Modular Representations:
McCoy et al. 2019 & Andreas 2019

Rami
Hope
Ekin
The Question

To what extent do learned representations (continuous vectors) of symbolic structures (sentences, trees) exhibit compositional structure?

How to Measure Compositionality?

What does it mean to be compositional?
Big Picture

**McCoy et al. 2019**

measures how well

an RNN
can be approximated by

a Tensor Product Representation

**Andreas 2019**

measures how well

the true representation-producing model
can be approximated by

a model that explicitly composes primitive model representations
Big Picture

McCoy et al. 2019 measures how well an RNN can be approximated by a Tensor Product Representation.

Andreas 2019 measures how well the true representation-producing model can be approximated by a model that explicitly composes primitive model representations.
RNNs Implicitly Implement Tensor Product Representations
(McCoy et al. 2019)
Hypothesis

Neural networks trained to perform symbolic tasks will implicitly implement filler/role representations.

(McCoy et al. 2019)
OUTLINE

TPDNs: A way to approximate existing vector representations as TPRs

Synthetic Data: Can TPDNs Approximate RNN Autoencoder Representations?
- Q1: Do TPDNs even work? Can they approximate learned representations?
- Q2: Do different RNN architectures induce different representations?

Natural Data: What About Naturally Occurring Sentences?
- Q1: Can TPDNs approximate learned representations of natural language?
- Q2: How encodings approximated by TPDNs compare with original RNN encodings when used as sentence embeddings for downstream tasks?
- Q3: What can we learn by comparing minimally distant sentences (analogies)?

Synthetic Data: When do RNNs learn compositional representations?
- Q1: Effect of the architecture?
- Q2: Effect of the Training Task?

(McCoy et al. 2019)
OUTLINE

TPDNs: A way to approximate existing vector representations as TPRs

(McCoy et al. 2019)
TPDNs (Tensor Product Decomposition Networks)

Step 1: Train RNN (e.g. autoencoder)

(McCoy et al. 2019)
TPDNs (Tensor Product Decomposition Networks)

Step 2. Train TPDN to learn RNN encoding

Sequence w/ Hypothesized Role Scheme

TPDN (encoder)

Target: Minimize MSE

TPDN Encoding

RNN Encoding

(McCoy et al. 2019)
TPDNs (Tensor Product Decomposition Networks)

Represent sequence as filler:role pairs

Look up Filler and Role embeddings

Bind the filler & role vectors:
Filler vec \( \otimes \) Role vec

Sum tensor products

Flatten

Apply linear transformation \( M \)

\[
M(\text{flatten}(\sum_i r_i \otimes f_i))
\]
TPDNs (Tensor Product Decomposition Networks)

Step 3. Use trained TPDN (encoder) to assess whether a learned representation has (implicitly) learned compositional structure

If the output of decoding is correct, conclude that the TPDN is approximating RNN encoder well
OUTLINE

TPDNs: A way to approximate existing vector representations as TPRs

Synthetic Data: Can TPDNs Approximate RNN Autoencoder Representations?

- Q1: Do TPDNs even work? Can they approximate learned representations?
- Q2: Do different RNN architectures induce different representations?

(McCoy et al. 2019)
Can TPDNs Approximate RNN Autoencoder Representations?

**Data:** Digit Sequences e.g. 4, 3, 7, 9

**Architectures:** GRU with 3 types of encoder-decoders:

- Unidirectional
- Bidirectional
- Treebased

(McCoy et al. 2019)
Can TPDNs Approximate RNN Autoencoder Representations?

**Role Schemes**

<table>
<thead>
<tr>
<th>Role Scheme</th>
<th>Example Sequence: (4, 3, 7, 9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unidirectional (left-to-right)</td>
<td>(4: \text{first} + 3: \text{second} + 7: \text{third} + 9: \text{fourth})</td>
</tr>
<tr>
<td>Unidirectional (right-to-left)</td>
<td>(4: \text{fourth-to-last} + 3: \text{third-to-last} + 7: \text{second-to-last} + 9: \text{last})</td>
</tr>
<tr>
<td>Bidirectional</td>
<td>(4: (\text{first, fourth-last}) + 3: (\text{second, third-last}) + 7: (\text{third, second-last}) + 9: (\text{fourth, last}))</td>
</tr>
<tr>
<td>Bag of words</td>
<td>(4: \text{r0} + 3: \text{r0} + 7: \text{r0} + 9: \text{r0})</td>
</tr>
<tr>
<td>Wickelroles</td>
<td>(4: #<em>3 + 3: 4_7 + 7: 3_9 + 9: 7_6 + 6: 9</em>{6})</td>
</tr>
<tr>
<td>Tree positions</td>
<td>(4: \text{LLL} + 3: \text{LLRL} + 7: \text{LLRR} + 9: \text{LR} + 6: \text{R})</td>
</tr>
</tbody>
</table>

(McCoy et al. 2019)
Can TPDNs Approximate RNN Autoencoder Representations?

**Hypothesis:** RNN autoencoders will learn to use role representations that parallel their architectures:

- unidirectional network  →  left-to-right roles
- bidirectional network   →  bidirectional roles
- tree-based network      →  tree-position roles

**Experiments:**

(6 Role schemes) X (3 Architectures) = 18 experiments

(McCoy et al. 2019)
Can TPDNs Approximate RNN Autoencoder Representations?

Results! Do the results match the hypothesis?

- Green checkmark: tree-based autoencoder
- Red question mark: Unidirectional auto-encoder
- Red cross: Bidirectional auto-encoder

Takeaways:
- Architecture affects Learned Representation
- Roles used sometimes (but not always) parallel the architecture
- Missing role hypotheses? Different structure-encoding scheme other than TPRs?

(McCoy et al. 2019)
OUTLINE

TPDNs: A way to approximate existing vector representations as TPRs

Synthetic Data: Can TPDNs Approximate RNN Autoencoder Representations?

- Q1: Do TPDNs even work? Can they approximate learned representations? [non-exhaustive YES]
- Q2: Do different RNN architectures induce different representations? [YES, but not always as expected]
OUTLINE

TPDNs: A way to approximate existing vector representations as TPRs

Synthetic Data: Can TPDNs Approximate RNN Autoencoder Representations?
- Q1: Do TPDNs even work? Can they approximate learned representations? [non-exhaustive YES]
- Q2: Do different RNN architectures induce different representations? [YES, but not always as expected]

Natural Data: What About Naturally Occurring Sentences?
- Q1: Can TPDNs approximate learned representations of natural language?
- Q2: How do TPDN encodings compare with the original RNN encodings as sentence embeddings for downstream tasks?
- Q3: What can we learn by comparing minimally distant sentences (analogies)?
Naturally Occurring Sentences

1. Can TPDNs approximate natural language RNN encodings?

<table>
<thead>
<tr>
<th>Models</th>
<th>Model Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InferSent</td>
<td>BiLSTM trained on SNLI</td>
</tr>
<tr>
<td>Skip-thought</td>
<td>LSTM trained to predict the sentence before or after a given sentence</td>
</tr>
<tr>
<td>SST</td>
<td>tree-based recursive neural tensor network trained to predict movie review sentiment</td>
</tr>
<tr>
<td>SPINN</td>
<td>tree-based RNN trained on SNLI</td>
</tr>
</tbody>
</table>
Naturally Occurring Sentences

1. Can TPDNs approximate natural language RNN encodings?

Sentence Embedding Evaluation Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Task Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SST</td>
<td>rating the sentiment of movie reviews</td>
</tr>
<tr>
<td>MRPC</td>
<td>classifying whether two sentences paraphrase each other</td>
</tr>
<tr>
<td>STS-B</td>
<td>labeling how similar two sentences are</td>
</tr>
<tr>
<td>SNLI</td>
<td>determining if one sentence entails or contradicts a second sentence, or neither</td>
</tr>
</tbody>
</table>
Naturally Occurring Sentences

1. Can TPDNs approximate natural language RNN encodings?

Evaluation (per task):

**Step 1:** Train classifier on top of RNN encoding to perform the task

**Step 2:** Freeze classifier and use to classify TPDN encodings

Metric: Proportion matching

(McCoy et al. 2019)
Naturally Occurring Sentences

1. Can TPDNs approximate natural language RNN encodings?

Results!

- “no marked difference between bag-of-words roles and other role schemes”

- “...except for the SNLI task” (entailment & contradiction prediction)
  - Tree-based model best-approximated with tree-based roles

- Skip-thought cannot be approximated well with any role scheme we considered

(McCoy et al. 2019)
What About Naturally Occurring Sentences?

3. Analogies: Minimally Distant Sentences

I see now − I see = you know now − you know

(I:0 + see:1 + now:2) − (I:0 + see:1 ) = (you:0 + know:1 + now:2) − (you:0 + know:1)

Both Simplify to:  now:2

Therefore:  I see now − I see = you know now − you know

Contingent On: the left-to-right role scheme  “role-diagnostic analogy”

(McCoy et al. 2019)
What About Naturally Occurring Sentences?

3. Analogies: Minimally Distant Sentences

Evaluation:

**Step 1:** Construct Dataset of analogies, where each analogy only holds for one role scheme

**Step 2:** Calculate *Euclidean Distance* between sentences in Analogy using TPDN approximations using different role schemes
What About Naturally Occurring Sentences?

3. Analogies: Minimally Distant Sentences

Results!

- InferSent, Skip-thought, and SPINN most consistent with bidirectional roles
- bag-of-words column shows poor performance by all models
What About Naturally Occurring Sentences?

3. Analogies: Minimally Distant Sentences

Takeaways

- Poor performance for bag-of-words: In controlled enough settings these models can be shown to have some more structured behavior even though evaluation on examples from applied tasks does not clearly bring out that structure.
- These models have a weak notion of structure, but that structure is largely drowned out by the non-structure-sensitive, bag-of-words aspects of their representations.
TPDNs: A way to approximate existing vector representations as TPRs

Synthetic Data: Can TPDNs Approximate RNN Autoencoder Representations?
  - Q1: Do TPDNs even work? Can they approximate learned representations?
  - Q2: Do different RNN architectures induce different representations?

Natural Data: What About Naturally Occurring Sentences?
  - Q1: Can TPDNs approximate learned representations of natural language?
  - Q2: How encodings approximated by TPDNs compare with original RNN encodings when used as sentence embeddings for downstream tasks?
  - Q3: What can we learn by comparing minimally distant sentences (analogies)?

Synthetic Data: When do RNNs learn compositional representations?
  - Q1: Effect of the architecture?
  - Q2: Effect of the Training Task?
When do RNNs Learn Compositional Structure?

1. Architecture

- Repeat synthetic data experiments with **different architecture for encoder vs. decoder**

Results!

- The **decoder** had much more influence on the role representation
- The encoder still had some influence

(McCoy et al. 2019)
2. Training Task

Tasks:

- autoencoding
- reversal
- sorting (note: does not require any structural information about the input)
- interleaving
When do RNNs Learn Compositional Structure?

2. Training Task

Results!

<table>
<thead>
<tr>
<th>Task</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoencoding</td>
<td>mildly bidirectional roles (favoring left-to-right)</td>
</tr>
<tr>
<td>reversal</td>
<td>right-to-left direction &gt;&gt; left-to-right</td>
</tr>
<tr>
<td>sorting</td>
<td>bag-of-words ~ rest of role schemes</td>
</tr>
<tr>
<td>interleaving</td>
<td>bidirectional roles &gt;&gt; unidirectional roles</td>
</tr>
</tbody>
</table>

Takeaways

- Model learns to discard/ignore structure when it is not needed for the task…
- that is, **RNNs only learn structure when it is needed**

(McCoy et al. 2019)
Conclusions

1. Recurrent neural networks can learn compositional representations of symbolic structures

   **but don’t always do so in practice**

2. Factors affecting whether RNNs learn compositional representations:
   - Architecture, e.g. decoder
   - Training Task

3. Popular sentence-encoding natural language models lack systematic structure

(McCoy et al. 2019)
Discussion

- Differences in the capabilities between TPDNs and RNNs

- When it works to replace an RNN Encoder with an TPDN Encoder, what does that mean? What about if it fails?

- What are the limitations of this approach with respect to measuring compositionality?

(McCoy et al. 2019)
Measuring Compositionality in Representation Learning
(Andreas 2019)
Big Picture

McCoy et al. 2019 measures how well an RNN can be approximated by a Tensor Product Representation.

Andreas 2019 measures how well the true representation-producing model can be approximated by a model that explicitly composes primitive model representations.
Outline

1. Motivation
2. Tree Reconstruction Error: A standard measure for compositionality
3. How measured compositionality relates to
   a. Learning dynamics
   b. Human judgements
   c. Out-of-distribution generalization
Motivation

- Philosophical motivators:
  - Fodor, Lewis, Carnap, Montague
  - Not very general

- Emergent communication lit
  - Not at all quantitative (ad-hoc human)

Finite semantics that maps onto the world is the desideratum, seems impossible

Algebraic interpretation of all semantics

Parts yield pure math/logical syntax without meaning

Whole + syntax

\[
\{\{\text{dark, blue}\}, \text{square}\} \quad \rightarrow \quad \text{aaxx}
\]

\[
\{\text{green, square}\} \quad \rightarrow \quad \text{aazz}
\]

\[
\{\text{blue, triangle}\} \quad \rightarrow \quad \text{bbmx}
\]

\[
\{\text{green triangle}\} \quad \rightarrow \quad \text{bbby}
\]
TRE: A Measure For Compositionality

A standard quantitative measure for learned (vector) representations
Symbolic Compositionality

TRE assumes the symbolic structure for the inputs known as derivation trees

Input ($x$)
- dark blue triangle
- yellow square
- green circle

Derivation ($d$)
- ((dark, blue), triangle)
- (yellow, square)
- (green, circle)

Derivation Oracle ($D$)
Traditional View on Compositionality of Representations

**Intuition:** Representations are compositional if each $f(x)$ is fully determined by the structure of $D(x)$

Define a composition operator: $\theta_a \asymp \theta_b \Rightarrow \theta$

**Exact Compositionality:**

$$D(x) = (D(x_a), D(x_b)) \Rightarrow f(x) = f(x_a) \asymp f(x_b)$$

Assumes that $f$ can produce representations for primitives!

(Andreas. 2019)
Problem with the Traditional View

How do we identify lexicon entries: the primitive parts from which representations are constructed?

How do we define the composition operator \( \star \)?

What do we do with languages but for which the homomorphism condition cannot be made to hold exactly?
TRE for Compositionality of Representations

Representations are compositional if each $f(x)$ is determined well approximated by the structure of $D(x)$

Define Learn a composition operator: $\theta_a \ast \theta_b \mapsto \theta$, and learn a compositional function $f_\eta$ given $D$ such that:

$$D(x) = (d_a, d_b) \Rightarrow f_\eta(D(x)) = f_\eta(d_a) \ast f_\eta(d_b)$$

Learn the primitive representations:

$$f_\eta(x) = \eta_i \text{ for all } D(x) \in D_0$$

(Andreas. 2019)
TRE for Compositionality of Representations

Find the closest compositional approximation \((f_\eta \circ D)\) to the true model \(f\) under a learned composition operator \((\star)\)

TRE is the approximation error between \(f\) and \(f_\eta \circ D\)!

(Andreas. 2019)
TRE for Compositionality of Representations

**Minimize** the approximation error on the training data w.r.t a $\delta$:

$$\eta^* = \arg \min \sum \delta(f(x), f_\eta(x))$$

Model representations are compositional if each $f(x)$ is well approximated by a compositional function, $f_{\eta^*}(x)$ under $D(x)$:

$$\text{TRE}(x) = \delta(f(x), f_{\eta^*}(x)) \ll 1$$

(Andreas. 2019)
Problems with TRE

If every \( x \in \mathcal{X} \) assigned a unique derivation. Then there is always some \( \ast \) that achieves \( \text{TRE}(\mathcal{X}) = 0 \), by setting \( f_n = f \), and defining \( \ast \) such that:

\[
f(x) = f(x_a) \ast f(x_b) \quad \text{for all} \quad x, x_a, x_b
\]

Pre-commitment to a limited family of \( \ast \) operators like linear operators
Discussion

- How does TPR approximation compare to TRE?

- TRE assumes unlabeled derivation tree for the inputs. How could we enable explicit filler/role structure in TRE framework?

- How can we relax assumptions on composition functions and known derivation oracle?
Figure 3: Relationship between reconstruction error \( \text{TRE} \) and mutual information \( I(\theta; X) \) between inputs and representations. (a) Evolution of the two quantities over the course of a single run. Both initially increase, then decrease. The color bar shows the training epoch. (b) Values from ten training runs. (c) Values from the second half of each training run, taken to begin when \( I(\theta; X) \) reaches a maximum. In (b) and (c), the observed correlation is significant: respectively \( (r = 0.70, p < 1e-10) \) and \( (r = 0.71, p < 1e-8) \).
Compositionality vs Human Judgements

- Bigrams \(<w_1, w_2>\)
- using FastText 100d vector
- instance based TRE
- Humans rated “most compositional” -- low TRE
  - application form, polo shirt, research project
- Humans rated “least compositional” -- high TRE
  - fine line, lip service, and nest egg.
- TRE values were anti-correlated with Human ratings (0-5)

(Andreas. 2019)
Compositionality vs Similarity Metrics

- Tree Distance vs TRE distance
- According to the distance function, two representations that are close together will definitionally have low TRE
- Even if representations are similar, and this can be captured by TRE, the functions that produce these representations may still be very different and we may not have the correct distance metric?
Compositionality vs Generalization

\[ \theta \star \theta' = A\theta + B\theta' \]

Figure 4: The communication task: A *speaker* model observes a pair of target objects, and sends a description of the objects (as a discrete code) to a *listener* model. The listener attempts to reconstruct the targets, receiving fractional reward for partially-correct predictions.
Figure 5: Relationship between TRE and reward. (a) Compositional languages exhibit lower generalization error, measured as the difference between train and test reward ($r = 0.50$, $p < 1e^{-6}$). (b) However, compositional languages also exhibit lower absolute performance ($r = 0.57$, $p < 1e^{-9}$). Both facts remain true even if we restrict analysis to “successful” training runs in which agents achieve a reward $> 0.5$ on held-out referents ($r = 0.6$, $p < 1e^{-3}$ and $r = 0.38$, $p < 0.05$ respectively).
Conclusions + qs for discussion

Do we believe these expts?

Compare to SCAN & CLUTTR

Could we apply TRE to discrete representations?

Davli (individual neurons represent something like the filler-role) & Weiss (is there structure in the clusters)?

“how to generalize TRE to the setting where oracle derivations are not available”
Discussion
:)  

bye!