The Challenge of Compositionality for AI is to develop an effective formalism for compositionality that resides in vector spaces. A proposed solution to this challenge is presented in a pair of recent papers written with several collaborators: Neurocompositional computing. Nascent forms of neurocompositionality give CNNs & Transformers their boosted power: 1G.

In their highly influential paper Connectionism and cognitive architecture: A critical analysis (Cognition, 1988) Jerry Fodor & Zenon Pylyshyn claimed that compositionality was a profound problem for neural networks (‘connectionism’)
- Debate at MIT in 1988 (bootleg recording available on youtube).

Sources


Thanks for financial support to NSF, Microsoft

Synopsis
✦ what: Neurocompositional Computing for AI
✦ why: deeper understanding in AI systems demands continuous compositional encodings
✦ key: continuous compositionality
✦ how: Vector-embed compositionality primitives in DL net: NECST computing
  - primitives: filler/role decomposition etc.
  - Vector embedding: Tensor Product Reps.
✦ assessment: Sufficiency of NECST:
  - Computability (in principle)
  - Learnability (in practice)
Current AI systems don’t understand the world is deeply compositional, to a good first approximation. DNNs’ power comes from continuity — in representations, processes & learning — need encodings that are simultaneously:

- continuous (learnable)
- compositional

i.e., continuous compositional structure

This defines neurocompositional computing.

Compositionality, intuitively

Information is encoded in structures composed of simpler encodings; understanding the whole is built by composing understanding of the parts e.g., understanding that plans are composed of sub-plans which are composed of sub-sub-plans scenes ... sub-scenes ... sub-sub-scenes phrases ... sub-phrases ... sub-sub-phrases inferences ... sub-inferences ... sub-sub-inferences formal expressions ... sub-expressions ... sub-sub-expressions

Synopsis

✦ what: Neurocompositional Computing for AI
✦ why: deeper understanding in AI systems demands continuous compositional encodings
✦ key: continuous compositionality
✦ how: Vector-embed compositionality primitives in DL net: NECST computing
  • primitives: filler/role decomposition etc.
  • Vector embedding: Tensor Product Reps.
✦ assessment: Sufficiency of NECST:
  • Computability (in principle)
  • Learnability (in practice)
they don’t have a strong bias pushing them to encode the world compositionally

which GOFAI symbolic AI did have ⇒ yielding robust (discrete) compositional generalization

but discreteness is very limiting

Continuous compositionality, intuitive examples

what is that?!?

current AI systems don’t understand that the world is deeply compositional, to a good first approximation

DNNs’ power comes from continuity — in representations, processes & learning — need encodings that are simultaneously

- continuous (learnable)
- compositional

i.e., *continuous compositional structure*

This defines *neurocompositional computing*

Synopsis

- what: Neurocompositional Computing for AI
- why: deeper understanding in AI systems demands continuous compositional encodings
- key: continuous compositionality

- how: Vector-embed compositionality primitives in DL net: NECST computing
  - primitives: filler/role decomposition etc.
  - Vector embedding: Tensor Product Reps.
- assessment: Sufficiency of NECST:
  - Computability (in principle)
  - Learnability (in practice)
Current AI systems don’t understand that the world is deeply compositional, to a good first approximation. DNNs’ power comes from continuity — in representations, processes & learning — need encodings that are simultaneously • continuous (learnable) • compositional

I.e., continuous compositional structure

This defines neurocompositional computing

Continuous compositionality, intuitive examples

- **Continuous content** in compositional structure

  French *ami* (‘friend’) pronounced as
tami nami zami ami

  *petit ami* un ami les amis joli ami

  Propose stored form in the mental dictionary is

  $[0.09*t + 0.09*n + 0.09*z]$ a m i

  which of t/n/z is pronounced (if any) is determined by the end of the previous word

  not a probabilistic mixture

- **Continuous structural relations**

  spatial relations in scenes

  1.2m-above-and-0.5m-to-the-left(painting, table)

Synopsis

✦ what: Neurocompositional Computing for AI

✦ why: deeper understanding in AI systems demands continuous compositional encodings

✦ key: continuous compositionality

• Vector embedding: Tensor Product Reps.

✦ assessment: Sufficiency of NECST:

  • Computability (in principle)
  • Learnability (in practice)

How do we employ the power of continuous vector representations to achieve compositional processing?

First: decompose ‘compositional structure’.
Then: embed in vector spaces so neural operations can compose and extract structure.

compositional structure composed of constituents each of which binds together
- content: what — ‘filler’
- form: where — ‘role’ (∼ relations)
constituent = binding filler:role
substructures: fillers can be entire structures extraction (unbinding): \( (S_1:L \& S_2:R) ÷ L = S_1 \)
compositionality: \( S = S_1:L \& S_2:R \Rightarrow \)
\[ f(S) = F(f(S ÷ L), f(S ÷ R)) \]
(systematicity: filler-independence)

Synopsis
✦ what: Neurocompositional Computing for AI
✦ why: deeper understanding in AI systems demands continuous compositional encodings
✦ key: continuous compositionality
✦ how: Vector-embed compositionality primitives in DL net: NECST computing
  • primitives: filler/role decomposition etc.
  • Vector embedding: Tensor Product Reps.
✦ assessment: Sufficiency of NECST:
  • Computability (in principle)
  • Learnability (in practice)

Filler:role decomposition

[lock able] =

= lock: [ ] & able: [ ]

= lock: \( L \) & able: \( R \)

ordered pair structural type is defined by roles:

\( L = [ - ] \quad R = [ - ] \)

ordered pair token defined by binding roles to fillers
Vector-embedding compositional structure

How do we employ the power of continuous vector representations to achieve compositional processing?
First: decompose ‘compositional structure’.
Then: embed in vector spaces so neural operations can compose and extract structure.

compositional structure composed of constituents each of which binds together
- content: what — ‘filler’
- form: where — ‘role’ (∼ relations)
constituent = binding filler:role
substructures: fillers can be entire structures
 extraction (unbinding): \( (S_1:L \& S_2:R) \div L = S_1 \)
compositionality: \( S = S_1:L \& S_2:R \Rightarrow f(S) = F(f(S\div L), f(S\div R)) \)
(systematicity: filler-independence)

Synopsis
✦ what: Neurocompositional Computing for AI
✦ why: deeper understanding in AI systems demands continuous compositional encodings
✦ key: continuous compositionality
✦ how: Vector-embed compositionality primitives in DL net: NECST computing
  • primitives: filler/role decomposition etc.
  • Vector embedding: Tensor Product Reps.
✦ assessment: Sufficiency of NECST:
  • Computability (in principle)
  • Note for discussion: standard NNs learn compositionality = TPRs !!
Vector-embedding compositional structure

- embed fillers as vectors: \( A \mapsto \vec{A} \) (standard)
- embed roles as vectors: \( L \mapsto \vec{L} \) (novel: key)

bind with tensor product: \( A:L \mapsto \vec{A} \otimes \vec{L} \)
aggregate with addition:
\[
A:L \& B:R \mapsto \vec{A} \otimes \vec{L} + \vec{B} \otimes \vec{R}
\]
unbind with inner product: \( S:L \mapsto \vec{S} \cdot \vec{L} \)
conditions on weight matrices computing \( f \)

**Synopsis**

- **what**: Neurocompositional Computing for AI
- **why**: Deeper understanding in AI systems
  - demands continuous compositional encodings
- **key**: Continuous compositionality
- **how**: Vector-embedding compositional primitives in DL net: NECST computing
  - primitives: filler/role decomposition etc.
  - Vector embedding: Tensor Product Reps.
- **assessment**: Sufficiency of NECST:
  - Computability (in principle)
  - Learnability (in practice)

**Tensor product, defined**
Collection of all products of elements of vectors

**Tensor Product Representations (TPRs)**
- Neurally Embedded Compositionaly Structured Tensor (NECST) computing
- key: continuous compositionality
- how: Vector-embedding compositionality primitives in DL net: NECST computing
  - primitives: filler/role decomposition etc.
  - Vector embedding: Tensor Product Reps.
- assessment: Sufficiency of NECST:
  - Computability (in principle)
  - Learnability (in practice)
NECST Computation

semantics
interpretation computation:

frog $\mapsto m_{\text{frog}}$
big frog $\mapsto f_{\text{big}}(m_{\text{frog}}) = [\lambda x. f_{\text{big}}(x)](m_{\text{frog}})$

$[\lambda x. P(x) \Rightarrow Q(x)](a) = [P(a) \Rightarrow Q(a)]$

Assessment: Sufficiency of NECST
- Computability (in principle)
- Learnability (in practice)
  - function-application
    (β-reduction in λ-calculus)
  - bind a variable to a value

Synopsis
- what: Neurocompositional Computing for AI
- why: deeper understanding in AI systems
  demands continuous compositional encodings
- key: continuous compositionality
- how: Vector-embed compositionality
  primitives in DL net: NECST computing
  - primitives: filler/role decomposition etc.
  - Vector embedding: Tensor Product Reps.
- assessment: Sufficiency of NECST:
  - Computability (in principle)
  - Learnability (in practice)

β-reduction
Basic operation of λ-calculus [function application]:

$$(\lambda x. B) A$$

Computable by neural network computation over TPRs!
Synopsis

✦ what: Neurocompositional Computing for AI
✦ why: deeper understanding in AI systems demands continuous compositional encodings
✦ key: continuous compositionality
✦ how: Vector-embed compositionality primitives in DL net: NECST computing
  • primitives: filler/role decomposition etc.
  • Vector embedding: Tensor Product Reps.
✦ assessment: Sufficiency of NECST:
  • Computability (in principle)
  • Learnability (in practice)

NECST Computation

syntactic compositionality:
Tree-Adjoining Grammar
✦ enables the level of complexity characteristic of human natural language syntax

Assessment: Sufficiency of NECST
• Computability (in principle)
• Learnability (in practice)
  ✦ function-application
    (β-reduction in λ-calculus)
  • bind a variable to a value
  ✦ tree adjoining (TAG)
Neurocompositionality↑ → compositional generalization↑

On simple symbol-manipulation tasks:
Data efficiency and success of out-of-distribution generalization
NECST: 100% at 700 examples
Transformer: ∼90% at 1500

Likelihood of perfect learning
NECST vs Transformer: improvement > 100%

Assessment: Sufficiency of NECST
• Computability (in principle)
• Learnability (in practice)

AI domains addressed by NECST models
• question-answering from Wikipedia text
• question-answering on simple narratives
• basic propositional reasoning
• problem-solving in math, programming
• generating text summaries

Synopsis
✦ what: Neurocompositional Computing for AI
✦ why: deeper understanding in AI systems demands continuous compositional encodings
✦ key: continuous compositionality
✦ how: Vector-embed compositionality primitives in DL net: NECST computing
  • primitives: filler/role decomposition etc.
  • Vector embedding: Tensor Product Reps.
✦ assessment: Sufficiency of NECST:
  • Computability (in principle)
  • Learnability (in practice)

2nd-generation NECST Models
Architectures
• RNNs with TPR hidden states
• TPR memory for tracking entity states, with provided mechanisms for retrieving, updating
• Transformer with TPR hidden states (TPT)
Synopsis
✦ what: Neurocompositional Computing for AI
✦ why: deeper understanding in AI systems demands continuous compositional encodings
✦ key: continuous compositionality
✦ how: Vector-embed compositionality primitives in DL net: NECST computing
  • primitives: filler/role decomposition etc.
  • Vector embedding: Tensor Product Reps.
✦ assessment: Sufficiency of NECST:
  • Computability (in principle)
  • Learnability (in practice)

Sources


Thanks for financial support to NSF, Microsoft