

Learning High-Order Word Representations

Konstantinos Kogkalidis

June 12, 2018

LoLa Fan Club

Outline

Motivation

Supervised Learning

Dataset

Intermediate Representations

Transferring Knowledge

Evaluation

Conclusion

Bag of Tricks

Motivation

Idea: structure-preserving map \mathcal{F}

$$\mathcal{F} : \mathcal{G} \rightarrow \mathbf{FdVect}$$

- Atomic types translated to vectors (order-one tensors)
- Complex types translated to (multi-)linear maps (higher order tensors)

Why Compositionality?

- Bridging of formal & distributional semantics
- Syntax-informed meaning derivations
- Modeling of functional words
- Formal treatment of ambiguous words
- Richer representations
- \vdots

Why Not Compositionality?

- ✓ Great properties
- ? How to obtain word representations?

Possible options:

Why Not Compositionality?

- ✓ Great properties
- ? How to obtain word representations?

Possible options:

1. Co-occurrence statistics

Why Not Compositionality?

✓ Great properties

? How to obtain word representations?

Possible options:

1. Co-occurrence statistics ✗
2. Unsupervised techniques (*a la word2vec*)

Why Not Compositionality?

✓ Great properties

? How to obtain word representations?

Possible options:

1. Co-occurrence statistics ✗
2. Unsupervised techniques (*a la word2vec*) ✗
3. Supervised learning

Why Not Compositionality?

- ✓ Great properties
- ? How to obtain word representations?

Possible options:

1. Co-occurrence statistics ✗
2. Unsupervised techniques (*a la word2vec*) ✗
3. Supervised learning ?

Problem Statement

Examine whether supervised learning can be used to find higher-order word representations (transitive verbs)

Supervised Learning

- Search over set of functions $A \rightarrow B$ parameterized over P
- Find optimal approximation \hat{f}_P to $f: A \rightarrow B$
- Use samples $(a, f(a)) \in A \times B$ to update P

Supervised Learning

Dataset

Sample space must be:

- Labeled
- constrained
- of large size
- of high quality

Raw text paraphrase pairs

Example pair

proposed by the president ~ suggested by the chairman

- Labeled ✓
- constrained ✗ (different syntactic types)
- of large size ✓
- of high quality ?

1. **Parse and filter by type** (transitive verb case)

- Labeled ✓
- constrained ✓
- of large size ✗ (>95% loss)
- of high quality ✗ (*parser-induced errors*)

2. **Back-translation**

- Labeled ✓
- constrained ✓
- of large size ✓
- of high quality ✗ (*translation-induced errors*)

3. **Filter by co-occurrence / mutual information**

- Labeled ✓
- constrained ✓
- of large size ✓
- of high quality ?

Verb / object dictionaries:

$$\mathcal{V} : \{v_1 : 1, v_2 : 2, \dots, v_N : N\}$$

$$\mathcal{O} : \{o_1 : 1, o_2 : 2, \dots, o_M : M\}$$

Paraphrase relation:

$$\mathcal{P} : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \rightarrow \{0, 1\} \quad (\text{binary classification})$$

$$\mathcal{P}(i, j, k, l) = \mathcal{P}(k, l, i, j) = \begin{cases} 1 & v_i o_j \sim v_k o_l \\ 0 & \text{otherwise} \end{cases}$$

Supervised Learning

Intermediate Representations

Objective Function

Our semantic interpretations are:

- Actions: $[a] = \mathbb{R}^A$
- Objects: $[np] = \mathbb{R}^{NP}$
- Transitive Verbs: $[a/np] = \mathbb{R}^{A \times NP}$

And our objective is to learn a **verb embedding function** ϵ_{verb} :

$$\epsilon_{verb} : \mathbb{N} \rightarrow \mathbb{R}^{A \times NP}$$

But instead we have samples from some $f : \mathbb{N}^4 \rightarrow \{0, 1\}$

Formulating the network

Solution

Formulate f_p to incorporate ε_{verb} .

$$f_p = f_1 \circ f_2 \circ \dots \circ \varepsilon_{verb} \circ \dots$$

Formulating the network

Solution

Formulate f_p to incorporate ε_{verb} .

$$f_p = f_1 \circ f_2 \circ \dots \circ \varepsilon_{verb} \circ \dots$$

Simplification (1)

Assume pre-trained **object embedding function** ε_{object} .

$$\varepsilon_{object} : \mathbb{N} \rightarrow \mathbb{R}^{300}$$

Filling the missing blocks

$$i \in \mathbb{N}$$

$$j \in \mathbb{N}$$

$$k \in \mathbb{N}$$

$$l \in \mathbb{N}$$

$$\hat{y} \in \mathbb{R}$$

Filling the missing blocks

$$i \in \mathbb{N}$$

$$j \in \mathbb{N}$$

$$k \in \mathbb{N}$$

$$l \in \mathbb{N}$$

ε_{object}

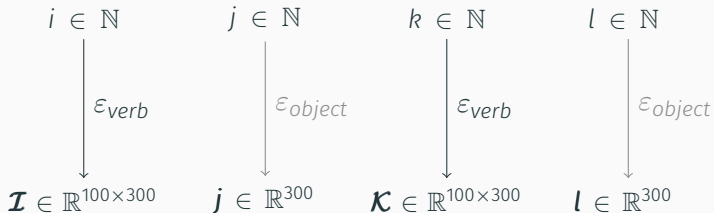
$$j \in \mathbb{R}^{300}$$

ε_{object}

$$l \in \mathbb{R}^{300}$$

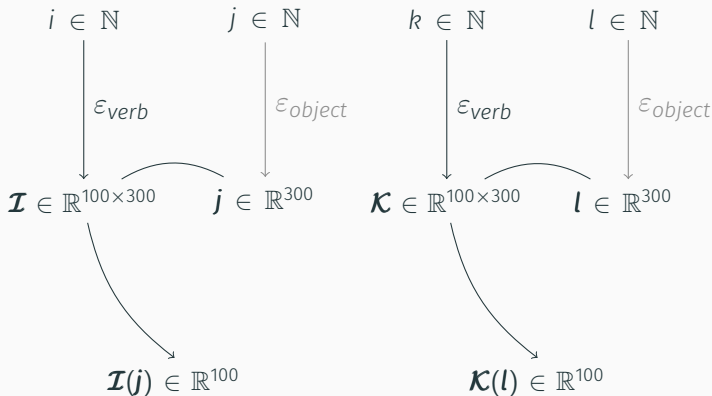
$$\hat{y} \in \mathbb{R}$$

Filling the missing blocks



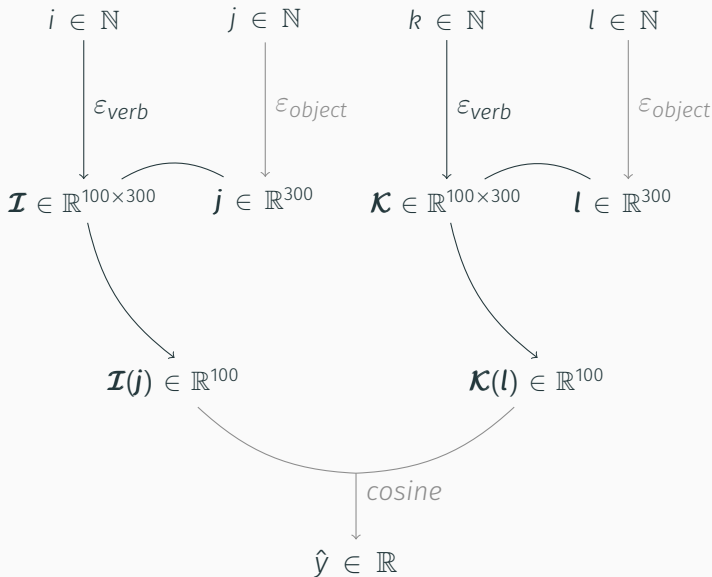
$$\hat{y} \in \mathbb{R}$$

Filling the missing blocks



$$\hat{y} \in \mathbb{R}$$

Filling the missing blocks



Error Signal

$$\cos(\mathbf{V}_i(\mathbf{j}), \mathbf{V}_k(\mathbf{l})) \approx \mathcal{P}(i, j, k, l) \quad \forall (i, j, k, l) \in \mathcal{V} \times \mathcal{O} \times \mathcal{V} \times \mathcal{O}$$

Error Signal

$$\cos(\mathbf{V}_i(\mathbf{j}), \mathbf{V}_k(\mathbf{l})) \approx \mathcal{P}(i, j, k, l) \quad \forall (i, j, k, l) \in \mathcal{V} \times \mathcal{O} \times \mathcal{V} \times \mathcal{O}$$

Considerations:

1. Network Size

- 1.000 verbs
 - $300 \times 100 = 30.000$ parameters per verb
- \Rightarrow 30.000.000 parameters for ε_{verb} to learn

Error Signal

$$\cos(\mathbf{V}_i(\mathbf{j}), \mathbf{V}_k(\mathbf{l})) \approx \mathcal{P}(i, j, k, l) \quad \forall (i, j, k, l) \in \mathcal{V} \times \mathcal{O} \times \mathcal{V} \times \mathcal{O}$$

Considerations:

1. Network Size

- 1.000 verbs
 - $300 \times 100 = 30.000$ parameters per verb
- \Rightarrow 30.000.000 parameters for ε_{verb} to learn

2. Quantifying over two spaces ...

Error Signal

$$\cos(\mathbf{V}_i(j), \mathbf{V}_k(l)) \approx \mathcal{P}(i, j, k, l) \quad \forall (i, j, k, l) \in \mathcal{V} \times \mathcal{O} \times \mathcal{V} \times \mathcal{O}$$

Considerations:

1. Network Size

- 1.000 verbs
 - $300 \times 100 = 30.000$ parameters per verb
- \Rightarrow 30.000.000 parameters for ε_{verb} to learn

2. Quantifying over two spaces ...

3. ... both of which are non-convex



... A "beast" to train ☹️

Supervised Learning

Transferring Knowledge

Finding an Oracle

Dataception: use our labeled dataset to create a new dataset

Finding an Oracle

Dataception: use our labeled dataset to create a new dataset

Simplification (2)

Assume another pre-trained verb embedding function ϵ'_{verb} .

$$\epsilon'_{verb} : \mathbb{N} \rightarrow \mathbb{R}^{300}$$

Finding an Oracle

Dataception: use our labeled dataset to create a new dataset

Simplification (2)

Assume another pre-trained **verb embedding function** ϵ'_{verb} .

$$\epsilon'_{verb} : \mathbb{N} \rightarrow \mathbb{R}^{300}$$

Oracle

We can now use a **paraphrase embedding function** ϵ_{par} .

$$\epsilon_{par} : \mathbb{R}^{300} \times \mathbb{R}^{300} \rightarrow \mathbb{R}^{100}$$

$$i \in \mathbb{N}$$

$$j \in \mathbb{N}$$

$$k \in \mathbb{N}$$

$$l \in \mathbb{N}$$

$$\hat{y} \in \mathbb{R}$$

Oracle Flow

$$i \in \mathbb{N}$$

$$j \in \mathbb{N}$$

$$k \in \mathbb{N}$$

$$l \in \mathbb{N}$$

ε_{object}

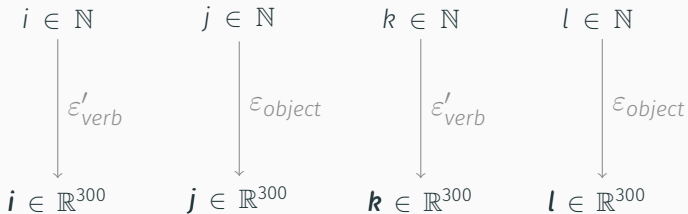
$$j \in \mathbb{R}^{300}$$

ε_{object}

$$l \in \mathbb{R}^{300}$$

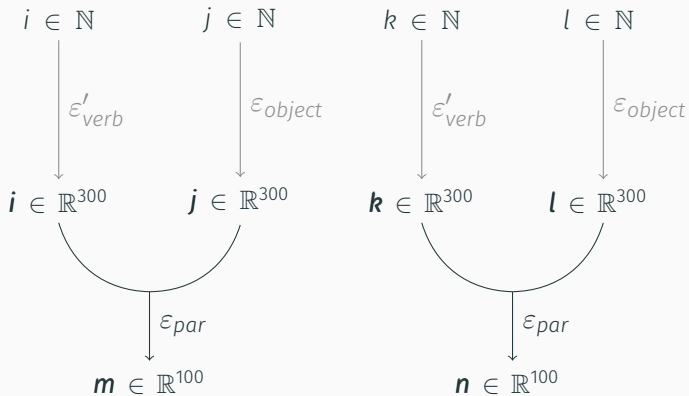
$$\hat{y} \in \mathbb{R}$$

Oracle Flow



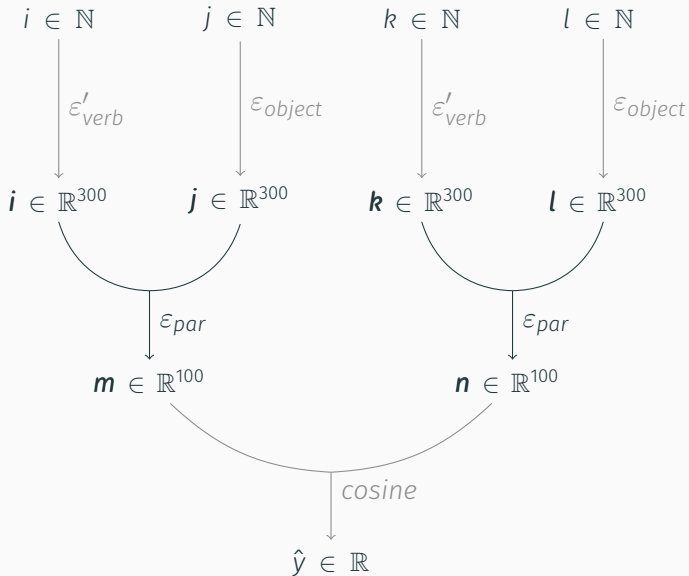
$$\hat{y} \in \mathbb{R}$$

Oracle Flow



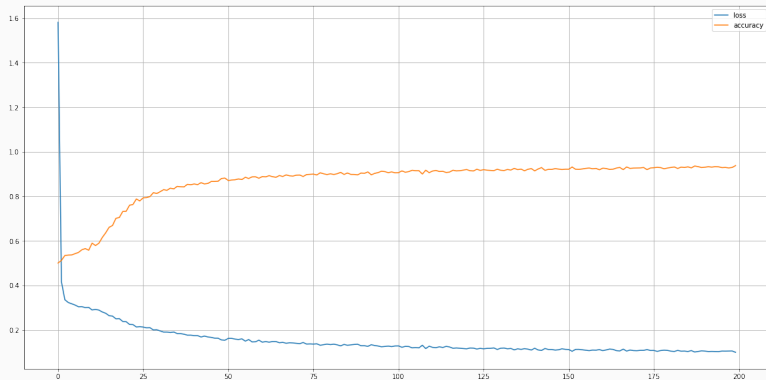
$$\hat{y} \in \mathbb{R}$$

Oracle Flow



Training the Oracle

ϵ_{par} : recurrent autoencoder (≈ 700.000 parameters)

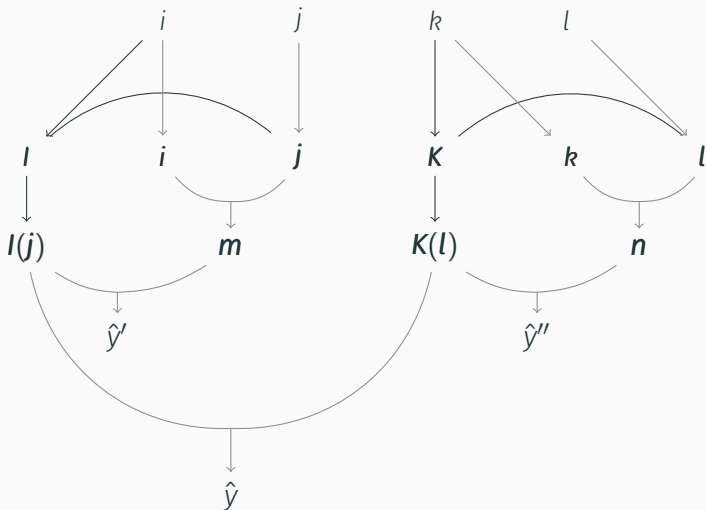


New Error Signal

$$\cos(\mathbf{V}_i(\mathbf{j}), \varepsilon_{par}(\mathbf{v}, \mathbf{j})) \approx 1 \quad \forall (i, j) \in \mathcal{V} \times \mathcal{O}$$

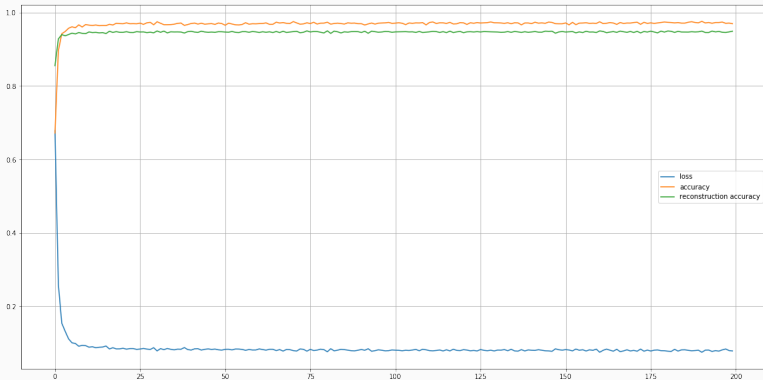
- ε_{par} gives us paraphrase embeddings '*for free*'
- We can use them to facilitate training
- Much smaller problem space

Composing Networks



Training the Original

ϵ_{verb} : tanh activated dense layer ($\approx 30.000.000$ parameters)





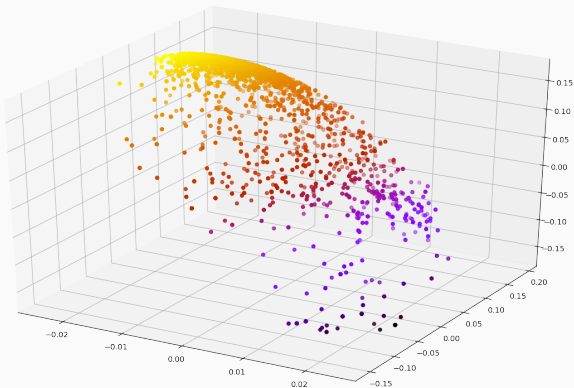
The beast has been tamed! 😊

Evaluation

Task-specific performance relates to the small-scale structure of the learned space:

Ground Truth \ Prediction	Oracle		Final	
	T	\perp	T	\perp
T	0.92	0.08	0.88	0.02
\perp	0.08	0.92	0.12	0.98

3D PCA on paraphrase embeddings



Conclusion

1. Uninformative error signal ?
2. Over-parameterization
 - a) Encoder / decoder architectures ?
 - b) Linearity constraint ✗
 - c) Chasing after an oracle ?
 - d) Bad scaling ✗

Next Steps

1. Directly evaluate verb matrices
2. More structural constraints (activity regularization)
3. Iterative learning
4. Other data formats:
 - Different syntactic types
 - Different labels / samples altogether
5. Different oracle architectures

Bag of Tricks

1. Negative Sampling
2. Curriculum Learning / Domain Adaptation
3. Loss Mixing