Modeling and Learning Semantic Co-Compositionality through Prototype Projections and Neural Networks

Masashi Tsubaki, Kevin Duh, Masashi Shimbo, Yuji Matsumoto

Nara Institute of Science and Technology (NAIST), Japan
Two contributions in our work

New model of compositionality in word vector space

Unsupervised word vector re-training algorithm considering compositionality
Two contributions in our work

- **New model of compositionality in word vector space**
- **Unsupervised word vector re-training algorithm considering compositionality**
From word to **phrase representation** with matrix-vector operation

[Mitchell and Lapata 08], [Baroni and Zamparell 10], [Socher+ 12], [Van de Cruys+ 13]
Modeling of compositionality in word vector space

From word to **phrase representation** with matrix-vector operation

[Mitchell and Lapata 08], [Baroni and Zamparell 10], [Socher+ 12], [Van de Cruys+ 13]
Modeling of compositionality in word vector space

From word to phrase representation with matrix-vector operation

[Mitchell and Lapata 08], [Baroni and Zamparell 10], [Socher+ 12], [Van de Cruys+ 13]
Modeling of compositionality in word vector space

From word to phrase representation with matrix-vector operation
[Mitchell and Lapata 08], [Baroni and Zamparell 10], [Socher+ 12], [Van de Cruys+ 13]

New model inspired by Co-Compositionality
Main Idea: Co-Compositionality [Pustejovsky 1995]

Co-compositionality

Verb and object are allowed to modify each other’s meanings and generate the overall semantics.
Main Idea: Co-Compositionality [Pustejovsky 1995]

Co-compositionality

Verb and object are allowed to modify each other’s meanings and generate the overall semantics.

\[ f(\text{run, company}) = \text{operate} \]

\[ f(\text{run, marathon}) = \text{race} \]
Main Idea: Co-Compositionality [Pustejovsky 1995]

Co-compositionality

Verb and object are allowed to modify each other’s meanings and generate the overall semantics.

\[ f(\text{run}_{\text{company}}, \text{company}) = \text{operate} \]

\[ f(\text{run}_{\text{marathon}}, \text{marathon}) = \text{race} \]
Main Idea: Co-Compositionality [Pustejovsky 1995]

Co-compositionality

Verb and object are allowed to modify each other’s meanings and generate the overall semantics.

\[ f( \text{run}_{\text{company}}, \text{company}_{\text{run}} ) = \text{operate} \]

\[ f( \text{run}_{\text{marathon}}, \text{marathon}_{\text{run}} ) = \text{race} \]
Main Idea: Co-Compositionality [Pustejovsky 1995]

Co-compositionality

Verb and object are allowed to modify each other’s meanings and generate the overall semantics.

\[ f( \text{run}_{\text{company}}, \text{company}_{\text{run}} ) = \text{operate} \]

\[ f( \text{run}_{\text{marathon}}, \text{marathon}_{\text{run}} ) = \text{race} \]
Our Model

Main Idea: Co-Compositionality [Pustejovsky 1995]

Verb and object are allowed to modify each other’s meanings and generate the overall semantics.

\[
f( \text{run}_{\text{company}}, \text{company}_{\text{run}} ) = \text{operate}
\]

\[
f( \text{run}_{\text{marathon}}, \text{marathon}_{\text{run}} ) = \text{race}
\]

Question

How do we implement co-compositionality in vector space?
Prototype Projection

Matrix-vector operation as an implementation for Co-Compositionality
Co-Compositionality with Prototype Projections

Our Model

run

company
Co-Compositionality with Prototype Projections

Our Model

Prototype verbs of “company”

run

company

VerbOf

start build buy
Co-Compositionality with Prototype Projections

Our Model

Prototype verbs of “company”

VerbOf

run

start
build
buy

≈

×

company

Our Model
Co-Compositionality with Prototype Projections

Our Model

Prototype verbs of “company”

VerbsOf

≈

run

company

Latent subspace formed by prototype verb vectors of company
Co-Compositionality with Prototype Projections

Our Model

Prototype verbs of “company”

\[ \text{run}_{\text{company}} \]

\[ P_{\text{company}} = V^T V \]

(Orthogonal projection matrix to V)

VerbOf

Latent subspace formed by prototype verb vectors of company
Co-Compositionality with Prototype Projections

Prototype Projection

\[ P_{\text{company}} = V^TV \]

(Orthogonal projection matrix to V)

Prototype verbs of “company”

Latent subspace formed by prototype verb vectors of company
Co-Compositionality with Prototype Projections

Our Model

Prototype Projection

Prototype verbs of “company”

Prototype objects of “run”

VerbOf

ObjectOf

\( \text{run}_{\text{company}} \)

\( \text{P}_{\text{company}} = V^T V \)

(Orthogonal projection matrix to \( V \))
Co-Compositional with Prototype Projections

Prototype Projection

Prototype verbs of “company”

Prototype objects of “run”

Our Model

Prototype verbs of “company”

Prototype objects of “run”

Orthogonal projection matrix to $V$
Co-Compositionality with Prototype Projections

Prototype Projection

\[
\text{run}_{\text{company}} 
\]

\[
\text{company}_{\text{run}}
\]

\[
P_{\text{company}} = V^TV
\]

\[
P_{\text{run}} = O^TO
\]

Prototype verbs of “company”

Prototype objects of “run”
Co-Compositionality with Prototype Projections

Our Model

Prototype Projection

Operate =

\[ \text{operate} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} \cdots \end{bmatrix} \]

\[ \text{run}_{\text{company}} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \]

\[ \text{run} = \begin{bmatrix} \cdots \end{bmatrix} \]

\[ \text{company}_{\text{run}} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \]

\[ P_{\text{company}}^{\text{run}} = V^TV \]

\[ P_{\text{run}}^{\text{company}} = O^TO \]

Prototype verbs of “company”

Prototype objects of “run”

Prototype verbs of “company”

Prototype objects of “run”
Tease out the proper semantics from aggregate representation by projection to latent space.

Assume various senses of “run” are aggregated in one vector.

Prototype verbs of “company”

Prototype verbs of “marathon”

Intuitive image of prototype projection.
## Evaluation : Verb disambiguation in subject-verb-object triples

### Evaluation dataset [Grefenstette and Sadrzadeh 11]

<table>
<thead>
<tr>
<th>Subj-Verb-Obj</th>
<th>Landmark verb</th>
<th>Similarity of human judgment</th>
</tr>
</thead>
<tbody>
<tr>
<td>People-run-company</td>
<td>operate</td>
<td>7</td>
</tr>
<tr>
<td>People-run-company</td>
<td>move</td>
<td>2</td>
</tr>
</tbody>
</table>

200 subject-verb-object triples judged by 25 participants
Evaluation : Verb disambiguation in subject-verb-object triples

Evaluation dataset [Grefenstette and Sadrzadeh 11]

<table>
<thead>
<tr>
<th>Subj-Verb-Obj</th>
<th>Landmark verb</th>
<th>Similarity of human judgment</th>
</tr>
</thead>
<tbody>
<tr>
<td>People-run-company</td>
<td>operate</td>
<td>7</td>
</tr>
<tr>
<td>People-run-company</td>
<td>move</td>
<td>2</td>
</tr>
</tbody>
</table>

200 subject-verb-object triples judged by 25 participants

Final co-compositional vector for subject-verb-object

\[ \text{subj} + \text{cocompositioned(verb, obj)} \]
Evaluation: Verb disambiguation in subject-verb-object triples

Evaluation dataset [Grefenstette and Sadrzadeh 11]

<table>
<thead>
<tr>
<th>Subj-Verb-Obj</th>
<th>Landmark verb</th>
<th>Similarity of human judgment</th>
</tr>
</thead>
<tbody>
<tr>
<td>People-run-company</td>
<td>operate</td>
<td>7</td>
</tr>
<tr>
<td>People-run-company</td>
<td>move</td>
<td>2</td>
</tr>
</tbody>
</table>

200 subject-verb-object triples judged by 25 participants

Final co-compositional vector for subject-verb-object

\[ \text{subj} + \text{cocompositioned}(\text{verb, obj}) \]

Models are evaluated by Spearman’s rank correlation between vectors’ computed similarity and human judgment
Implementation details

Extracted 20 prototype words from ukWaC corpus
Implementation details

Extracted 20 prototype words from ukWaC corpus
Implementation details

Extracted 20 prototype words from ukWaC corpus

Both high frequency and high similarity

VerbOf

ObjectOf

start
build
buy

run

company

firm
bank
hotel
Implementation details

Extracted 20 prototype words from ukWaC corpus

- VerbOf: run
- ObjectOf: company

- start ≈ build ≈ buy
- high frequency
- 80% of the top singular values

- firm, bank, hotel
- both high frequency and high similarity
Implementaiton details

Extracted 20 prototype words from ukWaC corpus

Word representation [Blacoe and Lapata 12]
①Distributional vector (2000 dim) ②Neural vector (50 dim)
## Baselines: Models compared to ours

<table>
<thead>
<tr>
<th>Add</th>
<th>$\text{sbj + verb + obj}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Mitchell and Lapata 08]</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Multiply</th>
<th>$\text{sbj \times verb \times obj}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Mitchell and Lapata 08]</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Grefenstette and Sadrzadeh 11</th>
<th>Mathematical model based on abstract categorical framework</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Van de Cruys+13</th>
<th>Multi-way interaction model based on non-negative matrix factorization</th>
</tr>
</thead>
</table>
Correlation with human judgment (Distributional vector)

Achieves high performance ($\rho = 0.41$)

![Bar chart showing correlation $\rho$ for various methods](chart.png)

- Add: 0.31
- Multiply: 0.35
- Grefenstette and Sadrzadeh 11: 0.21
- Van de Cruys+ 13: 0.37
- Our Model: 0.41
Correlation with human judgment (Neural vector)

State of the art performance ($\rho = 0.44$)

<table>
<thead>
<tr>
<th>Model</th>
<th>Correlation $\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>0.31</td>
</tr>
<tr>
<td>Multiply</td>
<td>0.3</td>
</tr>
<tr>
<td>Grefenstette and Sadrzadeh 11</td>
<td>0.21</td>
</tr>
<tr>
<td>Van de Cruys+ 13</td>
<td>0.37</td>
</tr>
<tr>
<td>Our Model</td>
<td>0.44</td>
</tr>
</tbody>
</table>
Correlation with human judgment (Neural vector)

State of the art performance ($\rho = 0.44$)

Co-Compositionality is useful for word sense disambiguation. Prototype projection is effective implementation for Co-Compositionality.
Two contributions in our work

- New model of compositionality in word vector space
- Unsupervised word vector re-training algorithm considering compositionality
Two contributions in our work

- New model of compositionality in word vector space
- Unsupervised word vector re-training algorithm considering compositionality
Re-training word representation with decomposition of phrase vector
Compositional Neural Language Model

Re-training word representation with decomposition of phrase vector

\[ s = u^T z \]

① Compute the score \( s \) of correct phrase
Compositional Neural Language Model

Re-training word representation with decomposition of phrase vector

\[ s_c = u^T z \]

① Compute the score \( s \) of correct phrase

② Compute the score \( s_c \) of corrupted incorrect phrase

Our Model

Compositional Neural Language Model
Re-training word representation with decomposition of phrase vector

Our Model

Compositional Neural Language Model

・Compute the score $s$ of correct phrase

・Compute the score $s_c$ of corrupted incorrect phrase

$J = \max \left(0, 1 - s + s_c\right)$

Correct score $>$ Incorrect score
Re-training word representation with decomposition of phrase vector

1. Compute the score $s$ of correct phrase
2. Compute the score $s_c$ of corrupted incorrect phrase
3. Minimize cost function by SGD, $u \rightarrow u_{\text{new}}$, $z \rightarrow z_{\text{new}}$

$$J = \max \left( 0, 1 - s + s_c \right)$$

Correct score $>\$ Incorrect score
Re-training word representation with decomposition of phrase vector

\[ s = u^T z \]

① Compute the score \( s \) of correct phrase

② Compute the score \( s_c \) of corrupted incorrect phrase

③ Minimize cost function by SGD, \( u \rightarrow u_{\text{new}}, \ z \rightarrow z_{\text{new}} \)

\[ J = \max \left( 0, 1 - s + s_c \right) \]

Correct score > Incorrect score
Re-training word representation with decomposition of phrase vector

1. Compute the score $s$ of correct phrase

2. Compute the score $s_c$ of corrupted incorrect phrase

3. Minimize cost function by SGD, $u \rightarrow u_{\text{new}}, z \rightarrow z_{\text{new}}$

4. New verb vector is $v_{\text{new}} = z_{\text{new}} - o$

$$J = \max \left( 0, 1 - s + s_c \right)$$

Correct score $>$ Incorrect score
Re-training word representation with decomposition of phrase vector

① Compute the score $s$ of correct phrase

② Compute the score $s_c$ of corrupted incorrect phrase

③ Minimize cost function by SGD, $u \rightarrow u_{\text{new}}, z \rightarrow z_{\text{new}}$

④ New verb vector is $v_{\text{new}} = z_{\text{new}} - o$

$J = \max(0, 1 - s + s_c)$

Correct score $>$ Incorrect score

New word representations considering compositionality
Compositional Neural Language Model

Our Model

\[ s = u^T z \]

\[ z = x + y \]
Compositional Neural Language Model

Our Model

s = u^T z

Co-Compositionality with Prototype Projection
Our Model

Compositional Neural Language Model

\[ s = u^T z \]

\[ z = x + y \]

Co-Compositionality with Prototype Projection
Our Model

Co-Compositional Neural Language Model

Compositional Neural Language Model with Prototype Projection

\[ s = u^T z \]

\[ z = x + y \]

Co-Compositionality with Prototype Projection

Masashi Tsubaki
Co-Compositional Neural Language Model

Compositional Neural Language Model with Prototype Projection

① Prototype projection for both verb and object

② Optimize parameters with same method as Compositional NLM

③ Minimize

\[
\min_v \left( \| x_{new} - P_{obj} v \|^2 + \lambda \|v\|^2 \right)
\]
Our Model

Co-Compositional Neural Language Model

Compositional Neural Language Model with Prototype Projection

Prototype projection for both verb and object

Optimize parameters with same method as Compositional NLM

Minimize

\[
\min_v \left( \left\| x_{new} - P_{obj} v \right\|^2 + \lambda \left\| v \right\|^2 \right)
\]
Co-Compositional Neural Language Model

With Prototype Projection

1. Prototype projection for both verb and object.
2. Optimize parameters with same method as Compositional NLM.
3. Minimize

\[
\min_v \left( \left\| x_{\text{new}} - P_{\text{obj}} v \right\|^2 + \lambda \| v \|^2 \right)
\]
Co-Compositional Neural Language Model

Our Model

New word representations considering co-compositionality
Evaluation: Verb disambiguation [Grefenstette and Sadrzadeh 11]

*Original* neural vector [Blacoe and Lapata 12]

VS.

*Re-trained* neural vector with our learning models
Evaluation: Verb disambiguation [Grefenstette and Sadrzadeh 11]

Original neural vector [Blacoe and Lapata 12] vs. Re-trained neural vector with our learning models

Training data
Extracted 5000 Verb-Obj pairs from ukWaC corpus

Hyper-parameters
Learning rate: 0.01, Regularization: 10^4
20 iterations (One iteration is one run through the training data)
Result and Discussion

Correlation with human judgment (Re-trained neural vector)

New state of the art performance ($\rho = 0.47$)

Higher performance with re-trained word representation
New model of compositionality in word vector space

Co-Compositionality with Prototype Projection

Unsupervised word vector re-training algorithm considering compositionality

Compositional & Co-Compositional Neural Language Models Achieve state of the art on verb disambiguation task
Examples

<table>
<thead>
<tr>
<th>verb</th>
<th>object</th>
<th>landmark</th>
<th>similarity(verb, landmark)</th>
<th>similarity(projected verb, landmark)</th>
</tr>
</thead>
<tbody>
<tr>
<td>run</td>
<td>company</td>
<td>operate</td>
<td>0.40</td>
<td>0.70</td>
</tr>
<tr>
<td>meet</td>
<td>criterion</td>
<td>satisfy</td>
<td>0.49</td>
<td>0.71</td>
</tr>
<tr>
<td>spell</td>
<td>name</td>
<td>write</td>
<td>0.04</td>
<td>0.50</td>
</tr>
</tbody>
</table>

Table 1: Examples of verb-object pairs. Original verb and landmark verb similarity, prototype projected verb and landmark verb similarity, as measure by cosine using Collobert and Weston’s word embeddings. *Meet* has a abstract meaning itself, but after prototype projection with matrix constructed by word vectors of $W(\text{VerbOf, criterion})$, *meet* is more close to meaning of *satisfy*. 
Results of the different compositionality models

<table>
<thead>
<tr>
<th>Model</th>
<th>$\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grefenstette and Sadrzadeh (2011)</td>
<td>0.21</td>
</tr>
<tr>
<td>Add (SDS)</td>
<td>0.31</td>
</tr>
<tr>
<td>Add (NLM)</td>
<td>0.31</td>
</tr>
<tr>
<td>Multiply (SDS)</td>
<td>0.35</td>
</tr>
<tr>
<td>Multiply (NLM)</td>
<td>0.30</td>
</tr>
<tr>
<td>Van de Cruys et al. (2013)</td>
<td>0.37</td>
</tr>
<tr>
<td>Erk and Padó (SDS)</td>
<td>0.39</td>
</tr>
<tr>
<td>Erk and Padó (NLM)</td>
<td>0.03</td>
</tr>
<tr>
<td>Co-Comp with $f=$Add (SDS)</td>
<td>0.41</td>
</tr>
<tr>
<td>Co-Comp with $f=$Add (NLM)</td>
<td><strong>0.44</strong></td>
</tr>
<tr>
<td>Co-Comp with $f=$Multiply (SDS)</td>
<td>0.37</td>
</tr>
<tr>
<td>Co-Comp with $f=$Multiply (NLM)</td>
<td>0.35</td>
</tr>
<tr>
<td>Upper bound</td>
<td>0.62</td>
</tr>
</tbody>
</table>

Table 3: Results of the different compositionality models on the similarity task. The number of prototype words $m = 20$ in all our models. Our model ($f=$Addition and NLM) achieves the new state-of-the-art performance for this task ($\rho = 0.44$).
The number of prototype words

Figure 5: The relation between the number of prototype words and correlation of SDS or NLM. In general, NLM has higher correlation than SDS and is more robust across the $m$. 
Table 5: Variants of the full co-compositional model, based on how subject, verb, and object vector representations are included. prpj indicates that prototype projection is used. + indicates that the vector is added without projection first. Blank indicates that the vector is not used in the final compositional score.

<table>
<thead>
<tr>
<th>Subj</th>
<th>Verb</th>
<th>Obj</th>
<th>NLM $\rho$</th>
<th>SDS $\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td>prpj</td>
<td>prpj</td>
<td>prpj</td>
<td>0.39</td>
<td>0.37</td>
</tr>
<tr>
<td>+</td>
<td>prpj</td>
<td>prpj</td>
<td>0.44</td>
<td>0.41</td>
</tr>
<tr>
<td>prpj</td>
<td>prpj</td>
<td>prpj</td>
<td>0.45</td>
<td>0.41</td>
</tr>
<tr>
<td>+</td>
<td>prpj</td>
<td>+</td>
<td>0.43</td>
<td>0.38</td>
</tr>
<tr>
<td>prpj</td>
<td>+</td>
<td>+</td>
<td>0.43</td>
<td>0.38</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0.31</td>
<td>0.31</td>
</tr>
</tbody>
</table>
## Composition operator and parameter

<table>
<thead>
<tr>
<th>Composition Operator</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add: $w_1 u + w_2 v$</td>
<td>$w_1, w_2 \in \mathbb{R}$</td>
</tr>
<tr>
<td>Multiply: $u^{w_1} \odot v^{w_2}$</td>
<td>$w_1, w_2 \in \mathbb{R}$</td>
</tr>
<tr>
<td>FullAdd: $W_1 u + W_2 v$</td>
<td>$W_1, W_2 \in \mathbb{R}^{n \times n}$</td>
</tr>
<tr>
<td>LexFunc: $A_u v$</td>
<td>$A_u \in \mathbb{R}^{n \times n}$</td>
</tr>
<tr>
<td>FullLex: $\sigma([W_1 A_u v, W_2 A_v u])$</td>
<td>$A_u, A_v \in \mathbb{R}^{n \times n}$</td>
</tr>
<tr>
<td>Ours (Add): $P_{(R,v)} u + P_{(R,u)} v$</td>
<td>$W_1, W_2 \in \mathbb{R}^{n \times n}$</td>
</tr>
<tr>
<td>Ours (Mult): $P_{(R,v)} u \odot P_{(R,u)} v$</td>
<td>SVD’s ($m, k$)</td>
</tr>
</tbody>
</table>

Table 6: Comparison of composition operators that combine two word vector representations, $u, v \in \mathbb{R}^n$ and their learning parameters. Our model only needs two hyper-parameters: the number of prototype words $m$ and dimensional reduction $k$ in SVD.