circ \rho(\text{right}) \circ \rho^2(\text{right}) \circ \rho^3(\text{left}) \circ \rho^4(\text{left}) \\
&+ g \circ \text{left} \circ \rho(\text{right}) \circ \rho^2(\text{right}) \circ \rho^3(\text{left}) \circ \rho^4(\text{right})
\end{align*}
\]

\[
\text{tree} \circ (\text{left} \circ \rho(\text{right}) \circ \rho^2(\text{left})) = b + \text{noise}
\]
Using the resonator network to solve tree search

\[ \text{tree} \odot c = \text{right} \odot \rho(\text{right}) \odot \rho^2(\text{left}) + \text{noise}. \]

We denote each factor estimate as \( \hat{x}^{(0)}, \hat{x}^{(1)}, \hat{x}^{(2)}, \hat{x}^{(3)}, \hat{x}^{(4)} \) and the codebook matrices as \( X_0, X_1, X_2, X_3, X_4 \). Each codebook matrix contains permuted versions of \text{left} and \text{right}, and 1: \( X_d = [\rho^d(\text{left}), \rho^d(\text{right}), 1] \) where \( d \) indicates the depth in the tree. The network is constructed analogous to equation 3.2, but with five factor estimates running in parallel instead of three. For instance, the update equation for the first estimate is

\[
\hat{x}^{(0)}(t + 1) = g(X_0X_0^T (s \odot \hat{x}^{(1)}(t) \odot \hat{x}^{(2)}(t) \odot \hat{x}^{(3)}(t) \odot \hat{x}^{(4)}(t))).
\]
Resonator networks performance comparison

![Graph showing performance comparison between search in superposition and Gradient Search.](Image)
Resonator network convergence properties
Resonator networks for simple scene inference
How to represent continuous manifolds with complex HD vectors

VSA binding in the complex domain can form index patterns of continuous variables.

(1) **Distinguishability**: Points far apart are encoded by dissimilar “indexing” patterns.

(2) **Smoothness**: Points close to each other are encoded by similar patterns that smoothly transition.

\[ e^{i\phi_1 x(t)} \]

\[ \mathbf{v} \odot \mathbf{v} = \mathbf{v}^2 \]

\[ \mathbf{v} \odot \mathbf{v} \odot \sqrt{\mathbf{v}} = \mathbf{v}^{2.5} \]

Spinning Phasors:

\[ \mathbf{v}^x(t) \]

“exponentiation trick”
Computing on Functions Using Randomized Vector Representations

E. Paxton Frady, Denis Kleyko, Christopher J. Kynm, Bruno A. Olshausen, Friedrich T. Sommer
Programming the resonator network

\[ X = [x^1, x^2, \ldots, x^k] \]

\[ Y = [y^1, y^2, \ldots, y^l] \]

\[ Z = [t^a, t^b, \ldots, t^z] \]

\[ t^a = \sum_{x,y} x^x \bigodot y^y \cdot \text{Template}(x, y) \]
Resonator networks for simple scene inference
How to scale on neuromorphic hardware? How might the brain be doing it?

- Complex-valued matrix operations
- Patterns of sparse spike-timing activity

Sparse binding operation?

• Want something like the binding operation, but with sparse vectors
  • Operation produces quasi-orthogonal vector from two inputs
• Want atomic vectors to be sparse, and binding of atomic vectors should stay sparse
• Want minimal scaling of synaptic connections
Sparse VSA strategies

\[ \text{\# Synapses} = \left( \frac{N}{K} \right)^N \quad \text{\# Neurons} = N \quad \text{\# Active} = K \]

Modern neuroscience of cortical circuits
Active dendritic mechanisms in pyramidal neurons
Overview of active dendritic processing
Conclusions

• Factorization is an important problem in perception
• Solving factorization problems requires significant computation
• Resonator networks use the strategy of searching in superposition to solve factorization problems efficiently
• VSAs can be extended to represent geometry and binding can be used to perform geometrical transformations
• Sparse and Complex VSAs bring us much closer to linking VSAs with neuroscience
• Binding operations in sparse VSAs resembles active dendritic integration in pyramidal cells
• Cortical microcircuit is a resonator network??
Thanks!

REDWOOD CENTER
for Theoretical Neuroscience