Compositional Architectures and Word Vector Representations for PP Attachment

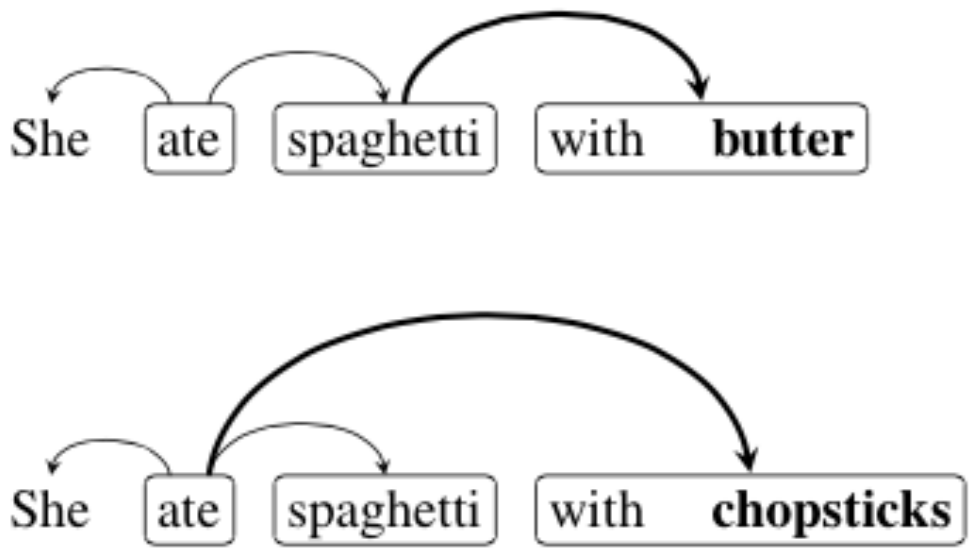
Yonatan Belinkov, Tao Lei, Regina Barzilay, Amir Globerson

NAACL 2015
(Published at TACL)

MIT CSAIL

THE HEBREW UNIVERSITY OF JERUSALEM

Background


She ate spaghetti with **butter**
She ate spaghetti with **chopsticks**

Applications

- PP attachments: major source of errors in syntactic parsing (Kummerfeld et al. 2012)
- Syntactic parsing: a core NLP module
  - Named entity recognition
  - Machine translation
  - Co-reference resolution
- Relation extraction

Historical Development

- Classic NLP task since the 1990s

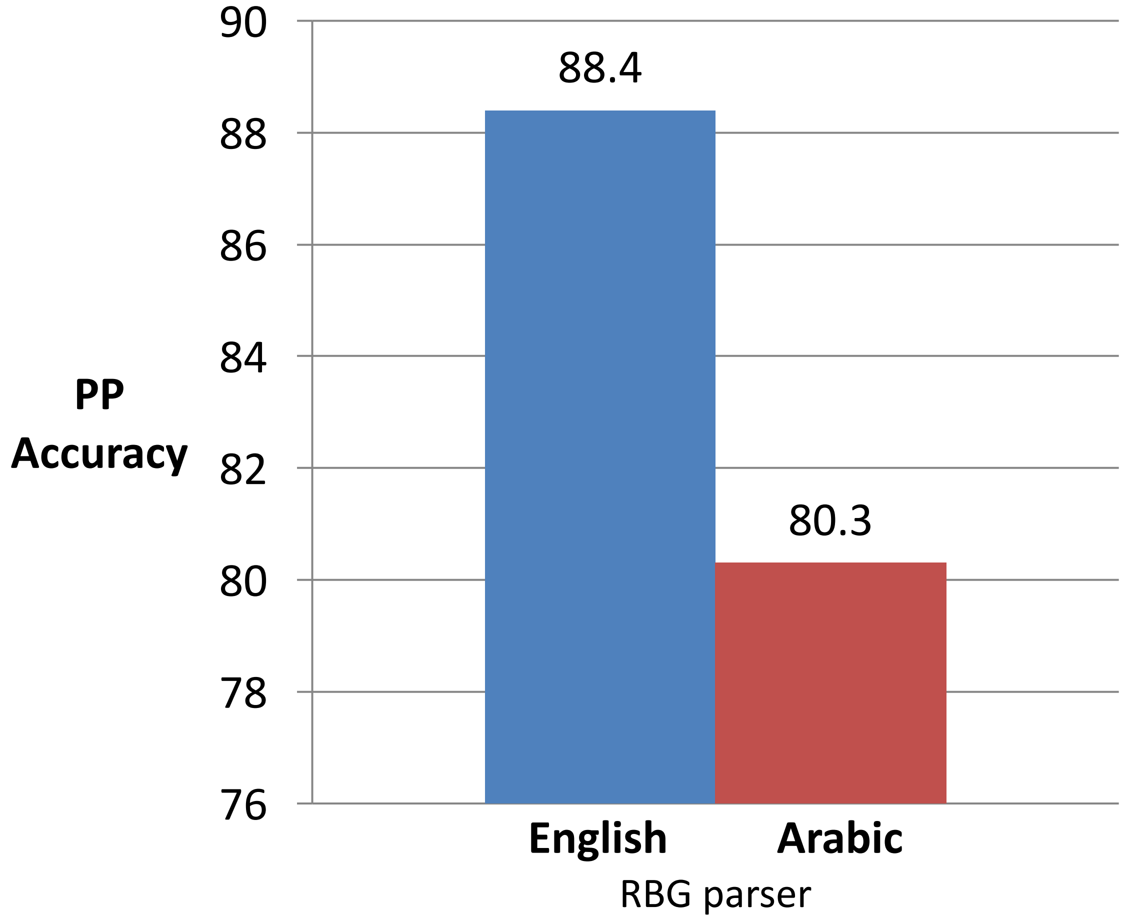
Historical Development

- Classic NLP task since the 1990s
- Improvement over time (Kummerfeld et al. 2012)
  - 32% error reduction in 15 years (since Collins 1997)

Historical Development

- Classic NLP task since the 1990s
- Improvement over time (Kummerfeld et al. 2012)
  - 32% error reduction in 15 years (since Collins 1997)
- But: still a major challenge in parsing
  - Largest source of errors across a range of parsers

Problem goes beyond English



Previous Work

- Problem formulation:
  - Constrained binary classifiers (Ratnaparkhi 1994; …)
  - Full-scale parsers

➡ Consider an un-constrained PP scenario

- Information sources:
  - Hand-crafted knowledge (Gamallo et al. 2003; …)
  - Statistics from raw text (Volk 2002), word vector representations (Šuster 2012; Socher et al. 2013)

➡ Combine word vectors with other representations

Setup

- Training: sentences, prepositions (*with*), children (*butter),* heads (*spaghetti*)

Setup

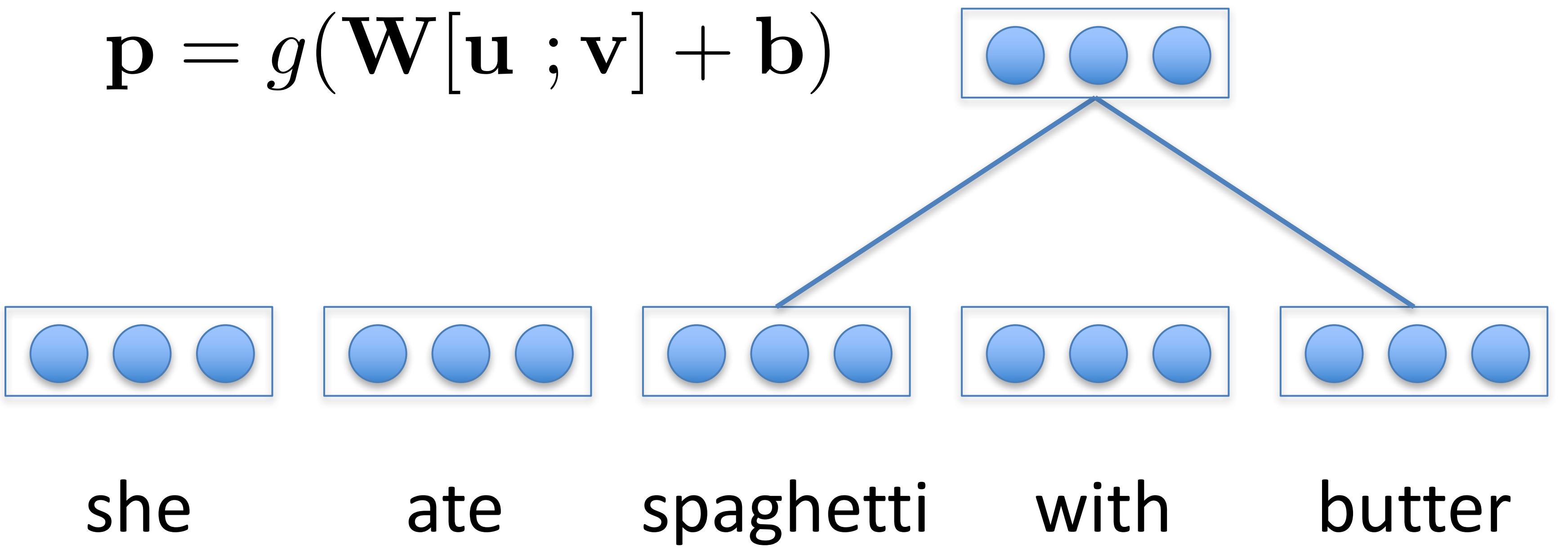
- Training: sentences, prepositions (*with*), children (*butter),* heads (*spaghetti*)

- Testing:
  - Given: sentences, prepositions, children
  - Predict: heads

Basic Model

$$\arg\max_{\mathbf{p}}(\mathbf{w} \cdot \mathbf{p})$$

$$\mathbf{p} = g(\mathbf{W}[\mathbf{u}\,;\mathbf{v}] + \mathbf{b})$$



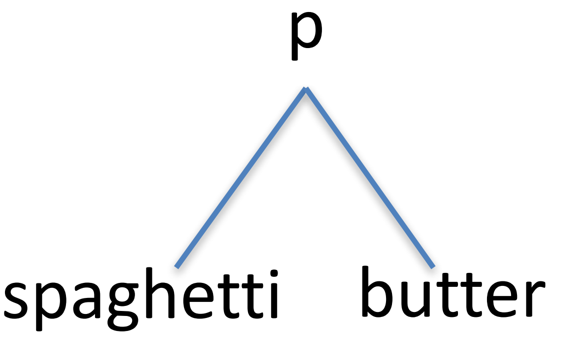
Basic Model

- Represent words as vectors: $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$

Basic Model

- Represent words as vectors: $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$
- Compose with a neural network:

$$\mathbf{p} = g(\mathbf{W}[\mathbf{u}\ ; \mathbf{v}] + \mathbf{b})$$

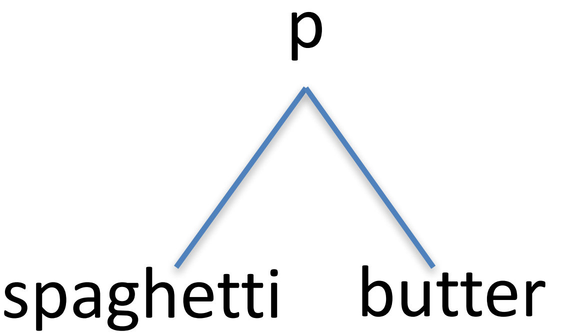


Basic Model

- Represent words as vectors: $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$
- Compose with a neural network:

$$\mathbf{p} = g(\mathbf{W}[\mathbf{u}\,;\mathbf{v}] + \mathbf{b})$$

- Score parent: $\arg\max_{\mathbf{p}}(\mathbf{w} \cdot \mathbf{p})$

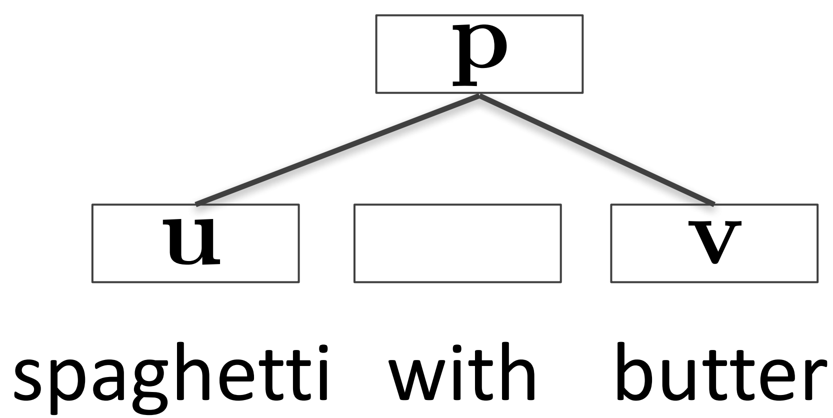


Basic Model

- Represent word as vectors: $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$
- Compose vectors with neural network:

$$\mathbf{p} = g(\mathbf{W}[\mathbf{u} \,; \mathbf{v}] + \mathbf{b})$$

- Score parent: $\arg\max_{\mathbf{p}}(\mathbf{w} \cdot \mathbf{p})$



Example

She ate spaghetti with butter

Example

- Candidate *spaghetti*

She ate *spaghetti* with butter

Example

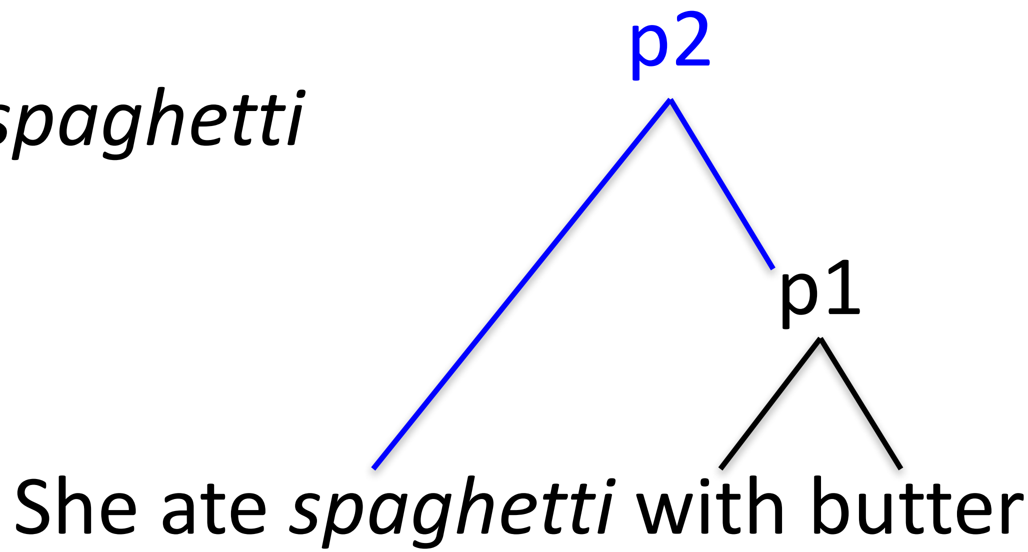
- Candidate *spaghetti*

p1

She ate *spaghetti* with butter

Example

- Candidate *spaghetti*

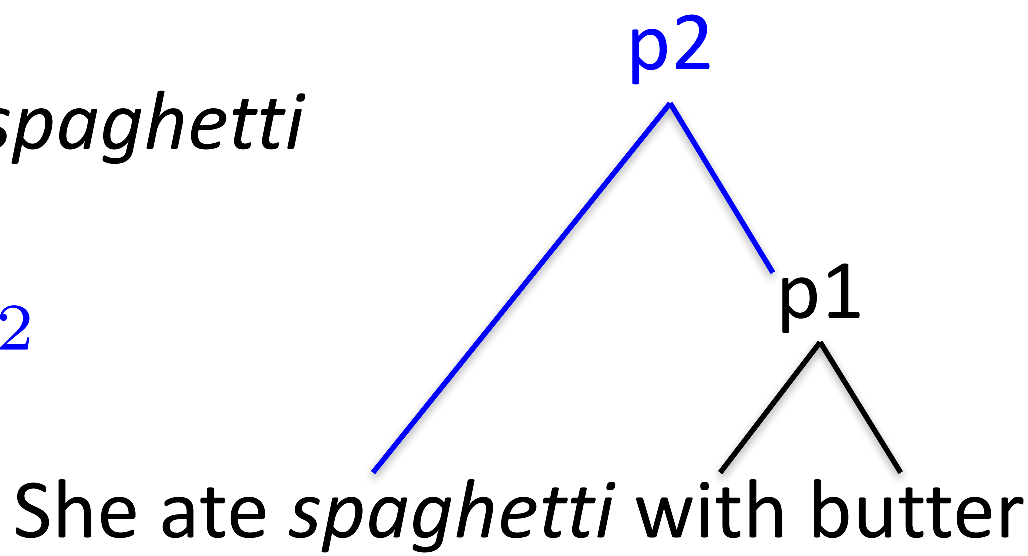


Example

- Candidate *spaghetti*

  Score:

  $$s = \mathbf{w} \cdot \mathbf{p}_2$$

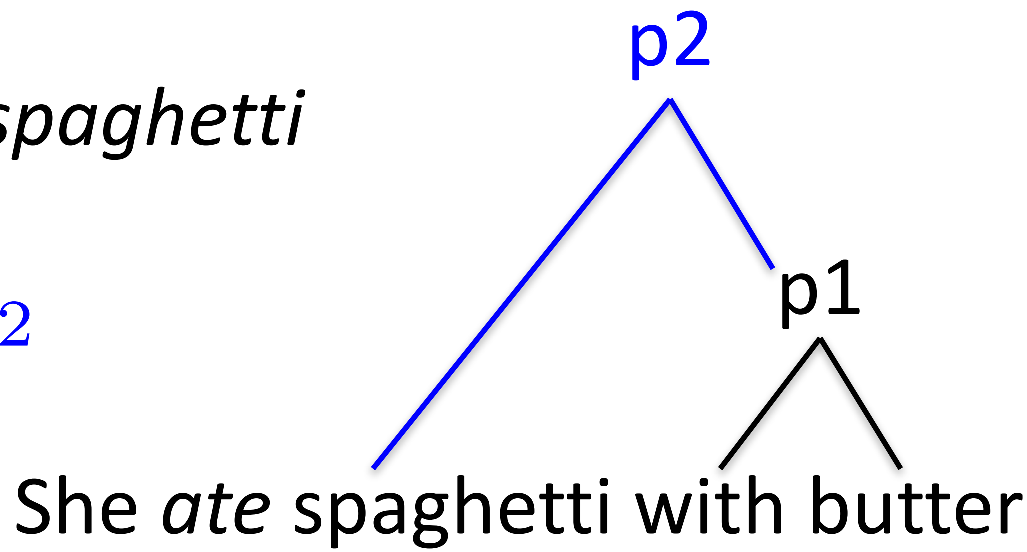


Example

- Candidate *spaghetti*

  Score:

  $$s = \mathbf{w} \cdot \mathbf{p}_2$$



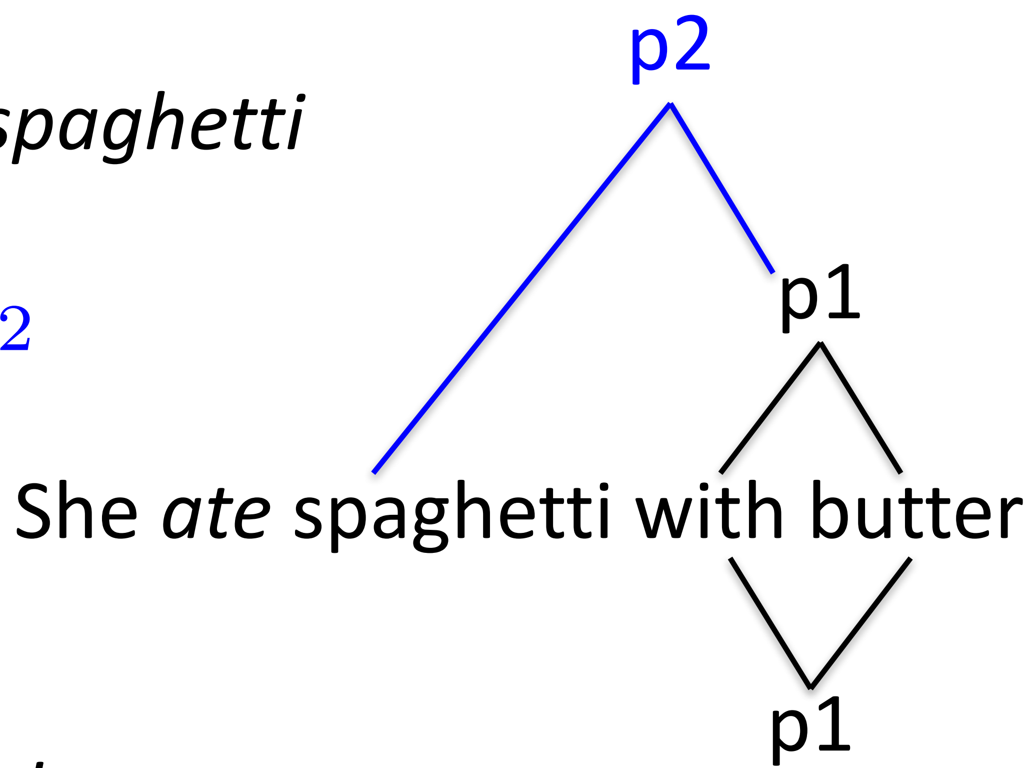
- Candidate *ate*

Example

- Candidate *spaghetti*

  Score:

  $$s = \mathbf{w} \cdot \mathbf{p}_2$$



- Candidate *ate*

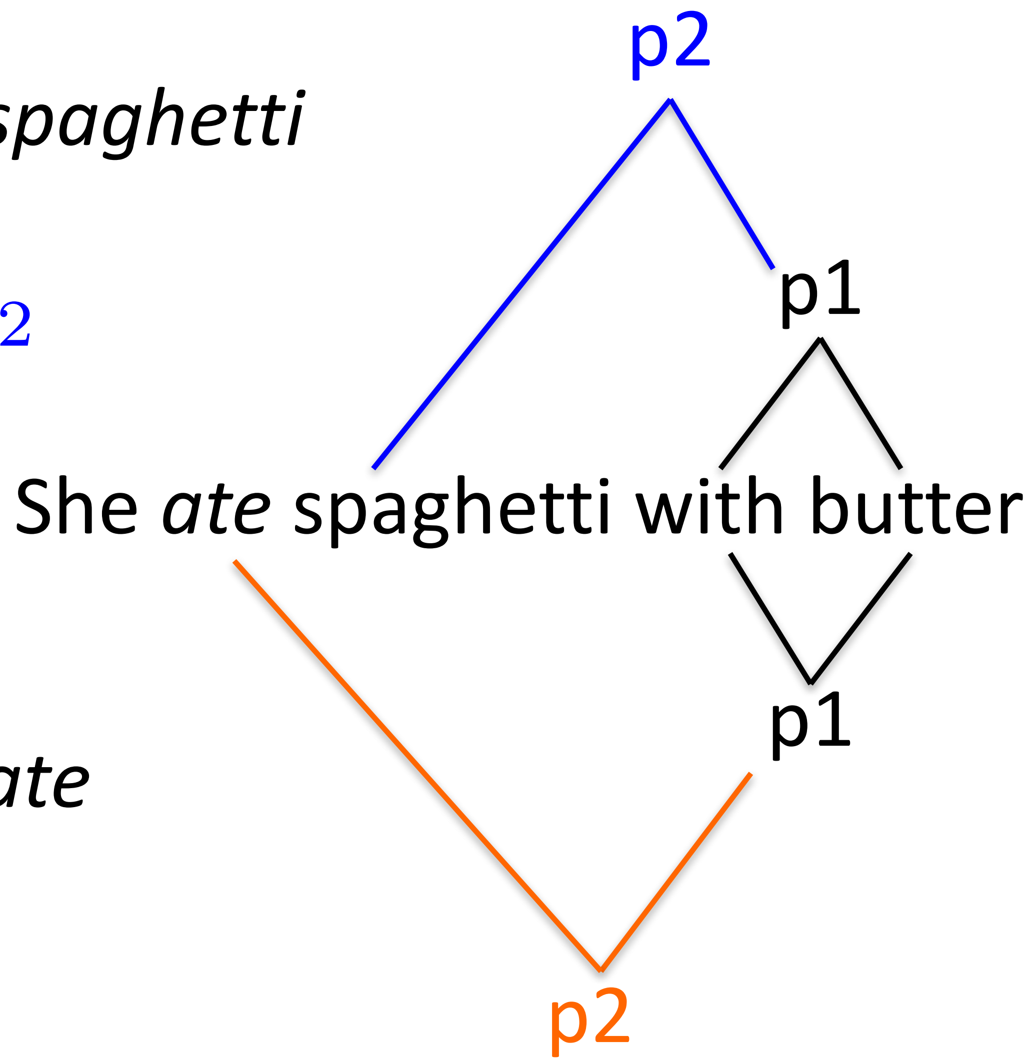
Example

- Candidate *spaghetti*

  Score:

  $$s = \mathbf{w} \cdot \mathbf{p}_2$$

- Candidate *ate*

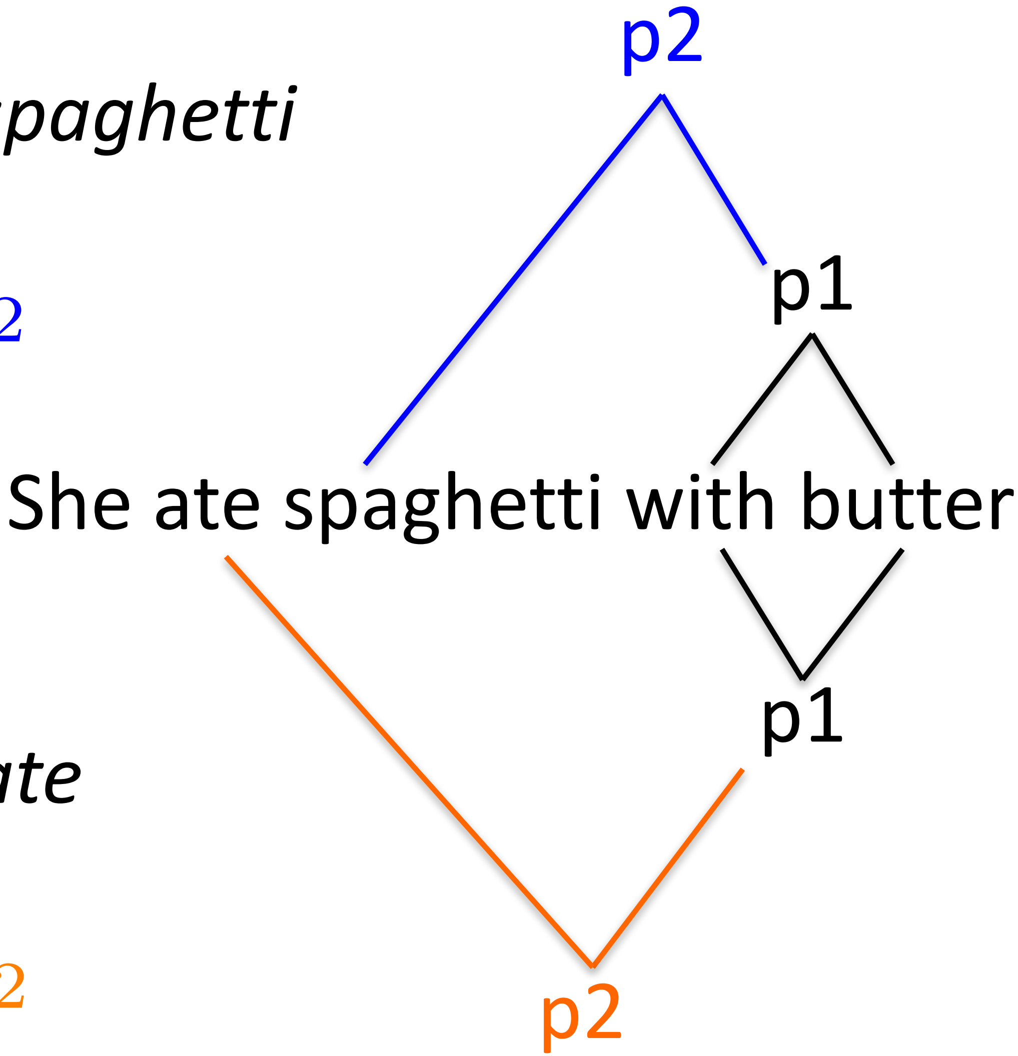


Example

- Candidate *spaghetti*

  Score:

  $s = \mathbf{w} \cdot \mathbf{p}_2$

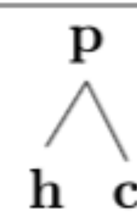
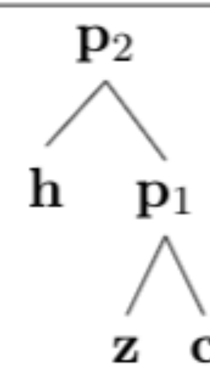
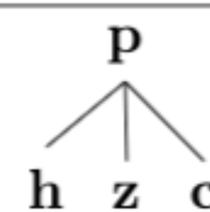


- Candidate *ate*

  Score:

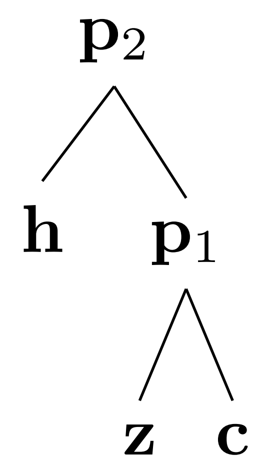
  $s = \mathbf{w} \cdot \mathbf{p}_2$

Composition Architectures

Model	Equations	Structure
Head-Child (HC)	$\mathbf{p} = g(\mathbf{W}[\mathbf{h}; \mathbf{c}] + \mathbf{b})$	
Head-Prep-Child (HPC)	$\mathbf{p}_1 = g(\mathbf{W}[\mathbf{z}; \mathbf{c}] + \mathbf{b})$ $\mathbf{p}_2 = g(\mathbf{W}[\mathbf{h}; \mathbf{p}_1] + \mathbf{b})$	
Head-Prep-Child-Ternary (HPCT)	$\mathbf{p} = g(\mathbf{W}^{Tern}[\mathbf{h}; \mathbf{z}; \mathbf{c}] + \mathbf{b})$	

Improving the Architecture

- Granularity

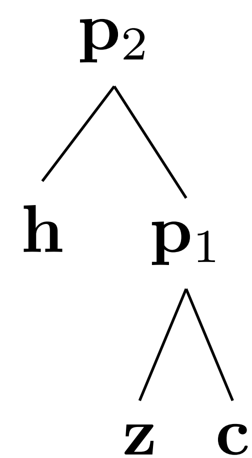


Improving the Architecture

- Granularity
  - Different matrices for different compositions

$$\mathbf{p}_1 = g(\mathbf{W}^{bottom}[\mathbf{z};\mathbf{c}] + \mathbf{b}^{bottom})$$

$$\mathbf{p}_2 = g(\mathbf{W}^{top}[\mathbf{h};\mathbf{p}_1] + \mathbf{b}^{top})$$



Improving the Architecture

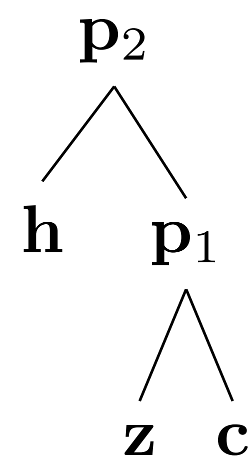
- Granularity
  - Different matrices for different compositions

$$\mathbf{p}_1 = g(\mathbf{W}^{bottom}[\mathbf{z};\mathbf{c}] + \mathbf{b}^{bottom})$$

$$\mathbf{p}_2 = g(\mathbf{W}^{top}[\mathbf{h};\mathbf{p}_1] + \mathbf{b}^{top})$$

  - Distance-dependent matrices

$$\mathbf{p}_2 = g(\mathbf{W}^{d}[\mathbf{h};\mathbf{p}_1] + \mathbf{b}^{d})$$



Improving the Architecture

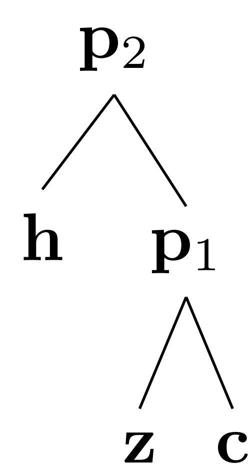
- Granularity
  - Different matrices for different compositions

$$\mathbf{p}_1 = g(\mathbf{W}^{bottom}[\mathbf{z};\mathbf{c}] + \mathbf{b}^{bottom})$$

$$\mathbf{p}_2 = g(\mathbf{W}^{top}[\mathbf{h};\mathbf{p}_1] + \mathbf{b}^{top})$$

  - Distance-dependent matrices

$$\mathbf{p}_2 = g(\mathbf{W}^{d}[\mathbf{h};\mathbf{p}_1] + \mathbf{b}^{d})$$



- Context
  - Concatenate vectors for neighbors

Training

- Given a corpus of pairs of sentences and attachments, *{$x^{(i)}$,$y^{(i)}$}*, minimize:

$$J(\theta) = \sum_{i=1}^{T} \sum_{z \in PR(x^{(i)})} \max_{h} \Big[s(x^{(i)}, z, h; \theta) \\ - s(x^{(i)}, z, y^{(i)}(z); \theta) + \Delta(h, y^{(i)}(z)) \Big]$$

Training

- Given a corpus of pairs of sentences and attachments, $\{x^{(i)}, y^{(i)}\}$, minimize:

$$J(\theta) = \sum_{i=1}^{T} \sum_{z \in PR(x^{(i)})} \max_{h} \Big[s(x^{(i)}, z, h; \theta) \\ - s(x^{(i)}, z, y^{(i)}(z); \theta) + \Delta(h, y^{(i)}(z)) \Big]$$

- Optimization: AdaGrad (Duchi et al. 2011)
- Regularization: Dropout (Hinton et al. 2012)

AdaGrad

- Adaptive gradient descent (Duchi et al. 2011)
- Update for parameter θ_i at time *t+1*:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{\sum_{t'=1}^{t} g_{t',i}^2}} g_{t,i}$$

AdaGrad

- Guarantees asymptotically sub-linear regret:

$$R(T) = \sum_{t=1}^{T} [f_t(\theta_t) - f_t(\theta^*)]$$

- Where:
  - θ^* is the optimal parameter set
  - f is the objective function

Dropout

- Regularization for neural networks (Hinton et al. 2012)

- Randomly dropout units from each layer:

$$\tilde{\mathbf{p}} = \mathbf{p} \odot \mathbf{r}$$

- **r** = random Bernoulli variable w/ parameter *ρ*
- In testing, scale matrices:

$$\tilde{\mathbf{W}} = \rho \mathbf{W}$$

Word Vector Representations

- Initial word vectors:
  - Trained from raw texts
  - Skip-gram model (Mikolov et al. 2013)
  - Similar words have similar vectors

Skip-gram Model

- For a corpus with T words $w_1,\ldots,w_T$, maximize:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c,j\neq 0}\log p(w_{t+j}|w_t)$$

- where:

$$p(w_{t+j}|w_t) = \frac{\exp({v'}_{w_{t+j}}^{T} v_{w_t})}{\sum_{w=1}^{W}\exp({v'}_{w}^{T} v_{w_t})}$$

- v_w, v'_w - input/output of neural network
- With some additional approximations

Alternative Representations

- Relearn vectors during training
  - Backpropagate errors from supervised data

Alternative Representations

- Relearn vectors during training
  - Backpropagate errors from supervised data
- Enrich vectors with external resources
  - WordNet, VerbNet, POS

Alternative Representations

- Relearn vectors during training
  - Backpropagate errors from supervised data
- Enrich vectors with external resources
  - WordNet, VerbNet, POS
- Exploit syntactic context
  - Dependency-based word vectors

    (Bansal et al. 2014, Levy and Goldberg 2014)

Data

- English (WSJ) and Arabic (SPMRL) datasets

	Arabic		English	
	Train	Test	Train	Test
# Attachments	42,387	3,197	35,359	1,951
Avg # Candidates	4.5	4.3	3.7	3.6
Vocab sizes				
Prepositions	13	10	72	46
Heads	8,225	2,936	10,395	2,133
Children	4,222	1,424	5,504	983

Data

- English (WSJ) and Arabic (SPMRL) datasets

	Arabic		English	
	Train	Test	Train	Test
# Attachments	42,387	3,197	35,359	1,951
Avg # Candidates	4.5	4.3	3.7	3.6
Vocab sizes				
Prepositions	13	10	72	46
Heads	8,225	2,936	10,395	2,133
Children	4,222	1,424	5,504	983

Data

- English (WSJ) and Arabic (SPMRL) datasets

	Arabic		**English**	
	Train	Test	Train	Test
# Attachments	42,387	3,197	35,359	1,951
Avg # Candidates	4.5	4.3	3.7	3.6
Vocab sizes				
Prepositions	13	10	72	46
Heads	8,225	2,936	10,395	2,133
Children	4,222	1,424	5,504	983

Data

- English (WSJ) and Arabic (SPMRL) datasets

	Arabic		English	
	Train	Test	Train	Test
# Attachments	42,387	3,197	35,359	1,951
Avg # Candidates	4.5	4.3	3.7	3.6
Vocab sizes				
Prepositions	13	10	72	46
Heads	8,225	2,936	10,395	2,133
Children	4,222	1,424	5,504	983

Baselines

- Closest candidate
- SVM

Baselines

- Closest candidate
- SVM
- Parsers
  - Malt (Niver et al. 2006)
  - MST (McDonald et al. 2005)
  - Turbo (Martins et al. 2010, 2013)
  - RBG (Lei et al. 2014)
  - RNN (Socher et al. 2013)
  - Charniak-RS (McClosky et al. 2006)

Results (PP attachment)

Simple baselines

System	Arabic	English
Closest candidate	62.7	81.7
SVM	**77.5**	**85.9**

Results (PP attachment)

	System	Arabic	English
Simple baselines	Closest candidate	62.7	81.7
	SVM	77.5	85.9
Parsers	Malt	75.4	79.7
	MST	76.7	86.8
	Turbo	76.7	88.3
	RBG	80.3	88.4
	RNN	68.9	85.1
	Charniak-RS	**80.8**	**88.6**

Results (PP attachment)

	System	Arabic	English
Top parsers	RBG	80.3	88.4
	Charniak-RS	80.8	88.6
Our model	HPCD	**82.6**	**88.7**

Results (PP attachment)

	System	Arabic	English
Top parsers	RBG	80.3	88.4
	Charniak-RS	80.8	88.6
Our model	HPCD	82.6	88.7
	RBG + HPCD	**82.7**	**90.1**

Results (parsing)

- Add PP predictions to a state-of-the-art parser
- Binary feature for each predicted attachment

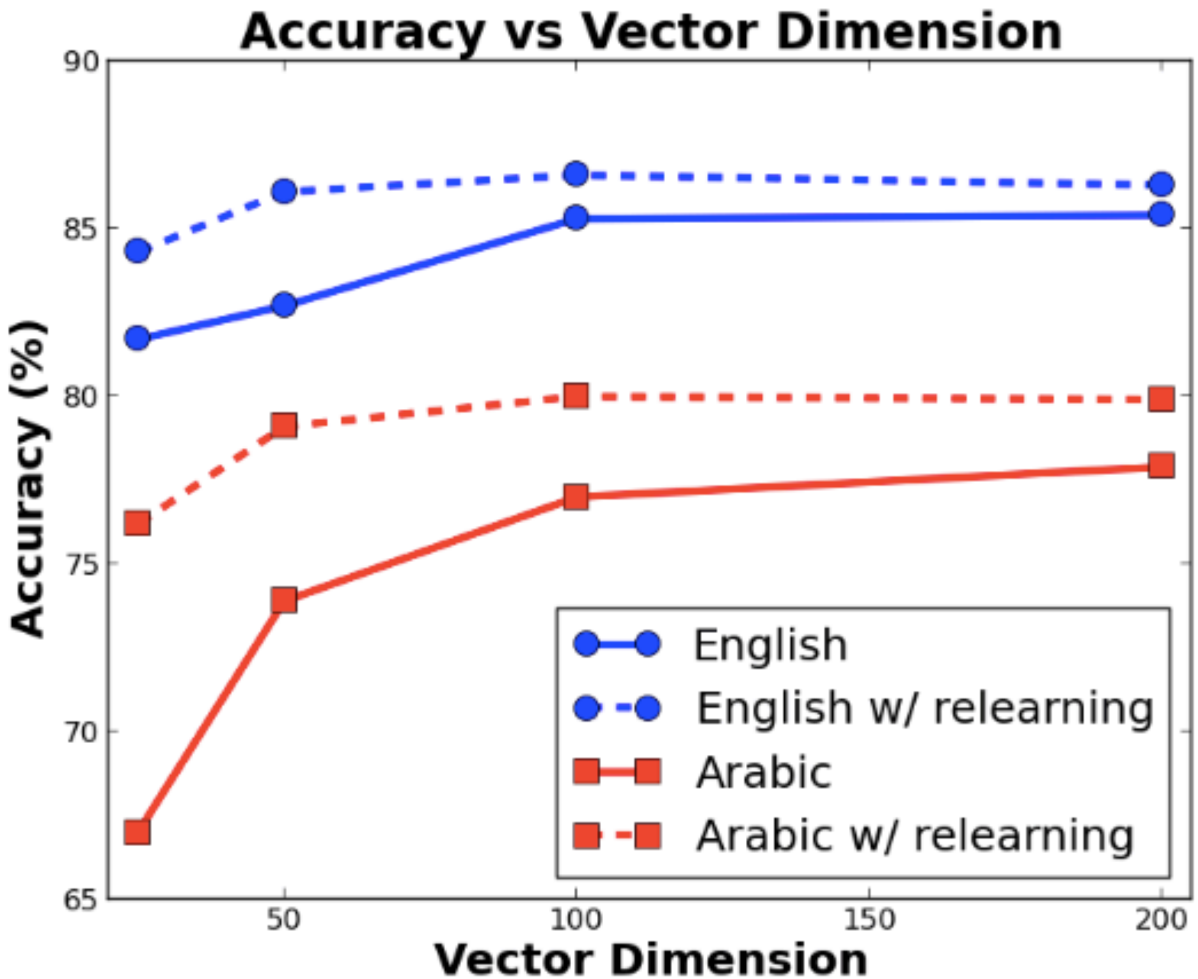
System	Arabic	English
RBG	87.70	93.96
RBG + predicted PP	87.95	94.05

Contribution of Model Components

- Word representations
  - Relearning word vectors
  - Enriching word vectors
  - Using syntactic word vectors

- Composition architectures

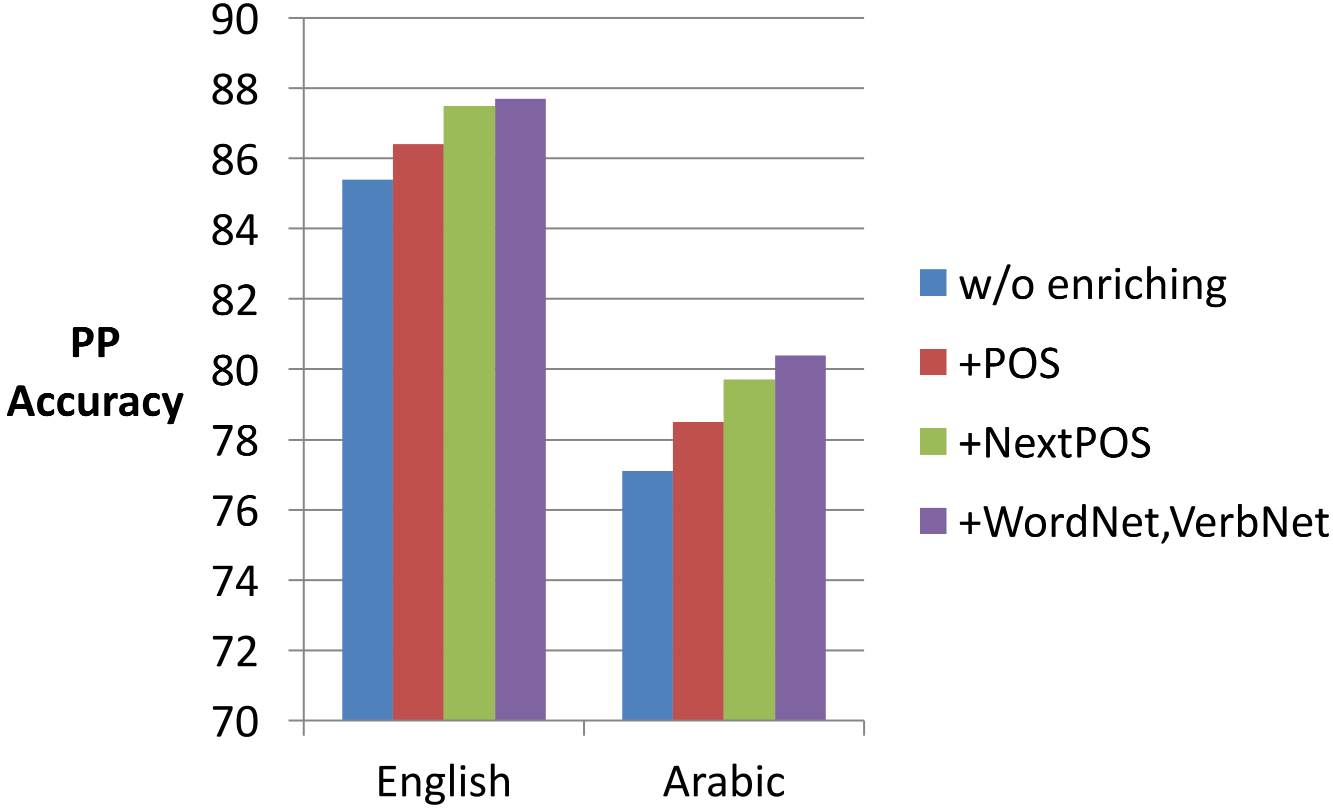
Effect of Relearning Word Vectors



Effect of Enriching Word Vectors

Representation	**Arabic**	**English**
w/o enriching	77.1	85.4
w/ enriching +POS +NextPOS +WordNet+VerbNet	 78.5 79.7 80.4	 86.4 87.5 87.7
w/ enriching+relearning	81.7	88.1
w/ enriching+relearning+syntactic	82.6	88.7

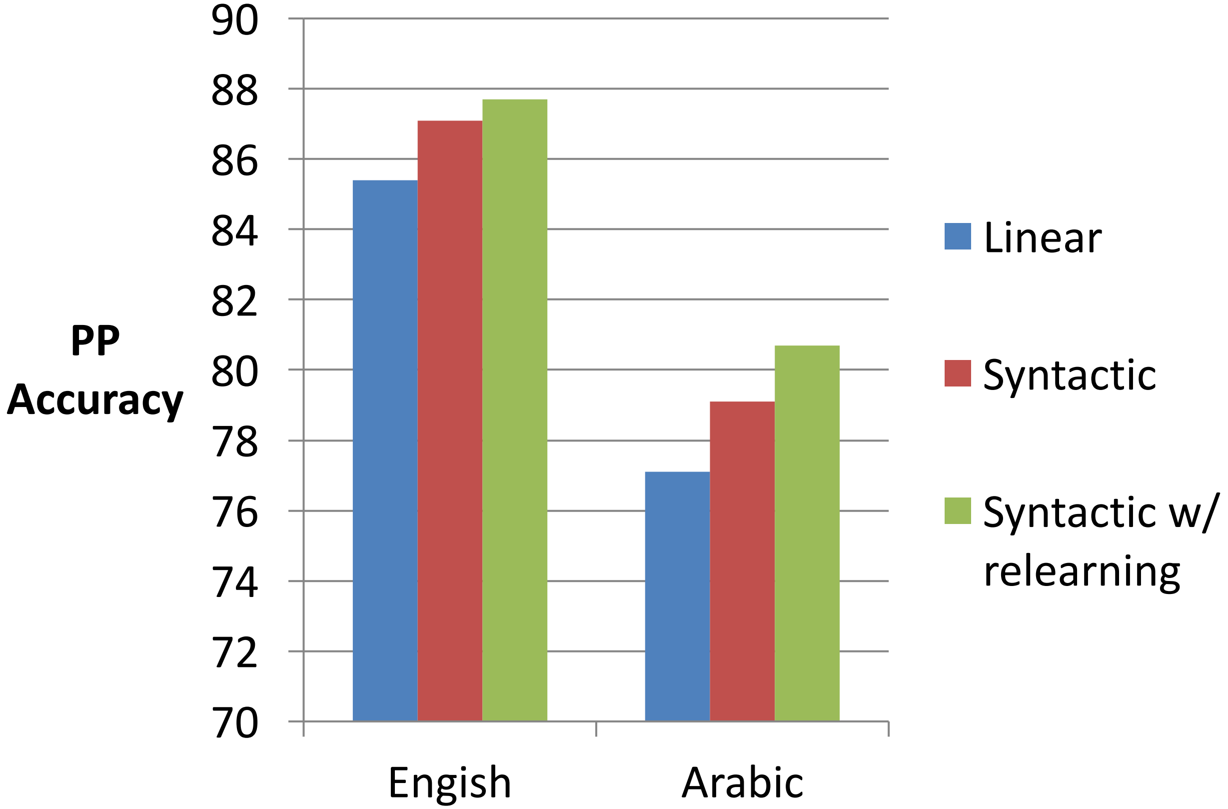
Effect of Enriching Word Vectors



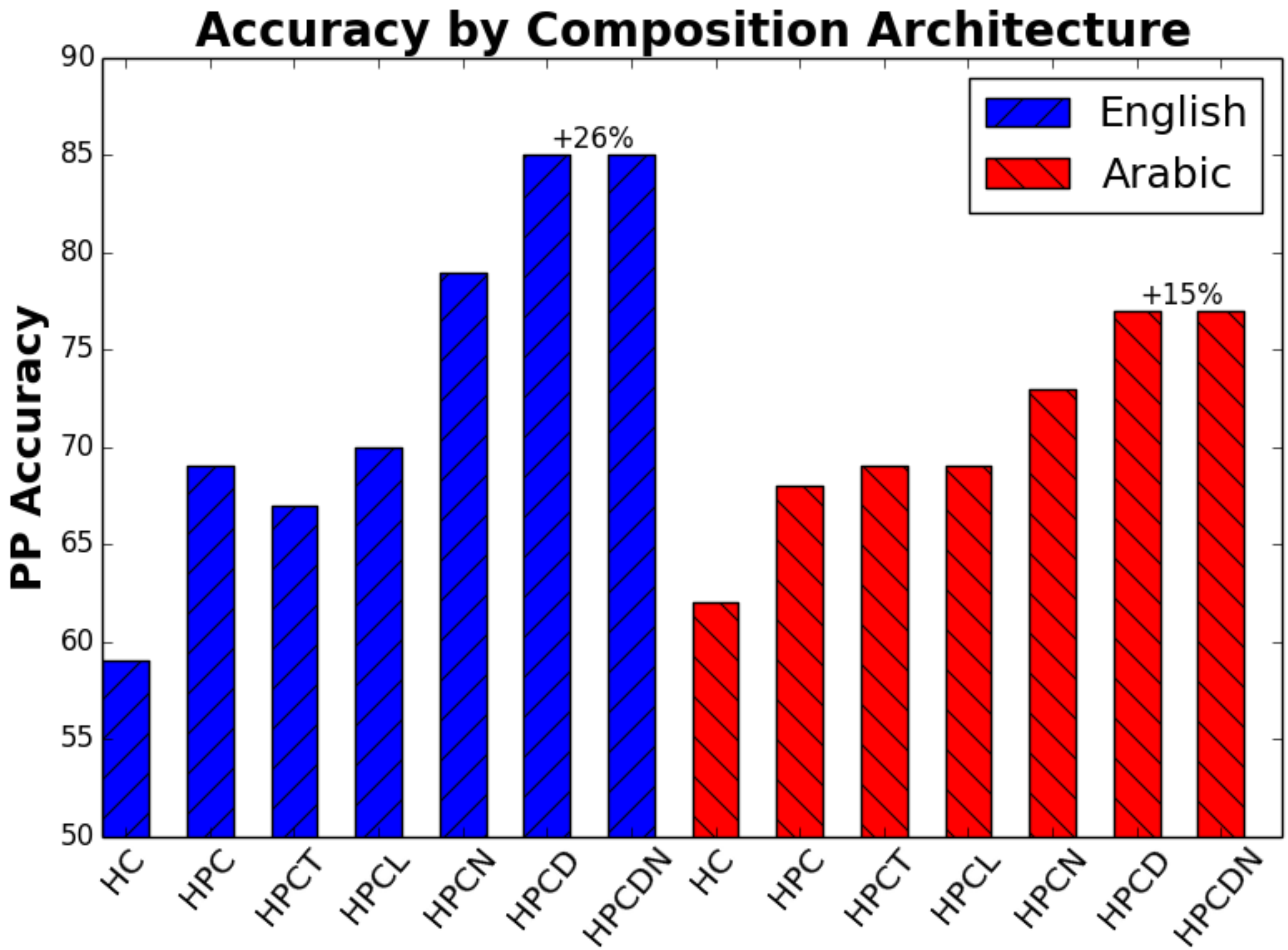
Effect of Syntactic Word Vectors

Representation	Arabic	English
Linear (standard)	77.1	85.4
Syntactic	79.1	87.1
Syntactic w/ relearning	80.7	87.7

Effect of Syntactic Word Vectors



Effect of Composition Architectures



Contributions

- Develop a compositional neural network model dedicated for PP attachment
- Explore utility of different word vector representations
- Improve performance of a state-of-the-art parser

- Code and data: http://groups.csail.mit.edu/rbg/code/pp