Prepositional Phrase Attachment over Word Embedding Products

Pranava Madhyastha[1], Xavier Carreras[2], Ariadna Quattoni[2]

The University Of Sheffield.

[1] University of Sheffield

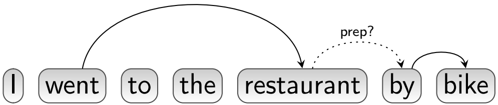
NAVER LABS

[2] Naver Labs Europe

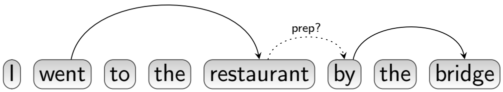
Prepositional Phrases

I went to the restaurant by bike v/s I went to the restaurant by the bridge

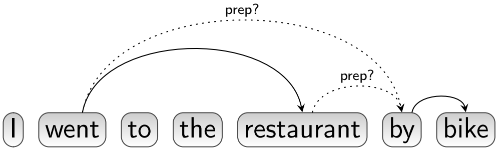
Prepositional Phrases



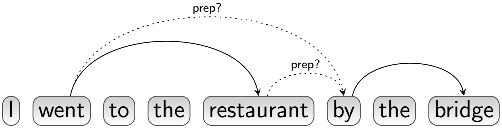
v/s



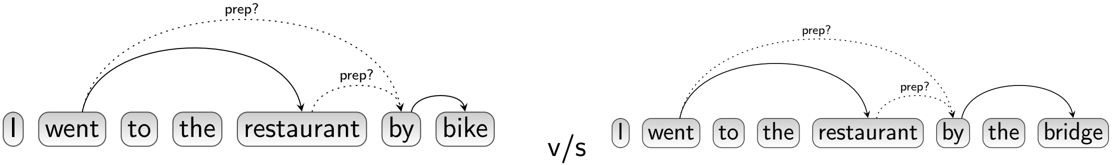
Prepositional Phrases



v/s



Prepositional Phrases



- s/bridge/Arno/ $\Rightarrow$?

Local PP Attachment Decisions

- Ratnaparkhi; 1998 approached PP attachment as a local classification decision
- Lose potential advantage for inference over the whole structure

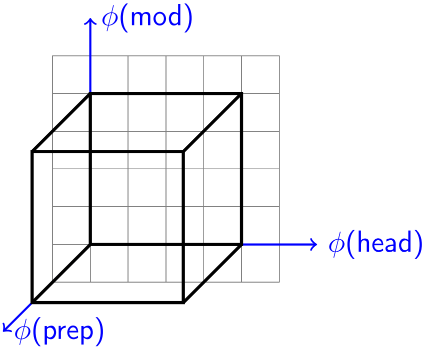
Local PP Attachment Decisions

- Ratnaparkhi; 1998 approached PP attachment as a local classification decision
- Lose potential advantage for inference over the whole structure
- Two settings:
  - Binary attachment problem[Ratnaparkhi; 1998]
  - Multi-class attachment problem[Belinkov+; 2015]

Word Embeddings

- Previous approaches: features in a binary decision setting
- Word Embeddings are rich source of information
- How far can we go with only word embeddings?

Resolving Ternary Relations using Word Embeddings

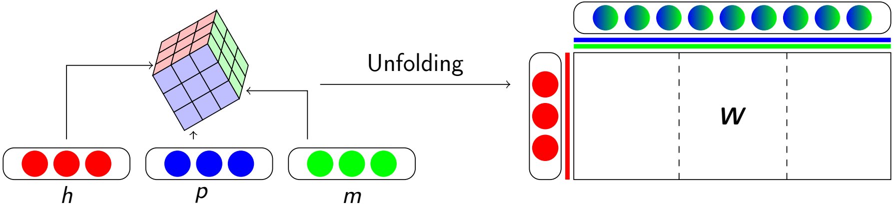


- The problem reduces to finding compatibility between: ϕ(head), ϕ(prep) and ϕ(mod)
- One simple approach is to do:

$$f(h, p, m) = \vec{w} \cdot [v_h \otimes v_p \otimes v_m]$$

- Exploit exploit each dimension alone - with a simple trick: append 1

Our Proposal: Unfolding the Tensors



- This results in:

$$f(h, p, m) = \boldsymbol{v}_h^\top \ \boldsymbol{W} \ [\boldsymbol{v}_p \otimes \boldsymbol{v}_m]$$

- We can further explore compositionality as:

$$f(h, p, m) = \alpha(h)^\top \boldsymbol{W} \beta(p, m)$$

Training the Model

- We use logistic loss
- Std. conditional Max. Likelihood optimization

$$\operatorname*{arg\,min}_{\boldsymbol{W}} \mathrm{logistic}(Tset, \boldsymbol{W}) + \lambda \|\boldsymbol{W}\|_{\star} \tag{1}$$

- As a structural constraint, we use rank-constrained regularization

Empirical Overview

- Exploring Word Embeddings
- Exploring Compositionality
- Binary Attachment Datasets
- Multiple Attachment Datasets
- Semi-supervised multiple attachment experiments with PTB and WTB

Exploring Word Embeddings

- Word2Vec based skip-gram embeddings ([Mikolov+; 2013])
- Skip-dep ([Bansal+; 2014]): uses word2vec with dependency contexts
- Exploration over different dimensions and data.

Exploring Word Embeddings

Word Embedding		Accuracy wrt. dimension (n)		
Type	Source Data	$n = 50$	$n = 100$	$n = 300$
Skip-gram	BLLIP	83.23	83.77	83.84
Skip-gram	Wikipedia	83.74	84.25	84.22
Skip-gram	NYT	84.76	85.06	85.15
Skip-dep	BLLIP	85.52	**86.33**	85.97
Skip-dep	Wikipedia	84.23	84.39	84.32
Skip-dep	NYT	85.27	85.48	–
Skip-gram & Skip-dep	BLLIP	–	83.44	–

- Attachment accuracy over RRR devset

Exploring Compositionality

$$f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W} \boldsymbol{\beta}(p, m)$$

Exploring the possibilities with $\boldsymbol{\beta}(p, m)$:

Exploring Compositionality

$$f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W} \boldsymbol{\beta}(p, m)$$

Exploring the possibilities with $\boldsymbol{\beta}(p, m)$:

$$\Rightarrow \text{Sum} : f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W}(w_p + w_m)$$

Exploring Compositionality

$$f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W} \boldsymbol{\beta}(p, m)$$

Exploring the possibilities with $\boldsymbol{\beta}(p, m)$:

$$\Rightarrow \text{Sum} : f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W}(w_p + w_m)$$

$$\Rightarrow \text{Concatenation} : f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W}(w_p; w_m)$$

Exploring Compositionality

$$f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W} \boldsymbol{\beta}(p, m)$$

Exploring the possibilities with $\boldsymbol{\beta}(p, m)$:

$$\Rightarrow \text{Sum} : f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W}(w_p + w_m)$$

$$\Rightarrow \text{Concatenation} : f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W}(w_p; w_m)$$

$$\Rightarrow \text{Products} : f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W}(w_p \otimes w_m)$$

Exploring Compositionality

$$f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W} \boldsymbol{\beta}(p, m)$$

Exploring the possibilities with $\boldsymbol{\beta}(p, m)$:

$$\begin{aligned}
\Rightarrow \text{Sum} &: f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W}(w_p + w_m) \\
\Rightarrow \text{Concatenation} &: f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W}(w_p; w_m) \\
\Rightarrow \text{Products} &: f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W}(w_p \otimes w_m) \\
\Rightarrow \text{Prep. Identities} &: f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W}_p w_m
\end{aligned}$$

Exploring Compositionality

Composition of p and m		Tensor Size	Acc.
Sum	$[\boldsymbol{v}_p + \boldsymbol{v}_m]$	$n \times n$	84.42
Concatenation	$[\boldsymbol{v}_p; \boldsymbol{v}_m]$	$n \times 2n$	84.94
p Identities	$[\boldsymbol{i}_p \otimes \boldsymbol{v}_m]$	$n \times \lvert\mathcal{P}\rvert * n$	84.36
Product	$[\boldsymbol{v}_p \otimes \boldsymbol{v}_m]$	$n \times n * n$	**85.52**

Empirical Results over Binary Attachment Datasets

Method	Word Embedding	Test Accuracy RRR	Test Accuracy WIKI	Test Accuracy NYT
Tensor product	Skip-gram, Wikipedia, $n = 100$	84.96	83.48	82.13
"	Skip-gram, NYT, $n = 100$	85.11	83.52	82.65
"	Skip-dep, BLLIP, $n = 100$	*86.13*	83.60	82.30
"	Skip-dep, Wikipedia, $n = 100$	85.01	83.53	82.10
"	Skip-dep, NYT, $n = 100$	85.49	**83.64**	*83.47*
[Stetina+; 1997(*)]		**88.1**	-	-
[Collins+; 1999]		84.1	72.7	80.9
[Belinkov+; 2014(*)]		85.6	-	-
[Nakashole+; 2015(*)]		84.3	79.3	**84.3**

Scores refer to Accuracy measures

Multiple Heads and Positional Information

$$f(h, p, m) = \boldsymbol{\alpha}(h)^\top \boldsymbol{W} \beta(p, m)$$

In this case, we can modify $\boldsymbol{\alpha}(h)$ to include the positional info:

- $\boldsymbol{\alpha}(h) = \boldsymbol{\delta}_h \otimes \boldsymbol{v}_h$, where $\boldsymbol{\delta}_h \in \mathbb{R}^{|H|}$ is the position vector

This can also be written as:

$$f(h, p, m) = \boldsymbol{v}_h^\top \boldsymbol{W}_{\delta_h} [\boldsymbol{v}_p \otimes \boldsymbol{v}_m] \qquad .$$

Experiments with Multiple Attachment Setting

	Test Accuracy	
	Arabic	English
Tensor product ($n=50$, ℓ_2)	-	87.8
Tensor product ($n=50$, ℓ_*)	-	88.3
Tensor product ($n=100$, ℓ_*)	*81.1*	*88.4*
Belinkov+; 2014 (basic)	77.1	85.4
Belinkov+; 2014 (syn)	79.1	87.1
Belinkov+; 2014 (feat)	80.4	87.7
Belinkov+; 2014 (full)	**82.6**	88.7
Yu+; 2016 (full)	-	**90.3**

Semi-supervised Experiments

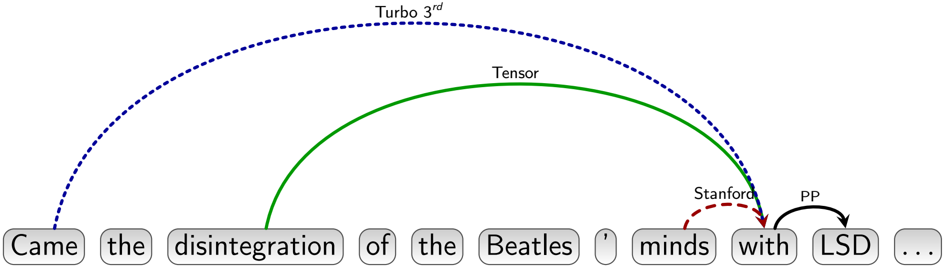
- Embeddings trained on *source* and *target* raw-data
- Model trained on the source data
- Compare against Stanford neural network based parser and Turbo parser

Semi-supervised Experiments with PTB vs WTB

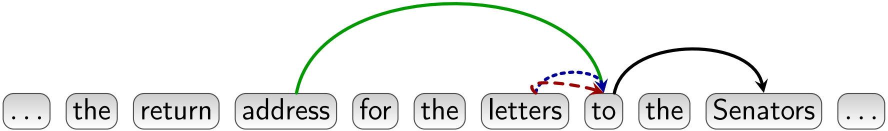
	PTB	Web Treebank Test					
	Test (2523)	A (868)	E (936)	N (839)	R (902)	W (788)	Avg (4333)
Tensor BLLIP	**89.0**	82.7	82.6	87.4	82.5	86.3	84.2
Tensor BLLIP+WTB	88.9	83.3	**85.2**	**90.1**	**85.9**	86.6	**86.1**
Chen+; 2014	87.3	79.3	79.7	85.7	82.2	83.8	82.0
Martins+; 2010 (2^{nd} order)	88.8	83.6	83.7	87.6	84.2	**87.8**	85.3
Martins+; 2010 (3^{rd} order)	88.9	**84.2**	84.5	87.6	84.4	87.6	85.6

Scores refer to Accuracy measures

What Seems to Work



Where it has a Problem



Conclusions

- We described a PP attachment model using simple tensor products
- Product of embeddings seem to be better at capturing compositions
- In out-of-domain datasets, we see the models shows a big promise
- Proposed models can serve as a building block to dependency parsing methods

Thank You

Learning with Structural Constraints

- Controlling the learned parameter metrix with regularization

Learning with Structural Constraints

- Controlling the learned parameter metrix with regularization
- We can obtain dense, dispersed parameter matrix with ℓ_2

Learning with Structural Constraints

- Controlling the learned parameter metrix with regularization

$$\underbrace{\left(\underbrace{\begin{bmatrix} \mathbf{u}_{11} & \cdots & \mathbf{u}_{1k} \\ \mathbf{u}_{21} & \cdots & \mathbf{w}_{2k} \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ \mathbf{u}_{n1} & \cdots & \mathbf{u}_{nk} \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_k \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} \mathbf{v}_{11} & \cdots & \cdots & \mathbf{v}_{1n} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{v}_{k1} & \cdots & \cdots & \mathbf{v}_{kn} \end{bmatrix}}_{V^\top} \right)}_{\mathrm{SVD}(W) = U\Sigma V^\top}$$

Learning with Structural Constraints

- Controlling the learned parameter metrix with regularization
- We can obtain dense, dispersed parameter matrix with ℓ_2
- Obtain a *low-rank* matrix with ℓ_* regularization

Learning with Structural Constraints

- Controlling the learned parameter metrix with regularization
- Bonus everything can be convex optimized with FOBOS!!!

A Quicklook at Optimization

Input: Gradient function g

$\boldsymbol{W}_{t+1} = \mathrm{argmin}_W \, ||\boldsymbol{W}_{t+0.5} - \boldsymbol{W}||_2^2 + \eta_t \lambda r(\boldsymbol{W})$;

Output: $\boldsymbol{W}_{t+1}$

$\boldsymbol{W}_1 = \mathbf{0}$

while $t < T$ **do**

    $\eta_t = \frac{c}{\sqrt{t}}$

    $\boldsymbol{W}_{t+0.5} = \boldsymbol{W}_t - \eta_t g(\boldsymbol{W}_t)$

    if ℓ_2 *regularizer* **then**

        $\boldsymbol{W}_{t+1} = \frac{1}{1+\eta_t\lambda} \boldsymbol{W}_{t+0.5}$

    else if ℓ_* *regularizer* **then**

        $U\Sigma V^\top = \mathrm{SVD}(\boldsymbol{W}_{t+0.5})$

        $\bar{\Sigma}_{i,i} = \max(\Sigma_{i,i} - \eta_t\lambda, 0)$

        $\boldsymbol{W}_{t+1} = U\bar{\Sigma}V^\top$

end

Data

- BLLIP with $\sim$1.8 million sentences and $\sim$43 million tokens of Wall Street Journal text (and excludes PTB evaluation sets);
- English Wikipedia $\sim$13.1 million sentences and $\sim$129 million tokens;
- The New York Times portion of the GigaWord corpus, with $\sim$52 million sentences and $\sim 1,253$ million tokens.