A Study on Compositional Semantics of Words in Distributional Spaces

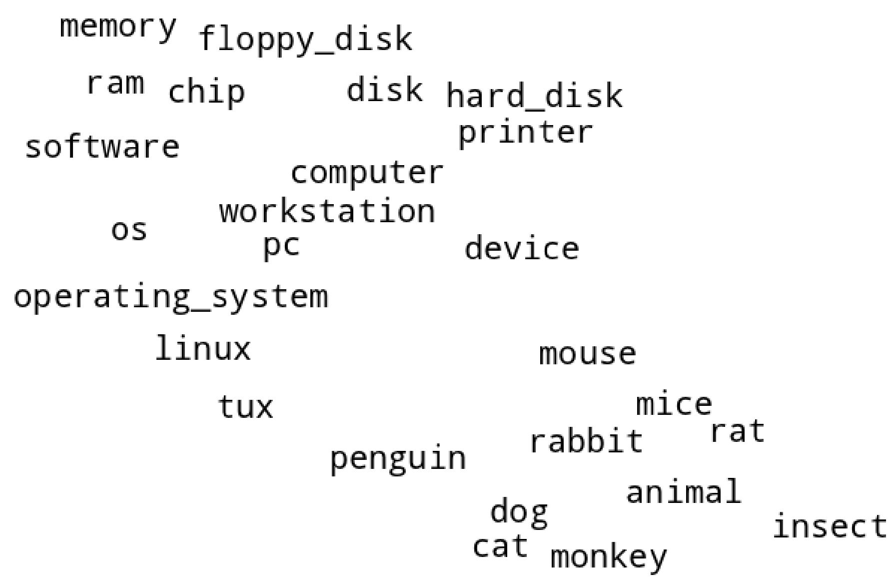
Pierpaolo Basile, Annalina Caputo,
Giovanni Semeraro
Dept. of Computer Science
University of Bari Aldo Moro
{basilepp}@di.uniba.it

6th IEEE International Conference on Semantic Computing
September 19-21, 2012 Palermo, Italy

Background...

You shall know a word by the company it keeps!

Meaning of a word is determined by its usage



…Background

Distributional Semantic Models (DSMs) defined as: <T, C, R, W, M, d, S>

- T: target elements (words)
- **C: context**
- R: relation between T and C
- W: weighting schema
- M: geometric space TxC
- d: space reduction M -> M’
- S: similarity function defined in M’

Motivations

- One definition of context at a time
  - encode syntactic information in DSMs
- Words are represented in isolation
  - syntactic role could be used as a glue to compose words

It’s raining cats and dogs = My cats and dogs are in the rain

Outline

- Simple DSMs and simple operators
- Syntactic dependencies in DSMs
  - Structured DSMs
  - Compositional operators
- Evaluation and results
- Final remarks

SIMPLE DSMS AND SIMPLE OPERATORS

Simple DSMs…

Term-term co-occurrence matrix (TTM): each cell contains the co-occurrences between two terms within a prefixed distance

	dog	cat	computer	animal	mouse
dog	0	4	0	2	1
cat	4	0	0	3	5
computer	0	0	0	0	3
animal	2	3	0	0	2
mouse	1	5	3	2	0

…Simple DSMs

Latent Semantic Analysis (LSA): relies on the Singular Value Decomposition (SVD) of the co-occurrence matrix

Random Indexing (RI): based on the Random Projection

Latent Semantic Analysis over Random Indexing (RI^{LSA})

Random Indexing

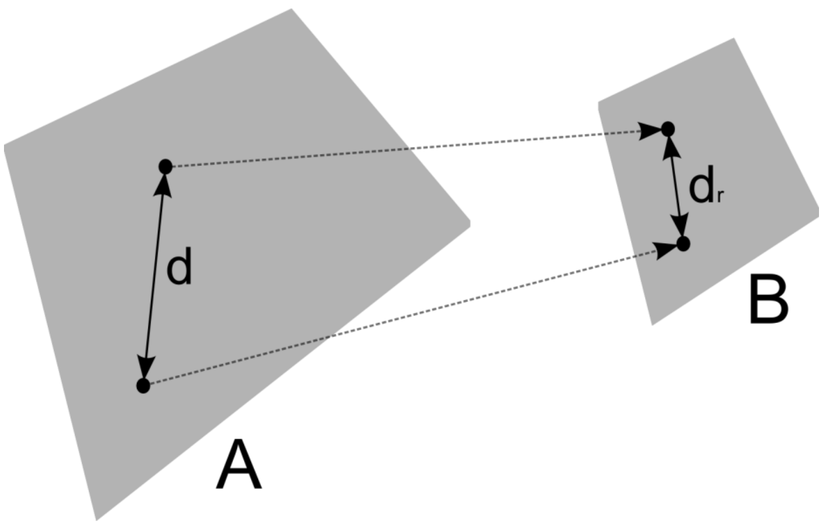
- Create and assign a context vector to each context element (e.g. document, passage, term, …)
- Term vector is the sum of the context vectors in which the term occurs
  - sometimes the context vector could be boosted by a score (e.g. term frequency, PMI, …)

Context Vector

0 0 0 0 0 0 0 -1 0 0 0 0 1 0 0 -1 0 1 0 0 0 0 1 0 0 0 0 -1

- sparse
- high dimensional
- ternary {-1, 0, +1}
- small number of randomly distributed non-zero elements

Random Indexing (formal)



$$B^{n,k} = A^{n,m} R^{m,k} \quad k << m$$

B nearly preserves the distance between points (Johnson-Lindenstrauss lemma)

$$d_r = c \times d$$

RI is a locality-sensitive hashing method which approximate the cosine distance between vectors

Random Indexing (example)

John eats a red apple

CV_{john} -> (0, 0, 0, 0, 0, 0, 1, 0, -1, 0)
CV_{eat} -> (1, 0, 0, 0, -1, 0, 0 ,0 ,0 ,0)
CV_{red} -> (0, 0, 0, 1, 0, 0, 0, -1, 0, 0)

$$TV_{apple}= CV_{john}+ CV_{eat}+CV_{red}=(1, 0, 0, 1, -1, 0, 1, -1, -1, 0)$$

Latent Semantic Analysis over Random Indexing

1. Reduce the dimension of the co-occurrences matrix using RI
2. Perform LSA over RI (LSARI)
   – reduction of LSA computation time: RI matrix contains less dimensions than co-occurrences matrix

Simple operators…

Addition (+): pointwise sum of components

Multiplication ($\circ$): pointwise multiplication of components

Addition and multiplication are commutative

- do not take into account word order

Complex structures represented summing or multiplying words which compose them

…Simple operators

Given two word vectors **u** and **v**

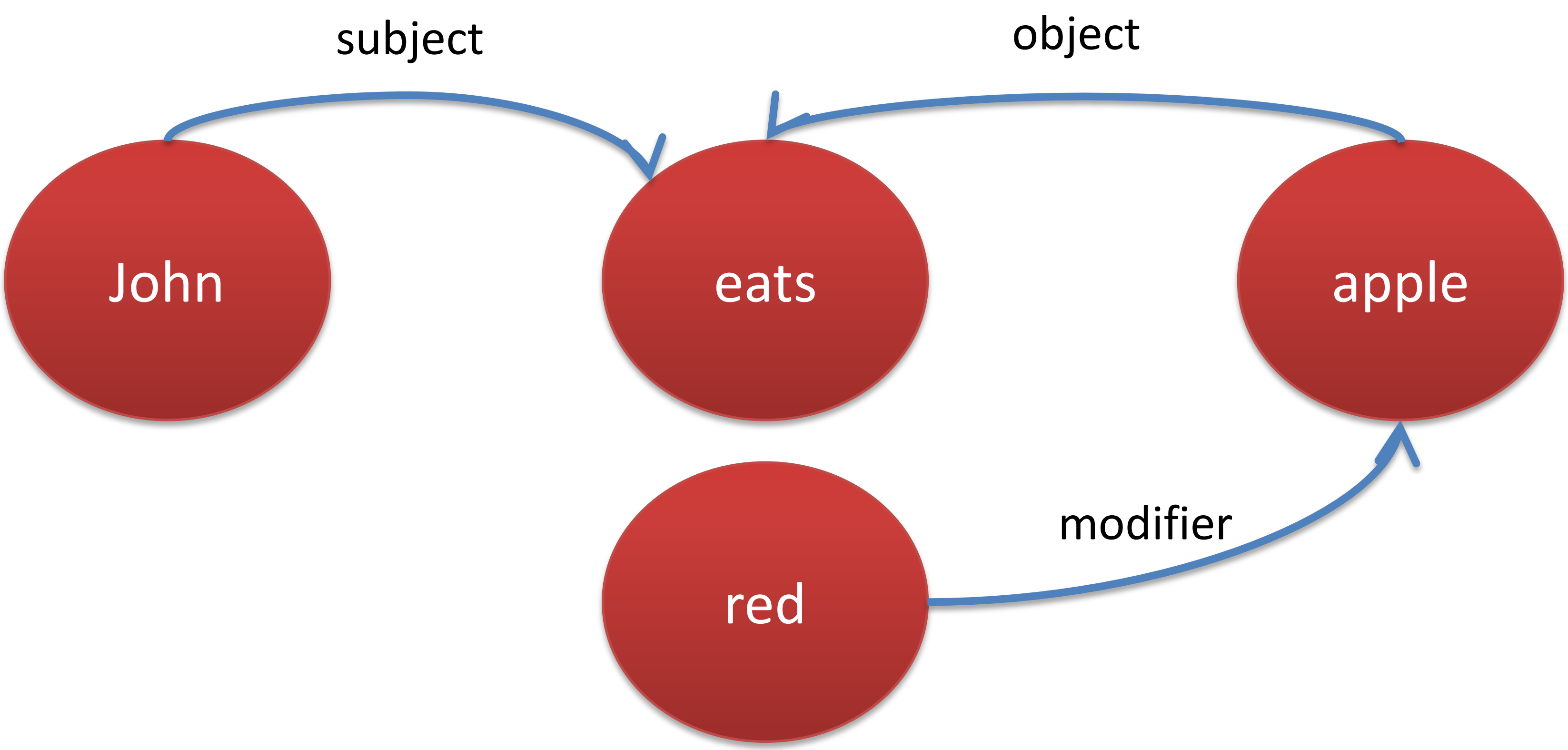
- composition by sum $\mathbf{p} = \mathbf{u} + \mathbf{v}$
- composition by multiplication $\mathbf{p} = \mathbf{u} \circ \mathbf{v}$

Can be applied to any sequence of words

SYNTACTIC DEPENDENCIES IN DSMS

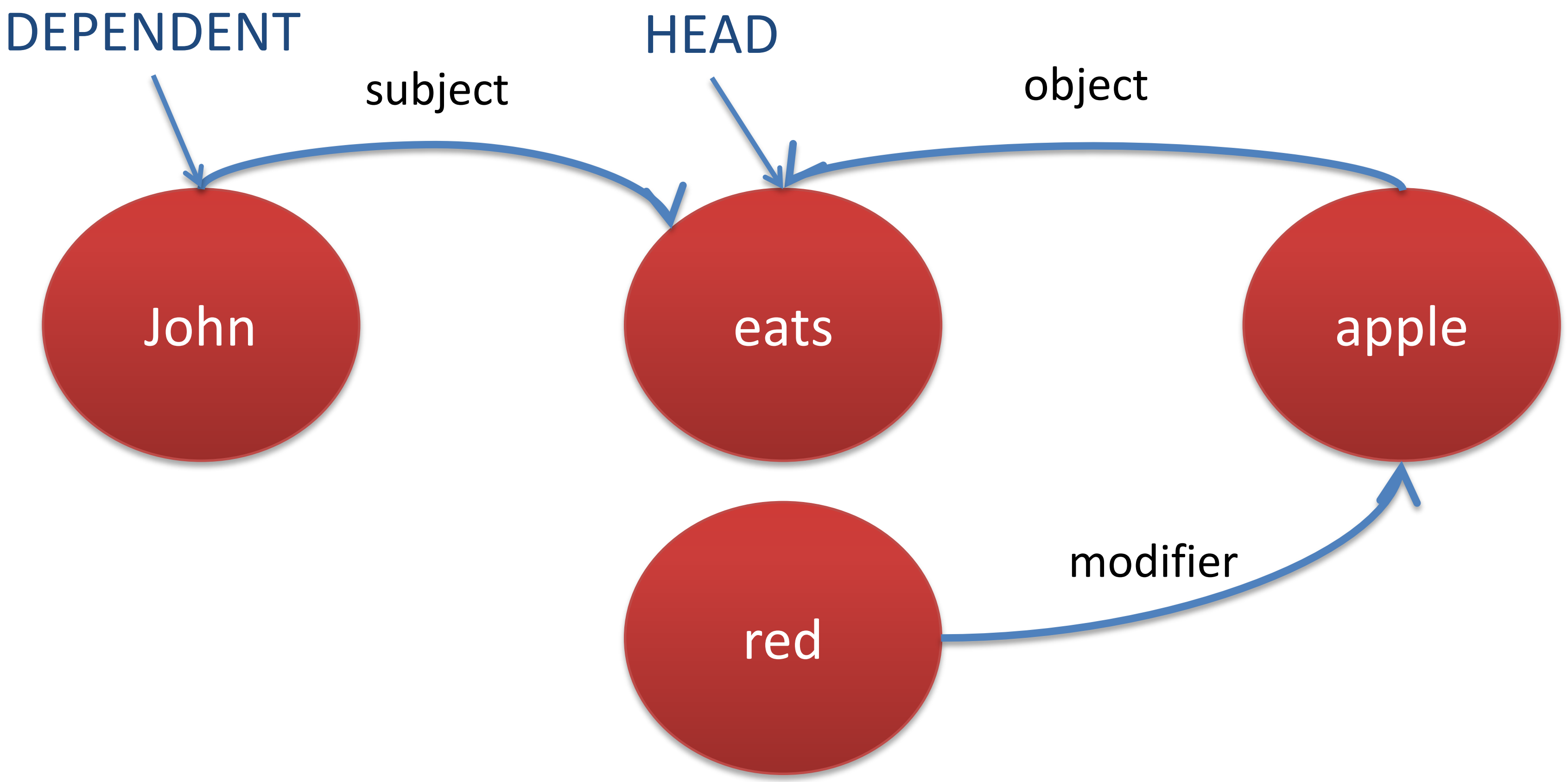
Syntactic dependencies…

John eats a red apple.



…Syntactic dependencies

John eats a red apple.



Representing dependences

Use filler/role binding approach to represent a dependency *dep(**u**, **v**)*

$$r_d \odot \mathbf{u} + r_h \odot \mathbf{v}$$

r_d and r_h are vectors which represent respectively the role of dependent and head

$\odot$ is a placeholder for a composition operator

Representing dependences (example)

obj(apple, eat)

$$r_d \odot \mathbf{apple} + r_h \odot \mathbf{eat}$$

role vectors

Structured DSMs

1. Vector permutation in RI (PERM) to encode dependencies
2. Circular convolution (CONV) as filler/binding operator to represent syntactic dependencies in DSMs
3. LSA over PERM and CONV carries out two spaces: $PERM^{LSA}$ and $CONV^{LSA}$

Vector permutation in RI (PERM)

Using permutation of elements in context vectors to encode dependencies

- right rotation of *n* elements to encode dependents (permutation)
- left rotation of *n* elements to encode heads (inverse permutation)

PERM (method)

Create and assign a context vector to each term

Assign a rotation function Π^{+1} to the dependent and Π^{-1} to the head

Each term is represented by a vector which is

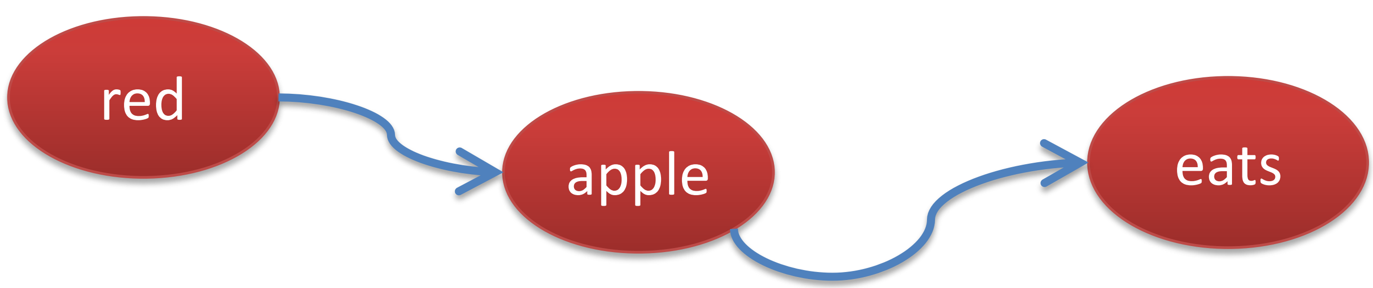
- the sum of the permuted vectors of all the dependent terms
- the sum of the inverse permuted vectors of all the head terms
- the sum of the no-permuted vectors of both dependent and head words

PERM (example…)

John eats a red apple

John -> (0, 0, 0, 0, 0, 0, 1, 0, -1, 0)
eat -> (1, 0, 0, 0, -1, 0, 0 ,0 ,0 ,0)
red -> (0, 0, 0, 1, 0, 0, 0, -1, 0, 0)
apple -> (1, 0, 0, 0, 0, 0, 0, -1, 0, 0)

$$TV_{apple} = \Pi^{+1}(CV_{red}) + \Pi^{-1}(CV_{eat}) + CV_{red} + CV_{eat}$$



PERM (...example)

John -> (0, 0, 0, 0, 0, 0, 1, 0, -1, 0)
eat -> (1, 0, 0, 0, -1, 0, 0 ,0 ,0 ,0)
red-> (0, 0, 0, 1, 0, 0, 0, -1, 0, 0)
apple -> (1, 0, 0, 0, 0, 0, 0, -1, 0, 0)

John eats a red apple

$TV_{apple} = \Pi^{+1}(CV_{red}) + \Pi^{-1}(CV_{eat}) + CV_{red} + CV_{eat} = \ldots$

right shift

left shift

...=(0, 0, 0, 0, 1, 0, 0, 0, -1, 0) + (0, 0, 0, -1, 0, 0, 0, 0, 1) +

+ (0, 0, 0, 1, 0, 0, 0, -1, 0, 0) + (1, 0, 0, 0, -1, 0, 0 ,0 ,0 ,0)

Convolution (CONV)

Create and assign a context vector to each term

Create two context vectors for head and dependent roles

Each term is represented by a vector which is

- the sum of the convolution between dependent terms and the dependent role vector
- the sum of the convolution between head terms and the head role vector
- the sum of the vectors of both dependent and head words

Circular convolution operator

Circular convolution

$$\mathbf{p} = \mathbf{u} \circledast \mathbf{v}$$

defined as:

$$p_j = \sum_{k=1}^{n} u_k v_{(j-k) \equiv (n+1)}$$

U=<1, 1, -1, -1, 1>
V=<1, -1, 1, -1, -1>
P=<-1, 3, -1, -1, -1>

		U_1	U_2	U_3	U_4	U_5
P_1	V_1	1	1	-1	-1	1
P_2	V_2	-1	-1	1	1	-1
P_3	V_3	1	1	-1	-1	1
P_4	V_4	-1	-1	1	1	-1
P_5	V_5	-1	-1	1	1	-1

Circular convolution by FFTs

Circular convolution is computed in O(n^2)

– using FFTs is computed in O($n\log n$)

Given f the discrete FFTs and f^{-1} its inverse

– $u \circledast v = f^{-1}(f(u) \circ f(v))$

CONV (example)

John eats a red apple

John -> (0, 0, 0, 0, 0, 0, 1, 0, -1, 0)
eat -> (1, 0, 0, 0, -1, 0, 0 ,0 ,0 ,0)
red-> (0, 0, 0, 1, 0, 0, 0, -1, 0, 0)
apple -> (1, 0, 0, 0, 0, 0, 0, -1, 0, 0)
r_d -> (0, 0, 1, 0, -1, 0, 0, 0, 0, 0)
r_h -> (0, -1, 1, 0, 0, 0, 0, 0, 0, 0)

$$\text{apple} = \text{eat} + \text{red} + (r_d \circledast \text{red}) + (r_h \circledast \text{eat})$$

Context vector for dependent role

Context vector for head role

Complex operators

Based on filler/role binding taking into account syntactic role: $\mathbf{r_d} \odot \mathbf{u} + \mathbf{r_h} \odot \mathbf{v}$

- $\mathbf{u}$ and $\mathbf{v}$ could be recursive structures

Two vector operators to bind the role:

- convolution ($\circledast$)
- tensor ($\otimes$)
- convolution ($\circledast^{+}$): exploits also the sum of term vectors

$$\mathbf{r_d} \circledast \mathbf{u} + \mathbf{r_h} \circledast \mathbf{v} + \mathbf{v} + \mathbf{u}$$

Complex operators (remarks)

Existing operators

- $t_1 \odot t_2 \odot \ldots \odot t_n$: does not take into account syntactic role
- $t_1 \circledast t_2$ is commutative
- $t_1 \otimes t_2 \otimes \ldots \otimes t_n$: tensor order depends on the phrase length
  - two phrases with different length are not comparable
- $t_1 \otimes r_1 \otimes t_2 \otimes r_2 \otimes \ldots \otimes t_n \otimes r_n$: also depends on the sentence length

System setup

- Corpus
  - WaCkypedia EN based on a 2009 dump of Wikipedia
  - about 800 million tokens
  - dependency parse by MaltParser
- DSMs
  - 500 vector dimension (LSA/RI/RI^{LSA})
  - 1,000 vector dimension (PERM/CONV/$PERM^{LSA}$/$CONV^{LSA}$)
  - 50,000 most frequent words
  - co-occurrence distance: 4

Evaluation

- GEMS 2011 Shared Task for compositional semantics
  - list of two pairs of words combination

    ```
    (support offer) (help provide) 7
    (old person) (right hand) 1
    ```

    - rated by humans
    - 5,833 rates
    - 3 types involved: noun-noun (NN), adjective-noun (AN), verb-object (VO)
  - GOAL: compare the system performance against humans scores
    - Spearman correlation

Results (simple spaces)…

	NN				AN				VO			
	TTM	LSA	RI	RI^{LSA}	TTM	LSA	RI	RI^{LSA}	TTM	LSA	RI	RI^{LSA}
+	.21	.36	.25	.42	.22	.35	.33	.41	.23	.31	.28	.31
$\circ$	.31	.15	.23	.22	.21	.20	.22	.18	.13	.10	.18	.21
$\circledast$	.21	.38	.26	.35	.20	.33	.31	.44	.15	.31	.24	**.34**
$\circledast^{+}$	.21	.34	.28	**.43**	.23	.32	.31	.37	.20	.31	.25	.29
$\otimes$	.21	.38	.25	.39	.22	.38	.33	.43	.15	.34	.26	.32
human	.49				.52				.55			

Simple Semantic Spaces

…Results (structured spaces)

	NN				AN				VO			
	CONV	PERM	$\mathrm{CONV}^{\mathrm{LSA}}$	$\mathrm{PERM}^{\mathrm{LSA}}$	CONV	PERM	$\mathrm{CONV}^{\mathrm{LSA}}$	$\mathrm{PERM}^{\mathrm{LSA}}$	CONV	PERM	$\mathrm{CONV}^{\mathrm{LSA}}$	$\mathrm{PERM}^{\mathrm{LSA}}$
$+$	.36	.39	**.43**	.42	.34	.39	.42	.45	.27	.23	.30	.31
$\circ$	.22	.17	.10	.13	.23	.27	.13	.15	.20	.15	.06	.14
$\circledast$	.31	.36	.37	.35	.39	.39	.45	.44	.28	.23	.27	.28
$\circledast^{+}$	.30	.36	.40	.36	.38	.32	**.48**	.44	.27	.22	.30	.32
$\otimes$	.34	.37	.37	.40	.36	.40	.45	.45	.27	.24	.31	.32
human	.49				.52				.55			

Final remarks

- Best results are obtained when complex operators/spaces (or both) are involved
- No best combination of operator/space exists
  - depend on the type of relation (NN, AN, VO)
- Tensor product and convolution provide good results in spite of previous results
  - filler/role binding is effective
- Future work
  - generate several r_d and r_h vectors for each kind of dependency
  - apply this approach to other direct graph-based representations

Thank you for your attention!

Questions?

<basilepp@di.uniba.it>