Deep Approaches to Semantic Matching for Text

Jun Xu
Institute of Computing Technology, Chinese Academy of Sciences
junxu@ict.ac.cn

Outline

- Problems with direct methods
- Deep matching models for text
  - Composition focused methods
  - Interaction focused methods
- Summary

Problems with direct methods

[Problem 1] ***The order information of words is missing***

Bag of words assumption:

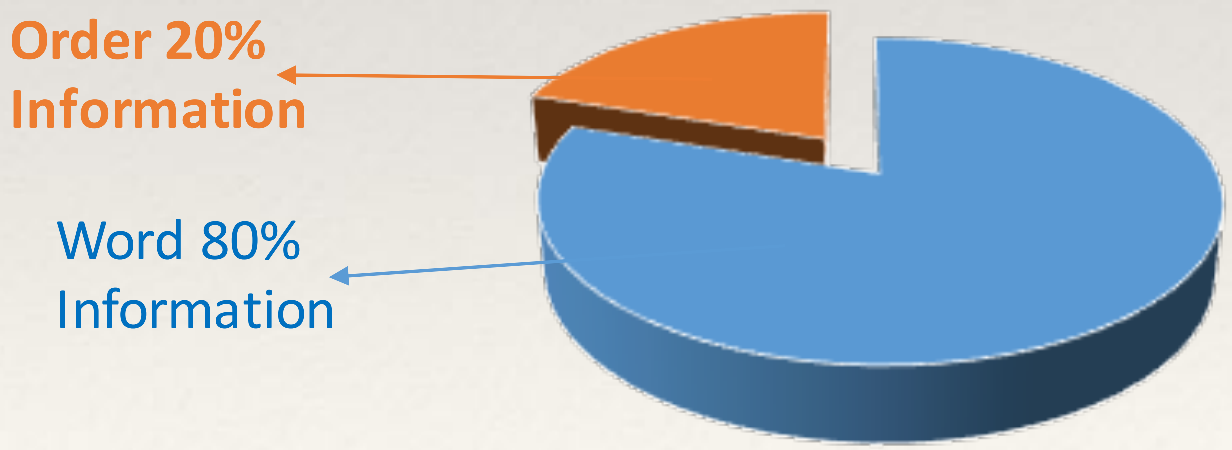
hot dog = dog hot

However:

hot dog ≠ dog hot

The importance of the words order

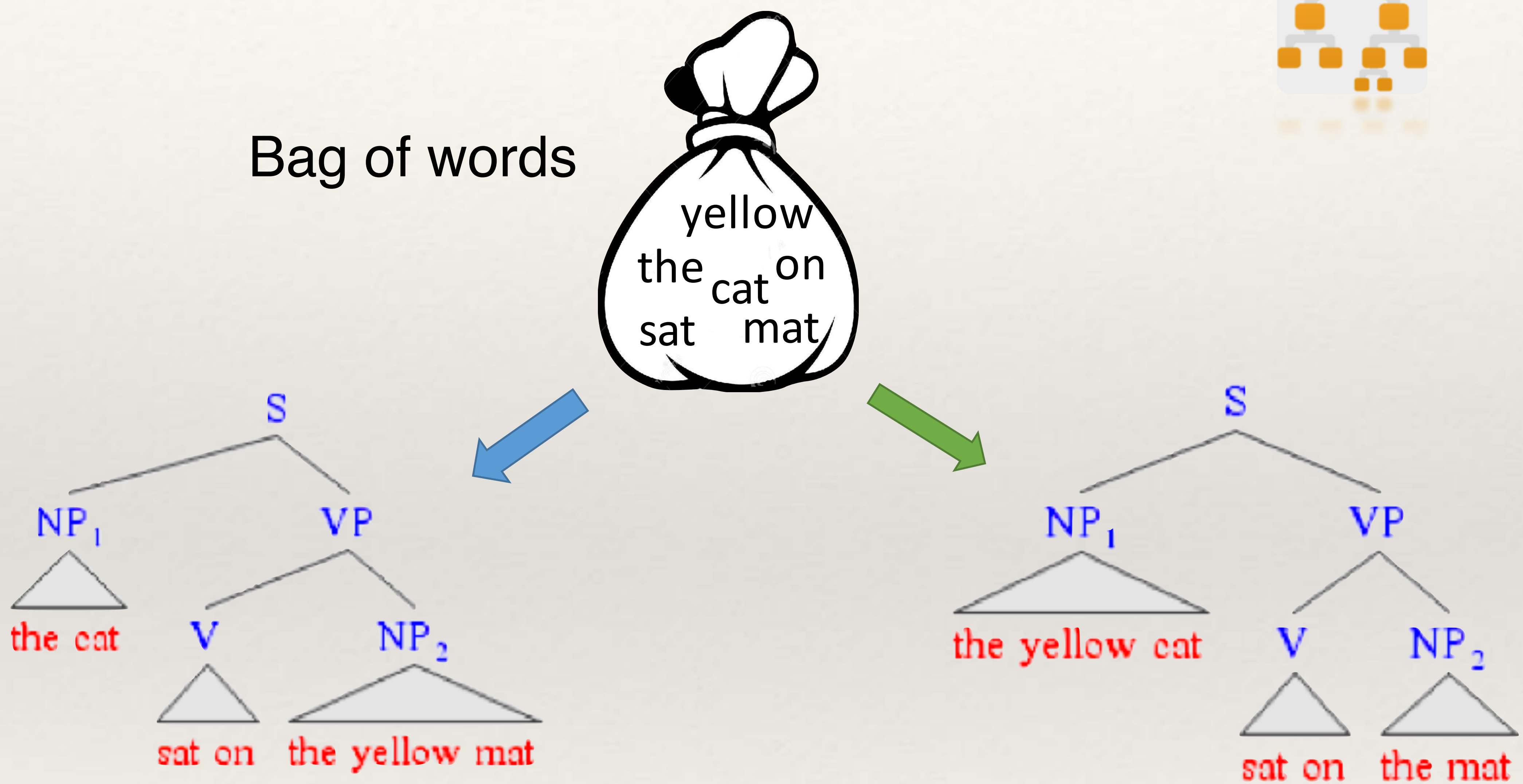
- Assume that comprehension vocabulary is 100,000 words, that sentences are 20 words long, and that word order is important only within sentences.
- Then the contributions, in bits are $\log_2(100000^{20})$ and $\log_2(20!)$ respectively, which works out to over 80% of the potential information in language being in the choice of words without regard to the order in which they appear.



Landauer T K. On the computational basis of learning and cognition: Arguments from LSA[J]. Psychology of learning and motivation, 2002, 41: 43-84.

Problems with direct methods

[Problem 2] ***Over simplified sentence representation***



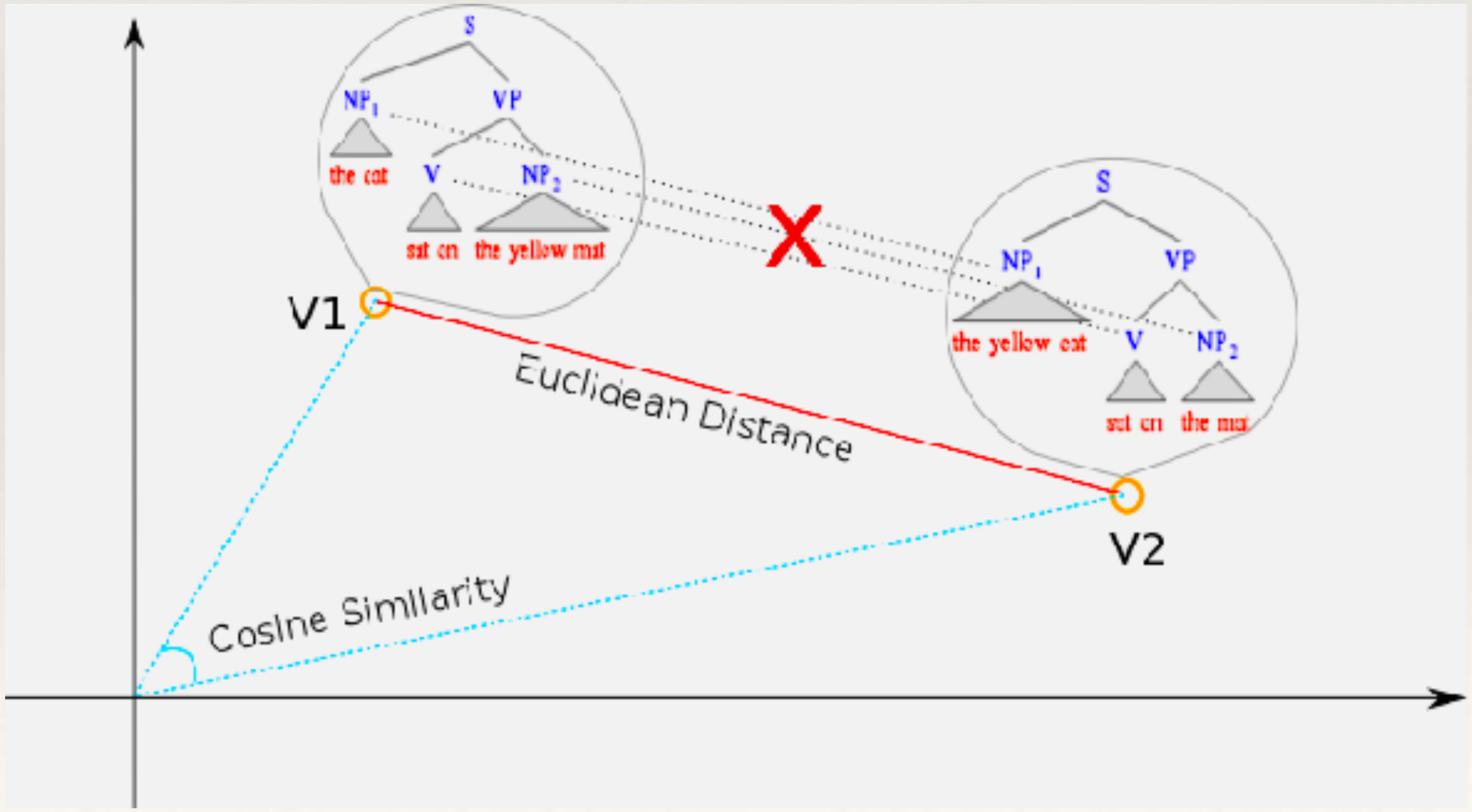
"The cat sat on the **yellow mat** = The **yellow cat** sat on the mat" under bag-of-words assumption

Problems with direct methods

[Problem 3] *Heuristic matching function*

- A vector for representing the whole sentence
- Based on distance measures between two vectors
  - Cosine, Euclidean distance …

Limited information of two vectors are taken into consideration

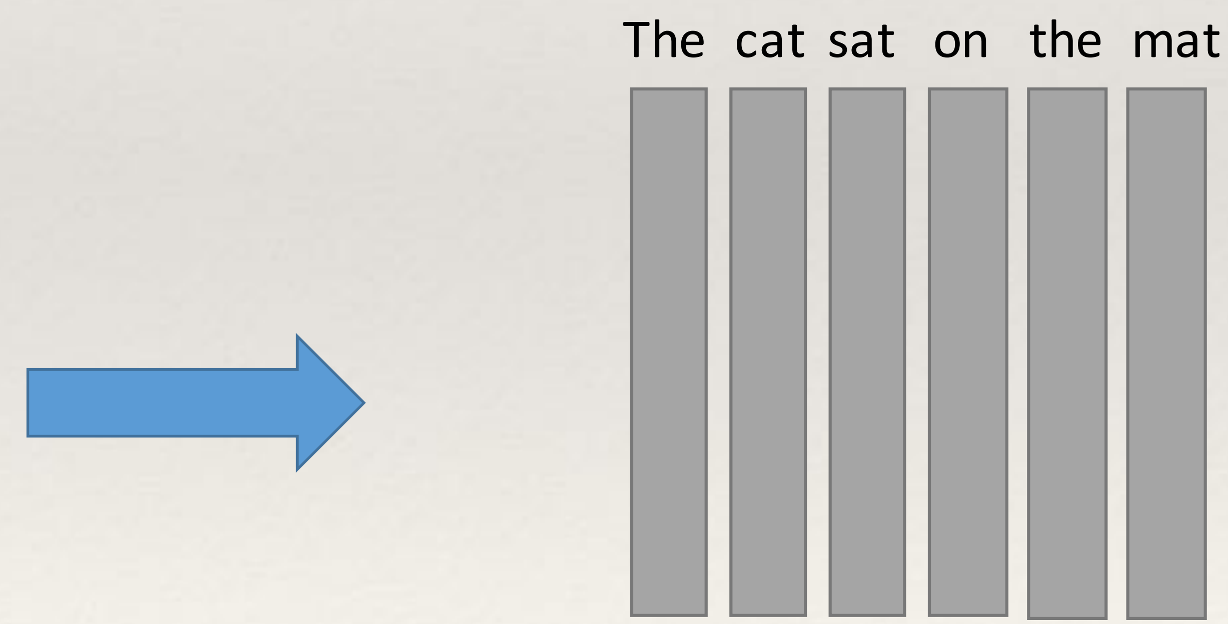


How to design deep semantic matching models for text?

Keeping order information

- A sequence of word embeddings
  - Convert each word to its embedding (e.g., word2vec)
  - Concatenate embeddings to a sequence

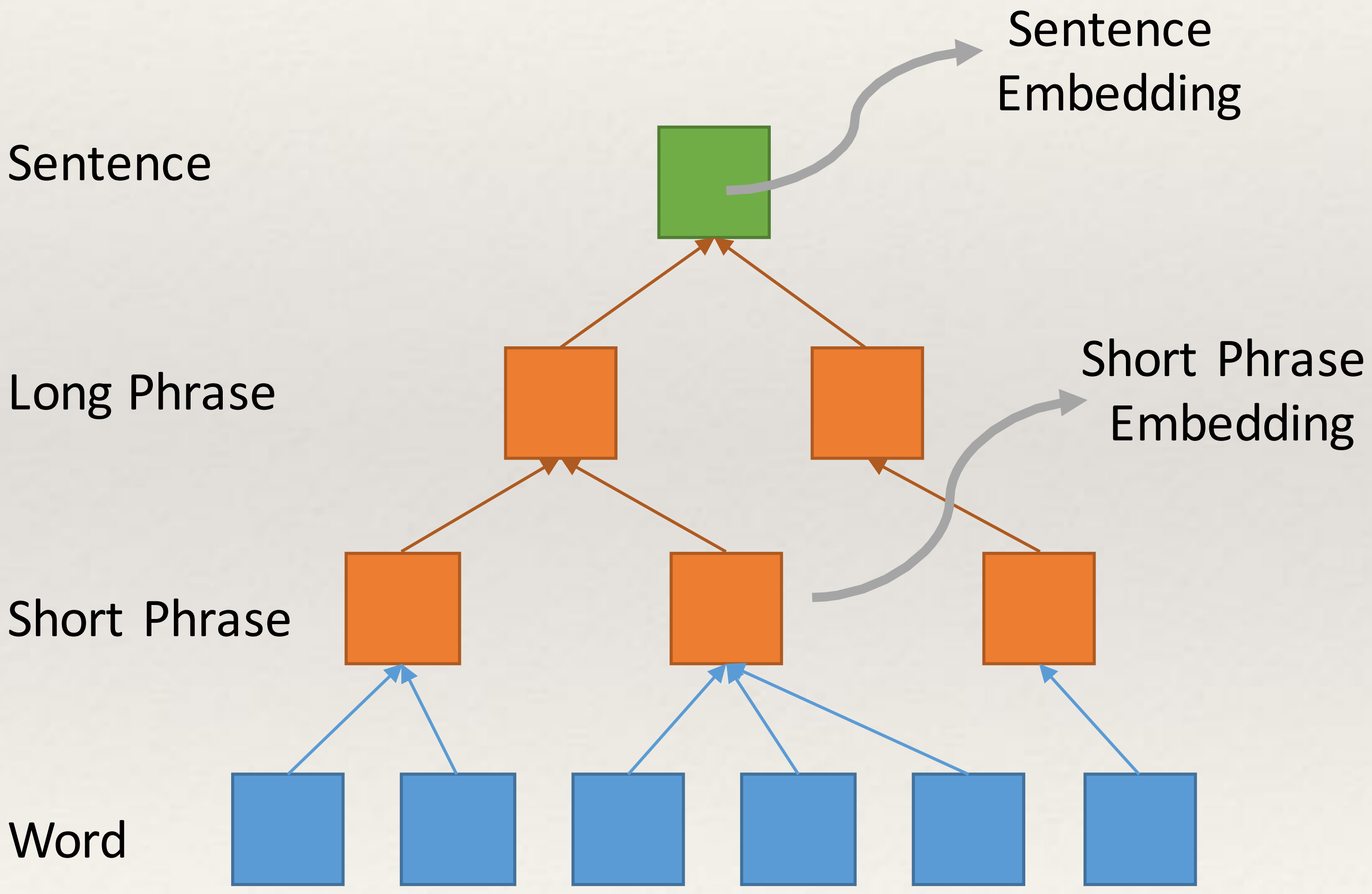
Bag of Word Embeddings



Sequence of Word Embeddings

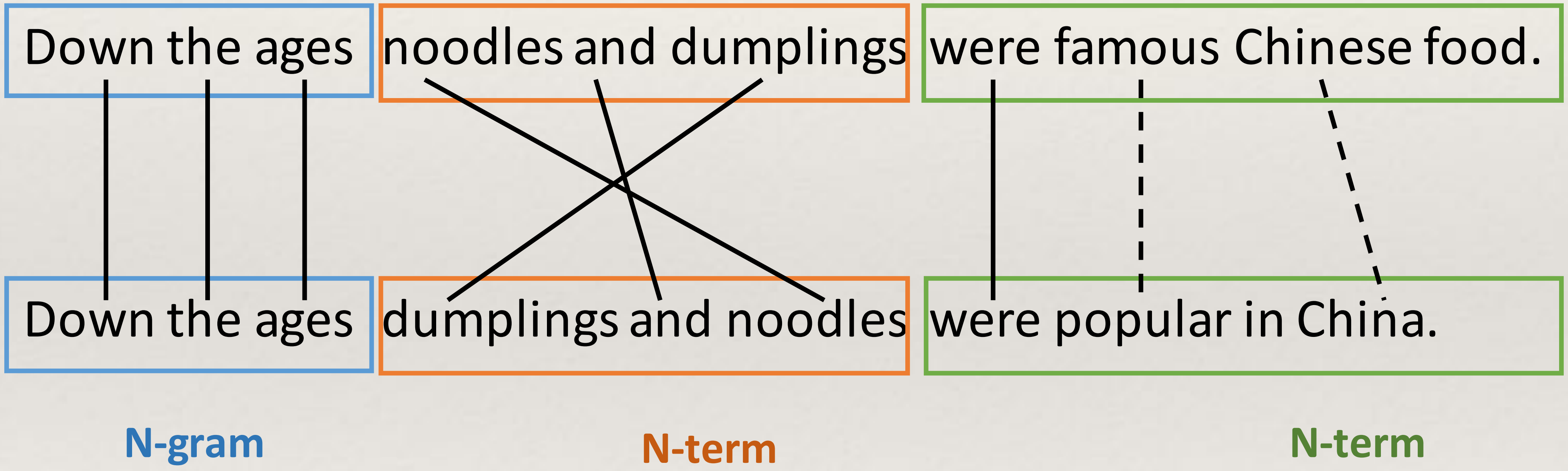
Rich sentence representation

- Hierarchical structure of sentence representation, e.g., different levels of embeddings



Powerful matching function

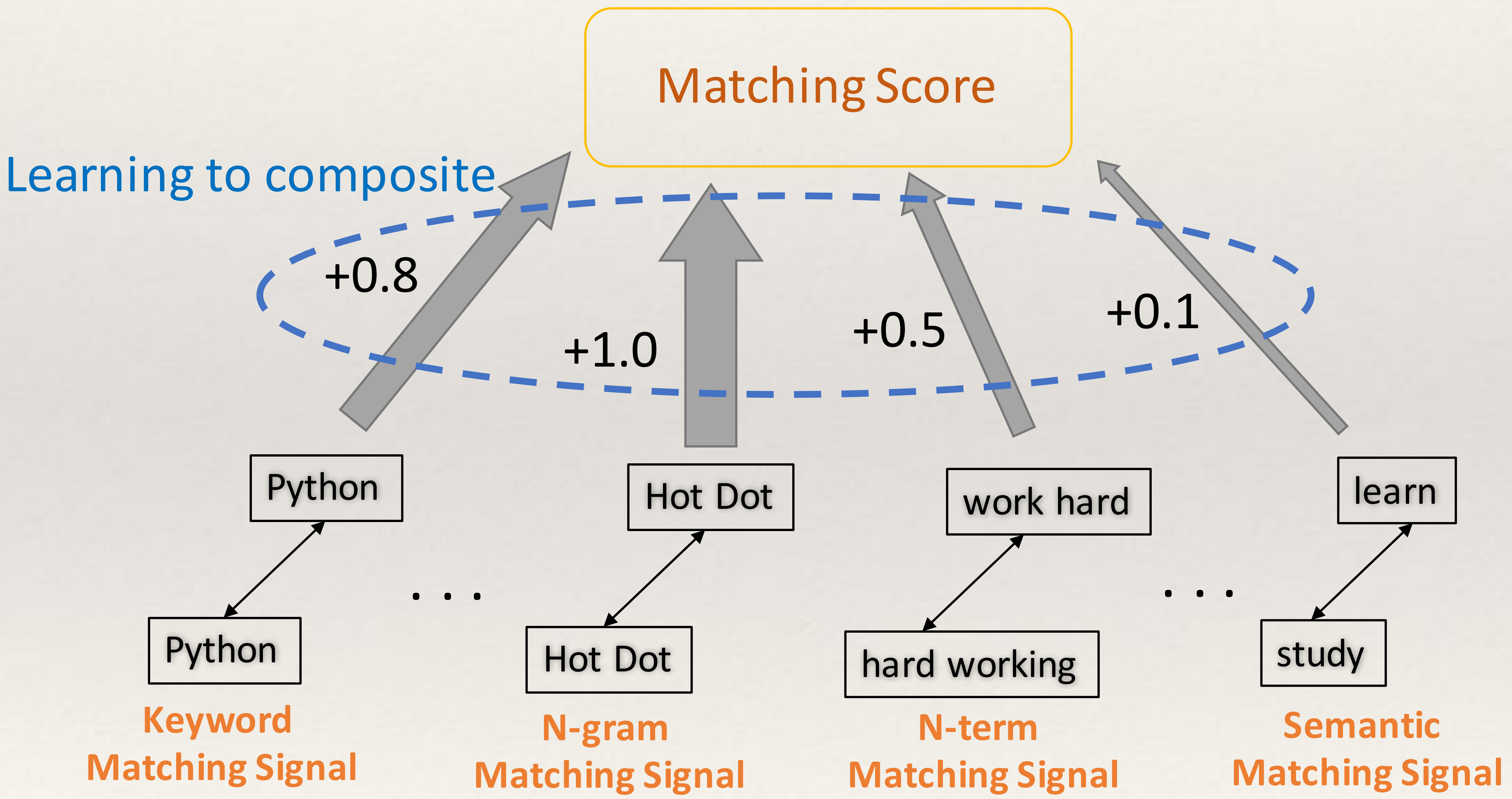
- Considering different levels / types of matching signals



Pang L, Lan Y, Guo J, et al. Text matching as image recognition//Proceedings of the 30th AAAI Conference on Artificial Intelligence. Phoenix, USA, 2016: 2793-2799.

Learning the matching function

- Data-driven approaches to determining the parameters



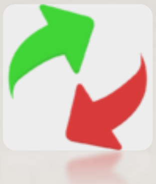
Outline

- Problems with direct methods
- Deep matching models for text
  - Composition focused
  - Interaction focused
- Summary

Existing deep text matching models

- Composition focused methods
  - **[Problem 1: order] [Problem 2: structure]**
  - Composite each sentence into one embedding
  - Measure the similarity between the two embeddings

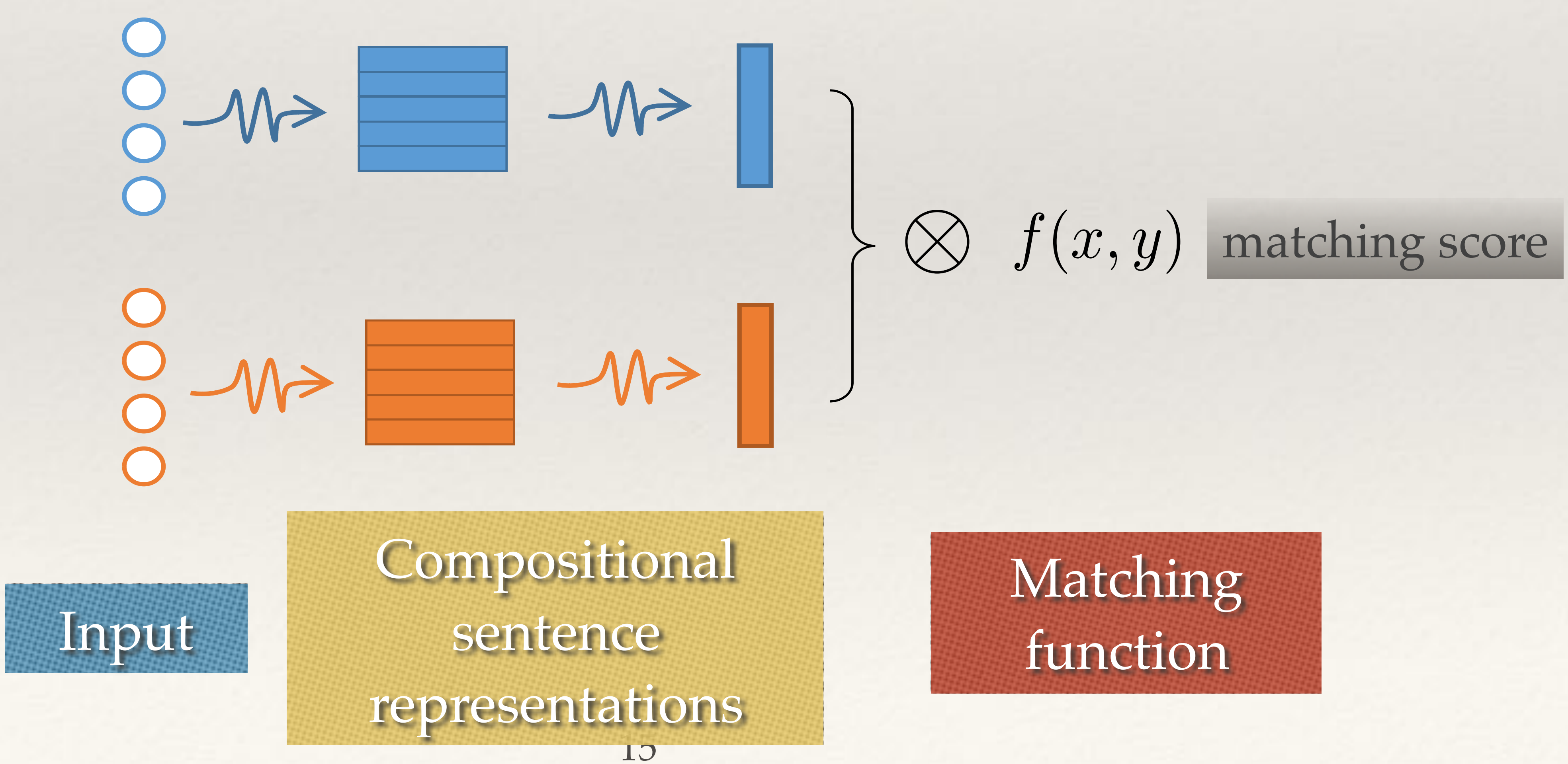

- Interaction focused methods
  - **[Problem 1: order] [Problem 3: matching function]**
  - Two sentences meet before their own high-level representations mature
  - Capture complex matching patterns




Composition Focused Methods

Composition focused methods

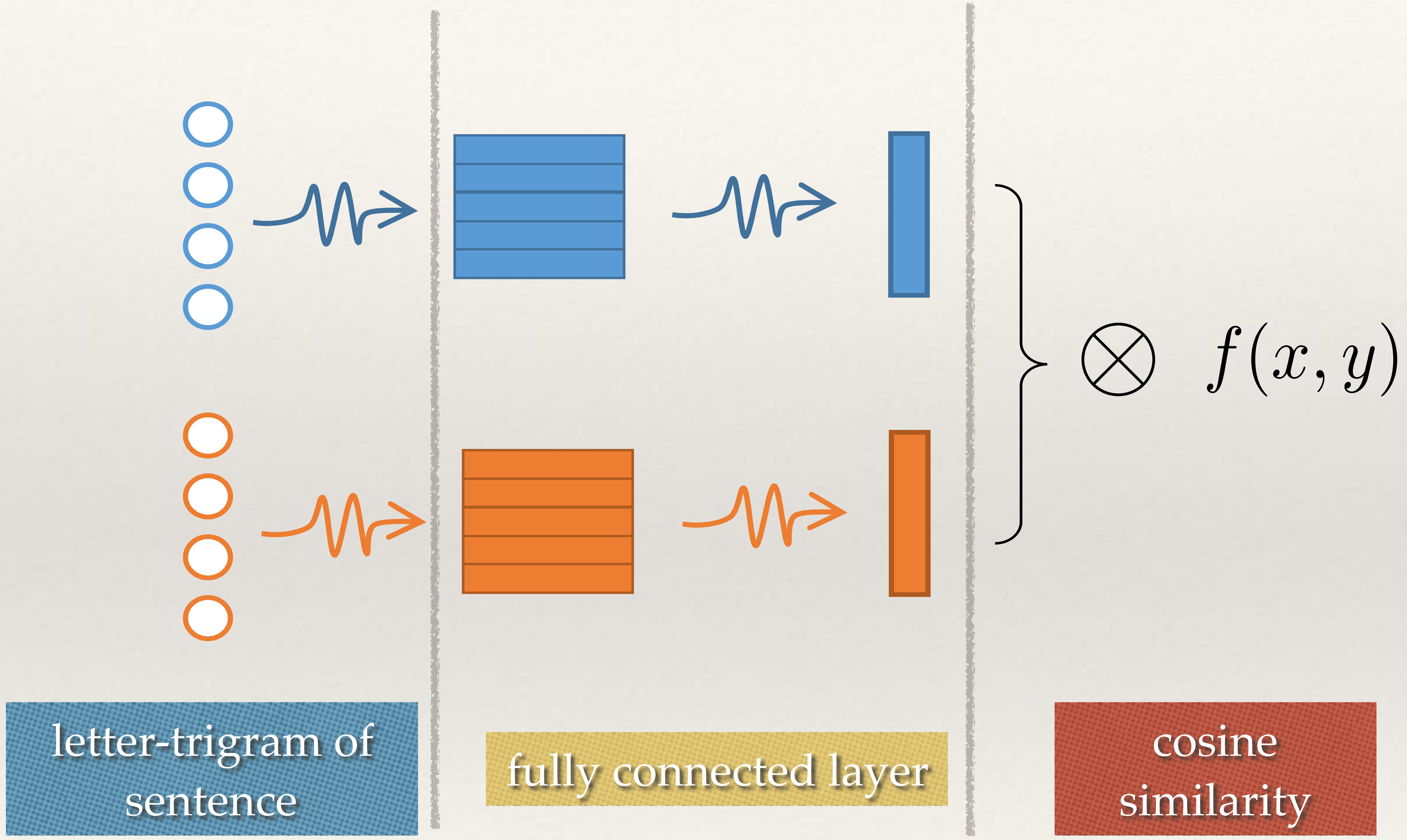
- Step 1: Composite sentence representation $\phi(\mathrm{x})$
- Step 2: Matching between the representations $F(\phi(\mathrm{x}), \phi(\mathrm{y}))$



Composition focused methods will be discussed

- Based on DNN
  - DSSM: Learning Deep Structured Semantic Models for Web Search using Click-through Data (Huang et al., CIKM '13)
- Based on CNN
  - CDSSM: A latent semantic model with convolutional-pooling structure for information retrieval (Shen Y et al., CIKM '14)
  - ARC I: Convolutional Neural Network Architectures for Matching Natural Language Sentences (Hu et al., NIPS '14)
  - CNTN: Convolutional neural tensor network architecture for community-based question Answering (Qiu et al., IJCAI '15)
- Based on RNN
  - LSTM-RNN: Deep Sentence Embedding Using the Long Short Term Memory Network: Analysis and Application to Information Retrieval (Palangi et al., TASLP '16)

Deep structured semantic model (DSSM)



Huang P-S, He X, Gao J, et al. Learning deep structured semantic models for web search using clickthrough data//Proceedings of the 22nd ACM international conference on CIKM. Amazon, India, 2013: 2333-2338

DSSM input: letter-trigram

- Bag of words representation
  - “candy store”: [0 0 0 1 0 0 0 1 0 0 0 …]
- Letter-trigram representation
  - “#candy# #store#” $\Longrightarrow$ #ca | can | and | ndy | dy# | #st | sto | tor | ore | re#
  - [0 0 1 0 0 … 0 1 0 1 … 0 0 …]
- Advantages:
  - Compact representation: # words: 500K $\Longrightarrow$ # letter-trigram: 30K
  - Generalize to unseen words
  - Robust to noisy inputs, e.g., misspelling, inflection …

DSSM sentence representation: DNN

Model: DNN for capturing the compositional sentence representation

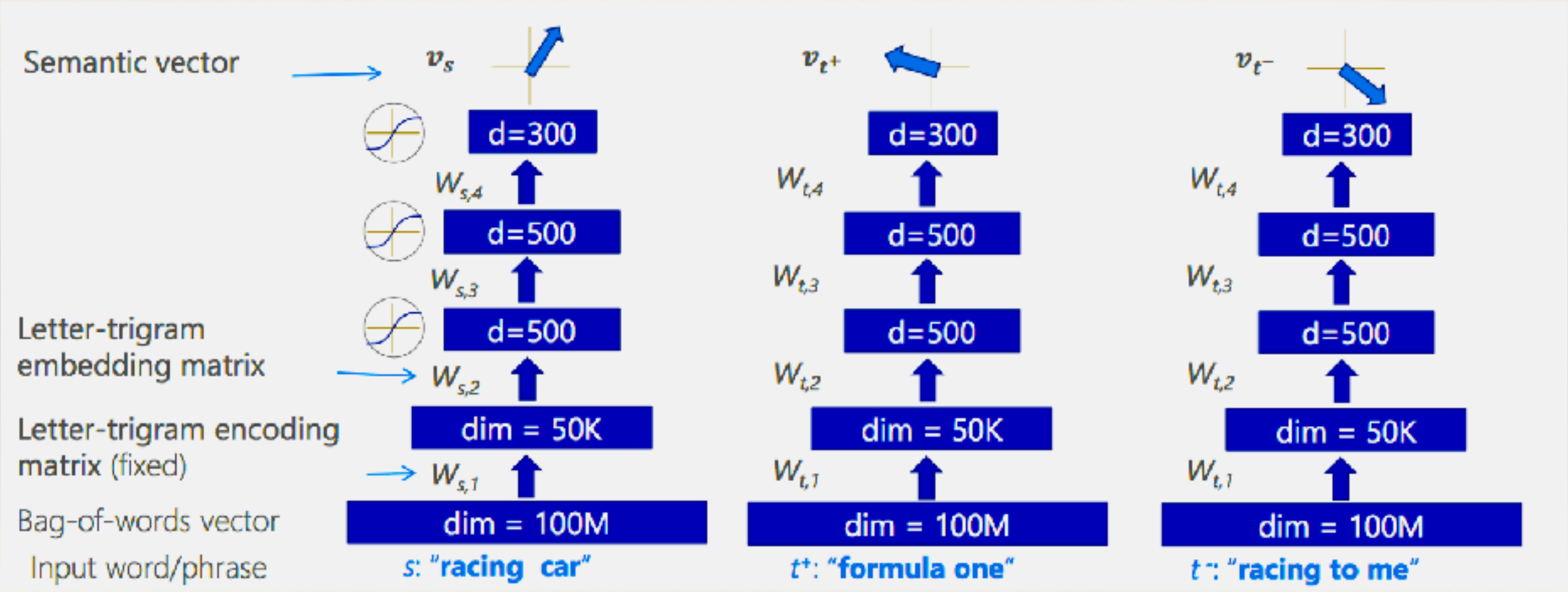


Figure from He et al., CIKM '14 tutorial

DSSM matching function

- Cosine similarity between semantic vectors

$$S = \frac{x^T \cdot y}{|x| \cdot |y|}$$

- Training
  - A query q and a list of docs $D = \{d^+, d_1^-, \dots, d_k^-\}$
  - d^+ relevant doc, $d_1^-, \dots, d_k^-$ irrelevant docs
  - Objective: $P(d^+|q) = \frac{\exp(\gamma \cos(q, d^+))}{\sum_{d \in D} \exp(\gamma \cos(q, d))}$
  - Optimizing with SGD

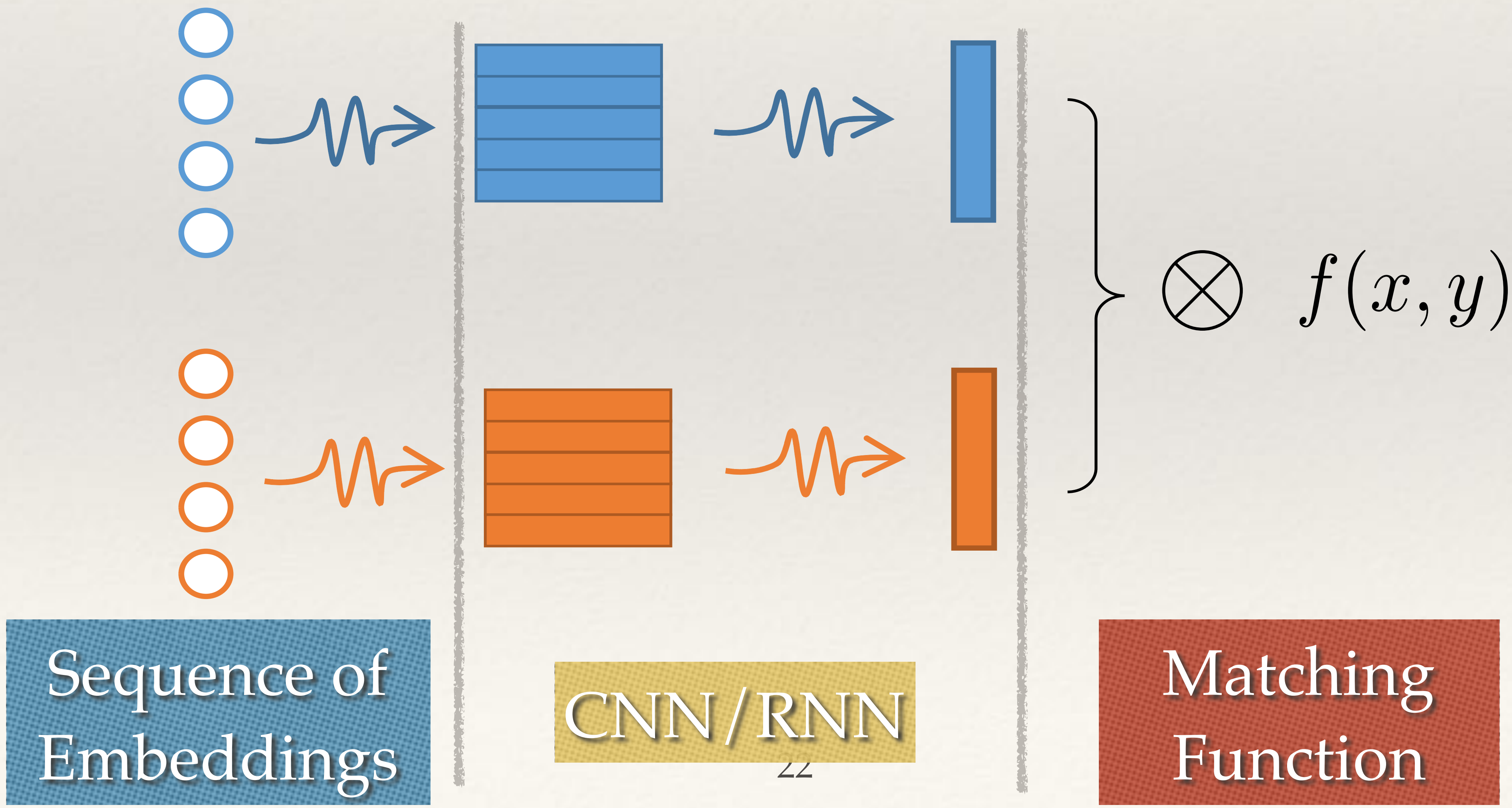
DSSM: short summary

- Input: sub-word units (i.e. letter-trigram) as input for scalability and generalizability
- Representation: mapping sentences to vectors (i.e. DNN): semantically similar sentences close to each other
- Matching: cosine similarity as the matching function

- Problem: bag of letter-trigrams as inputs, **the order information of words ignored**

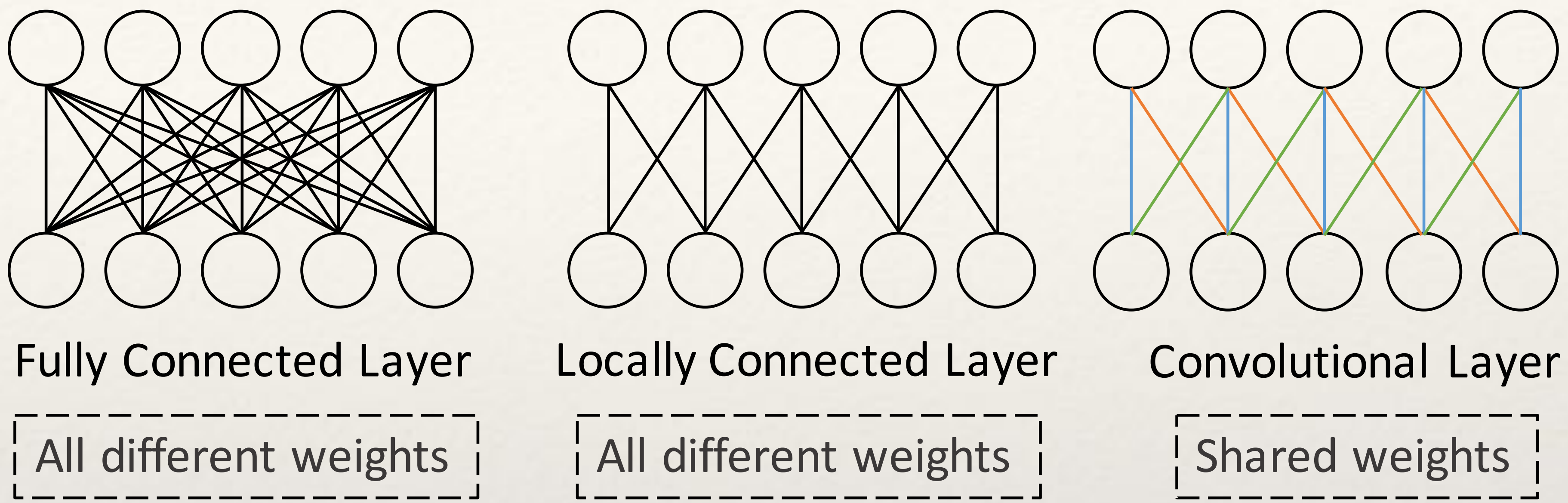
Capturing the order information

- Input: **word sequence** rather than bag of letter-trigrams
- Model:
  - **Convolutional** based methods can keep **locally order**
  - **Recurrent** based methods can keep **long dependence relations**

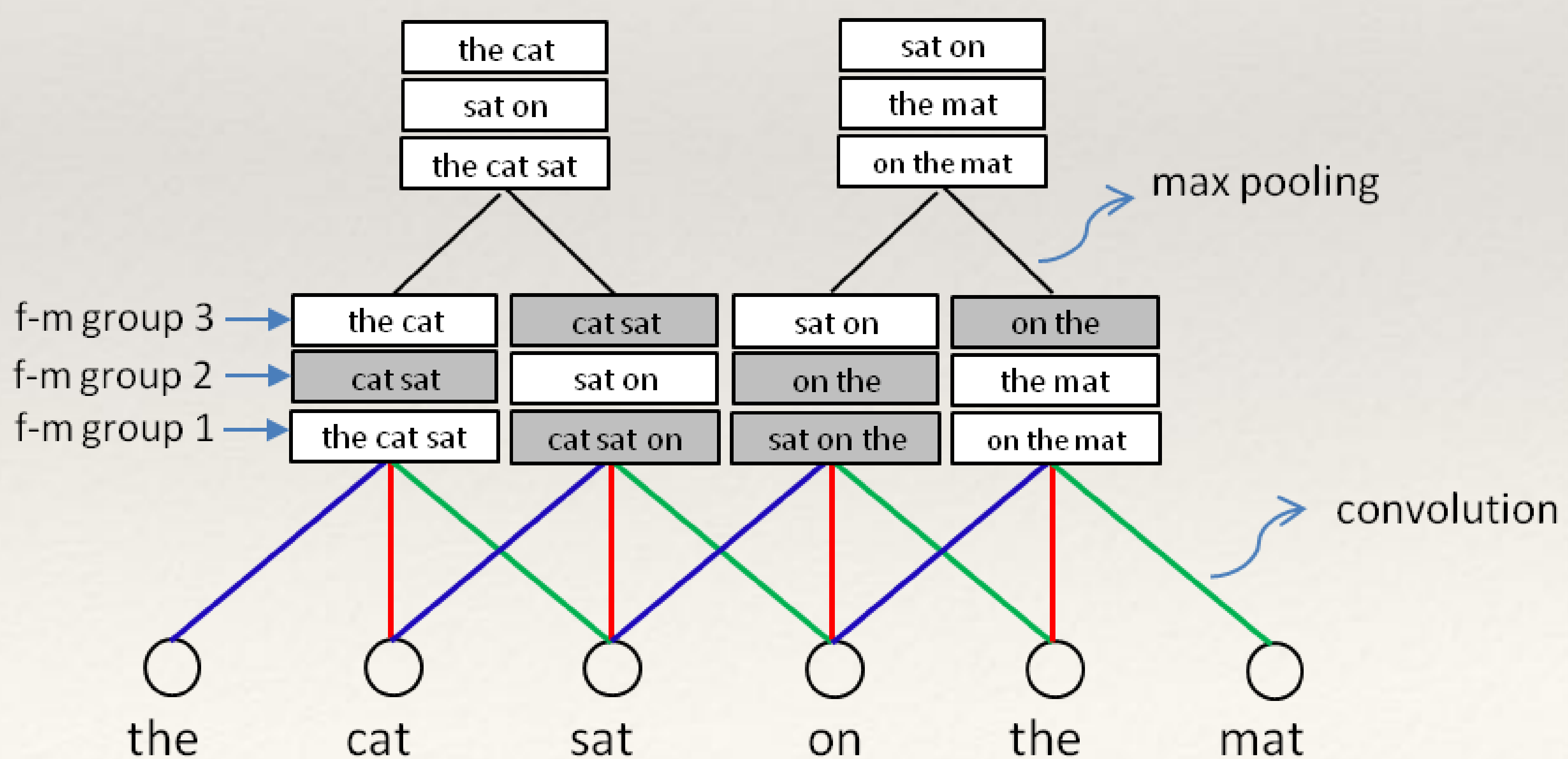


CNN can model the order information

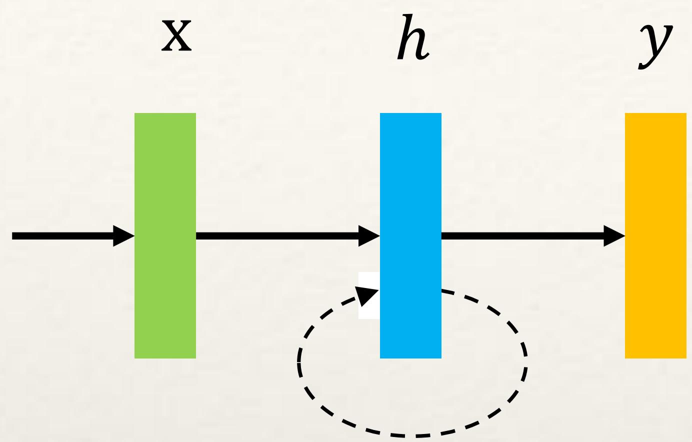
- Inspired by the cat's visual cortex [Hubel68].



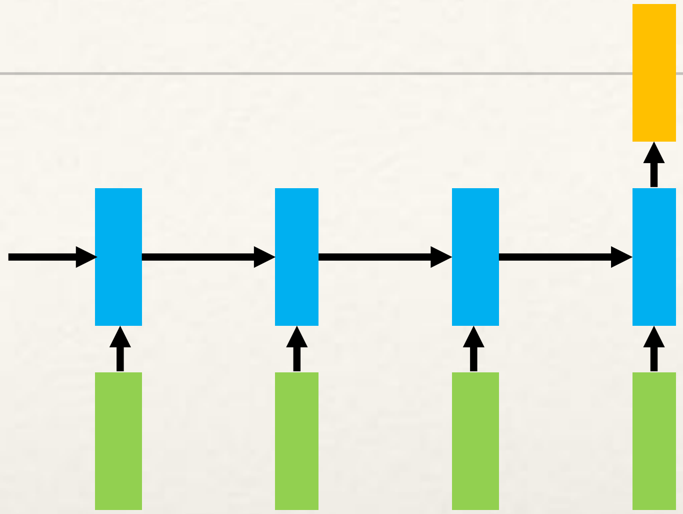
- Convolution & max pooling operations on text



RNN can model the order information



RNN – Self Recurrent Link



Expand RNN

- RNNs implement dynamical systems
- RNNs can approximate arbitrary dynamical systems with arbitrary precision
- Training: back propagation through time

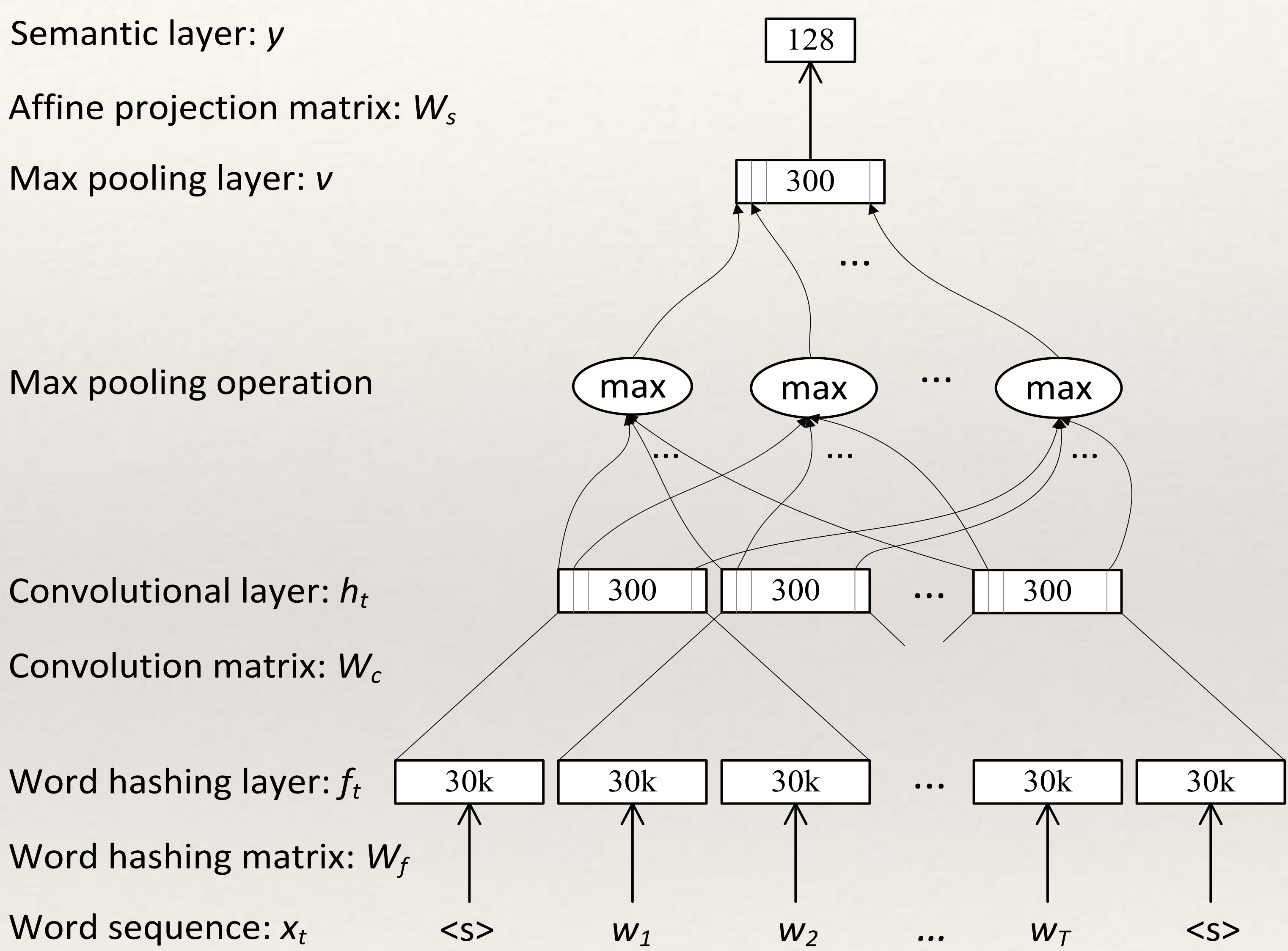
$$s(t) = f(Uw(t) + Ws(t-1) + b)$$

- Two popularly used variations: long-short term memory (LSTM) and gated recurrent unit (GRU)

Using CNN: CDSSM

- Input: encode **each word** as bag of letter-trigram
- Model: the convolutional operation in CNN compacts each **sequence of k words**

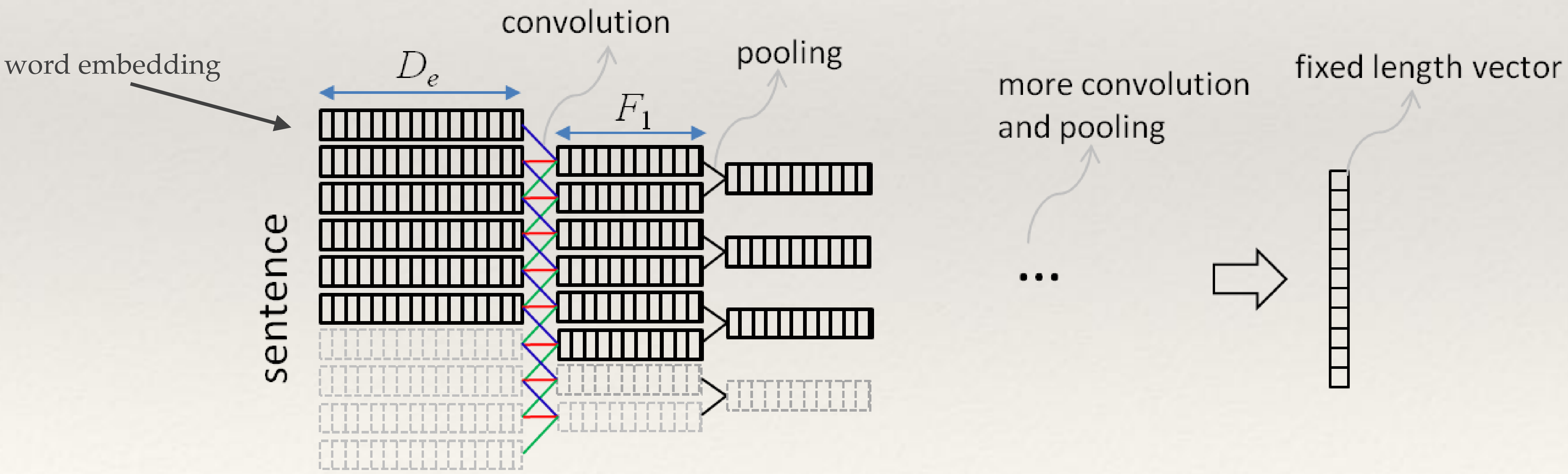
each word's letter-trigrams



Shen Y, He X, Gao J, et al. A latent semantic model with convolutional-pooling structure for information retrieval//Proceedings of the 23rd ACM international conference on CIKM. New York, USA, 2014: 101-110.

Using CNN: ARC-I / CNTN

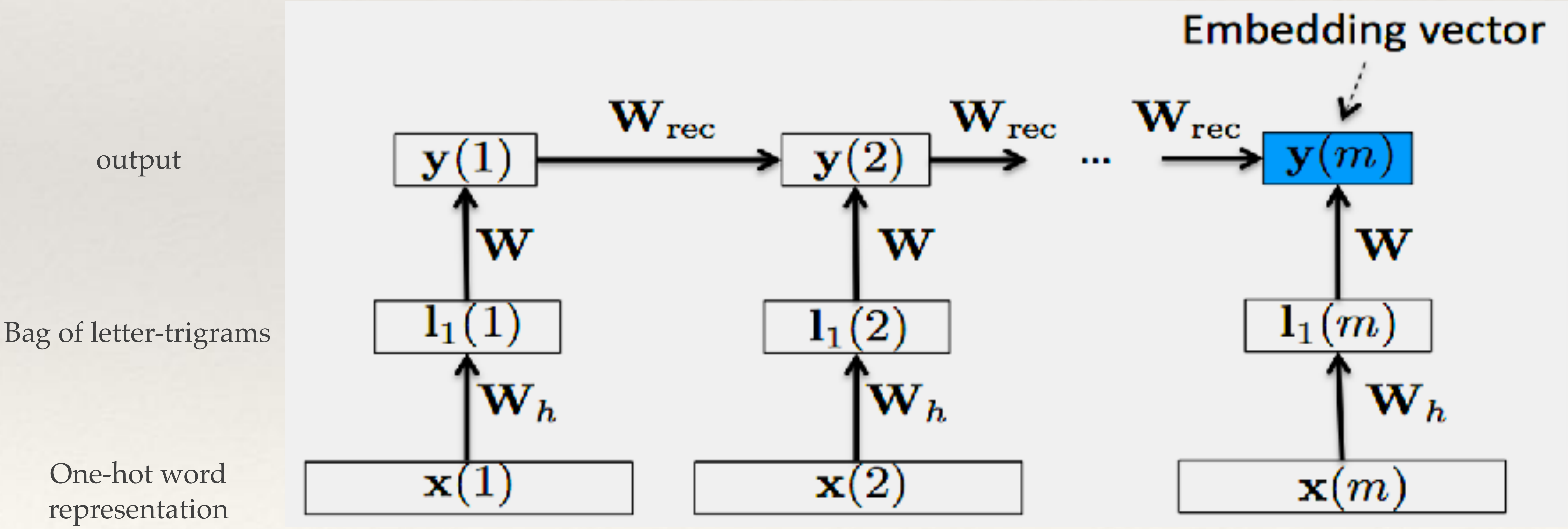
- Input: sequence of word embeddings
  - Word embeddings from word2vec model train on large dataset
- Model: CNN compacts each sequence of k words



Qiu X, Huang X. Convolutional neural tensor network architecture for community-based question answering//Proceedings of the 24th (IJCAI), Buenos Aires, Argentina, 2015: 1305-1311.

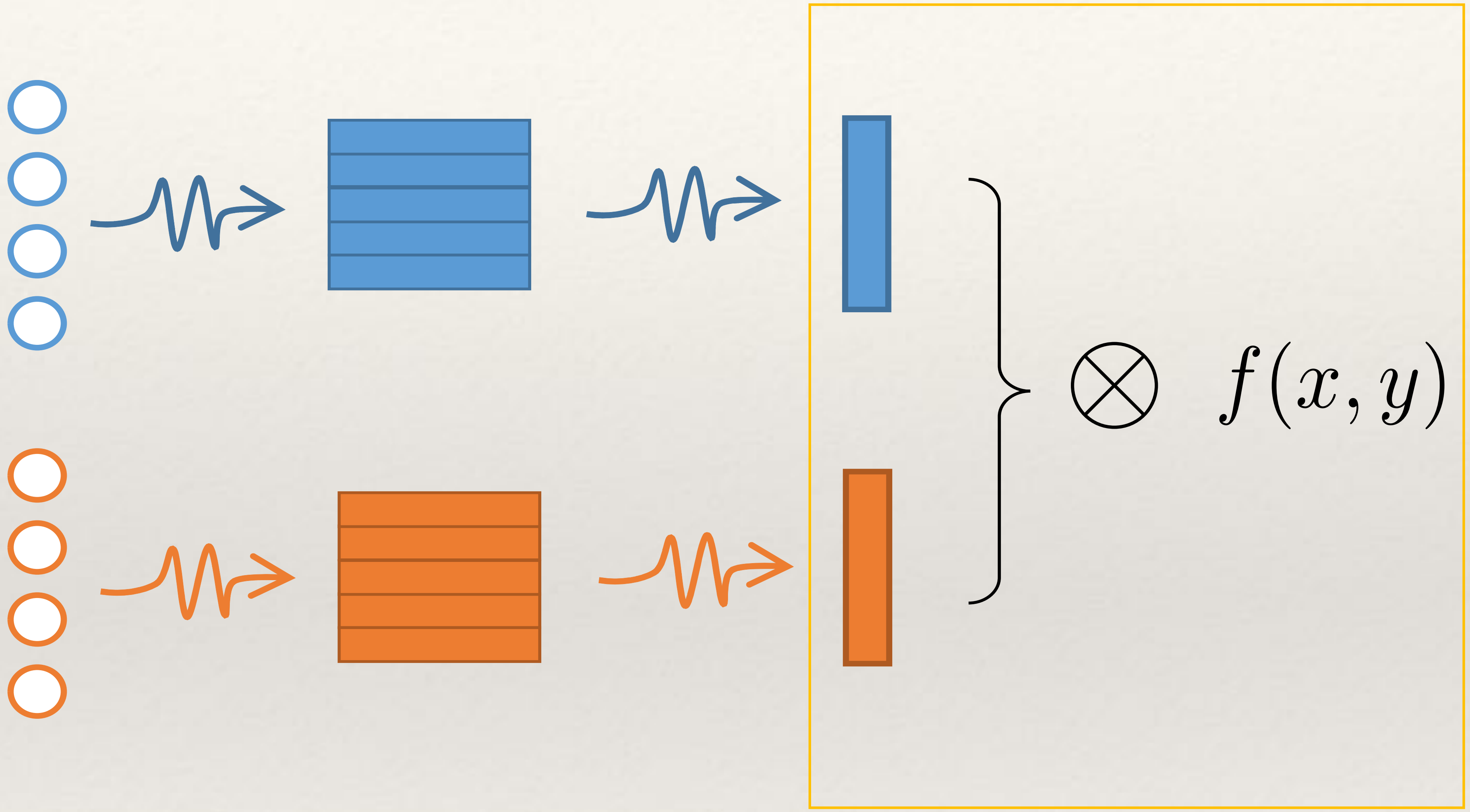
Using RNN: LSTM-RNN

- Input: sequence letter trigrams
- Model: long-short term memory (LSTM)
    - The last output as the sentence representation



Palangi H, Deng L, Shen Y, et al. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2016, 24(4): 694-707.

Matching functions



Heuristic: cosine, dot product
Learning: MLP, Neural tensor networks

Matching functions (cont')

- Given the representations of two sentences: *x* and *y*.
- Similarity between these two embeddings:
  - Cosine Similarity (DSSM, CDSSM, RNN-LSTM)

$$S = \frac{x^T \cdot y}{|x| \cdot |y|}$$

  - Dot Product

$$S = x^T \cdot y$$

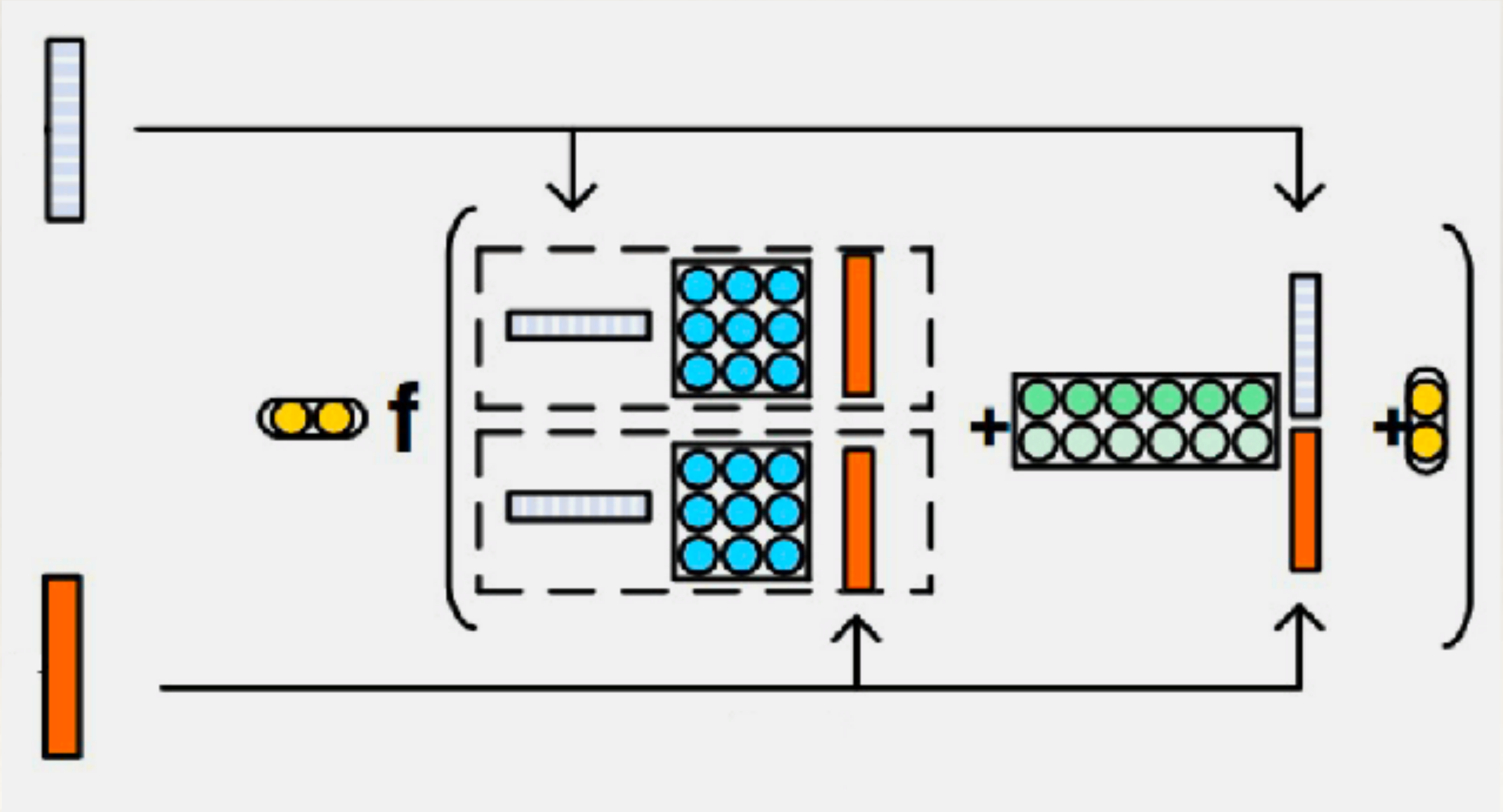
  - Multi-Layer Perception (ARC-I)

$$S = W_2 \cdot \left(W_1 \cdot \begin{bmatrix} x \\ y \end{bmatrix} + b_1 \right) + b_2$$

Matching functions (cont')

- Neural Tensor Network (CNTN)

$$S = u^T f(x^T M^{[1:r]} y + V \begin{bmatrix} x \\ y \end{bmatrix} + b)$$



Performance evaluation based on QA task

- Dataset: Yahoo! Answers
- Contain 60,564 (question, answer) pairs
- Example:
    - *Q: How to get rid of memory stick error of my sony cyber shot?*
    - *A: You might want to try to format the memory stick but what is the error message you are receiving.*

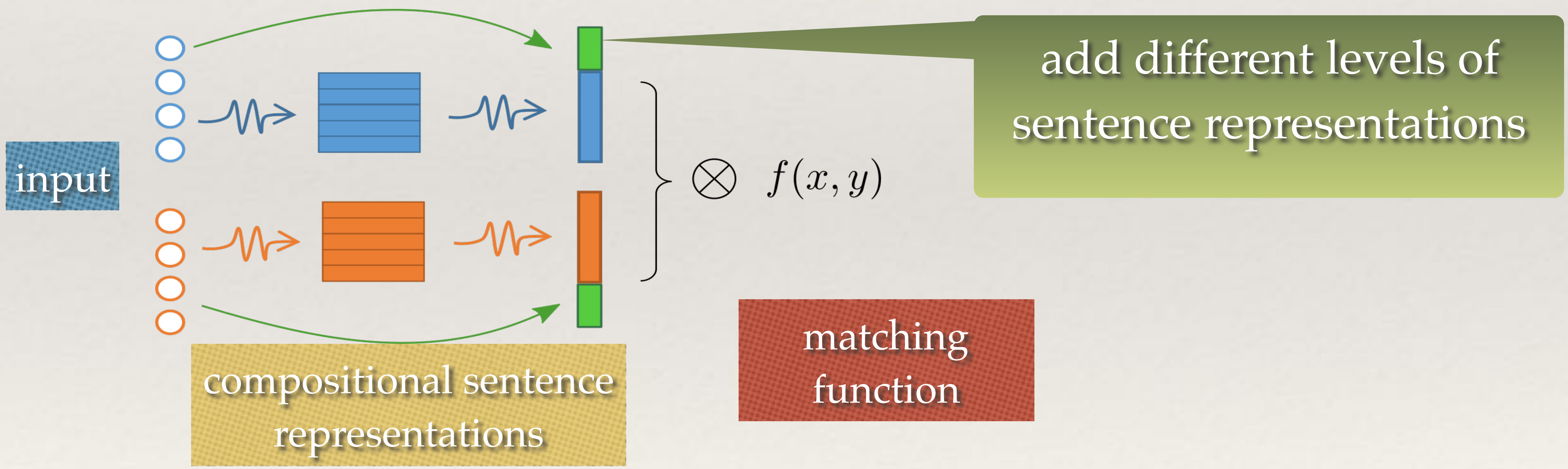
Experimental results

	Model	P@1	MRR
Statistic Traditional	Random	0.200	0.457
	BM25	0.579	0.726
Comosition Focused	ARC-I	0.581	0.756
	CNTN	0.626	0.781
	LSTM-RNN	0.690	0.822

- Composition focused methods outperformed the baselines
    - Semantic representation is important
- LSTM-RNN is the best performed method
    - Modeling the order information does help

Extensions to composition focused methods

- Problem: sentence representations are too coarse to conduct exact text matching tasks
  - Experience in IR: combining topic level and word level matching signals usually achieve better performances
- Add fine-grained matching signals in composition focused methods



- MultiGranCNN: An Architecture for General Matching of Text Chunks on Multiple Levels of Granularity. (Yin W, Schütze T, Hinrich. ACL2015)
- U-RAE: Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection, (Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, Christopher D. Manning, NIPS2011)
- MV-LSTM: A Deep Arhitecture for Semantic Matching with Multiple Positional Sentence Representations. (Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. AAAI 2016)

Performance evaluation on QA task

	Model	P@1	MRR
Statistic Traditional	Random	0.200	0.457
	BM25	0.579	0.726
Comosition Focused	ARC-I	0.581	0.756
	CNTN	0.626	0.781
	LSTM-RNN	0.690	0.822
	uRAE	0.398	0.652
	MultiGranCNN	0.725	0.840
	MV-LSTM	0.766	0.869

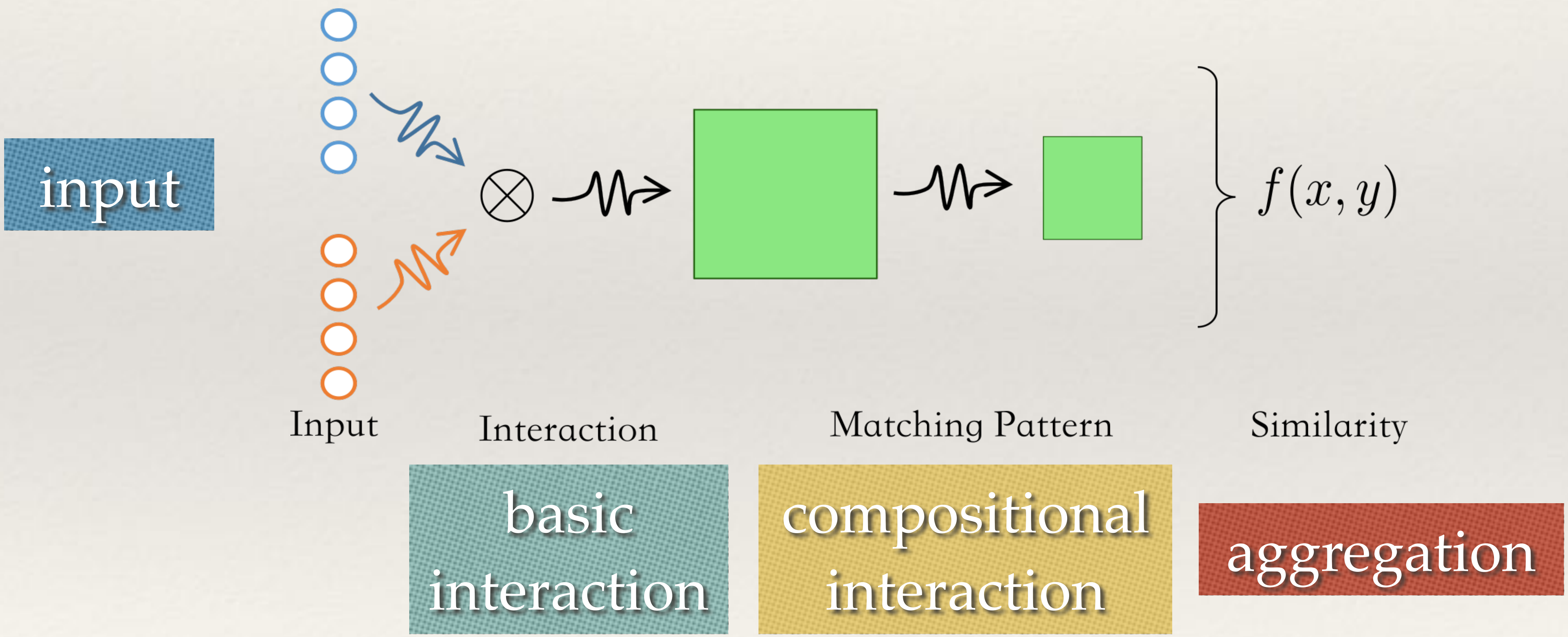
- MultiGranCNN and MV-LSTM achieved the best performance
  - Fine-grained matching signals are useful

Outline

- Problems with direct methods
- Deep matching models for text
  - Composition focused
  - Interaction focused
- Summary

Interaction focused methods

- Step 1: Construct basic low-level interaction signals
- Step 2: Aggregate matching patterns

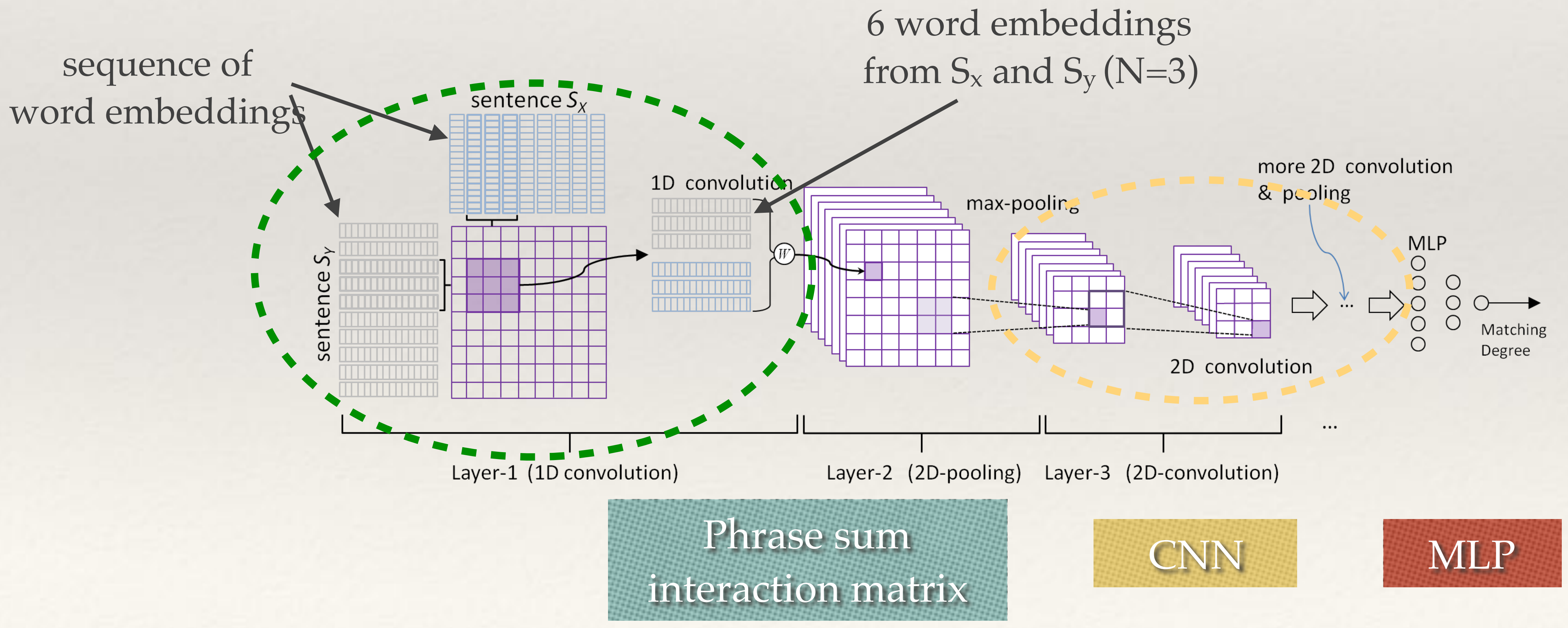


Interaction focused methods will be discussed

- ARC II: Convolutional Neural Network Architectures for Matching Natural Language Sentences (Hu et al., NIPS'14)
- MatchPyramid: Text Matching as Image Recognition. (Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. AAAI 2016)
- Match-SRNN: Modeling the Recursive Matching Structure with Spatial RNN. (Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. IJCAI 2016)

ARC-II

- Let two sentences meet before their own high-level representations mature
- Basic interaction: phrase sum interaction matrix
- Compositional interaction: CNN to capture the local interaction structure
- Aggregation: MLP



Hu B, Lu Z, Li H, et al. Convolutional neural network architectures for matching natural language sentences//Proceedings of the Advances in NIPS, Montreal, Canada, 2014: 2042-2050.

ARC-II (cont')

- Order preservation
  - Both the convolution and pooling have order preserving property

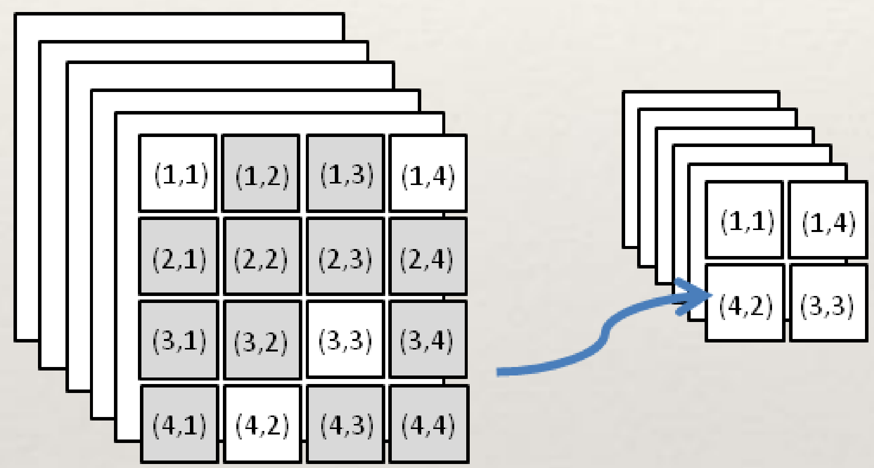
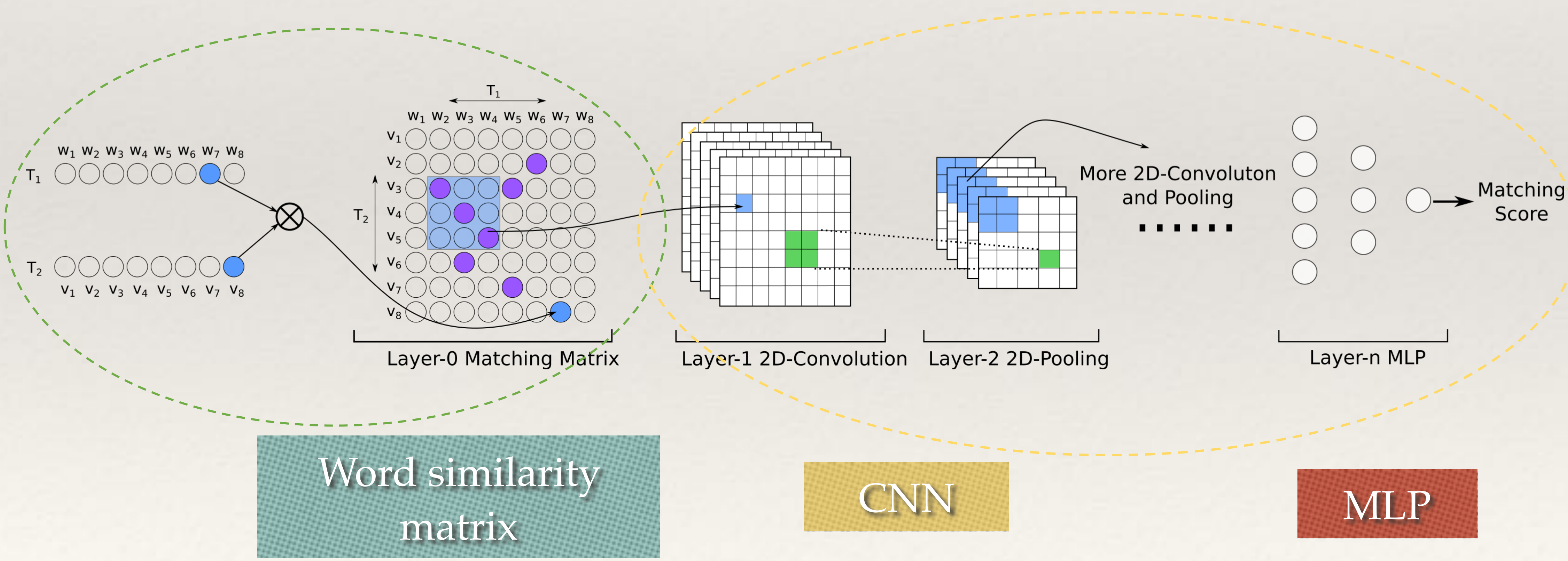


Figure 5: Order preserving in 2D-pooling.

- However, the **word level matching signals are lost**
  - 2-D matching matrix is construct based on the embedding of the words in two N-grams

MatchPyramid

- Inspired by image recognition task
- Basic interaction: word-level matching matrix
- Compositional interaction: hierarchical convolution
- Aggregation: MLP



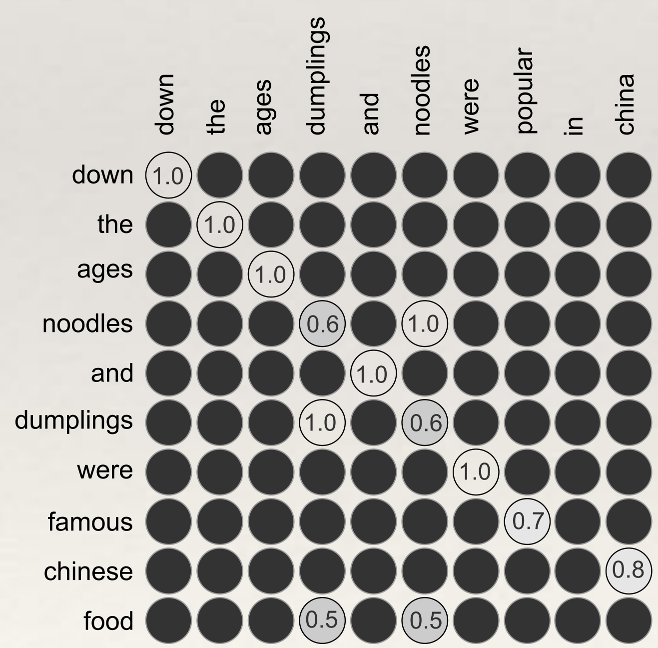
Pang L, Lan Y, Guo J, et al. Text matching as image recognition//Proceedings of the 30th AAAI Conference on Artificial Intelligence. Phoenix, USA, 2016: 2793-2799

MatchPyramid: the matching matrix

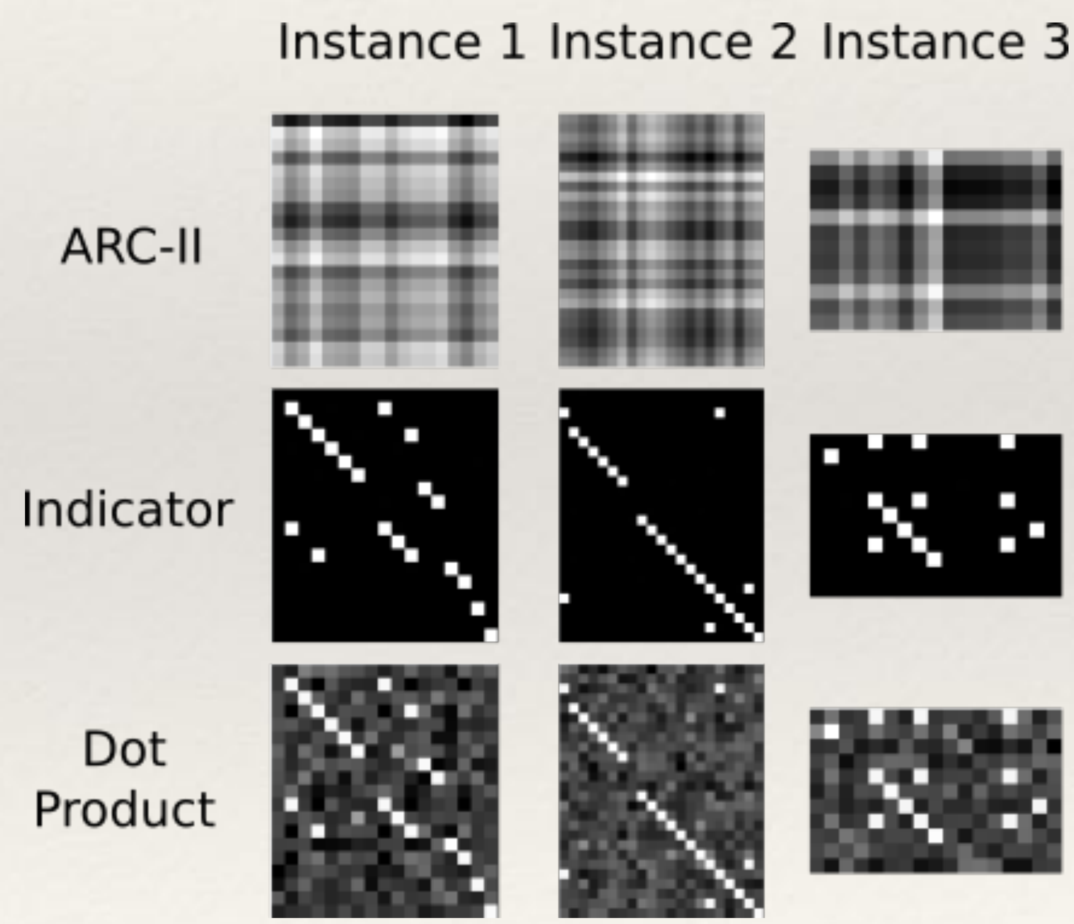
- Basic interaction: word similarity matrix
    - Strength of the word-level matching
    - Positions of the matching occurs

$$\mathrm{M}_{ij} = w_i \otimes v_j$$

(a) Indicator

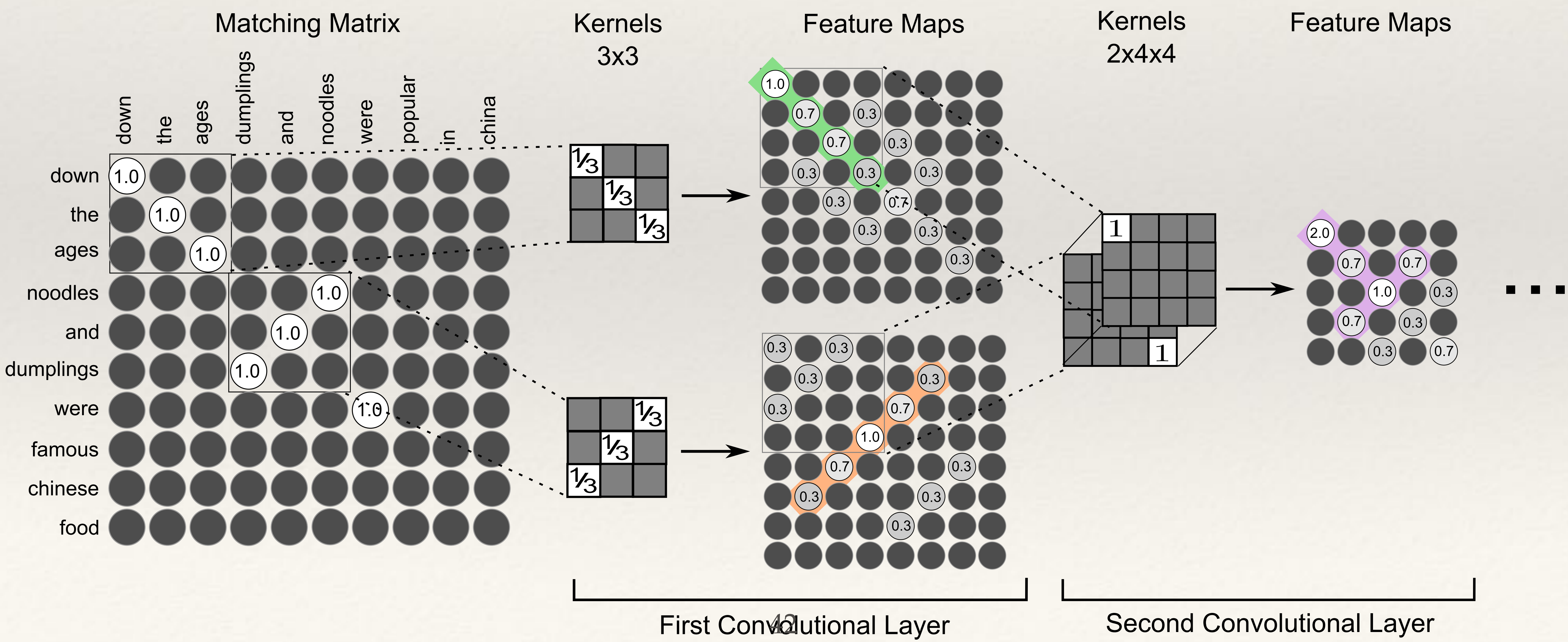


(b) Cosine



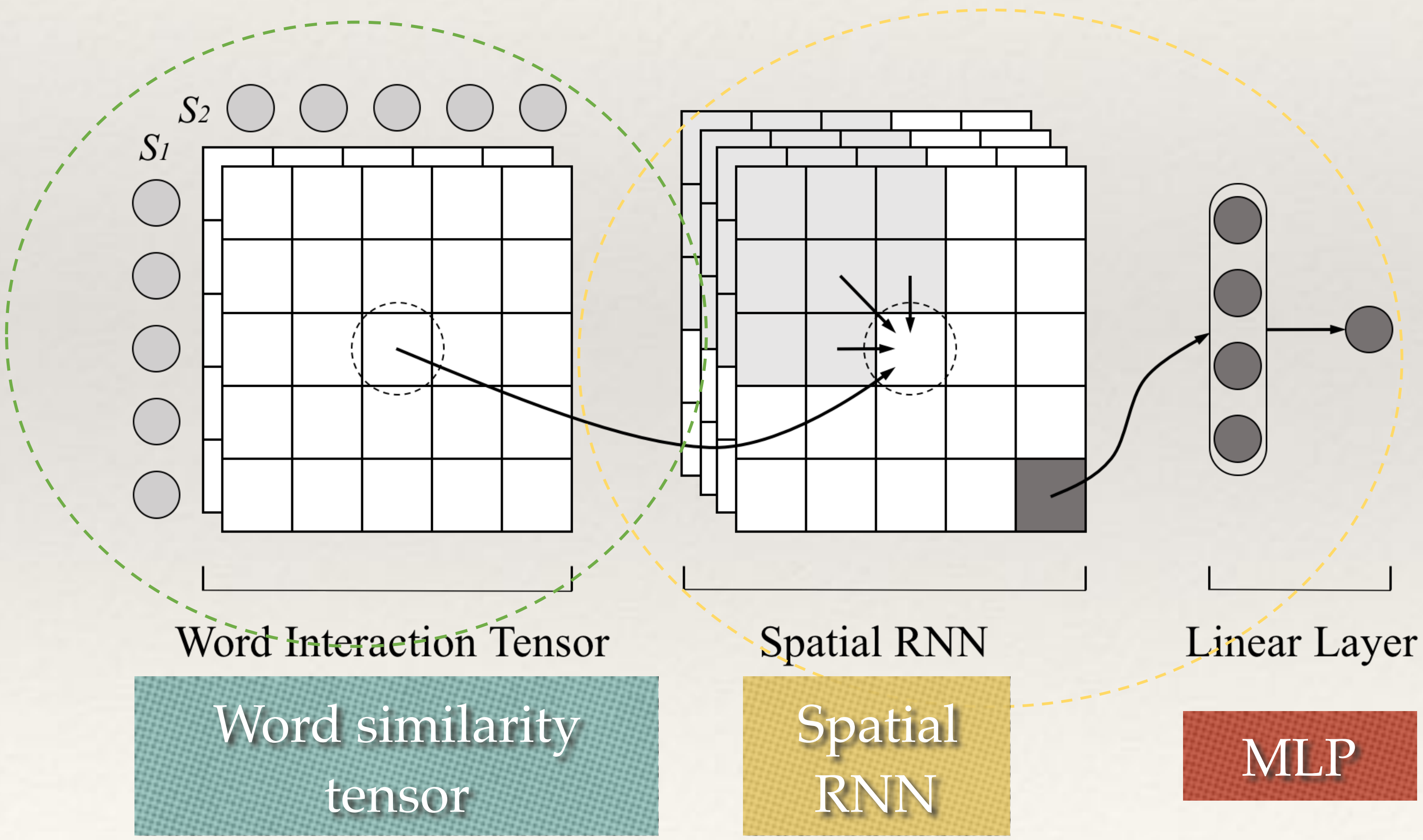
MatchPyramid: the hierarchical convolution

- Compositional interaction: CNN constructs different levels of matching patterns, based on word-level matching signals



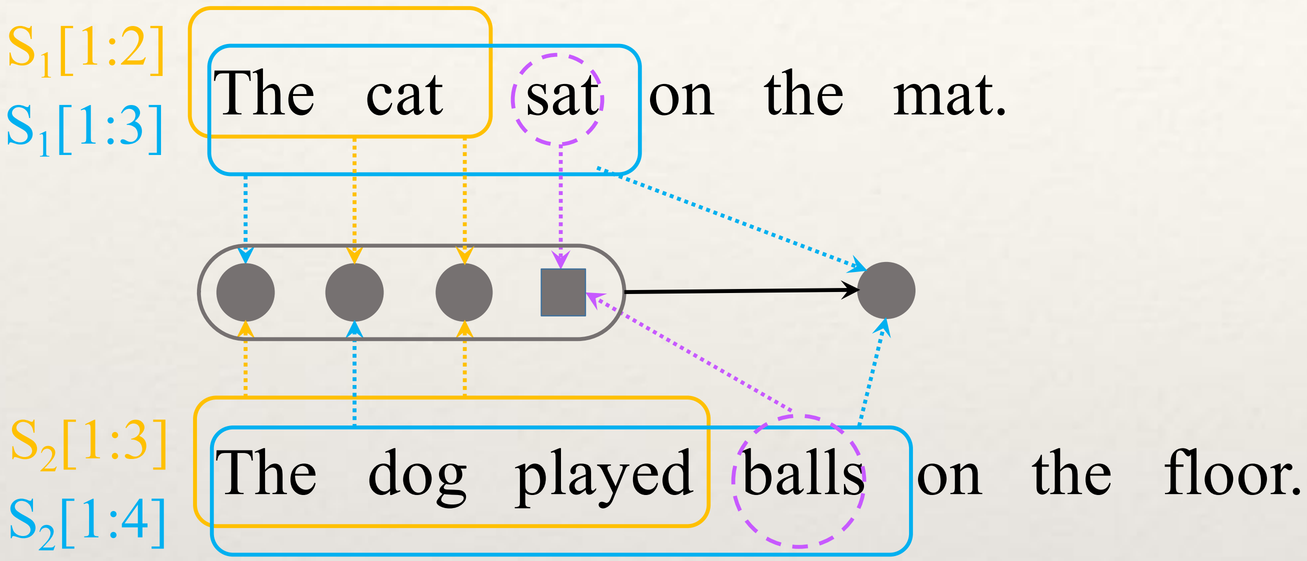
Match-SRNN

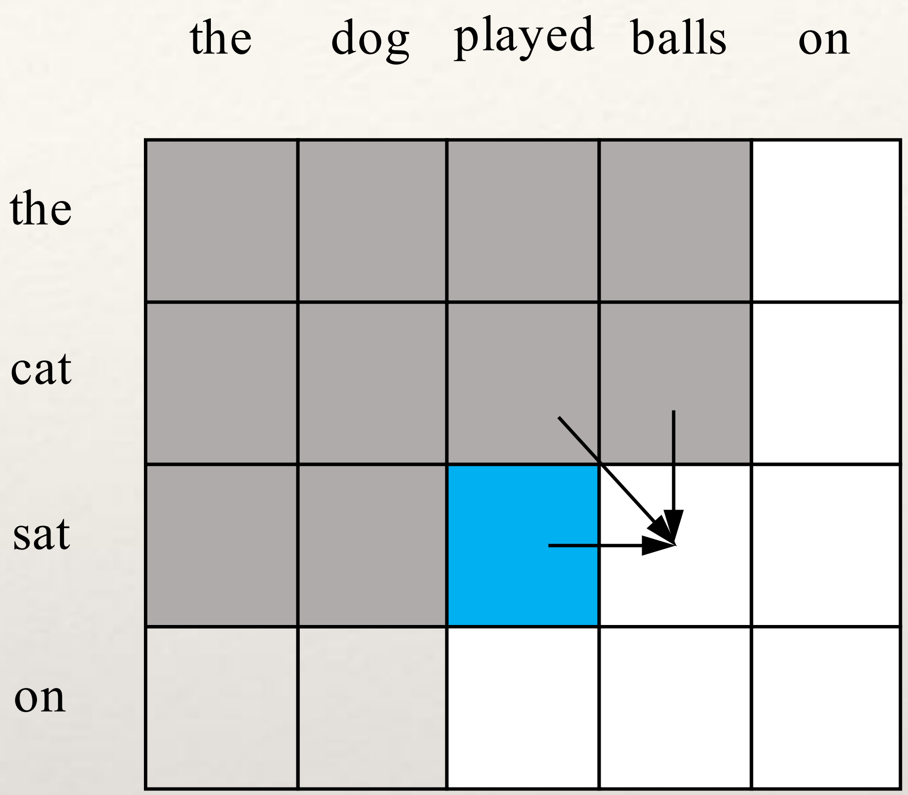
- Spatial recurrent neural network (SRNN) for text matching
- Basic interaction: word similarity tensor
- Compositional interaction: recursive matching
- Aggregation: MLP



Wan S, Lan Y, Guo J, et al. Match-SRNN: Modeling the recursive matching structure with spatial RNN//Proceedings of the 25th IJCAI, New York, US, 2016: 1022-1029.

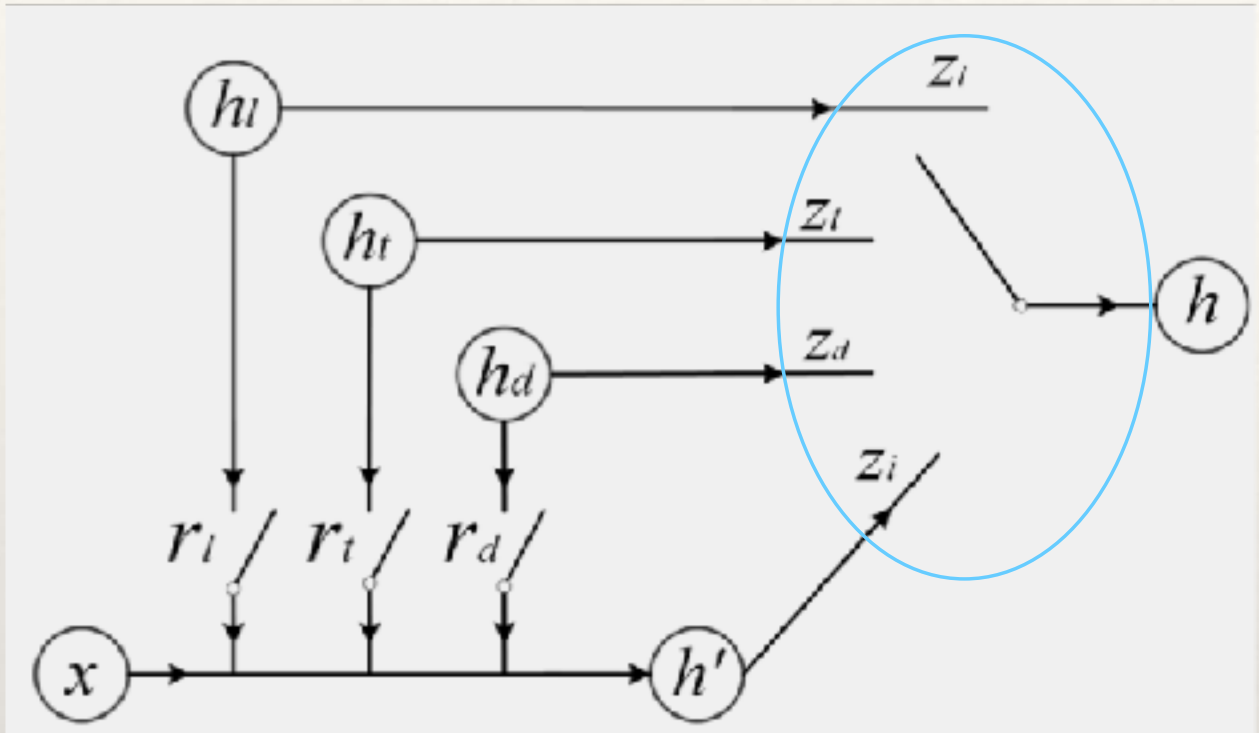
Match-SRNN: recursive matching structure





- Matching scores are calculated recursively (from top left to bottom right)
- All matchings between sub sentences have been utilized
  - sat ←→ balls
  - The cat ←→ the dog played
  - The cat ←→ The dog played balls
  - The cat sat ←→ The dog played

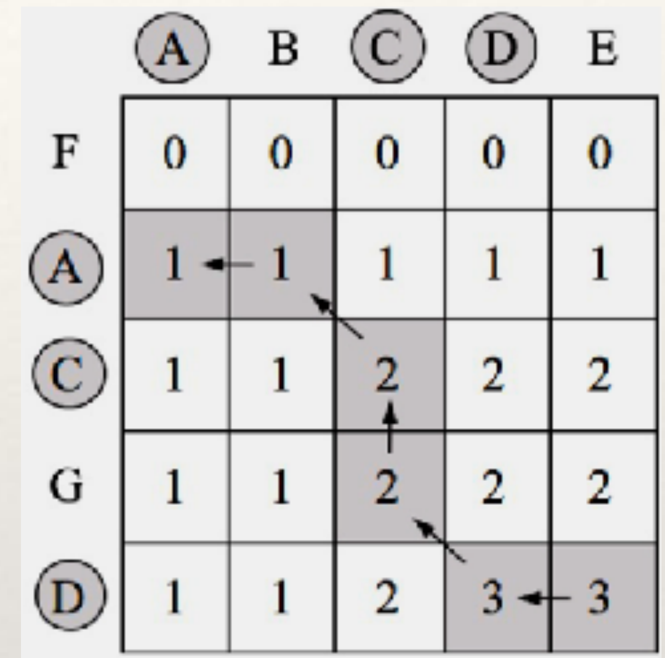
Using spatial GRU (two dimensions)



Softmax function is used to select connections among these four choices softly

$$
\begin{aligned}
q^T &= [h_{i-1,j}^T, h_{i,j-1}^T, h_{i-1,j-1}^T, s_{ij}^T]^T, \\
r_l &= \sigma(W^{(r_l)} q + b^{(r_l)}), \\
r_t &= \sigma(W^{(r_t)} q + b^{(r_t)}), \\
r_d &= \sigma(W^{(r_d)} q + b^{(r_d)}), \\
r^T &= [r_l^T, r_t^T, r_d^T]^T, \\
z_i^{'} &= W^{(z_i)} q + b^{(z_i)}, \\
z_l^{'} &= W^{(z_l)} q + b^{(z_l)}, \\
z_t^{'} &= W^{(z_t)} q + b^{(z_t)}, \\
z_d^{'} &= W^{(z_d)} q + b^{(z_d)}, \\
&\boxed{[z_i, z_l, z_t, z_d] = \mathrm{SoftmaxByRow}([\mathrm{z}_\mathrm{i}^{'}, \mathrm{z}_\mathrm{l}^{'}, \mathrm{z}_\mathrm{t}^{'}, \mathrm{z}_\mathrm{d}^{'}]),} \\
h_{i,j}^{'} &= \phi(W s_{ij} + U(r \odot [h_{i,j-1}^T, h_{i-1,j}^T, h_{i-1,j-1}^T]^T) + b), \\
h_{i,j} &= z_l \odot h_{i,j-1} + z_t \odot h_{i-1,j} + z_d \odot h_{i-1,j-1} + z_i \odot h_{i,j}^{'}.
\end{aligned}
$$

Connection to LCS



- Longest common sub-sequence (LCS)
  - S1: A B C D E
  - S2: F A C G D
  - LCS: A C D
- Solving LCS with dynamic programming (DP)
  - Step function: $c[i, j] = \max(c[i, j-1], c[i-1, j], c[i-1, j-1] + \mathbb{I}_{\{x_i = y_j\}})$
  - Backtrace: depends on the selection of "max" operation

Connection to LCS

- Match-SRNN can be explained with(LCS)
- Simplified Match-SRNN
  - Only exact word-level matching signals
  - Remove the reset gate r and set hidden dimension to 1

$$h_{ij} = z_l \cdot \boxed{h_{i,j-1}} + z_t \cdot \boxed{h_{i-1,j}} + z_d \cdot \boxed{h_{i-1,j-1} + z_i \cdot h'_{ij}}$$

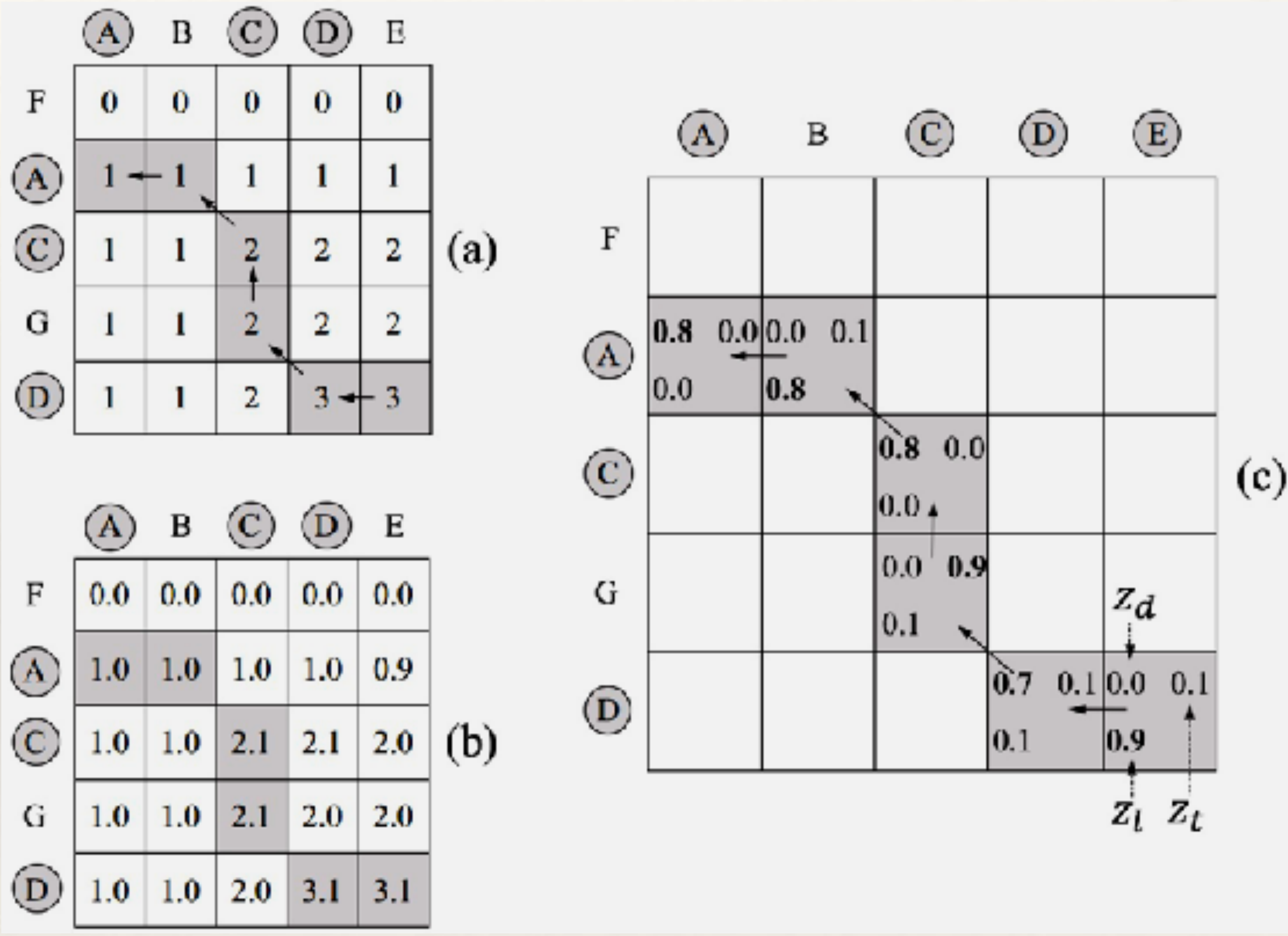
- Simplified Match-SRNN simulates LCS

$$c[i,j] = \max(\boxed{c[i,j-1]}, \boxed{c[i-1,j]}, \boxed{c[i-1,j-1] + \mathbb{I}_{\{x_i = y_j\}})}$$

  - Since that *z* is obtained by SOFTMAX
  - Backtrace by the value of *z* in simplified Match-SRNN

Simulation

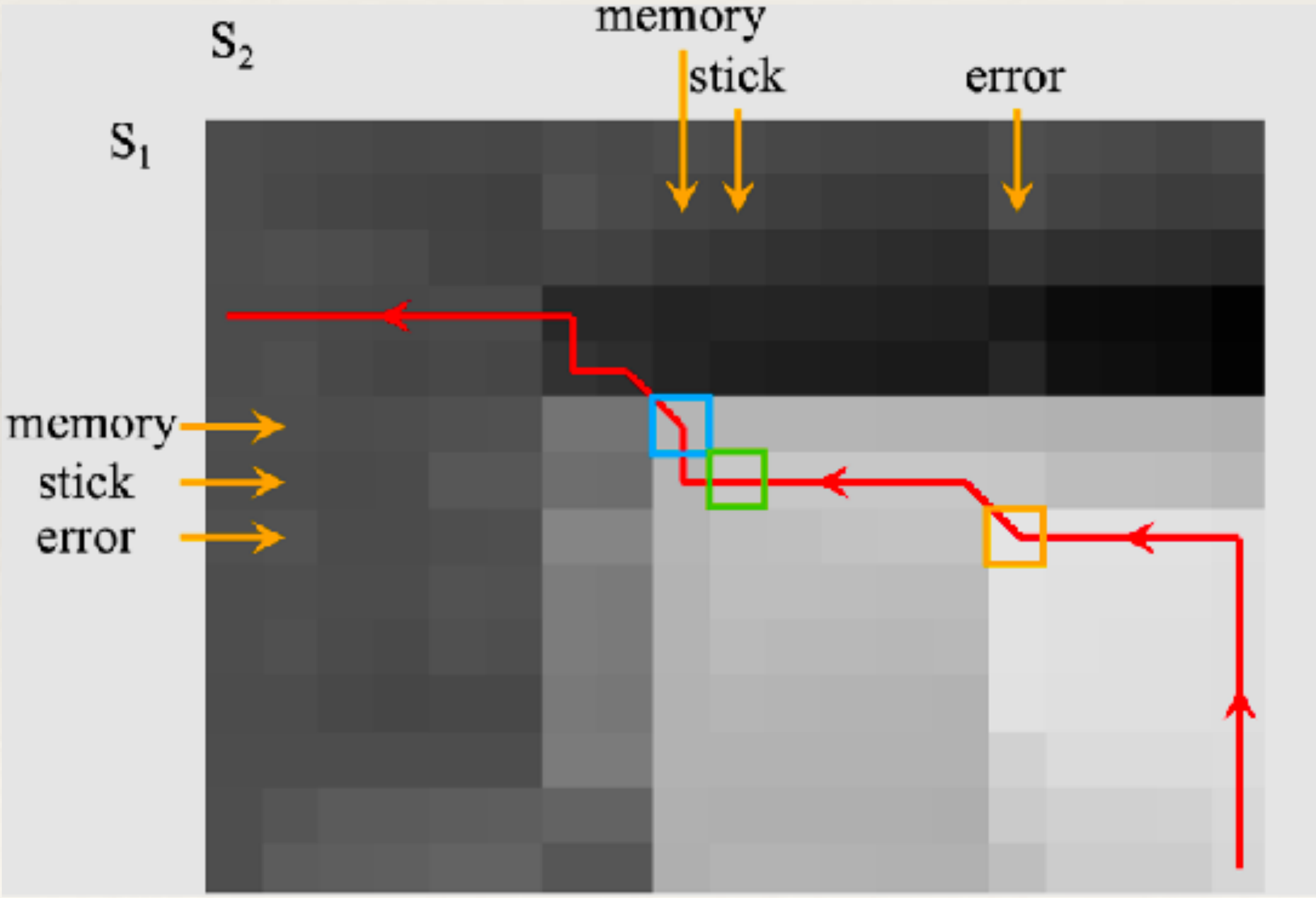
- Simulation data
  - Random sampled sequence
  - Ground truth obtained by DP
  - The label is the length of LCS



Match-SRNN simulates LCS well!

Real Data

- *Question*: *"How to get rid of memory stick error of my sony cyber shot?"*
- *Answer*: *"You might want to try to format the memory stick but what is the error message you are receiving."*



Performance evaluations on QA task

	Model	P@1	MRR
Statistic traditional	Random	0.200	0.457
	BM25	0.579	0.726
Composition focused	ARC-I	0.581	0.756
	CNTN	0.626	0.781
	LSTM-RNN	0.690	0.822
	uRAE	0.398	0.652
	MultiGranCNN	0.725	0.840
	MV-LSTM	0.766	0.869
Interaction focused	DeepMatch	0.452	0.679
	ARC-II	0.591	0.765
	MatchPyramid	0.764	0.867
	Match-SRNN	0.790	0.882

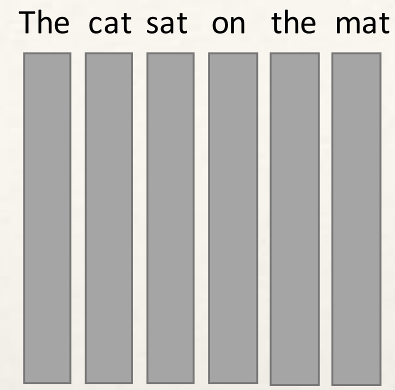
- Interaction focused methods outperformed the composition focused ones
  - Low level interaction (word level) signals are also important
- Match-SRNN performs the best
  - Powerful recursive matching structure

Outline

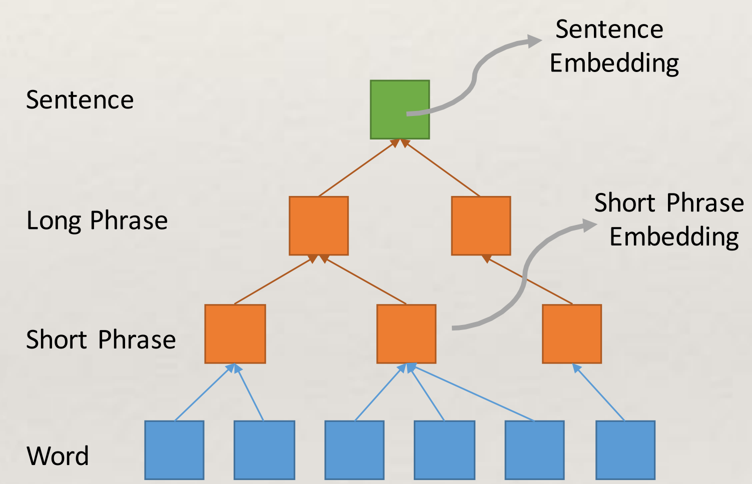
- Problems with direct methods
- Deep matching models for text
  - Composition focused
  - Interaction focused
- Summary

Summary

- Order of words

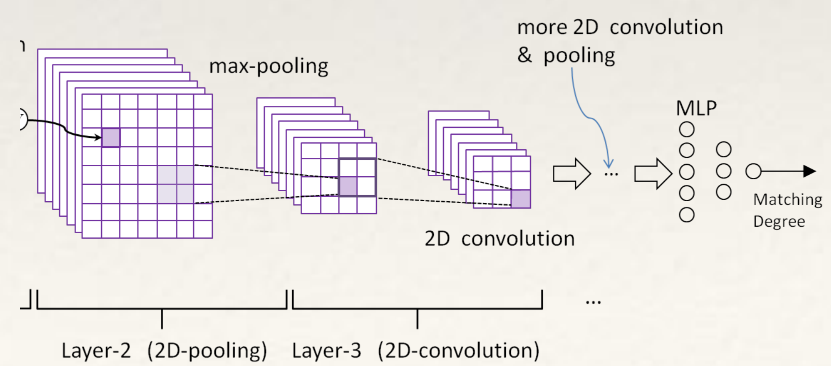


- Structure of sentence



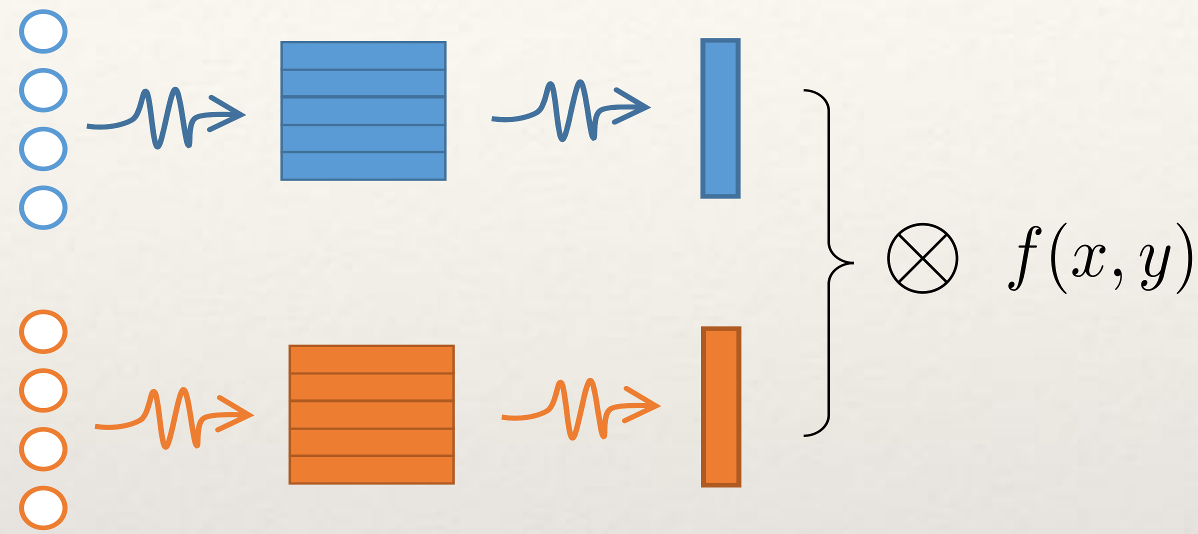
- Matching function

$$S = \frac{x^T \cdot y}{|x| \cdot |y|}$$

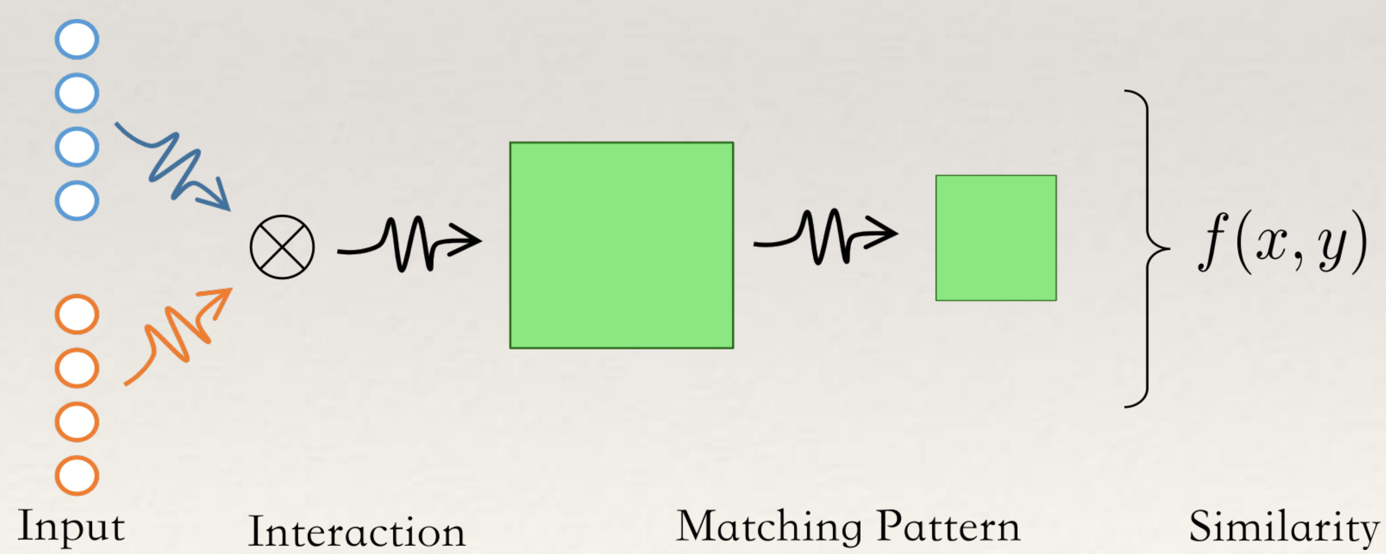


Summary (cont')

- Composition focused



- Interaction focused



Challenges

- Data: building benchmarks
  - Current: lack of large scale text matching data
  - Deep learning models have a lot of parameters to learn
- Model: leveraging human knowledge
  - Current: most models are purely data-driven
  - Prior information (e.g., large scale knowledge base and other information) should be helpful
- Application: domain specific matching models
  - Current: matching models are designed for a general goal (similarity)
  - Different applications have different matching goal
  - For example, in IR, relevance ≠ similarity

Thanks!